



L^AT_EX_pedia

LORENZO PANTIERI

Lorenzo Pantieri
*L^AT_EX*pedia
Copyright © 2017-2018

COLOPHON

Questo lavoro è stato realizzato con L^AT_EX su Mac usando ArsClassica, uno stile ispirato a *Gli elementi dello stile tipografico* di Robert Bringhurst.

I nomi commerciali, i loghi e i marchi registrati menzionati nella guida appartengono ai rispettivi proprietari, i pacchetti e le relative documentazioni ai rispettivi autori.

CONTATTI

✉ lorenzo.pantieri@gmail.com

La citazione è un utile sostituto dell'arguzia.

– Oscar Wilde

Dedicato a tutti gli appassionati di \LaTeX .

INDICE

PREFAZIONE XIII

INTRODUZIONE XVII

I PER COMINCIARE 1

1 STORIA E FILOSOFIA 3

1.1 Storia 3

1.2 Filosofia 5

2 INSTALLARE E AGGIORNARE 13

2.1 Ferri del mestiere 13

2.2 Windows 16

2.3 Mac 17

2.4 Linux 17

3 BASI 19

3.1 Realizzare un documento 19

3.2 Codifiche e lingue 22

3.3 File con cui si ha a che fare 24

3.4 File sorgente 24

3.5 Classi di documento 32

3.6 Gestire la pagina 34

3.7 Strutturare il documento 36

3.8 Stili di pagina 39

3.9 Indice generale, titoli e profondità 40

3.10 Riferimenti incrociati 43

3.11 Collegamenti ipertestuali e indirizzi elettronici 44

3.12 Pacchetti 45

3.13 Unità di misura tipografiche 47

3.14 Documenti di grandi dimensioni 48

II TESTO 51

4 TESTO 53

4.1 Struttura del testo 53

4.2 Comporre i capoversi 53

4.3 Caratteri particolari e simboli 58

4.4 Modificare stile e corpo del font 60

4.5 Titoli e frontespizi 62

4.6 Note a margine e a piè di pagina 63

4.7 Evidenziare le parole 64

4.8 Ambienti testuali 65

4.9 Acronimi e glossari 70

5 ACRONIMI 71

5.1 Definire gli acronimi 71

5.2 Acronimi nel testo 72

5.3 Opzioni 72

III MATEMATICA E CODICI 75

- 6 MATEMATICA 77
 - 6.1 Formule in linea e in display 77
 - 6.2 Nozioni introduttive 80
 - 6.3 Operatori 87
 - 6.4 Parentesi 89
 - 6.5 Vettori e matrici 90
 - 6.6 Spezzare formule lunghe 92
 - 6.7 Raggruppare più formule 93
 - 6.8 Modificare stile e corpo del font 95
 - 6.9 Enunciati e dimostrazioni 97
 - 6.10 Altre scienze 100
- 7 CODICI 103
 - 7.1 Introduzione 103
 - 7.2 Impostazioni 106
 - 7.3 Personalizzazioni 107
 - 7.4 Tecniche avanzate 111

IV TABELLE, FIGURE E DISEGNI 115

- 8 TABELLE E FIGURE 117
 - 8.1 Strumenti fondamentali 117
 - 8.2 Oggetti in testo e fuori testo 118
 - 8.3 Tabelle 123
 - 8.4 Figure 140
 - 8.5 Disposizioni particolari 144
 - 8.6 Nuovi oggetti mobili 148
- 9 DISEGNI 151
 - 9.1 Introduzione 151
 - 9.2 Ferri del mestiere 152
 - 9.3 Disegnare il percorso 157
 - 9.4 Nodi 166
 - 9.5 Personalizzare il tratto 173
 - 9.6 Riempimenti 179
 - 9.7 Trasparenze 181
 - 9.8 Trasformazioni 182
 - 9.9 Approfondimenti 184
 - 9.10 Universo grafico 186
- 10 GRAFICI DI FUNZIONE 191
 - 10.1 Grafici e tipografia 191
 - 10.2 Ferri del mestiere 191
 - 10.3 Funzioni espresse analiticamente 197
 - 10.4 Curve e superfici date per coordinate 208
 - 10.5 Curve e superfici campionate da file 209
 - 10.6 Altri sistemi di riferimento 210
 - 10.7 Diagrammi a barre 212

V	BIBLIOGRAFIA E INDICE ANALITICO	215
11	BIBLIOGRAFIA	217
11.1	Bibliografia manuale	217
11.2	Bibliografia automatica	219
12	INDICE ANALITICO	239
12.1	Indici semplici	239
12.2	Indici complessi	240
VI	PERSONALIZZAZIONI E REVISIONE	243
13	PERSONALIZZAZIONI	245
13.1	Comandi e ambienti personali	245
13.2	Personalizzare il testo	247
13.3	Specialità	249
13.4	Altre personalizzazioni	253
14	REVISIONE FINALE	257
14.1	Problemi orizzontali	257
14.2	Problemi verticali	258
14.3	Altre indicazioni	261
VII	DOCUMENTI PARTICOLARI	263
15	VIDEOPRESENTAZIONI	265
15.1	Introduzione	265
15.2	Fondamentali	265
15.3	Presentazioni personalizzate	275
16	CURRICULUM	277
16.1	Introduzione	277
16.2	Stile e contenuti	277
16.3	Indicazioni	278
16.4	Curriculum a regola d'arte	278
17	LETTERE	283
17.1	Lettere standard	283
17.2	Lettere conformi allo stile italiano	285
VIII	LINGUE ESOTICHE	289
18	LATINO E GRECO	291
18.1	Scrivere in latino	291
18.2	Scrivere in greco	293
19	CIRILLICO	301
19.1	Introduzione	301
19.2	Codifica	301
19.3	Caratteri speciali	302
20	EBRAICO E ARABO	305
20.1	Scrivere in ebraico	305
20.2	Scrivere in arabo	306

IX	SPECIALITÀ	309
21	CAPILETTERA	311
21.1	Introduzione	311
21.2	Personalizzare il capolettera	311
21.3	Scrivere in gotico	316
22	CLASSICTHESIS	317
22.1	Basi	317
22.2	Personalizzazioni	326
22.3	Proporzioni di pagina	332
23	ARSCCLASSICA	335
23.1	Introduzione	335
23.2	Principi	335
23.3	Regole	336
A	NORME TIPOGRAFICHE	337
A.1	Accento e apostrofo	337
A.2	Punteggiatura e spaziatura	338
A.3	Stile del font	340
A.4	Trattamento del testo	341
B	GALLERIA DEGLI ORRORI	345
B.1	Basi	345
B.2	Tabelle e figure	348
B.3	Matematica	351
B.4	Bibliografia	356
C	BREVE STORIA DELLA SCRITTURA	357
C.1	Origini della scrittura	357
C.2	Nascita della tipografia	369
C.3	Tipografia digitale	374
	ACRONIMI	379
	ELENCO DEI SITI INTERNET	381
	BIBLIOGRAFIA	383
	INDICE ANALITICO	385

ELENCO DELLE FIGURE

Figura 1	Donald Knuth	3
Figura 2	Leslie Lamport	4
Figura 3	Editor al lavoro	14
Figura 4	Breve documento	29
Figura 5	Frontespizio	63
Figura 6	Figura collocata a mano	123
Figura 7	Figura con didascalia laterale	145
Figura 8	Figura composta da più sottofigure	146
Figura 9	Figura immersa nel testo	147
Figura 10	Esempio di disegno programmato	152
Figura 11	Coordinate cartesiane e polari	155
Figura 12	Esempio di percorso grafico	157
Figura 13	Percorso chiuso	159
Figura 14	Curva di Bézier cubica	162
Figura 15	Elementi di un nodo	167
Figura 16	Galleria d'esempi grafici/1	188
Figura 17	Galleria d'esempi grafici/2	189
Figura 18	Sistemi di riferimento	194
Figura 19	Azimut ed elevazione	205
Figura 20	Stili bibliografici	227
Figura 21	Bibliografie suddivise per fonte	231
Figura 22	Bibliografie separate per sezione	234
Figura 23	Indicizzazione delle voci bibliografiche	236
Figura 24	Font a larghezza variabile	248
Figura 25	Font a larghezza fissa	248
Figura 26	Presentazione semplice	267
Figura 27	Presentazione complessa	269
Figura 28	Blocchi di una presentazione	271
Figura 29	Diapositive particolari	273
Figura 30	Diapositive su più colonne	274
Figura 31	Temi predefiniti per una presentazione	276
Figura 32	Curriculum	280
Figura 33	Lettera standard	284
Figura 34	Lettera conforme allo stile italiano	286
Figura 35	Tastiera italo-greca	295
Figura 36	Font disegnati da Hermann Zapf	325
Figura 37	Font Minion Pro	326
Figura 38	Robert Bringhurst	327
Figura 39	Enunciati personalizzati	330
Figura 40	Proporzioni di pagina di ClassicThesis	333
Figura 41	Arte rupestre del Paleolitico	357
Figura 42	Scrittura dell'Isola di Pasqua	359
Figura 43	Codice di Hammurabi	359
Figura 44	Scrittura maya	360
Figura 45	Ideogrammi giapponesi	361
Figura 46	Stele fenicia	365

Figura 47	Iscrizione sul Pantheon di Roma	366
Figura 48	Manoscritto miniato	367
Figura 49	Carta, penna e calamaio	367
Figura 50	Forme scrittorie	368
Figura 51	Bibbia di Gutenberg	369
Figura 52	Proporzioni di pagina	370
Figura 53	Pagina-tipo di Gutenberg	371
Figura 54	Caratteri mobili	372
Figura 55	Penna stilografica	373
Figura 56	Penna biro	374
Figura 57	Macchina per scrivere	374
Figura 58	Personal computer	375
Figura 59	Tablet	375
Figura 60	Font ispirati a caratteri storici	376
Figura 61	e-book	378

ELENCO DELLE TABELLE

Tabella 1	Cronologia di \TeX e \LaTeX	5
Tabella 2	Modi di interazione tra \LaTeX e utente	21
Tabella 3	File ausiliari	25
Tabella 4	Principali tipi di comando	26
Tabella 5	Caratteri speciali di \LaTeX	27
Tabella 6	Scorciatoie da tastiera	28
Tabella 7	Opzioni delle classi standard	33
Tabella 8	Comandi di sezionamento	37
Tabella 9	Struttura generale di un libro o una tesi	38
Tabella 10	Corrispondenza fra livelli e sezioni	42
Tabella 11	Elementi richiamabili in un documento	43
Tabella 12	Unità di misura tipografiche	48
Tabella 13	Lunghezza orientativa delle sezioni	53
Tabella 14	Virgolette, tratti e puntini di sospensione	58
Tabella 15	Loghi frequenti	59
Tabella 16	Accenti e caratteri particolari	60
Tabella 17	Comandi per modificare lo stile del font	61
Tabella 18	Dichiarazioni per modificare il corpo del font	62
Tabella 19	Simboli insiemistici	82
Tabella 20	Lettere greche	83
Tabella 21	Accenti in modo matematico	84
Tabella 22	Frecce	85
Tabella 23	Simboli logici	86
Tabella 24	Spazi in modo matematico	87
Tabella 25	Operatori predefiniti	88
Tabella 26	Simboli speciali	92
Tabella 27	Simboli di relazione	93
Tabella 28	Stili matematici	95
Tabella 29	Alcuni linguaggi riconosciuti	107
Tabella 30	Preferenze di collocazione per gli oggetti mobili	122
Tabella 31	Tabella che non rispetta le regole generali	124
Tabella 32	Tabella che rispetta le regole generali	124
Tabella 33	Descrittori standard per le tabelle	125
Tabella 34	Tabella con colonna di larghezza prefissata	127
Tabella 35	Tabella di larghezza prefissata	128
Tabella 36	Tabella con colonne della stessa larghezza	128
Tabella 37	Tabella con colonna di soli numeri	129
Tabella 38	Tabella con cella multicolonna	130
Tabella 39	Tabella con celle multiriga	131
Tabella 40	Tabella con celle multiriga e multicolonna	132
Tabella 41	Tabelle con colonne diversamente allineate	135
Tabella 42	Tabella di sola matematica	136
Tabella 43	Tabella con corpo di carattere ridotto	137
Tabella 44	Tabella ripartita su più pagine	138
Tabella 45	Alcuni programmi utili per lavorare con \LaTeX	142
Tabella 46	Principali chiavi per includere immagini	144

Tabella 47	Collocare una figura immersa nel testo	148
Tabella 48	Funzioni matematiche predefinite	156
Tabella 49	Ancore di un nodo	169
Tabella 50	Spessori di linea	173
Tabella 51	Tipi di linea predefiniti	174
Tabella 52	Colori	176
Tabella 53	Motivi di riempimento	181
Tabella 54	Sistemi di riferimento	192
Tabella 55	Alcuni marcatori disponibili	208
Tabella 56	Voci nell'indice analitico	240
Tabella 57	Parole fisse italiane	253
Tabella 58	Diacritici e punteggiatura del greco antico	296
Tabella 59	Lettere cirilliche nella tastiera italiana	301
Tabella 60	Caratteri cirillici	303
Tabella 61	Consonanti ebraiche	305
Tabella 62	Vocali ebraiche e altri simboli	306
Tabella 63	Consonanti arabe	307
Tabella 64	Font Palatino	321
Tabella 65	Classi standard e non standard	322
Tabella 66	Opzioni non standard	323
Tabella 67	Font Computer Modern	324
Tabella 68	Lettere greche	331
Tabella 69	Cronologia della scrittura	358
Tabella 70	Scrittura lineare B	362
Tabella 71	Geroglifici egizi	363
Tabella 72	Scrittura cuneiforme	364
Tabella 73	Alfabeto fenicio	365

PREFAZIONE

Lorenzo Pantieri si è cimentato in un'*enciclopedia* su \LaTeX , sviluppando e ampliando l'ottima guida *L'arte di scrivere con \text{\LaTeX}* (l'*Arte* per antonomasia), scritta in collaborazione con Tommaso Gordini. Lorenzo ha aggiunto molte altre parti che aveva elaborato personalmente in forma di appunti. Ora questo materiale è accessibile all'intera comunità degli utilizzatori italiani di \LaTeX , al pari dell'*Arte* e di *\text{\LaTeX} per l'impaziente* (una versione ridotta dell'*Arte*).

Il linguaggio \LaTeX è di per sé molto vasto. Le migliaia di pacchetti esistenti lo estendono ancora di più. È impossibile descrivere compiutamente il numero sterminato di funzioni disponibili. Perfino il testo "ufficiale" *\text{\LaTeX} Companion*, che ha più di mille pagine, non è e non può essere completo.

Non è facile scrivere di \LaTeX , poiché il sistema viene aggiornato continuamente e ogni settimana vengono modificati o aggiunti una cinquantina di pacchetti. Oggi un'installazione completa di questo programma comprende più di *tremila* pacchetti. Quindi per scrivere una guida su \LaTeX o si resta sul generale o la si aggiorna spesso. Questa *\text{\LaTeX}pedia* ha raggiunto un ottimo compromesso fra la presentazione generale e l'aggiornamento frequente.

L'autore ha concentrato il suo testo in poco più di quattrocento pagine: un grosso lavoro, ma contenuto in un'opera ancora maneggevole e facilmente leggibile sia per il neofita che per l'utente esperto. Come nelle altre sue opere, la forma grafica è legata a un disegno gradevolissimo, una sua elaborazione ispirata a *Gli elementi dello stile tipografico* di Robert Bringhurst. Gli argomenti proposti sono suddivisi in parti di specializzazione via via crescente, in modo che la lettura procede dal basilare all'avanzato. Gli ottimi indici generale e analitico permettono di usare il testo come opera di consultazione e di approfondimento.

Il risultato è eccellente. Io stesso conto di servirmene, come ho già fatto con l'*Arte*, perché mi risparmia molto tempo rispetto alla ricerca della documentazione di ogni pacchetto, spesso troppo sintetica e priva dei dettagli necessari.

Ringrazio l'autore per aver scritto questa *\text{\LaTeX}pedia*: sarà utilissima come e più dell'*Arte* per chi ama comporre i propri testi secondo i canoni dell'arte tipografica tramandata da secoli.

Torino, 22 gennaio 2017

Claudio Beccari

RINGRAZIAMENTI

Ringrazio innanzitutto i membri dello staff del Gruppo Utilizzatori Italiani di \TeX e \LaTeX , e poi tutti quelli che hanno discusso con me sul forum del Gruppo, in particolare Elia Arnese Feffin, Fabiano Busdraghi, Gustavo Cevolani, Rosaria D'Addazio, Agostino De Marco, Massimiliano Dominici, Gloria Faccanoni, Claudio Fiandrino, Heinrich Fleck, Massimo Guiggiani, Roberto Giacomelli, Tommaso Gordini, Gianluca Gorni, Maurizio Himmelmann, Jerónimo Leal, Paride Legovini, Lapo Filippo Mori, Gianluca Pignalberi, Luigi Scarso, Marco Stara, Andrea Tonelli, Emiliano Giovanni Vavassori ed Emanuele Vicentini, per l'aiuto fornito nella redazione di questo lavoro, le spiegazioni dettagliate, la pazienza e la precisione nei suggerimenti, le soluzioni fornite, la competenza e la disponibilità: grazie mille, ragazzi! Rivolgo un ringraziamento particolare al professor Enrico Gregorio, per i suoi insegnamenti e per avermi concesso l'onore di scrivere la prefazione all'*Arte*. Un "grazie" altrettanto speciale va al professor Claudio Beccari, per le indicazioni durante la stesura di un'opera che senza la sua pazienza non avrebbe mai raggiunto la forma attuale. Ringrazio infine Daniel Gottschlag, André Miede, Ivan Valbusa e Carlo Zoffoli, per le idee e per i consigli grafici.




INTRODUZIONE

*Abbiamo visto che la programmazione è un'arte,
perché richiede conoscenza, applicazione, abilità e ingegno,
ma soprattutto per la bellezza degli oggetti che produce.*

– Donald Ervin Knuth

Questo lavoro, rivolto agli utenti di \LaTeX di lingua italiana, presenta \LaTeX in modo (relativamente) approfondito.

Queste note derivano principalmente dalle numerose discussioni presenti sul forum del \GUIT (Gruppo Utilizzatori Italiani di \TeX e \LaTeX ,  **GUIT**), che è sempre un eccellente riferimento per tutti i temi trattati in questa guida.

L'esposizione degli argomenti è articolata in nove parti. La prima, la cui lettura è propedeutica a quella del resto della guida, comprende tre capitoli.

IL PRIMO CAPITOLO traccia una breve storia di \LaTeX , indicandone idee di fondo e peculiarità.

IL SECONDO CAPITOLO spiega come installare e aggiornare \LaTeX sul proprio calcolatore.

IL TERZO CAPITOLO presenta alcune nozioni indispensabili per comprendere il funzionamento del programma.

La seconda parte spiega come scrivere un documento di semplice testo ed è articolata in due capitoli.

IL QUARTO CAPITOLO descrive gli strumenti per trattare il testo.

IL QUINTO CAPITOLO mostra come gestire gli acronimi.

La terza parte spiega come comporre la matematica e i linguaggi di programmazione ed è articolata in due capitoli.

IL SESTO CAPITOLO esplora uno dei principali punti di forza di \LaTeX : la composizione di formule matematiche.

IL SETTIMO CAPITOLO riguarda la scrittura di codici e linguaggi di programmazione in un documento \LaTeX .

La quarta parte si occupa della rappresentazione grafica di dati ed è articolata in tre capitoli.

L'OTTAVO CAPITOLO spiega come comporre le tabelle, includere le figure in un documento e gestire la loro collocazione sulla pagina.

IL NONO CAPITOLO offre gli strumenti essenziali per disegnare direttamente con \LaTeX .

IL DECIMO CAPITOLO mostra come tracciare il grafico di una funzione espressa analiticamente.

La quinta parte, dedicata alla bibliografia e all'indice analitico, è articolata in due capitoli.

L'UNIDCESIMO spiega come realizzare e gestire la bibliografia.

IL DODICESIMO mostra come generare l'indice analitico.

La sesta parte spiega come personalizzare e rifinire un documento ed è articolata in due capitoli.

IL TREDICESIMO CAPITOLO espone alcuni suggerimenti per fare in modo che \LaTeX produca risultati diversi da quelli predefiniti.

IL QUATTORDICESIMO CAPITOLO dà alcuni suggerimenti per migliorare l'impaginazione del documento.

La settima parte mostra come comporre documenti particolari (videopresentazioni, curriculum e lettere) ed è articolata in tre capitoli.

IL QUINDICESIMO CAPITOLO fornisce gli elementi essenziali per realizzare una videopresentazione.

IL SEDICESIMO CAPITOLO spiega come scrivere un curriculum.

IL DICIASSETTESIMO CAPITOLO mostra come scrivere una lettera.

L'ottava parte, riservata alle lingue esotiche, è articolata in tre capitoli.

IL DICOTTESIMO CAPITOLO spiega come scrivere in latino e in greco.

IL DICIANNOVESIMO CAPITOLO spiega come scrivere in cirillico.

IL VENTESIMO CAPITOLO spiega come scrivere in ebraico e in arabo.

La nona parte tratta alcune specialità tipografiche ed è articolata in tre capitoli.

IL VENTUNESIMO CAPITOLO mostra come inserire diversi tipi di capilettera.

IL VENTIDUESIMO CAPITOLO spiega come produrre un documento con lo stile ClassicThesis.

IL VENTITREESIMO CAPITOLO spiega come produrre un documento con lo stile ArsClassica.

Concludono l'opera tre appendici.

L'APPENDICE A descrive le principali norme tipografiche italiane.

L'APPENDICE B raccoglie alcuni esempi tratti da documenti trovati in Rete, con l'obiettivo di mostrare come *non* si scrive in \LaTeX .

L'APPENDICE C, infine, traccia una breve storia della scrittura, di cui \LaTeX rappresenta una delle tappe più felicemente riuscite.

Questo *non* è un manuale su \LaTeX , ma un tentativo di riordinare appunti accumulatisi nel tempo, via via che divenivo abituale utente di questo programma. Come semplice appassionato non ho nulla da insegnare; d'altra parte ho studiato \LaTeX e l'ho usato intensamente, acquisendo una certa esperienza che ci piacerebbe condividere con altri utenti.

Se avete idee su argomenti da aggiungere o modificare in questo libro, o se vi dovesse capitare di notare errori, sia di battitura che di sostanza (ed è probabile che ce ne siano parecchi, e del primo e del secondo tipo), fareste davvero una cosa gradita comunicandomelo, così da poterli correggere nelle versioni successive del lavoro.

Ecco lo spirito che ci ha guidati in questo lavoro: spero che possiate usare \LaTeX con il mio stesso piacere.

Parte I

PER COMINCIARE

Questo capitolo presenta una breve storia di \TeX e \LaTeX , e ne indica idee di fondo e peculiarità. Se ne descrivono inoltre i principali pro e contro, i luoghi comuni, le scorciatoie da non prendere e i più comuni errori da evitare all’inizio.

1.1 STORIA

1.1.1 \TeX

\TeX è un programma di composizione tipografica realizzato da Donald Ervin Knuth, professore di Informatica all’Università di Stanford (USA), e distribuito con una licenza di software libero.

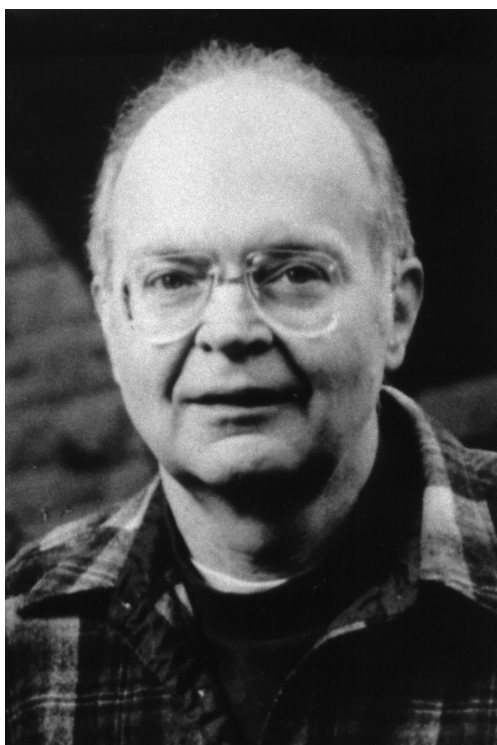


Figura 1: Donald Ervin Knuth

Nel 1977, Knuth cominciò a scrivere il suo “motore” di tipocomposizione per esplorare le potenzialità degli strumenti di stampa digitale che allora cominciavano a muovere i primi timidi passi in campo editoriale.

La qualità tipografica delle proprie pubblicazioni (e soprattutto del capolavoro *The Art of Computer Programming*, in più volumi ricchi di formule matematiche), constatava Knuth, era in crescente peggioramento: a quel tempo, infatti, gran parte della matematica si componeva con la macchina per scrivere, alzando e abbassando il carrello per indici ed esponenti e cambiando testina per i simboli. Con \TeX , il Professore sperava di mettere un freno a questa tendenza.

Il programma ha visto la luce nel 1982, e da allora ha subito costanti aggiornamenti e miglioramenti. \TeX , rinomato per essere molto stabile e per girare su diversi tipi di calcolatore, è stato rivisto per l’ultima volta nel 2014. Il suo numero di versione converge a π (attualmente è 3,14159265).

The \TeX book [Knuth, 1984], scritto da Donald Knuth, è il manuale d’uso di \TeX e uno dei libri più completi su questo programma. Attualmente \TeX è un marchio registrato della \mathcal{AMS} (American Mathematical Society).

1.1.2 Etimologia

Knuth ha nascosto un trabocchetto nel nome del programma: $\text{T}_{\text{E}}\text{X}$, infatti, si pronuncia *tèch* (aspirando il *ch* finale) e non com'è scritto, perché è una parola greca scritta in greco maiuscolo (in lettere minuscole si scriverebbe $\tau\epsilon\chi$). Si tratta di un'antichissima radice indoeuropea comune non soltanto ai greci τέκτων (pron. *tèkton*, "artefice") e τέχνη (pron. *tèchne*, insieme "arte" e "mestiere"), per esempio, ma viva ancora oggi negli usatissimi *tecnica*, *politecnico*, *architetto* e in numerose altre parole italiane. L'etimologia appena spiegata illumina la scelta di Knuth: $\text{T}_{\text{E}}\text{X}$ sarebbe stato il nome perfetto per un programma che compone documenti "allo stato dell'arte".

Knuth dice che «se $\text{T}_{\text{E}}\text{X}$ è ben pronunciato, lo schermo del calcolatore si appanna leggermente». La lettera *X*, infatti, corrisponde a una forte aspirazione non esistente in italiano, ma presente in molte lingue attualmente parlate nel mondo: oltre che in greco, si trova nel tedesco *Bach*, nello scozzese *Loch*, nello spagnolo *Juan* e *Mexico*, nel russo *Хорошо* (pron. *harasciò*, "bene"), nel cinese 你好 (pron. *nǐ hǎo*, "ciao"), solo per fare qualche esempio.

Lo stesso Knuth, però, ammette le diverse pronunce nazionali (che convergono in un *tèk* universale), rassicurando gli utenti che non è certo per il fatto di sentire $\text{T}_{\text{E}}\text{X}$ detto come ognuno preferisce che andrà su tutte le furie. Tuttavia, indica nella Grecia il Paese in cui oggi si può ascoltare la pronuncia più corretta di questa parola.

1.1.3 \LaTeX

\LaTeX ($\text{La}(\text{mport})\text{T}_{\text{E}}\text{X}$) è un programma di composizione tipografica realizzato da Leslie Lamport e liberamente disponibile, che usa $\text{T}_{\text{E}}\text{X}$ come motore di tipocomposizione.

\LaTeX non è $\text{T}_{\text{E}}\text{X}$, però. Per dare l'idea della differenza tra i due programmi, si potrebbe paragonare $\text{T}_{\text{E}}\text{X}$ a un corpo, e \LaTeX al più popolare degli "abiti" (fatto, però, di istruzioni in linguaggio $\text{T}_{\text{E}}\text{X}$) che nel corso degli anni gli sono stati confezionati addosso per avvicinarlo al pubblico con sembianze amichevoli.

\LaTeX è stato progettato per automatizzare in una volta sola tutte le più comuni operazioni necessarie per realizzare un documento e, tramite le proprie impostazioni predefinite, permette all'utente di impaginare il suo lavoro ai più elevati livelli di professionalità secondo canoni tipografici consolidati.

Lamport, che collaborava con Knuth allo sviluppo di $\text{T}_{\text{E}}\text{X}$, cominciò a scrivere \LaTeX alla fine degli anni Settanta del secolo scorso, quando $\text{T}_{\text{E}}\text{X}$ non era ancora stato pubblicato.

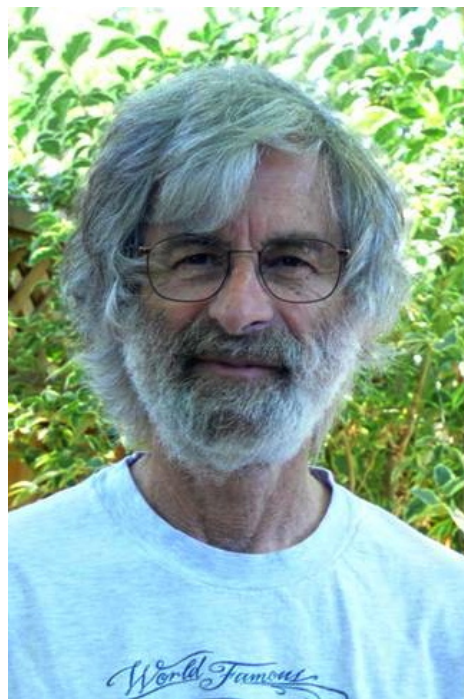


Figura 2: Leslie Lamport

Tabella 1: Cronologia di T_EX e L^AT_EX

1977	Knuth comincia a scrivere T _E X
1978	Lamport comincia a scrivere il primo nucleo di L ^A T _E X
1982	Prima versione pubblica di T _E X
1984	Knuth pubblica la prima edizione di <i>The T_EXbook</i>
1985	Lamport pubblica L ^A T _E X 2.09 e la prima edizione di <i>L^AT_EX. A Document Preparation System</i>
1994	L ^A T _E X 2 _ε
2014	Ultima revisione di T _E X
2016	Ultima revisione di L ^A T _E X

La prima versione pubblica di L^AT_EX risale al 1985, e da allora il programma è stato continuamente aggiornato e migliorato. Per molti anni il suo numero è rimasto fissato a 2.09 e le successive revisioni sono state identificate con le loro date.

Nel 1994, finalmente, un gruppo di programmatori guidato da Frank Mittelbach lo ha aggiornato in modo sostanziale, includendovi tutte le versioni successive alla 2.09 e numerosi altri miglioramenti. Per distinguerla da quella precedente, la nuova edizione è stata battezzata L^AT_EX 2_ε, e costituisce l'oggetto di questa guida. *L^AT_EX. A Document Preparation System* [Lamport, 1994], scritto da Leslie Lamport, è il manuale ufficiale di L^AT_EX.

Il futuro L^AT_EX₃ si profila come un progetto a lungo termine: i costanti aggiornamenti di L^AT_EX, la cui ultima versione pubblica è del 2016, ne costituiscono le tappe di avvicinamento.

1.2 FILOSOFIA

1.2.1 Composizione sincrona e asincrona

La caratteristica che più differenzia L^AT_EX dagli altri elaboratori di testo è il fatto che l'introduzione del testo e la sua composizione avvengono *in tempi diversi*.

Per modificare un documento scritto con un comune *word processor* (come Microsoft Word), l'utente agisce direttamente sul testo già composto così come gli appare sul monitor, e ogni sua azione si traduce in una variazione *immediata* di quel testo. Perciò questo tipo di composizione è detto "composizione sincrona". Per essere *davvero* sincrono e operare con un ritardo trascurabile tra modifica del testo e visualizzazione, però, il programma deve puntare tutto sulla rapidità della presentazione, ciò che di fatto impedisce di ottenere una composizione perfetta per via della molto più accurata elaborazione del materiale richiesta. È anche vero, però, che oggi i programmi di videoscrittura sono estremamente rapidi e che la qualità della loro composizione migliora sensibilmente a ogni nuova versione, ma il compromesso tra velocità e qualità esiste sempre.

La "composizione asincrona", invece, consiste nell'introdurre il testo in un editor concentrandosi unicamente su struttura logica e contenuto del documento (senza perciò preoccuparsi di come apparirà), per darlo "in pasto" a un compositore (L^AT_EX, in questo caso) che lo impagina solo *successivamente*. L'utente, naturalmente, può modificare il proprio lavoro in qualunque mo-

do anche dopo la composizione, ma sempre tenendo bene a mente che \LaTeX non si limita ad aggiustarlo *nel punto* in cui è stato modificato e basta, come farebbe un programma qualunque, ma riorganizza sempre *l'intero capoverso* (e, di conseguenza, l'intera pagina) nel migliore dei modi.

Va da sé che questo secondo tipo di composizione è migliore del primo, perché la velocità di visualizzazione passa in secondo piano a tutto vantaggio della qualità: \LaTeX elabora il testo sempre *nel suo complesso*, e per questo motivo può fare le scelte d'impaginazione migliori.

1.2.2 Istruzioni di marcatura

L'idea di Lamport era forte. E \LaTeX centra in pieno l'obiettivo: l'utente può (quasi *deve*, si direbbe) astrarsi dai dettagli estetici che con un altro elaboratore di testo sarebbe costretto a introdurre a mano, per indirizzare più efficacemente le proprie energie su *contenuto* e scansione delle parti del proprio lavoro.

\LaTeX pretende dall'utente considerazioni sul *cosa*: «il mio documento sarà composto da un certo numero di capitoli, ciascuno diviso in paragrafi numerati, avrà indice generale e analitico, delle figure e qualche tabella». E questo è tutto.

Al *come* pensa \LaTeX , e lo fa molto bene. Per esempio, uno stesso file sorgente può generare in teoria documenti radicalmente diversi soltanto cambiandone la classe o caricando un pacchetto che agisce in modo globale su di esso. (S'è detto *in teoria* perché, nella pratica, qualche aggiustamento manuale si rende di fatto *sempre* necessario.)

Un file da comporre con \LaTeX è scritto in una "lingua" particolare costituita da *marcatori* (o *etichette logiche*, *mark-up* in inglese), ovvero istruzioni che il programma deve eseguire per trattare nel modo specificato dall'utente la porzione di testo a cui si riferiscono. Le etichette logiche sono, in una parola, i *comandi* (chiamati anche, in gergo, *macro*, abbreviazione di *macroistruzioni*) e gli *ambienti*.

Nonostante faccia dei marcatori uno dei propri punti di forza (per cui molte delle tradizionali operazioni sul testo vengono automatizzate), \LaTeX definisce anche gli strumenti per regolare finemente (e a mano, questa volta) il risultato durante la revisione finale del documento, passaggio ancora insostituibile. In tipografia, infatti, le variabili sono troppe e troppo variegate per poter essere gestite in una volta sola da un unico programma; e l'utente deve accettarlo, se pretende un prodotto di alta qualità.

Il file prodotto con l'editor, insomma, è un codice scritto in una sorta di linguaggio di programmazione, dato che contiene sia il vero e proprio testo del documento, sia le istruzioni di marcatura (i comandi) che ordinano al programma di tipocomposizione di comporre quello che gli si dà in pasto secondo lo stile particolare scelto per scrivere il documento.

Non si spaventi chi non conosce bene l'inglese: i comandi di \LaTeX sono in un inglese molto semplice, e anche se abbreviati risultano di solito comprensibilissimi e facilmente memorizzabili.

Un esempio

Per dare l'idea di come appare un documento da comporre con \LaTeX , si riportano alcune righe di *codice sorgente* (o più semplicemente *sorgente*),

cioè l'insieme di testo, numeri, simboli e istruzioni di marcatura da scrivere nell'editor.

```
Due matrici  $n \times n$  complesse  $A$  e  $B$  si dicono simili
se esiste una matrice  $n \times n$  invertibile  $T$  tale che
\begin{equation}
B=T^{-1}AT
\end{equation}
```

Il sorgente viene composto da \LaTeX che, attraverso \TeX , produce il documento tipocomposto (*typeset*). Se il risultato non soddisfa, non si può modificare direttamente il documento a schermo, ma bisogna correggere il sorgente e poi ricomporlo.

L'esempio seguente riporta a sinistra lo stesso sorgente appena visto e a destra il risultato della composizione.

```
Due matrici  $n \times n$ 
complesse  $A$  e  $B$  si dicono
simili se esiste una
matrice  $n \times n$ 
invertibile  $T$  tale che
\begin{equation}
B=T^{-1}AT
\end{equation}
```

Due matrici $n \times n$ complesse A
e B si dicono *simili* se esiste una
matrice $n \times n$ invertibile T tale che

$$B = T^{-1}AT \quad (1.1)$$

Nei prossimi capitoli si spiegheranno tutte le istruzioni usate nell'esempio. Tuttavia, anche con pochi rudimenti di inglese si capisce facilmente quello che il linguaggio di marcatura ha specificato.

1.2.3 \LaTeX : pro e contro

I vantaggi di \LaTeX rispetto agli altri elaboratori di testo sono innumerevoli. Di seguito se ne elencano alcuni. \LaTeX :

- compone documenti al massimo grado di professionalità e presenta caratteristiche di qualità e stabilità sconosciute agli altri elaboratori di testo;
- si occupa dell'impaginazione del documento, mentre l'utente pensa a struttura e contenuto;
- genera strutture complesse come riferimenti incrociati, indici e bibliografie con grande efficienza e flessibilità;
- gestisce in modo impeccabile e tuttora ineguagliato la composizione tipografica di formule matematiche e capoversi;
- è gratuito, multilingue e multiplatforma;
- ha una struttura modulare che permette di estenderne le capacità per eseguire compiti tipografici non direttamente gestiti dal programma;
- rende davvero difficile creare documenti mal strutturati e tipograficamente scadenti;
- appassiona talmente... che ci si può dimenticare la tazza del caffè sul foglio!

Innanzitutto, \LaTeX presenta anche alcuni svantaggi.

- Con \LaTeX ci vuole attitudine all'astrazione.
- Con \LaTeX la gratificazione non è istantanea, ma ritardata.
- \LaTeX «non lavora bene per chi ha venduto la propria anima» [Oetiker *et al.*, 2011].
- Solo gli esperti possono permettersi di uscire dagli stili predefiniti.
- Il fatto che \LaTeX raggiunga un'elevatissima qualità soltanto grazie alla composizione asincrona può talvolta rivelarsi un difetto anziché un pregio (almeno se non si considerano i motivi di questo comportamento e non si adatta di conseguenza il proprio metodo di lavoro). \LaTeX , infatti, gestisce il capoverso in modo del tutto particolare: rimuoverne anche una sola parola ne determina *sempre* la riorganizzazione completa. Con risultati a volte indesiderati: il capoverso potrebbe risultare addirittura di una riga più lungo, se \LaTeX decidesse che quella è la soluzione migliore, con le ovvie conseguenze se ciò dovesse accadere durante l'ultimissima revisione del documento.
- \LaTeX non gestisce pagine più grandi di 33 metri quadrati e non dà denaro alle case di software.

In breve, è opportuno usare \LaTeX quando [Mittelbach *et al.*, 2007]:

- l'utente preferisce pensare per strutture logiche;
- i documenti da preparare richiedono coerenza interna;
- i documenti da preparare non hanno un formato completamente definito o potrebbero essere presentati parallelamente in vesti diverse;
- i documenti da preparare contengono molta matematica;
- il materiale è corposo.

Viceversa, si sceglierà un sistema di videoscrittura tradizionale quando:

- l'utente preferisce pensare per strutture visive;
- l'utente non si trova del tutto a proprio agio nel lavorare con un linguaggio di programmazione (un editor per \LaTeX aiuta, ma...);
- i documenti da preparare richiedono più flessibilità visuale che consistenza (volantini, biglietti d'invito, dépliant, brochure, eccetera);
- il materiale non è corposo.

In ultima battuta: prima di affidarsi a \LaTeX si consiglia caldamente di vagliare *a monte* e con la massima attenzione il tipo di prodotto che s'intende realizzare.

1.2.4 Luoghi comuni

Molte delle persone che si avvicinano a \LaTeX dopo anni di tormenti alle prese con altri elaboratori di testo si stupiscono quando scoprono che era disponibile da oltre tre decenni, pur non avendone mai sentito parlare. Non si tratta certo di una cospirazione ai loro danni, ma solo di «un segreto ben conservato e noto solo a pochi milioni di persone», come disse una volta un utente anonimo.

Knuth e Lamport hanno generosamente reso i propri programmi di pubblico dominio fin da subito, e perciò per molto tempo \LaTeX è vissuto indisturbato tra le mura delle università senza che nessuno sentisse il bisogno di pubblicizzarlo al di fuori. Oggi, tuttavia, \LaTeX è divenuto popolarissimo: innumerevoli editori pubblicano documenti nel suo formato, e centinaia di migliaia di utenti scrivono con esso milioni di documenti ogni giorno.

In questi anni, su \LaTeX sono fioriti molti luoghi comuni: per evitare possibili incomprensioni, conviene in questa sede esaminarne i più diffusi.

Leggenda: \LaTeX ha solo un font

\LaTeX può usare *ogni* font di tipo TrueType, Type1, OpenType, PostScript e METAFONT. Ciò è più di quanto viene offerto dalla maggior parte degli altri sistemi di composizione tipografica. Il font standard di \LaTeX è il Computer Modern e non il Times New Roman, quindi alcuni restano turbati da un documento diverso dal solito.

Leggenda: \LaTeX è un software solo per Unix

Si sente anche dire che \LaTeX è un software solo per Unix, o solo per Mac, eccetera. È tutto il contrario: \LaTeX funziona sulla maggior parte dei calcolatori oggi sul mercato (supercomputer e palmari compresi): dai PC con Windows ai Mac ai sistemi Unix/Linux. Se \LaTeX non gira sul proprio computer è solo perché, molto probabilmente il sistema operativo installato è troppo vecchio.

Leggenda: \LaTeX è obsoleto

Proprio il contrario. L'incessante lavoro di migliaia di appassionati in tutto il mondo lo tiene costantemente aggiornato, e quasi ogni giorno si aggiungono agli archivi ufficiali nuove caratteristiche. È dunque indiscutibilmente più aggiornato della maggior parte degli altri sistemi di videoscrittura.

Leggenda: \LaTeX non è WYSIWYG

Dipende. Se per WYSIWYG (*What You See Is What You Get*, “ciò che vedi è ciò che ottieni”) s'intende un software in grado di produrre nel documento finito testo e immagini disposti esattamente come si vede sullo schermo del calcolatore, \LaTeX è un programma WYSIWYG della migliore qualità.

Se invece per WYSIWYG s'intende un programma di videoscrittura in cui l'utente agisce direttamente sul testo già composto così come gli appare a schermo e ogni sua azione si traduce in un'immediata variazione di quel testo, \LaTeX non è WYSIWYG ma WYSIWM (*What You See Is What You Mean*, “ciò che vedi è ciò che intendi”), perché compone il documento in modo asincrono.

Leggenda: L^AT_EX è troppo difficile

Questa frase si è sentita dire da fisici in grado di dividere gli atomi, da matematici che sanno dimostrare la trascendenza di π , da uomini d'affari che sanno leggere un foglio di bilancio, da storici che hanno compreso la politica bizantina e da linguisti che sanno decifrare la scrittura lineare B. L^AT_EX non è immediato come i normali elaboratori di testo, certo, ma quel po' di studio e di pratica iniziali richiesto (possibilmente su documenti elementari) verranno presto ricompensati dalla qualità dei risultati.

Leggenda: L^AT_EX è solo per matematici e scienziati

Niente affatto. Sebbene sia cresciuto nei campi della matematica e dell'informatica, due delle sue maggiori aree di espansione sono quelle umanistica ed economica, specie da quando ha preso piede l'XML, che ha sollevato nuove esigenze nell'ambito della tipocomposizione automatica.

1.2.5 Non tutte le strade portano a L^AT_EX

Molti utenti, per i motivi più vari, vorrebbero beneficiare della potenza e delle qualità di L^AT_EX senza pagare il piccolo prezzo dichiarato poco sopra, e si rivolgono ai vari programmi in circolazione che dichiarano di poter risolvere il problema "L^AT_EX è difficile" rendendolo "facile" con interfacce grafiche intuitive e ricche pulsantiere per soddisfare ogni esigenza di scrittura (come non averci pensato!), e garantendo contemporaneamente l'eccellenza della composizione tipografica tipica di T_EX.

Gli autori di questa guida *scoraggiano con vigore* l'uso di quei programmi che tentano di nascondere L^AT_EX all'utente mascherandolo da normale word processor. Il più accreditato di essi è L_XX, ma è provato che appena gli si chiede qualcosa di non previsto dai suoi sviluppatori ci si trova in guai seri, e spesso la soluzione al problema si rivela più difficile di quanto non sarebbe stato usare L^AT_EX fin dal principio. E lo stesso vale in genere per programmi analoghi. Per esempio, scrivere

```
\section{titolo}
```

non può essere davvero più complicato che premere un pulsante Sezione.

1.2.6 Errori da evitare

S'intendano i punti seguenti come un promemoria dei più comuni errori tipici dell'utente alle prime armi e se ne faccia tesoro.

- *Non leggere con attenzione una buona guida di base.*
Non si può pensare di cominciare a usare L^AT_EX senza affiancargli un manuale. Le guide sono scritte per essere lette e, se ben fatte, permettono di evitare gran parte degli errori iniziali.
- *Installare una distribuzione di L^AT_EX non completa.*
Sebbene possa sembrare inutile occupare diversi gigabyte di spazio sul disco per installare un sistema T_EX *completo*, sapendo bene che non lo si sfrutterà per intero praticamente mai, tuttavia si consiglia di farlo ugualmente: ci si mette al riparo da eventuali problemi e si è certi

di avere a disposizione tutti i pacchetti che servono. La capienza dei moderni dischi fissi non dovrebbe preoccupare in questo senso.

- *Lavorare con la distribuzione non aggiornata.*
È sempre bene aggiornare la distribuzione periodicamente: gli aggiornamenti non aggiungono solo nuove funzioni a pacchetti e a classi, ma risolvono anche eventuali malfunzionamenti lamentati dagli utenti. L'unica circostanza in cui potrebbe essere sconsigliabile farlo è quando si sta ultimando la revisione di un documento importante: le migliorie potrebbero disattivare alcune funzioni fondamentali per il documento e ci potrebbe non essere abbastanza tempo per studiare quelle nuove e apportare al sorgente i cambiamenti richiesti. In tal caso, meglio attendere di aver terminato il lavoro.
- *Pretendere di scrivere un intero sorgente unicamente premendo pulsanti con il mouse.*
Anche se le ricche pulsantiere che alcuni editor offrono per le varie funzioni *potrebbe* far sembrare più semplice scrivere codice L^AT_EX, tuttavia si sconsiglia fortemente di abituarcisi. Si scriva il codice sempre a mano imparandone il più possibile: non si fa più fatica ed è enormemente più utile: sarà possibile lavorare senza problemi anche su editor diversi dal proprio.
- *Pretendere, come primo documento, di riprodurre layout e caratteri del proprio libro preferito o quelli di altri word processor, riempiendo il sorgente di molte e avanzate personalizzazioni.*
Questo è forse l'errore più tipico del principiante. Come si scoprirà nelle prossime pagine, L^AT_EX è *molto* diverso dai soliti programmi di videoscrittura, e la cosa migliore è sempre affidarsi a classi e pacchetti che propongano layout e combinazioni di font predefinite tra cui scegliere. Fare pasticci in questo senso è più facile di quel che sembra, e usare tutti i font con la stessa facilità di altri programmi non è così semplice: le personalizzazioni sono pane per i denti dell'utente avanzato.
- *Non cercare la risposta alla domanda «come si fa?» innanzitutto nella documentazione dei pacchetti.*
AmMESSO di aver individuato il pacchetto che fa al caso proprio, la sua documentazione (quasi sempre in inglese) spiega tutto quello che c'è da sapere per ottenere ciò che serve. Se non la si legge, delle due l'una: o non si sa l'inglese o si è troppo pigri. Nei (pochi) casi in cui la documentazione risulti criptica, si può chiedere aiuto pubblicamente al forum del G_{IT}, in modo che la soluzione al problema rimanga visibile a tutti.

2

INSTALLARE E AGGIORNARE

Questo capitolo spiega come procurarsi tutto l'occorrente per usare \LaTeX , come installarlo nel proprio calcolatore e come aggiornarne la distribuzione (che in questa guida si considererà *sempre* nell'ultima versione disponibile). I programmi descritti sono gratuiti e, dove non altrimenti indicato, disponibili per tutti e tre i sistemi operativi considerati in questo capitolo (Windows, Mac e Linux).

2.1 FERRI DEL MESTIERE

Per creare un documento con \LaTeX sono indispensabili almeno tre cose:

- un editor di testi con cui scrivere il file sorgente;
- il programma \LaTeX , che lo elabora e produce il documento tipocomposto;
- un programma per visualizzare il documento finito.

Si eviti da subito l'errore, molto frequente all'inizio, di confondere \LaTeX (un "motore" del tutto invisibile all'utente) con l'editor (ciò che effettivamente appare sullo schermo): i due programmi sono completamente *indipendenti*, tanto che in teoria si può usare un editor qualunque, da quelli più elementari già presenti nel proprio computer a quelli più complessi capaci di gestire numerosi linguaggi di programmazione.

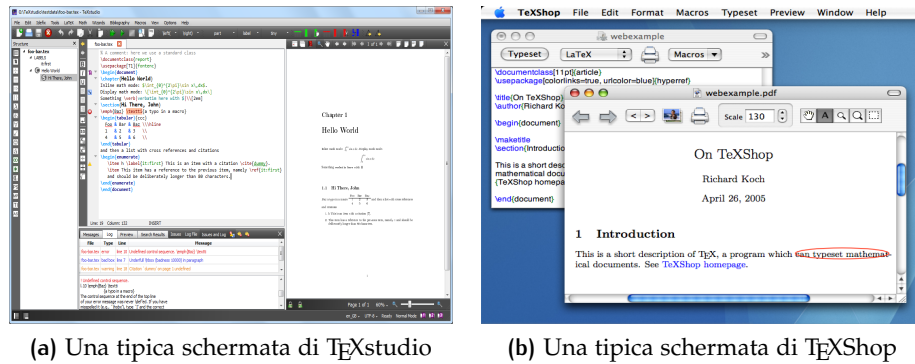
Altrettanto spesso si confondono i due programmi `latex` e `pdflatex`: il primo produce file `.dvi` (oggi indispensabili in circostanze così limitate da non meritare una trattazione in questa guida), il secondo file `.pdf`. La confusione nasce perché il nome \LaTeX viene usato indifferentemente per indicare entrambi, e in qualche editor si chiama così anche il pulsante che avvia la composizione.

In questa guida con \LaTeX s'intende *sempre* il programma `pdflatex`, e con le espressioni *documento composto* e *documento finito* un file in formato PDF.

2.1.1 Editor e visualizzatore

Anche se oggi molti software di videoscrittura tradizionali prevedono estensioni per trasformare il testo immesso in codice \LaTeX , si consiglia di usare senz'altro un editor *dedicato*, cioè destinato esclusivamente a \LaTeX e ideato per facilitarne il più possibile l'interazione con l'utente: i vantaggi sono molti e sostanziali. I più diffusi tra essi, inoltre, comprendono anche un visualizzatore di PDF, eliminando così i possibili problemi derivanti dall'usarne uno esterno.

Gli editor che gestiscono codice \LaTeX sono moltissimi, come si può scoprire con una veloce ricerca in Rete. Quelli consigliati in questa guida sono:



(a) Una tipica schermata di TeXstudio

(b) Una tipica schermata di TeXShop

Figura 3: Editor al lavoro

- TeXstudio (🐞 **TEXSTUDIO**), multiplatforma;
- TeXShop (🐞 **TEXSHOP**), specifico per Mac.

Questi ambienti di lavoro, adatti sia ai principianti che a utenti esperti, fanno parte di TeX Live (si veda il paragrafo successivo), e perciò sono *già installati* sulle macchine (tranne che su Linux).

L'interfaccia grafica di TeXstudio e TeXShop è molto semplice: se ne può vedere un esempio nella figura 3. I programmi sono compatibili con la tecnologia SyncTeX: sorgente e anteprima cioè, sono *sincronizzati*, e si può “saltare” da un punto dell'uno al corrispondente punto dell'altro (e viceversa).

Commenti speciali e gestione di documenti in più file

Un grande pregio che TeXstudio e TeXShop condividono con pochissimi altri editor è la capacità di interpretare i *commenti speciali*, cioè “righe magiche” che cominciano con la stringa % ! da scrivere *come prima cosa* nel sorgente e che, come i commenti veri e propri (si veda il paragrafo 3.4.5 a pagina 30), vengono ignorati da L^AT_EX ma *non* dall'editor, che in modo automatico si autoconfigura corrispondentemente. Non sono obbligatori, ma sono una buona abitudine: fissando *una volta per tutte* le impostazioni più importanti del documento e avendo la precedenza sulle impostazioni dell'editor, evitano all'utente i problemi descritti nel paragrafo 3.2 a pagina 22 e in [Beccari e Gordini, 2016].

Eccone la sintassi generale riconosciuta da TeXstudio e TeXShop:

```
%_!TEX_⟨parola chiave⟩_=_⟨valore⟩
```

dove:

- $\langle \text{parola chiave} \rangle$ è una delle quattro voci spiegate più sotto;
- $\langle \text{valore} \rangle$ dipende dalla $\langle \text{parola chiave} \rangle$.

Si noti che maiuscole, minuscole e spazi, qui evidenziati con il segno `_`, vanno *sempre* rispettati (TeXShop permette di inserire queste righe anche automaticamente con appositi pulsanti).

Le possibili parole chiave sono:

encoding dichiara la codifica con cui è scritto il sorgente: si metta sempre UTF-8 (verificando di aver caricato *anche* inputenc nello stesso modo, si veda il paragrafo 3.2 a pagina 22) e non si avranno problemi;

program dichiara il motore di composizione, che di solito è `pdflatex` o `xelatex` (in tutte minuscole);

root dichiara la condizione di file principale e di file secondario nei documenti suddivisi in più file (si veda poco sotto);

spellcheck attiva il controllo ortografico della lingua del documento: per l'italiano si scriva `it-IT` (dopo aver installato i dizionari italiani).

\TeX studio e \TeX Shop gestiscono facilmente anche progetti suddivisi in più file come una tesi o un libro (si veda il paragrafo 3.14 a pagina 48) permettendo di comporre comodamente il documento dalla finestra in cui ci si trova anziché tornare ogni volta al file principale. Immaginando di lavorare a questa guida, il cui file principale si chiama `latexpedia.tex`, basta scrivere

```
% !TEX root = latexpedia.tex
```

in testa a ciascun file da includervi se file principale e secondario stanno nella stessa cartella, oppure

```
% !TEX root = ../latexpedia.tex
```

se quest'ultimo è in una sottocartella.

Infine si dà il blocco completo dei commenti speciali con la sintassi di \TeX Shop per un documento in italiano da comporre con \LaTeX o \XeLaTeX (si veda il paragrafo 18.2.2 a pagina 297):

```
% !TEX encoding = UTF-8 Unicode
% !TEX TS-program = pdflatex
% !TEX TS-program = xelatex
% !TEX spellcheck = it-IT
% !TEX root = <nome del file principale>.tex
```


Si noti che:

- se si compone con \XeLaTeX il primo e il secondo commento vanno omessi;
- se si compone con \LaTeX va omesso il terzo;
- se il documento è in un solo file, l'ultimo commento non serve.

Per motivi di spazio, i commenti speciali *saranno omessi* in tutti i codici di questa guida.

2.1.2 \TeX Live

\LaTeX è uno, ma prende forma in differenti versioni (non così tante) che si chiamano *distribuzioni*. Una distribuzione è una raccolta di file e altro software (il programma \LaTeX vero e proprio, uno o più editor dedicati e di solito altri programmi accessori) autosufficiente per produrre un documento finito. Si può installare direttamente da Internet, dal disco rigido dopo averla scaricata oppure da DVD.

Il luogo di riferimento nel Web da cui scaricare il materiale ufficiale su \LaTeX è CTAN (*Comprehensive \TeX Archive Network*, “rete di archivi completi per \TeX ”,  CTAN), una rete di server dislocati in tutto il mondo, uguali tra

loro e ciascuno contenente una copia integrale del sito originale (si chiamano anche *mirror*, “specchio”): ci si può così servire dal mirror più vicino, evitando di sovraccaricare la Rete e abbreviando i tempi dell’operazione.

In questa guida si consiglia di installare T_EX Live (🐧 **TEXLIVE**): è una distribuzione affidabile, mantenuta da decine di sviluppatori e aggiornata annualmente. Gli utenti di Windows possono contare sull’alternativa di MiK_TE_X, che gira solo su quel sistema operativo. Qualunque delle due si scelga, si abbia cura di farne un’installazione *completa*: in caso contrario, potrebbero presentarsi alcuni fastidiosi problemi (si veda la sezione 3.2 a pagina 22).

Una distribuzione si modifica nel tempo, perché quasi ogni giorno molti dei *pacchetti* che la compongono (si veda il paragrafo 3.12 a pagina 45) vengono perfezionati, altri di nuovi vengono aggiunti agli archivi in Rete e altri ancora rimossi. Quelli già presenti riescono a soddisfare praticamente tutte le esigenze di scrittura, ma per mantenere la propria distribuzione sempre efficiente bisogna *aggiornarla* alle ultime versioni dei pacchetti e *installare* quelli nuovi continuamente creati e messi a disposizione.

Nei paragrafi seguenti si spiegherà:

- come installare T_EX Live sui diversi sistemi operativi;
- come usare il programma `tlmgr` (*TeX Live ManaGeR*) per aggiornarla e aggiungervi automaticamente i nuovi pacchetti; si raccomanda di farlo senz’altro subito dopo aver installato la distribuzione e poi con una certa regolarità (una volta alla settimana potrebbe andare bene).

2.2 WINDOWS

Installare

La procedura descritta di seguito permette di installare T_EX Live sul proprio calcolatore.

1. Si scarichi il file `texlive2016.iso` da <http://tug.org/texlive/Images/> e lo si decomprima.
2. Si apra la cartella risultante, si esegua il file `install-tl` al suo interno *come amministratore* e si attenda finché non compare l’omonima finestra.
3. Si seguano le istruzioni sullo schermo senza cambiare nulla e si attenda la fine dell’operazione.


Aggiornare

L’interfaccia grafica di `tlmgr` su Windows è T_EX Live Manager.

1. Si avvii il programma (Start → Programmi → TeX Live 2016 → TeX Live Manager).
2. Si preme il pulsante **Aggiorna installati** e si attenda la fine dell’aggiornamento.

2.3 MAC

Installare

Su Mac si raccomanda di installare \TeX Live tramite Mac \TeX ( **MACTEX**), un installer che provvede a tutto il necessario.

1. Si scarichi il file MacTeX.pkg da <http://www.tug.org/mactex/>.
2. Lo si esegua e se ne seguano le istruzioni fino a completare l'installazione.

Aggiornare

L'interfaccia grafica di \TeX Live su Mac è \TeX Live Utility.

1. Si avvii il programma (Applicazioni \rightarrow \TeX \rightarrow \TeX Live Utility).
2. Si scelga la voce Update All Packages dal menù Actions e si attenda la fine dell'aggiornamento.

2.4 LINUX

A causa di alcuni limiti imposti dagli sviluppatori, non si può installare e aggiornare \TeX Live sulle distribuzioni Linux con la semplicità degli altri sistemi operativi. Le istruzioni descritte nella guida di [Gregorio, 2013] però, permettono di farlo con estrema tranquillità. La procedura lì descritta, cui si rimanda, funziona su una distribuzione di Ubuntu correttamente installata e, con qualche modifica, anche su Fedora (e simili) e OpenSuSe (e simili).

3 | BASI

3.1 REALIZZARE UN DOCUMENTO

In questo paragrafo si mostrano le semplici fasi per realizzare un documento con \LaTeX , dalla scrittura del sorgente alla stampa.

Scrivere il codice

Si crei una cartella nella quale mettere *tutti* i file del documento. Dopo di che, con l'editor scelto si apra un nuovo file e si scriva il seguente codice (tutte le istruzioni verranno spiegate in questo e nel successivo capitolo):

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}

\begin{document}
Ecco il mio primo documento con \LaTeX.
\end{document}
```

Infine si registri il file come `primo.tex` (`.tex` è l'estensione dei sorgenti \LaTeX).

Comporre

Si componga il codice premendo l'apposito pulsante. Via via che \LaTeX elabora il sorgente, ne rendiconta dettagliatamente la composizione in un file apposito con estensione `.log` (è il *log*, in gergo) conservato in una cartella del sistema, e contemporaneamente ne mostra sullo schermo una versione ridotta, più o meno come la seguente (per brevità se ne sono omesse alcune parti):

```
This is pdfTeX, Version 3.14159265-2.6-1.40.17 (TeX Live 2016)
./primo.tex
LaTeX2e <2016/03/31> patch level 3
Babel <3.9r> and hyphenation patterns for 83 language(s) loaded
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/article.cls
Document Class: article 2014/09/29 v1.4h Standard LaTeX document class
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/fontenc.sty
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/tlenc.def))
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/inputenc.sty
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/utf8.def
(/usr/local/texlive/2016/texmf-dist/tex/latex/base/tlenc.dfu)))
(/usr/local/texlive/2016/texmf-dist/tex/generic/babel/babel.sty
(/usr/local/texlive/2016/texmf-dist/tex/generic/babel/italian.ldf))
Output written on primo.pdf (1 page, 15596 bytes).
SyncTeX written on primo.synctex.gz.
Transcript written on primo.log.
```

Si noti che premere il pulsante di composizione dell'editor equivale a eseguire dalla riga di comando l'istruzione

```
pdflatex primo.tex
```

L'interfaccia a riga di comando (che su Windows si chiama *Prompt dei comandi*, su Mac *Terminale* e su Linux *Terminale* o *Console*) si avvia con modalità proprie di ogni sistema operativo.

In questo modo, si è consapevoli del fatto che è \LaTeX (e non l'editor) il programma che elabora il codice `.tex`, che emette i messaggi e che produce il PDF, lavorando "dietro le quinte". La composizione non è dunque semplicemente "il tempo che bisogna attendere per vedere il PDF", ma il processo durante il quale il programma comprende le intenzioni dell'utente (esprese con i comandi), e le trasforma in un file (tipo)grafico.

Errori e warning

Per quanto scrupolosi si possa essere, di tanto in tanto qualche errore s'infiltra nel sorgente. Quando \LaTeX s'imbatte in qualcosa che non capisce o che non può fare, arresta la composizione in corso e mostra quello che non va in un messaggio, che può essere di due tipi:

- un messaggio di vero e proprio *errore* (nomi di comandi scritti scorrettamente, parentesi dimenticate, caratteri speciali usati per sbaglio, per esempio): la composizione si arresta e il programma rimane in attesa di istruzioni da parte dell'utente;
- un messaggio di *avviso* (o *warning*, in gergo), meno grave del primo (righe troppo lunghe o troppo corte, pagine riempite troppo o troppo poco, riferimenti incrociati irrisolti, per esempio): il programma semplicemente lo notifica ma porta a termine la composizione.

Gli errori vanno corretti *obbligatoriamente*. In caso contrario, se sono pochi e non gravi si potrà comunque completare la composizione, anche se in genere con un risultato diverso da quello atteso; se invece sono gravi (ne basta anche uno solo), la composizione potrebbe bloccarsi del tutto. Ai warning, semplici commenti su questioni di secondaria importanza, ci si potrà dedicare a lavoro ultimato.

Un esempio chiarirà le idee. Il nome di un comando scritto scorrettamente provoca l'arresto della composizione e la comparsa di un messaggio di questa forma (uguale per tutti i messaggi d'errore di questo tipo):

```
! Undefined control sequence.
l.6 Ecco il mio primo documento con \latex
?
```

Quando trova un errore, il programma segnala all'utente:

- la natura del messaggio (un errore, in questo caso) e il suo emittente (\TeX , in questo caso: un errore emesso da \LaTeX comincia sempre con `! LaTeX Error:`),
- la natura dell'errore (in questo caso, `Undefined control sequence`, "sequenza di controllo non definita", ovvero "comando sconosciuto");

Tabella 2: Modi di interazione tra \LaTeX e utente

Chiave	Ordina a \LaTeX di
Invio	ignorare l'errore e riprendere la composizione (va dato a ogni successivo errore)
q	continuare la composizione senza visualizzarne le informazioni e arrestarla solo quando un file non viene trovato (<i>carry on quietly</i>)
r	fare come nel caso precedente, ma informando l'utente sulla composizione (<i>run</i>)
s	ignorare gli errori e arrestare la composizione solo quando un file non viene trovato (<i>scroll</i>)
e	arrestare la composizione mostrando il documento composto fino a quel punto e tornare all'editor con il cursore nella riga che contiene l'errore (<i>edit</i>)
h	soccorrere l'utente con un messaggio d'aiuto (<i>help</i>)
i	attendere dall'utente la stringa corretta senza modificare il sorgente (<i>insert</i>)
x	arrestare immediatamente la composizione (<i>exit</i>)

- la riga (*line*) esatta del codice sorgente in cui si trova l'errore (l.6);
- il testo contenuto nella riga appena segnalata (o parte di esso), alla fine del quale si trova l'errore in questione: se il testo continuasse, la riga incriminata verrebbe spezzata in corrispondenza del punto problematico e continuata subito sotto.

Come comportarsi? Se si sta usando un editor interattivo (come quelli consigliati in questa guida), sarà possibile comunicare a \LaTeX le proprie intenzioni in uno dei modi mostrati nella tabella 2, dando poi Invio. A essere precisi, un errore come quello appena mostrato può avere due cause: o è sbagliato il nome del comando (quello giusto è \LaTeX e non \latex : attenzione alle maiuscole!); oppure, sebbene scritto correttamente, si sta usando un comando definito da un pacchetto non caricato. Essendo giusta la prima ipotesi, basta dare x e Invio, correggere il codice, registrare il file e ricomporre (oppure, con alcuni editor, ricomporre direttamente).

Si noti che, per quanto generalmente molto chiari, i messaggi di \LaTeX potrebbero richiedere un po' d'esercizio iniziale per essere interpretati correttamente a prima vista. In particolare, si ricordi che le notifiche d'errore possono provenire da più d'una fonte (\TeX , \LaTeX , la classe in uso, uno dei pacchetti), che le cause per uno stesso errore possono essere più d'una e che la riga indicata si riferisce al sorgente *in quel momento* elaborato: lo si tenga presente se il proprio documento è suddiviso in più file. A fronte di errori ricorrenti e apparentemente inspiegabili, infine, un metodo spesso efficace è quello di eliminare i file ausiliari generati dalla composizione e ricomporre il documento.

Per una panoramica su errori più frequenti e possibili soluzioni, si vedano [Lamport, 1994] e [Goossens *et al.*, 2004].

Visualizzare e stampare

Se si usa un editor dedicato e il sorgente non contiene (più) errori, la composizione andrà a buon fine e il visualizzatore di PDF si attiverà automa-

ticamente mostrando il documento finito, che si potrà poi stampare con le modalità proprie del programma.

3.2 CODIFICHE E LINGUE

3.2.1 Le codifiche di \LaTeX

\LaTeX è un programma nato per scrivere documenti *in inglese* ad alto contenuto matematico, e a questo scopo è stato originariamente corredato di una dotazione minima di caratteri (lettere, numeri e pochi altri segni) del tutto sufficiente, tant'è vero che il codice

```
\documentclass{<...>}
\begin{document}
...
\end{document}
```

permette di scrivere senza problemi un documento in quella lingua. La corretta scrittura di parole straniere, tuttavia, potrebbe richiedere di caricare i pacchetti descritti in questo paragrafo anche in un documento completamente in inglese.

Com'è noto, l'inglese si scrive senza accenti o caratteri particolari, contrariamente alla vasta maggioranza delle altre lingue che usano l'alfabeto latino. Gli unici caratteri "complicati" dell'italiano sono le vocali accentate, presenti (non proprio tutte) anche su una tastiera italiana. Affinché \LaTeX le "accetti" se inserite nell'editor con i tasti appositi, restituendole adeguatamente nel documento composto e sillabando correttamente le parole che le contengono, *immediatamente dopo* la dichiarazione di classe vanno caricati altri due pacchetti nell'ordine seguente:

```
\documentclass[<...>]{<...>}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
```

dove:

- fontenc (*font encoding*, "codifica dei font") si occupa dei font che si vedranno nel documento composto e fornisce a \LaTeX i caratteri particolari di una certa lingua già disegnati. Si noti che per funzionare al meglio il pacchetto richiede una distribuzione *completa* di \LaTeX , altrimenti i caratteri potrebbero apparire "sgranati" sullo schermo (ma non a stampa).
- T1 è la codifica dei font usati per scrivere in italiano e in molte altre lingue occidentali: per lingue o alfabeti particolari si usano altre codifiche. In documento multilingue l'*ultima* codifica dichiarata è quella della lingua principale: [T2A, T1] per un documento in italiano e russo, per esempio.
- inputenc (*input encoding*, "codifica di input") serve a \LaTeX per interpretare correttamente i caratteri immessi nell'editor.
- utf8 è la *codifica di input*, in gergo, che permette di scrivere nell'editor i segni di numerosi alfabeti direttamente dalla tastiera, evitando di

dover caricare ogni volta la codifica adatta alla lingua del documento. Se il proprio editor non supporta *pienamente* utf8, si usi latin1.

Non ci sono alternative: senza i due pacchetti appena descritti le vocali accentate non vengono visualizzate nel documento composto. Caricandoli, inoltre, il vecchio metodo di comporre *a mano* i caratteri con l'accento e fonte di non pochi problemi di sillabazione nelle parole che li contenevano, mostrato nell'esempio seguente,

Basta! Non se ne pu\`o pi\`u!
Perch\`e non c'\`e piet\`a
per chi deve scrivere cos\`i?

Basta! Non se ne può più! Perché
non c'è pietà per chi deve scrivere
così?

non serve più, e il codice è decisamente più pulito:

Però: caricando il pacchetto
giusto la vita diventerà così
facile che c'è molto più gusto!

Però: caricando il pacchetto giusto
la vita diventerà così facile che c'è
molto più gusto!

Si noti che, a patto di usare la tastiera “giusta” e \LaTeX (si veda il paragrafo 18.2.2 a pagina 297), la codifica Unicode permette di scrivere direttamente nel codice anche caratteri di alfabeti non latini.

3.2.2 Problemi con le codifiche

Un ulteriore aspetto da tenere bene a mente, spesso fonte di grattacapi e danni a volte irrimediabili, è il rapporto tra la codifica con cui è scritto un sorgente e quella con cui è impostato l'editor in uso, che *devono* coincidere, altrimenti aprendo un file sconosciuto si potrebbero vedere caratteri bizzarri sullo schermo. Se ciò si verificasse, *non* si componga né si registri il file *per nessun motivo*, ma lo si chiuda e si seguano le procedure descritte in [Beccari e Gordini, 2016], a cui si rimanda per individuare e risolvere i principali problemi in questo senso. Comporre il file, infatti, potrebbe danneggiarlo più o meno seriamente. In ogni caso, si legga la documentazione del proprio editor: potrebbe rivelarsi molto utile per evitare di questi pasticci.

3.2.3 \LaTeX e le lingue

Il terzo e ultimo pacchetto da caricare *sempre* è babel, che agisce su parole fisse (cioè le voci generate automaticamente dai comandi raccolti nella tabella 57), date, convenzioni tipografiche e sulla scelta delle regole di sillabazione, che per italiano e principali lingue europee sono già comprese in \TeX Live. Come opzioni prende una o più lingue e per un documento in italiano lo si carica nel modo seguente:

```
\usepackage[⟨...⟩,italian]{babel}
```

dove l'*ultima* dichiarata è la lingua principale del documento. Inoltre, per ciascuna lingua definisce nuovi comandi per semplificare l'immissione dei caratteri particolari nazionali, come si può vedere nella documentazione del pacchetto o in [Gregorio, 2010].

Si dà, infine, il tipico inizio di un sorgente per un documento in italiano con la corretta sequenza dei pacchetti da caricare:

```
\documentclass[⟨...⟩]{⟨...⟩}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
```

Il pacchetto babel definisce alcuni comandi molto utili per trattare correttamente ciascuna lingua in un documento multilingue. Supponendo di dover scrivere un documento in italiano con alcune parti in inglese, babel andrà caricato così:

```
\usepackage[english,italian]{babel}
```

Per singole parole o brevi frasi in lingua straniera è disponibile il comando

```
\foreignlanguage{⟨lingua⟩}{⟨testo⟩}
```

Per porzioni di testo in lingua più consistenti è disponibile l'ambiente

```
\begin{otherlanguage*}{⟨lingua⟩}
...
\end{otherlanguage*}
```

che però non traduce parole fisse e date. Se invece se ne avesse bisogno, si usi l'analogo ambiente `otherlanguage`.

3.3 FILE CON CUI SI HA A CHE FARE

Guardando nella cartella di lavoro dopo la prima composizione, si noterà che oltre al file `.tex` iniziale e agli eventuali file esterni come le immagini ce ne sono altri: sono i *file ausiliari* prodotti dalla composizione, e questo è un altro aspetto per cui \LaTeX è molto diverso dai programmi tradizionali. Il loro numero dipende dalla complessità del documento, ma è importante sapere che a parte il file `.bib` *non* vanno toccati: \LaTeX li crea e se ne serve automaticamente.

La tabella 3 a fronte ne raccoglie e descrive i principali.

3.4 FILE SORGENTE

Un sorgente di \LaTeX è un file di puro testo che contiene sia il testo vero e proprio del documento sia i comandi che istruiscono \LaTeX su come trattarlo.

3.4.1 Comandi e ambienti

Un comune programma di videoscrittura a composizione sincrona e \LaTeX presentano per certi versi un funzionamento simile: entrambi ricevono dall'utente sia il testo sia le istruzioni per impostarne l'aspetto. La differenza sostanziale, però, è che il primo le nasconde, proponendo all'utente menù da cui scegliere comandi "preconfezionati" che mostrano immediatamente i propri effetti; \LaTeX , invece, richiedendo di scrivere esplicitamente i comandi, mantiene queste istruzioni in superficie nei modi spiegati di seguito.

Tabella 3: Principali file ausiliari di \LaTeX

Prodotti da	Estensione	Descrizione
Utente	.bib	Database bibliografico
	.jpg, .pdf, .png	Formati grafici per \LaTeX
	.tex	File sorgente
Classi, pacchetti e stili	.bst	Stile bibliografico per Biber
	.cls	Classe di documento
	.sty	Pacchetto
Composizione	.aux	Trasporta informazioni generiche
	.lof	Indice delle figure
	.log	Rendiconta l'ultima composizione
	.lot	Indice delle tabelle
	.toc	Indice generale
Pacchetti e programmi	.bbl	Bibliografia creata con Biber
	.blg	Rendiconto di Biber
	.idx	Voci dell'indice analitico
	.ind	Prodotto di MakeIndex
	.ilg	Rendiconto di MakeIndex
	.out	Segnalibri ipertestuali
Output	.pdf	Prodotto di \LaTeX

Comandi e dichiarazioni

Un *comando* è un'istruzione che ordina a \LaTeX di trattare in un certo modo una porzione più o meno ampia di testo. Si possono classificare i comandi di \LaTeX in base a forma e funzione.

Per quanto riguarda la *forma* si distinguono tre tipi di comando, a seconda che siano costituiti da:

- Un solo carattere non alfabetico. Questi comandi sono quattro in tutto: spazio `\`, tilde `~`, circonflesso `^` e trattino basso `_`.
- Una barra rovescia `\` seguita da un solo carattere non alfabetico (cioè non compreso fra A-Z o a-z). Un comando di questo tipo termina al primo carattere non alfabetico e uno o più spazi (che contano per un solo spazio) immediatamente dopo *non* vengono ignorati. Alcune tra le combinazioni più usate sono: `\{`, `\}`, `\%`, `\$`, `_`, `\&`, `\#`, `\~`.
- Una barra rovescia `\` seguita da una sequenza di caratteri alfabetici. Un comando di questo tipo termina al primo carattere non alfabetico e uno o più spazi immediatamente dopo vengono ignorati. Si noti che questi comandi distinguono maiuscole e minuscole. Alcuni esempi: `\LaTeX`, `\emph`, `\documentclass`.

I comandi del terzo tipo che “producono testo” (come `\LaTeX`, `\TeX`, `\Ars`, `\dots`, `\today` e pochissimi altri) richiedono di essere “terminati”, pena una spaziatura errata dopo di sé. La cosa migliore è usare la barra rovescia o un gruppo vuoto `\{ }`, come mostrano gli esempi seguenti:

Tabella 4: Principali tipi di comando in base a numero e tipo di argomenti richiesto (F indica gli argomenti facoltativi, O quelli obbligatori)

Argomenti		Esempio
F	O	
		<code>\LaTeX</code>
1		<code>\item[...]</code>
	1	<code>\emph{...}</code>
1	1	<code>\documentclass[...]{...}</code>
2	1	<code>\subfloat[...][...]{...}</code>
1	2	<code>\pdfbookmark[...]{...}{...}</code>
	3	<code>\multicolumn{...}{...}{...}</code>

```
\Ars è la rivista del Gruppo
Utilizzatori Italiani di
\TeX e \LaTeX. \[lex]
\Ars\ è la rivista del Gruppo
Utilizzatori Italiani di
\TeX{} e \LaTeX.
```

```
4rsTeXnicaè la rivista del Gruppo
Utilizzatori Italiani di TExe LATEX.
4rsTeXnica è la rivista del Gruppo
Utilizzatori Italiani di TEx e LATEX.
```

La scrittura *sempre* corretta di questi comandi è una di quelle contenute nel secondo esempio. Si noti che un qualunque segno di punteggiatura immediatamente dopo il comando elimina la necessità dello spazio esplicito.

Per quanto riguarda la *funzione* si distinguono due tipi di comando, a seconda della porzione di testo su cui hanno effetto:

- Comandi come `\textit{<testo>}` ordinano a \LaTeX di trattare in un certo modo solo il `<testo>` scritto tra le parentesi graffe.
- Comandi come `\itshape`, detti *dichiarazioni*, ordinano a \LaTeX di trattare in un certo modo *tutto* il testo successivo al punto in cui vengono dati.

In altre parole, una dichiarazione è un comando che imposta uno o più aspetti generali della composizione, e può essere data:

- nel preambolo, e allora ha effetto sull'intero documento e si annulla soltanto con un'altra dichiarazione;
- nel corpo del documento, e allora va data in un *gruppo* (cioè una porzione di testo racchiusa di solito da parentesi graffe o comandi di inizio e fine ambiente).

Sono esempi di dichiarazione: `\small`, `\linespread`, `\appendix`.

Nell'esempio seguente si vedono all'opera alcuni comandi visti fin qui:

```
Data odierna: \today. \[
Sarò lì in \emph{un} minuto. \[
Tutto il {\itshape testo
seguito è in corsivo}.
```

```
Data odierna: 6 gennaio 2018.
Sarò lì in un minuto.
Tutto il testo seguente è in corsivo.
```

Si noti che il comando `\today` produce la data in cui si compone il documento secondo le convenzioni della lingua in uso.

Tabella 5: Caratteri speciali di \LaTeX

Carattere	Funzione	Codice
<code>\</code>	Comincia un comando	<code>\textbackslash</code>
<code>{ }</code>	Delimitano un gruppo	<code>\{ \}</code>
<code>\$</code>	Delimita la matematica in linea	<code>\\$</code>
<code>^</code>	Esponente matematico	<code>\^{}</code>
<code>_</code>	Pedice matematico	<code>_</code>
<code>&</code>	Separa le celle in una tabella	<code>\&</code>
<code>#</code>	Numero dell'argomento	<code>\#</code>
<code>~</code>	Spazio indivisibile	<code>\~{}</code>
<code>%</code>	Commento	<code>\%</code>

La giustapposizione degli elementi di un comando prende il nome di *sintassi* del comando. Ciò che va tra parentesi graffe si chiama *argomento obbligatorio*, mentre ciò che va tra parentesi quadre si chiama *argomento facoltativo*. Se gli elementi da scrivere nello stesso gruppo di parentesi sono più d'uno, vanno separati con una virgola *senza ulteriori spazi*. La tabella 4 nella pagina precedente mostra i principali tipi di comando in base a numero e tipo di argomenti che richiedono.

Ambienti

Un *ambiente* è una porzione di codice delimitata da un comando d'apertura e uno di chiusura, che \LaTeX tratta in un certo modo. La sintassi di un ambiente generico è:

```
\begin{<ambiente>}[<...>]{<...>}
...
\end{<ambiente>}
```

dove:

- `<ambiente>` è il nome dell'ambiente;
- se presenti, argomenti facoltativi e obbligatori si scrivono dopo il solo comando d'apertura;
- l'ambiente va separato dal resto del testo con una riga bianca prima e dopo se ciò che contiene *non* appartiene al flusso del discorso (una figura, per esempio); non va separato in caso contrario.

\LaTeX permette di annidare gli ambienti, purché l'ordine di chiamata venga rispettato:

```
\begin{ambiente}
...
\begin{Ambiente}
...
\end{Ambiente}
...
\end{ambiente}
```

Tabella 6: Scorciatoie da tastiera (italiana) per alcuni caratteri frequenti

Carattere	Windows	Mac	Linux
‘	Alt + 96	⌘ 9	Alt Gr + ’
{	Alt + 123	⌘ ⇧ [Alt Gr + 7
	Alt Gr + Maiusc + [Alt Gr + Maiusc + [
}	Alt + 125	⌘ ⇧]	Alt Gr + 8
	Alt Gr + Maiusc +]		Alt Gr + Maiusc +]
~	Alt + 126	⌘ 5	Alt Gr + `

3.4.2 Caratteri speciali

\LaTeX interpreta in modo particolare alcuni caratteri molto richiesti nella scrittura del codice: sono i cosiddetti *caratteri speciali*, così chiamati perché non possono essere stampati normalmente se non come mostra la tabella 5 nella pagina precedente. La loro alta frequenza, però, si scontra con il fatto che su nessuna tastiera, a parte quella inglese internazionale, ci sono tutti, e perciò la loro scrittura richiede combinazioni di tasti o codici numerici particolari.

Si noti che:

- i caratteri { ‘ ~ } mancano sulla tastiera italiana: la tabella 6 indica le scorciatoie da prendere in questi casi (su Windows, il codice relativo va digitato *sul tastierino numerico*);
- si distingue con attenzione ‘ (virgoletta alta aperta, accento grave) da ’ (apostrofo, virgoletta alta chiusa, accento acuto);
- il comando `\textbackslash` non sostituisce la sequenza `\\`, come potrebbe sembrare: sono infatti due comandi distinti con distinte funzionalità (verranno considerati nei prossimi capitoli).

3.4.3 Struttura del file sorgente

\LaTeX si aspetta di trovare il sorgente da elaborare strutturato in un certo modo. Elementi fondamentali sono almeno una dichiarazione di classe

```
\documentclass{<...>}
```

e le dichiarazioni d’inizio e fine documento:

```
\begin{document}
...
\end{document}
```

Tutte le istruzioni scritte tra `\documentclass` e `\begin{document}` *inclusi* costituiscono il *preambolo del documento* (o semplicemente *preambolo*) e comprendono:

- il caricamento di pacchetti che estendono le capacità di \LaTeX ;
- le definizioni di comandi e ambienti personalizzati, che si consiglia di organizzare come indicato nel paragrafo 3.14 a pagina 48;
- le opzioni generali del documento.

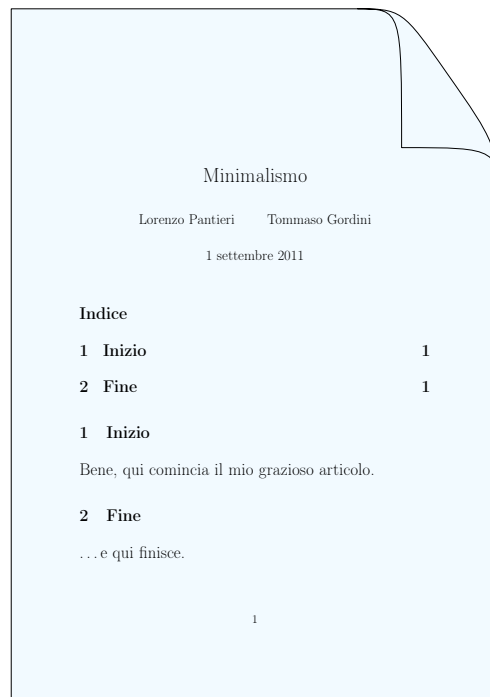


Figura 4: Documento elementare composto con \LaTeX

Si noti che un sorgente \LaTeX richiede *un solo* preambolo.

Fra `\begin{document}` e `\end{document}` va scritto il *corpo del documento*, cioè il vero e proprio testo che \LaTeX elaborerà e mostrerà nel documento finito. La figura 4 mostra il risultato della composizione del codice seguente (si possono intercalare o meno righe bianche per evidenziarne la struttura e facilitarne l'individuazione delle parti):

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}

\begin{document}

\author{Lorenzo Pantieri \and Tommaso Gordini}
\title{Minimalismo}
\maketitle

\tableofcontents

\section{Inizio}
Bene, qui comincia il nostro grazioso articolo\dots

\section{Fine}
\dots e qui finisce.

\end{document}
```

Dove:

- `\begin{document}` segnala l'inizio del documento;
- `\author` e `\title` (che si possono dare anche prima del comando precedente) ne specificano rispettivamente nome dell'autore e titolo;

- `\and` si spiega da sé;
- `\maketitle` produce il contenuto dei due comandi precedenti, *dopo* i quali deve essere dato;
- `\tableofcontents` produce l'indice generale dopo *due* composizioni;
- `\section{<titolo>}` produce un titolo di sezione (un paragrafo, in questo caso);
- `\dots` produce i puntini di sospensione ... ;
- `\end{document}` segnala la fine del documento.

L^AT_EX ignora tutto ciò che si trova dopo `\end{document}`: questo spazio, perciò, potrebbe essere un buon posto per appuntare un promemoria sul documento in lavorazione.

3.4.4 Spazi e righe vuote

Il modo in cui L^AT_EX tratta spazi, tabulazioni e righe vuote nel sorgente è particolarissimo, e decisamente diverso da quello di tutti i comuni elaboratori di testo. Infatti:

- una tabulazione è considerata come uno spazio;
- più spazi consecutivi sono considerati come un solo spazio;
- spazi o tabulazioni all'inizio di una riga vengono ignorati;
- una sola interruzione di riga è trattata come uno spazio;
- una riga vuota tra due righe di testo separa due capoversi;
- più righe vuote consecutive sono trattate come una sola riga vuota.

L'esempio seguente mostra all'opera i casi appena descritti:

Non ha alcuna importanza se si mettono uno o tanti spazi dopo una parola.

E neppure se si mettono tanti spazi all'inizio di una riga o se la s'interrompe.

Le cose cambiano se si saltano una o più righe, perché in questo modo si comincia un nuovo capoverso.

Non ha alcuna importanza se si mettono uno o tanti spazi dopo una parola. E neppure se si mettono tanti spazi all'inizio di una riga o se la s'interrompe.

Le cose cambiano se si saltano una o più righe, perché in questo modo si comincia un nuovo capoverso.

3.4.5 Commenti

Quando L^AT_EX incontra un carattere di percento % (tranne che nella forma `\%`) elaborando un sorgente, ignora il resto della riga, l'interruzione di riga, e tutti gli spazi bianchi all'inizio della riga successiva. Questo carattere è utile,

dunque, per appendere una nota o un promemoria (che non verrà stampato) proprio lì dove serve.

Talvolta però va usato per spezzare parole troppo lunghe o per dividere righe in cui non sono permessi spazi bianchi o interruzioni (come nel caso di comandi troppo lunghi per stare in una sola riga dell’editor). Ecco lo all’opera:

```
Ecco un % semplice,
% ma istruttivo
esempio: Supercal%
          ifragilist%
          ichespiralidoso.
```

Ecco un esempio: Supercalifragili-
stichespiralidoso.

Per commenti più lunghi si può usare l’ambiente `comment` definito dall’omonimo pacchetto:

```
Ecco un altro
\begin{comment}
semplice,
ma utile
\end{comment}
esempio per includere commenti
nel proprio documento.
```

Ecco un altro esempio per includere commenti nel proprio documento.

3.4.6 Sorgenti ordinati

Spesso gli utenti di \LaTeX sottovalutano l’importanza di un sorgente “pulito” (con rientri, incolonnamenti, eccetera) e commentato. *Non* è indispensabile, ma si consiglia di farlo ugualmente: l’ordine ne facilita la gestione, specie se a uno stesso progetto lavorano più persone, e rende più facile individuare eventuali errori.

Durante la stesura si raccomanda di usare i commenti, di suddividere con chiarezza il documento e di aiutarsi eventualmente con rientri, incolonnamenti, a capo e righe vuote supplementari. Ulteriori consigli in questo senso verranno forniti nelle prossime pagine al momento opportuno.

Si realizzano ora le indicazioni precedenti in un sorgente “ben scritto”:

Esempio di articolo composto con \LaTeX

```
% Un articolo scritto con LaTeX
\documentclass[a4paper,11pt]{article}
\usepackage[T1]{fontenc}           % codifica dei font
\usepackage[utf8]{inputenc}       % lettere accentate da tastiera
\usepackage[italian]{babel}       % lingua del documento
\usepackage{lipsum}               % genera testo fittizio
\usepackage{url}                  % per scrivere gli indirizzi Internet

\begin{document}

\author{Lorenzo Pantieri \and Tommaso Gordini}
\title{Il titolo}
\maketitle

\begin{abstract}
\lipsum[1]
```

```

\end{abstract}

\tableofcontents

\section{Un paragrafo}
\lipsum[1]

\subsection{Un sottoparagrafo}
\lipsum[1]

\section{Un paragrafo}
\label{sec:esempio}
\lipsum[1]

% Bibliografia
\begin{thebibliography}{9}
\bibitem{pantieri:arte}
Pantieri, Lorenzo e Tommaso Gordini (2017),
\emph{L'arte di scrivere con \LaTeX},
\url{http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf}.
\end{thebibliography}

\end{document}

```

3.5 CLASSI DI DOCUMENTO

La prima informazione che \LaTeX si aspetta di trovare nel sorgente è il *tipo di documento* che si desidera realizzare (la *classe*, in gergo), da specificare di norma come prima cosa con il comando

```
\documentclass[⟨opzioni⟩]{⟨classe⟩}
```

dove:

- $\langle\text{opzioni}\rangle$ sono le impostazioni generali del documento;
- $\langle\text{classe}\rangle$ è la classe di documento scelta.

Di seguito si elencano le principali classi di documento *standard* (cioè definite dal programma):

article per scrivere articoli;

report per scrivere relazioni o tesi suddivise in capitoli;

book per scrivere libri;

letter per scrivere lettere.

Esistono numerose altre classi non standard per i documenti più diversi. Tra quelle più diffuse ci sono memoir (che permette molta libertà nel personalizzare il documento), toptesi e suptesi (per tesi di laurea e dottorato), beamer (per presentazioni). Per esempio, questa guida è scritta con lo stile ArsClassica basato sulle classi KOMA-Script (per maggiori dettagli si vedano i capitoli 22 a pagina 317 e 23 a pagina 335).

Le $\langle\text{opzioni}\rangle$ date a `\documentclass` si dicono anche *globali*, perché agiscono sull'intero documento. Di seguito si descrivono quelle più comuni per

Tabella 7: Opzioni più comuni delle classi standard di \LaTeX . I simboli ●, ◐ e ○ indicano rispettivamente che l'opzione è predefinita, applicabile (anche se non predefinita), non applicabile.

Opzione	book	report	article
10pt	●	●	●
letterpaper	●	●	●
oneside	◐	●	●
twoside	●	◐	◐
openany	◐	●	○
openright	●	◐	○
titlepage	●	●	◐
final	●	●	●

le tre classi `article`, `report` e `book` (la classe `letter` meriterebbe una trattazione a parte, perciò non viene considerata). Si noti che alcune di esse hanno un'applicabilità limitata, come mostra la tabella 7.

10pt, 11pt, 12pt Impostano la dimensione del font principale del documento. Omettendo l'opzione, il valore predefinito è 10pt.

a4paper, a5paper, ... Definiscono le dimensioni del foglio, che per impostazione predefinita è nel formato americano `letterpaper`. Le altre opzioni possibili sono `executivepaper`, `legalpaper` e `b5paper`.

oneside, twoside Specificano se verrà composto un documento a singola o doppia facciata rispettivamente. Per impostazione predefinita, le classi `article` e `report` sono a singola facciata e la classe `book` è a doppia facciata.

openany, openright L'opzione `openany`, predefinita nella classe `report`, fa cominciare un capitolo nella successiva pagina a disposizione; l'opzione `openright`, predefinita nella classe `book`, lo fa cominciare sempre in una pagina destra. Entrambe non sono disponibili nella classe `article`, che non ammette la suddivisione in capitoli.

twocolumn Dà a \LaTeX le istruzioni per comporre l'intero documento su due colonne (si veda anche il paragrafo 13.3 a pagina 249).

titlepage, notitlepage Specificano se dopo il titolo del documento debba avere inizio una nuova pagina (come accade con `report` e `book`) o no (come accade con `article`) rispettivamente.

fleqn Allinea le formule a sinistra rispetto a un margine rientrato.

legno Mette la numerazione delle formule a sinistra anziché a destra.

draft, final L'opzione `draft` evidenzia le righe composte in modo non ottimale con un rettangolino nero ■ accanto, facilitandone l'individuazione sulla pagina. Ciò non accade con `final`. Si noti che entrambe influenzano il comportamento degli altri pacchetti caricati o addirittura li disabilitano del tutto (si veda il paragrafo 14.1 a pagina 257 per risolvere il problema).

Un tipico sorgente, allora, potrebbe cominciare con la riga

```
\documentclass[a4paper,11pt,twoside]{article}
```

che ordina a \LaTeX di impaginare il documento come un articolo, su carta di formato A4, con un carattere di 11 punti e impostato per la stampa fronte/retro.

3.6 GESTIRE LA PAGINA

3.6.1 Il tormentone dei margini

I margini della pagina in tipografia rivestono funzioni importantissime, prima fra tutte quella di delimitare in modo chiaro il testo. Il lettore così potrà individuarlo agevolmente sulla pagina e appoggiare i pollici su uno spazio sufficientemente confortevole per maneggiare comodamente il documento [Bringhurst, 1992]. È perciò che nei documenti impostati per la stampa in fronte/retro i margini esterni sono più ampi di quelli interni, che appaiono duplicati perché adiacenti.

La maggior parte degli utenti europei, che stampa su carta in formato A4, ritiene troppo ampi i margini predefiniti da \LaTeX nelle classi standard, e che di conseguenza la pagina non sia sufficientemente riempita. Prima di buttarsi nella frenesia dell' "allarghiamo un po' questa strettissima pagina", però, è doveroso riflettere.

I margini di \LaTeX derivano da convenzioni tipografiche ampiamente verificate e accettate, e mettono l'utente nelle vantaggiose condizioni di potersene servire per ottenere risultati professionali *già alla prima composizione* e senza doverci mettere le mani. Modificarli, perciò, significherebbe dover studiare un (bel) po' di tipografia prima di raggiungere risultati accettabili.

L'esperienza dimostra che leggere diventa tanto più difficile quanto più numerosi sono i caratteri in una singola riga di testo: l'occhio è costretto a compiere movimenti più ampi e si affatica presto (perciò quotidiani e riviste sono stampati su più colonne). Robert Bringhurst ha codificato quest'esperienza nella sua celebre "regola", che considera ottimale il numero di circa 66 caratteri per riga (spazi inclusi), indipendentemente dal font usato. L'ampiezza media in punti di un carattere corrisponde al rapporto tra la lunghezza dell'alfabeto latino minuscolo "abcdefghijklmnopqrstuvwxyz" e il numero di lettere che lo compongono (26). Se si considera che per riempire meglio la pagina \LaTeX usa già *in partenza* una riga più lunga del limite stabilito da Bringhurst, è chiaro che modifiche in questo senso vanno evitate il più possibile.

In alcune circostanze, tuttavia, può essere desiderabile o necessaria una maggiore copertura della pagina: tra i numerosi pacchetti scritti a questo scopo se ne consigliano due.

Il pacchetto LayAureo (se ne veda la documentazione in italiano) definisce un layout di pagina pronto per l'uso, permettendo di impostare facilmente anche lo spazio per la rilegatura con la chiave `binding=<dimensione>`. Il pacchetto agisce semplicemente caricandolo e non è personalizzabile: dunque, o piace o non piace.

Se servissero proporzioni di pagina ancora diverse (perché la propria facoltà impone un modello di tesi particolare, per esempio) da applicare

anche a una sola pagina, può risolvere il problema il pacchetto `geometry`, completamente configurabile.

Si immagini di dover comporre un documento in formato A4 con margini superiore e inferiore di 3 cm, sinistro e destro (che nella stampa in fronte/re-ro diventano interno ed esterno) di 3,5 cm e di voler destinare alla rilegatura uno spazio di 5 mm. Il codice da scrivere nel preambolo è il seguente:

```
\usepackage{geometry}
\geometry{a4paper,top=3cm,bottom=3cm,left=3.5cm,right=3.5cm,%
heightrounded,bindingoffset=5mm}
```

Tra le opzioni del pacchetto che come il precedente, si noti bene, agisce anche solo caricandolo, si consiglia sempre anche `heightrounded`, che modifica ulteriormente di poco le dimensioni della gabbia del testo per farle contenere un numero intero di righe.

Si eviti *nel modo più assoluto*, invece, di toccare comandi interni di \LaTeX come `\textwidth`, `\oddsidemargin`, eccetera, perché la loro azione non tiene in nessun conto le proporzioni di pagina [Fairbairns, 2014].

Talvolta il rilegatore potrebbe non sapere dove tagliare il foglio: il pacchetto `crop` stampa sul documento i crocini di taglio.

3.6.2 Interlinea e riempimento della pagina

Interlinea

Non pochi editori, relatori e regolamenti di facoltà impongono di impaginare pubblicazioni e tesi in modo non professionale, a partire dalla questione dell'*interlinea*. L'interlinea standard di \LaTeX garantisce un risultato tipografico ottimale e non andrebbe modificata senza una ragione precisa. Per farlo si consiglia il pacchetto `setspace`, che modifica lo *scartamento* (o *avanzamento di riga*), ovvero lo spazio fra le righe di base di due righe adiacenti; il pacchetto definisce tre scartamenti *globali* da impostare *nel preambolo* come segue:

- `\singlespacing` (scartamento 1);
- `\onehalfspacing` (scartamento 1,5);
- `\doublespacing` (scartamento 2).

Si può modificare l'interlinea soltanto in alcune parti del documento con gli ambienti `singlespace`, `onehalfspace` e `doublespace`, da usare nel modo consueto.

Se, infine, ne servisse una ancora diversa, si può dare *nel preambolo* il comando standard

```
\linespread{\fattore di scala}
```

che moltiplica lo scartamento per il $\langle \text{fattore di scala} \rangle$.

Riempimento della pagina

In mancanza di istruzioni specifiche \LaTeX cerca sempre di riempire la gabbia del testo per tutta la sua altezza. Se non riesce a farlo perché non c'è abbastanza testo, "stiracchia" il materiale che ha a disposizione inserendo dello spazio aggiuntivo dove può: tra i capoversi, tra le voci degli elenchi e così via. Si può disattivare questo comportamento ottenendo dello spazio bianco in fondo alla pagina scrivendo nel preambolo `\raggedbottom`.

3.7 STRUTTURARE IL DOCUMENTO

La tabella 8 nella pagina successiva raccoglie le principali istruzioni che producono una sezione nel documento e ne descrive il comportamento nelle classi standard. Non si considera la classe `letter` poiché non prevede alcun tipo di sezionamento.

3.7.1 Sezionare il corpo del testo e modificarne la numerazione

Per suddividere un documento in sezioni basta dare *nel corpo del testo* i comandi elencati nella prima parte della tabella, a proposito dei quali si noti quanto segue.

- I nomi inglesi dei comandi corrispondono alle unità di suddivisione del testo in vigore nei Paesi anglosassoni, e alcuni di essi non hanno corrispondenti italiani.
- Il comando `\section` produce una sezione equivalente al nostro *paragrafo*, e `\paragraph` (“capoverso”, in inglese) una sezione non numerata non equivalente né al nostro *paragrafo* né al nostro *capoverso* (che ammette al massimo un titoletto o una breve indicazione). Le stesse considerazioni valgono anche per i rimanenti comandi.
- Il comando `\part` non influenza la numerazione dei capitoli.
- Si consiglia di evitare suddivisioni così fini come quelle permesse dagli ultimi tre comandi e di assicurarsi che ogni sezione contenga *almeno due* sezioni di livello immediatamente inferiore: in caso contrario, l’unico comando di sezionamento presente diventa superfluo.

Nel documento saranno numerate automaticamente le sezioni fino a quelle di livello 2 compreso (si veda la tabella 10 a pagina 42). Per modificare la numerazione predefinita, si veda il paragrafo 3.9.3 a pagina 41.

3.7.2 Altri sezionamenti

Le altre istruzioni mostrate nella tabella 8 a fronte producono le corrispondenti sezioni descritte nei prossimi capitoli. Si noti che l’ambiente `abstract`, destinato a ospitare il sommario (o riassunto) del lavoro, è ammesso solo nelle classi `article` e `report` perché di solito nei libri è sostituito dall’introduzione.

3.7.3 Materiale iniziale, principale e finale

Oltre ai comandi appena illustrati, la sola classe `book` prevede tre dichiarazioni che agiscono al più alto livello possibile e costituiscono una specie di “supersezionamento”. Vanno date *sempre* nel corpo del documento e si comportano come segue:

- `\frontmatter` (“materiale iniziale”) non numera le sezioni e numera le pagine con numeri romani minuscoli (i, ii, iii, eccetera);
- `\mainmatter` (“materiale principale”) numera le sezioni e le pagine con numeri arabi (la numerazione della pagina riprende da 1);

Tabella 8: Istruzioni considerate in questa guida che producono una sezione nel documento e loro comportamento nelle classi standard. I simboli indicano che la sezione possiede la caratteristica sempre (●), mai (○), solo nella classe article (◐), solo nelle classi article e report (◑). Tra parentesi i pacchetti richiesti.

A che cosa serve	Istruzione	La sezione prodotta ha				Sezione
		numero	titolo	testatine	posto nell'indice	
Corpo del testo	<code>\part</code>	●	●	○	●	Parte
	<code>\chapter</code>	●	●	●	●	Capitolo
	<code>\section</code>	●	●	●	●	Paragrafo
	<code>\subsection</code>	●	●	○	●	Sottoparagrafo
	<code>\subsubsection</code>	◐	●	○	◐	Sotto-sottoparagrafo
	<code>\paragraph</code>	○	●	○	○	Sezione di livello ancora più basso
	<code>\subparagraph</code>	○	●	○	○	Sezione al più basso livello possibile
	Indici	<code>\tableofcontents</code>	○	●	●	○
<code>\listoffigures</code>		○	●	●	○	Elenco delle figure
<code>\listoftables</code>		○	●	●	○	Elenco delle tabelle
<code>\printindex</code>		○	●	●	○	Indice analitico (makeidx)
Bibliografia	<code>thebibliography</code>	○	●	●	○	Bibliografia manuale
	<code>\printbibliography</code>	○	●	●	○	Bibliografia automatica (biblatex)
Varie	<code>abstract</code>	○	◐	○	○	Sommario

Tabella 9: Struttura generale di un libro o una tesi. Le voci in corsivo sono obbligatorie, quelle in tondo sono facoltative, quelle asteriscate *non* devono comparire nell'indice generale.

Supersezionamento	Sezione
Materiale iniziale	<i>Frontespizio</i>
	Colophon*
	Dedica*
	Sommario*
	<i>Indice generale*</i>
	Elenco delle figure*
	Elenco delle tabelle*
	Altri elenchi*
	Prefazione
	Ringraziamenti*
	Introduzione non numerata
Materiale principale	Introduzione numerata
	<i>Capitoli</i>
	Una o più appendici numerate
Materiale finale	Una o più appendici non numerate
	Glossario
	<i>Bibliografia</i>
	Indice analitico

- `\backmatter` (“materiale finale”) non numera le sezioni e continua la numerazione araba delle pagine dal materiale principale.

3.7.4 Appendici

Per produrre le appendici basta dare la dichiarazione `\appendix`, che cambia i numeri dei capitoli (o dei paragrafi, se la classe impostata è `article`) in lettere. Il pacchetto `appendix` permette eventualmente di personalizzarle (se ne veda la documentazione).

3.7.5 Struttura generale di un libro o una tesi

La tabella 9 mostra una *possibile* successione dei componenti di una pubblicazione di una certa consistenza, come un libro o una tesi di laurea o dottorato. Lo schema è tratto con qualche variazione da [Mori, 2007], cui si rimanda per gli approfondimenti.

Si noti che:

- l'introduzione va senz'altro nel materiale principale se è a propria volta divisa in sezioni, mentre va in quello iniziale se è breve e contiene solo una sintetica esposizione dell'argomento;
- le appendici vanno valutate caso per caso in base a numero e importanza: se è una sola e poco importante può andare nel materiale finale; se invece è funzionale al corpo principale del documento va in quello principale.

Si ricordi, infine, che una sezione concepita come un capitolo non va *mai* messa nel materiale finale.

3.8 STILI DI PAGINA

Lo *stile di pagina* è l'organizzazione del contenuto di testatina e piede scelta per il documento, e va indicato nell'argomento `<stile>` del comando

```
\pagestyle{<stile>}
```

L^AT_EX prevede tre stili di pagina predefiniti e uno personalizzabile, descritti di seguito.

plain Mette i numeri di pagina nel piede, lasciando vuota la testatina. È lo stile predefinito nelle classi `article` e `report`.

empty Lascia testatina e piede vuoti.

headings Lascia il piede vuoto e compone le testatine come segue: il numero di pagina è sempre posto nel margine esterno, seguito dal titolo del capitolo corrente nella testatina di sinistra e preceduto dal titolo del paragrafo corrente in quella di destra. È lo stile predefinito nella classe `book` e agisce nello stesso modo nelle classi `report` e `article` impostate con `twoside`, con la differenza che nella seconda le testatine riportano i titoli di paragrafo e sottoparagrafo correnti, rispettivamente. Se invece s'imposta la classe con `oneside`, la testatina riporta soltanto il titolo della suddivisione maggiore.

myheadings È simile a `headings` nel risultato e va usato quando non si vuole che le testatine dipendano dai titoli delle sezioni (capitolo e paragrafo) correnti. L'utente deve specificarne il contenuto a ogni nuovo capitolo (o paragrafo, se la classe è `article`), dando `\markboth` per comporre entrambe oppure `\markright` per comporre soltanto quella di destra.

Si può cambiare lo stile della pagina *corrente* con il comando

```
\thispagestyle{<stile>}
```

Gestire le testatine

Nelle classi standard le testatine vengono prodotte dai comandi mostrati nella tabella 8 a pagina 37. Presenza o meno sulla pagina, contenuto e stile si possono regolare facilmente, ma *si raccomanda di non abusare* di queste possibilità e di attenersi alle scelte tipografiche della classe in uso. Si ricorda che le dichiarazioni di sezionamento descritte nel paragrafo 3.7.3 a pagina 36 *non* influenzano il comportamento delle testatine, che saranno dunque sempre presenti anche nelle sezioni non numerate prodotte da `\frontmatter`.

Per eliminare le testatine da una sezione che *l'utente* non ha numerato, invece, basta impostare per quella sezione lo stile di pagina `plain` (si veda il paragrafo 3.8) e ripristinare poi lo stile generale come segue:

```
\chapter*{Prefazione}
\pagestyle{plain}
...
```

```
\chapter{Introduzione}
\pagestyle{headings}
...
```

Si noti bene che se ci si dimentica di farlo, la sezione non numerata porterà le testatine prodotte dall'ultimo comando "utile" in questo senso.

Se comunque le si volessero, vanno inserite a mano con `\markboth`

```
\markboth{\MakeUppercase{\langle testatina di sinistra \rangle}}%
         {\MakeUppercase{\langle testatina di destra \rangle}}
```

che si usa come segue:

```
\chapter*{Prefazione}
\markboth{\MakeUppercase{Prefazione}}{\MakeUppercase{Prefazione}}
```

dove `\MakeUppercase` produce il proprio argomento in tutte maiuscole (come nelle testatine standard della classe book).

Eliminare testatine inutili e aggiungere pagine bianche

Per eliminare testatine e piedi *comunque* presenti nelle pagine bianche alla fine di un capitolo usando le classi standard con l'opzione `openright` (scelta consigliata), basta caricare il pacchetto `emptypage`.

Per aggiungere una pagina bianca dopo la pagina corrente, basta caricare il pacchetto `afterpage` e dare

```
\afterpage{\null\thispagestyle{empty}\clearpage}
```

3.9 INDICE GENERALE, TITOLI E PROFONDITÀ

3.9.1 Indice generale, miniindici e indici abbreviati

Il comando

```
\tableofcontents
```

produce *nel punto in cui viene dato* la sezione contenente l'indice generale con relativi titolo e testatina. Si noti che per ottenerlo nel documento finito servono *due* composizioni successive.

In particolari circostanze potrebbe essere utile inserire anche un *miniindice* fra titolo del capitolo e inizio del testo: il pacchetto `minitoc` permette di farlo automaticamente (se ne veda la documentazione), ma non se ne abusi.

Il pacchetto `shorttoc`, invece, genera indici abbreviati della profondità scelta (come un *Sommario* o un *Indice degli argomenti*), utili in documenti particolarmente corposi per descriverne il contenuto senza scendere nei dettagli.

3.9.2 Gestire i titoli

Titoli non numerati né indicizzati

Di tutti i comandi di sezionamento elencati nella tabella 8 a pagina 37 esiste anche una *variante asterisco* formata dal nome del comando con un asterisco alla fine, che genera titoli *non numerati* e che nemmeno andranno a finire

nell'indice generale, di qualunque livello essi siano. Il titolo precedente, per esempio, si è ottenuto con il comando

```
\subsection*{Titoli non numerati né indicizzati}
```

Per mandare nell'indice anche i titoli che normalmente non ci finirebbero, *subito dopo* il relativo comando di sezionamento basta dare

```
\addcontentsline{<indice>}{<livello>}{<titolo>}
```

dove:

- *<indice>* è il tipo di indice in cui far comparire la voce in questione (normalmente si sceglie *toc*, *lot* o *lof* per l'indice generale, delle tabelle e delle figure rispettivamente);
- *<livello>* è il nome del livello di sezionamento in questione (si noti che *paragraph* e *subparagraph* non sono ammessi);
- *<titolo>* è il titolo di sezione che finirà nell'indice.

Applicato al titolo di questa sezione, il codice sarebbe:

```
\subsection*{Titoli non numerati né indicizzati}
%\phantomsection
\addcontentsline{toc}{subsection}{Titoli non numerati né indicizzati}
```

Si noti che se *hyperref* è caricato, *subito prima* di `\addcontentsline` va dato anche `\phantomsection` per evitare possibili errori nei collegamenti ipertestuali e nei segnalibri del documento finito (in tal caso, si decommenti la riga corrispondente).

Titoli alternativi nell'indice generale

Nell'indice generale finiscono i titoli scritti nell'argomento dei comandi di sezionamento. Se, però, un titolo è troppo lungo per starci agevolmente (si noti che un titolo non dovrebbe *mai* andare a capo) o si hanno particolari esigenze, lo si può sostituire con un titolo alternativo più breve, da inserire nell'argomento facoltativo degli stessi comandi:

```
\chapter[Leggilo! È emozionante!]{Questo è un titolo lunghissimo
e particolarmente noioso}
```

Si noti che il titolo breve comparirà anche nelle testatine, se previste dalla classe di documento in uso e che, ovviamente, non si può usare se il comando è asteriscato.

3.9.3 Regolare la profondità dell'indice generale

La numerazione delle sezioni e ciò che compare nell'indice generale sono regolati dai valori numerici dei due contatori *secnumdepth* e *tocdepth* rispettivamente, entrambi impostati a 2 di default: le sezioni saranno numerate se di livello minore o uguale al valore di *secnumdepth*, mentre i loro titoli andranno nell'indice se di livello minore o uguale al valore di *tocdepth* (si veda la tabella 10 nella pagina seguente).

Tabella 10: Corrispondenza fra livelli e sezioni

Livello	Sezione
- 1	<code>\part</code>
0	<code>\chapter</code>
1	<code>\section</code>
2	<code>\subsection</code>
3	<code>\subsubsection</code>
4	<code>\paragraph</code>
5	<code>\subparagraph</code>

Per esigenze particolari (come le richieste di relatore o editore, per esempio) si potrebbe dover modificare questo comportamento predefinito. L'indipendenza dei due contatori permette di farlo facilmente, ottenendo praticamente qualsiasi risultato: si potranno avere documenti con numerazioni dettagliate e indici più snelli e viceversa.

Per esempio, scrivendo *nel preambolo*

```
\setcounter{secnumdepth}{3}
\setcounter{tocdepth}{1}
```

nel documento saranno numerate le sezioni fino al terzo livello, ma nell'indice generale compariranno solo quelle fino al primo. Come sempre, però, si raccomanda di non abusare di questa possibilità e di limitarla ai casi davvero necessari. In linea generale si sconsiglia vivamente di impostare *secnumdepth* a più di 3 e *tocnumdepth* a più di 2 (l'indice assomiglierebbe più a un elenco telefonico che a un indice vero e proprio).

3.9.4 Breve indice

In un documento di grandi dimensioni può essere utile accompagnare l'indice generale (*table of contents*) con uno più breve (*short table of contents*), che fornisce una visione d'insieme degli argomenti trattati. Il pacchetto `shorttoc` permette di farlo.

Il pacchetto definisce il comando `\shorttoc`, che si usa come segue:

```
\shorttoc{<titolo>}{<profondità>}
```

Dove:

- `<titolo>` è il titolo dell'indice abbreviato;
- `<profondità>` è il livello delle unità di sezionamento i cui titoli verranno inseriti nell'indice abbreviato (ha lo stesso significato del contatore di `LaTeX` `tocdepth`): impostandola a 0 verranno inclusi solo i titoli dei capitoli (e delle parti, se ve ne sono), mentre impostandola a 1 verranno inclusi solo i titoli dei capitoli e dei paragrafi (e delle parti).

Per esempio, scrivendo

```
\shorttoc{Argomenti}{0}
```

verrà creato un breve indice intitolato "Argomenti", che riporta solo i titoli dei capitoli (e delle parti).

Tabella 11: Elementi considerati in questa guida che ammettono un riferimento incrociato

<code>\part</code>	<code>\paragraph*</code>	<code>table</code>	<code>align**</code>	n. di pagina
<code>\chapter</code>	<code>\subparagraph*</code>	<code>equation</code>	<code>enumerate†</code>	
<code>\section</code>	<code>\footnote</code>	<code>multline**</code>	<code>sidecap†</code>	
<code>\subsection</code>	<code>\newtheorem</code>	<code>split**</code>	<code>subfig†</code>	
<code>\subsubsection</code>	<code>figure</code>	<code>gather**</code>	<code>wrapfloat†</code>	

* Solo se numerato.

** Richiede il pacchetto `amsmath`.

‡ Si richiama un elemento della lista.

† Rappresenta l'ambiente o gli ambienti definiti.

3.10 RIFERIMENTI INCROCIATI

Nei documenti si trovano spesso riferimenti incrociati a sezioni, figure, tabelle, teoremi e altri elementi. Per realizzarli si usano i comandi standard

```
\label{<etichetta>}
\ref{<etichetta>}
\pageref{<etichetta>}
```

dove:

- `\label` assegna agli elementi mostrati nella tabella 11 un'etichetta arbitraria e *univoca*, che si consiglia di scrivere sempre nella forma *<abbreviazione>: <parola chiave>* (dove la prima è un'abbreviazione dell'elemento in questione, come `tab` per una tabella, e la seconda una stringa identificativa) e avendo cura di evitare i caratteri accentati (scrivendo in francese, inoltre, si sconsiglia di mettere i due punti nelle etichette);
- `\ref` produce il numero dell'elemento messo in *<etichetta>*;
- `\pageref` produce il numero di pagina in cui l'elemento compare.

Per produrre i riferimenti incrociati nel documento sono necessarie *due* composizioni successive, altrimenti al loro posto si vedranno altrettanti `??`.

Per esempio, se s'identifica questo paragrafo con

```
\section{Riferimenti incrociati}
\label{sec:rif-inc}
```

poi ci si può riferire a esso con

```
Ecco un riferimento a questo
paragrafo: "si veda il
paragrafo~\ref{sec:rif-inc}".
```

Ecco un riferimento a questo paragrafo: "si veda il paragrafo 3.10".

Si noti che è una buona abitudine unire il riferimento alla parola precedente con uno *spazio indivisibile*: garantisce in genere risultati tipografici ottimali.

Molto spesso, specialmente quando il riferimento si trova a una o più pagine di distanza dall'oggetto, è utile avere un'indicazione completa *anche* del numero di pagina. Lo si può ottenere con il comando `\vref` del pacchetto `varioref` dopo aver messo la lingua principale del documento come opzione al comando di chiamata:

```
\usepackage[italian]{varioref}
```

A questo punto `\vref` si usa come di consueto:

Si veda il
paragrafo~\vref{sec:pacchetti}.

Si veda il paragrafo 3.12 nella
pagina successiva.

Se il documento contiene molti riferimenti incrociati potrebbe essere utile controllare la correttezza delle etichette o averle sempre sotto controllo: il pacchetto `showkeys` le visualizza nel margine della pagina.

3.11 COLLEGAMENTI IPERTESTUALI E AL WEB

Collegamenti ipertestuali e al Web

Il pacchetto `hyperref`, che di regola va caricato *per ultimo*, crea i collegamenti ipertestuali all'interno del documento, rendendo cliccabili i riferimenti incrociati visti nella sezione precedente, quelli a voci bibliografiche, a indirizzi Internet e molto altro. Se ne possono specificare le opzioni come di consueto oppure, se numerose, si può usare il comando `\hypersetup`:

```
\usepackage{hyperref}
\hypersetup{<chiave>=<valore>,<...>}
```

Il pacchetto permette di gestire collegamenti e segnalibri in modo molto fine: se ne veda la documentazione.

Per impostazione predefinita, `hyperref` circonda il collegamento con un riquadro colorato che *non* viene stampato. Si può avere il testo del collegamento colorato (con colori predefiniti o a piacere) scrivendo:

```
\usepackage[colorlinks]{hyperref}
```

Questo è utile per un documento da leggere a schermo o da stampare a colori, ma si ricordi che la stampa in bianco e nero restituisce i colori come sfumature di grigio, a volte poco leggibili.

Si possono avere tutti i collegamenti in nero e senza riquadri scrivendo semplicemente

```
\hypersetup{hidelinks}
```


Per gestire ancora più finemente i segnalibri, che con l'opzione `bookmarks` `hyperref` gestisce automaticamente, compresi quelli delle sezioni aggiunte all'indice generale con `\addcontentsline`, si segnala il pacchetto `bookmarks`, da caricare *dopo* `hyperref` (se ne veda la documentazione).

Oltre ai collegamenti ipertestuali per i riferimenti incrociati, `hyperref` permette anche di realizzare collegamenti al Web con il comando `\href`:

```
\href{<indirizzo Internet>}{<testo del collegamento>}
```

Scrivendo

Visita il sito del
`\href{http://www.guitex.org/}%
{\GuIT*}`.

Visita il sito del .

basta cliccare sul logo  per accedere al sito omonimo.

Indirizzi Internet e di posta elettronica

Il pacchetto `url` (caricato automaticamente da `hyperref`) definisce il comando `\url`, utile per scrivere un indirizzo Internet:

<code>\url{http://www.guitex.org/}</code>	http://www.guitex.org/
---	---

Per i collegamenti a un indirizzo di posta elettronica conviene definire nel preambolo un apposito comando `\mail` (si veda il paragrafo 13.1 a pagina 245),

```
\newcommand{\mail}[1]{\href{mailto:#1}{\texttt{#1}}}
```

da usare come segue:

<code>\mail{lorenzo.pantieri@gmail.com}</code>	lorenzo.pantieri@gmail.com
--	--

3.12 PACCHETTI

Scrivendo un documento, prima o poi ci s’imbatte in problemi che \LaTeX non riesce a risolvere da solo. Il suo linguaggio standard, per esempio, non gestisce l’inclusione delle immagini, né sillaba documenti in lingue diverse dall’inglese, né ancora permette di modificare facilmente i margini di pagina. Per aggirare “ostacoli” di questo tipo si sfrutta la struttura modulare del programma, che estende le proprie capacità di base tramite moduli aggiuntivi chiamati *pacchetti*.

3.12.1 Caratteristiche

Che cosa sono?

Fondamentalmente, un pacchetto è un file “di stile” (con estensione `.sty`) scritto in linguaggio \LaTeX , contenente istruzioni che permettono di svolgere alcune operazioni.

Come sapere se servono?

In genere, se per ottenere il risultato sperato si deve faticare troppo, probabilmente qualcuno che si è già trovato nella stessa situazione ha provveduto a creare un pacchetto per semplificare il lavoro.

\TeX Live non comprende *tutti* i pacchetti presenti su CTAN. Infatti, componendo un sorgente può capitare che \LaTeX produca un messaggio di errore del tipo

```
Can't find file steroid.sty
```

Ciò significa che è stato caricato un pacchetto (`steroid`, nell’esempio considerato) non presente nella distribuzione. Si risolve il problema seguendo le istruzioni contenute nel paragrafo 2.1.2 a pagina 15.

Viceversa, può accadere di usare un comando definito da un pacchetto che ci si è dimenticati di caricare: si otterrà un messaggio di errore di comando sconosciuto, che purtroppo non aiuta a indovinare il pacchetto che serve.

E se serve un pacchetto che non c'è in T_EX Live?

Per usare un pacchetto che *non può* esserci nella distribuzione (perché non ne è prevista l'inclusione in T_EX Live, o è un pacchetto personale o coperto da una licenza particolare, o è una versione sperimentale che si vuole comunque provare, o ancora perché l'aggiornamento o la pubblicazione cadono nel periodo di "congelamento") la strada più semplice è copiarne i file nella cartella di lavoro. (Ogni anno, qualche mese prima del rilascio della nuova versione gli aggiornamenti vengono "congelati". Gli archivi in Rete, tuttavia, continuano a funzionare come al solito.)

Come scovare il pacchetto che fa al proprio caso?

Questo è l'unico aspetto del lavoro con L^AT_EX in cui gusto, abilità e fortuna la fanno da padroni: cercando su T_EX Catalogue ([www.TEXCATALOGUE](http://www.texcatalogue.org)) si trovano preziosi riferimenti e soluzioni per risolvere moltissimi problemi.

3.12.2 Caricamento e precauzioni

Come caricarli?

I pacchetti si caricano *nel preambolo* con il comando

```
\usepackage[⟨opzioni⟩]{⟨pacchetto⟩}
```

dove:

- *⟨opzioni⟩* è una voce o un elenco di voci separate con la virgola costituite da un solo elemento o un'espressione del tipo *⟨chiave⟩=⟨valore⟩* che specificano le impostazioni del pacchetto;
- *⟨pacchetto⟩* è il nome del pacchetto, che va scritto *sempre* in tutte minuscole (LayAureo si scrive così, ma si carica come *layaureo*, per intenderci).

Si noti che con uno stesso comando `\usepackage` si possono caricare più pacchetti *senza opzioni*, separandone i nomi con la virgola.

Quali precauzioni prendere?

Non si possono caricare i pacchetti in un ordine casuale, anche se ciò è permesso entro certi limiti. Sequenze di caricamento ben precise, come si è già visto per `fontenc`, `inputenc` e `babel` sono richieste anche per altri pacchetti, ma non è questa la sede per elencarle tutte (né lo si potrebbe fare per evidenti limiti di spazio). I messaggi d'errore notificati dal programma in tal senso sono chiari, di solito, ma le precauzioni da prendere non sono mai troppe. Di seguito si danno alcuni consigli che dovrebbero limitare i problemi.

- L^AT_EX richiede di caricare (direttamente o indirettamente) i pacchetti *solo nel preambolo* e *una volta sola*, con *tutte* le opzioni che servono.
- Molti pacchetti ne caricano automaticamente degli altri: lo si può scoprire leggendo la documentazione. Non sapendolo e ricaricando un pacchetto, si ottiene un errore.

- Talvolta il caricamento dei pacchetti sottostà a vincoli precisi: alcuni vanno caricati prima di altri e viceversa, pena un errore. La documentazione del pacchetto indicato nell'errore potrebbe contenere informazioni utili: di solito basta modificare l'ordine di caricamento o eliminare la doppia chiamata.


3.12.3 Usarli al meglio: la documentazione

Chi scrive o aggiorna un pacchetto per \LaTeX ne scrive anche, e quasi sempre in inglese, la *documentazione*, che spesso si compone di due parti distinte:

- il manuale d'uso, che dichiara lo scopo del pacchetto e ne descrive i comandi;
- il codice che costituisce il pacchetto, destinato a chi voglia eventualmente svilupparlo (nel caso di pacchetti molto corposi, il codice costituisce un file a sé).

I pacchetti contenuti in ogni distribuzione di \LaTeX sono già corredati della relativa documentazione (quasi sempre un PDF omonimo), facilmente raggiungibile con il programma `texdoc`, integrato in \TeX Live. Il programma si lancia dalla riga di comando o con le scorciatoie che di solito ogni editor definisce a questo scopo e prevede numerose opzioni di ricerca: eseguendo

```
texdoc <nome del pacchetto>
```

da una posizione qualunque sul proprio computer, in un attimo si apre il relativo manuale. Si segnala anche la versione online di `texdoc` ( **TEXDOC**), nella quale i pacchetti sono ordinati in categorie per una più agevole ricerca delle informazioni.

3.12.4 Altri file

In alcuni casi, i pacchetti che si scaricano da Internet (in forma di archivi compressi) non contengono file `.sty` e PDF di documentazione, ma un file con estensione `.ins` e uno con estensione `.dtx`. Basta procedere così:

- Il file `.dtx` contiene la documentazione del pacchetto. Eseguendo \LaTeX su di esso, si ottiene il manuale.
- Eseguendo \LaTeX sul file `.ins`, invece, si ottiene il file `.sty` (o anche più d'uno, a seconda dei casi).

Si noti che alcuni pacchetti contengono soltanto il file `.dtx`: in questo caso con la prima istruzione si ottiene *anche* il file `.sty`.

A questo punto, per poterli usare basta sistemare i file `.sty` ottenuti come descritto nel paragrafo [3.12.1](#) a pagina [45](#).

3.13 UNITÀ DI MISURA TIPOGRAFICHE

Nei prossimi capitoli spesso si useranno istruzioni che richiedono di esprimere una lunghezza in una qualche *unità di misura tipografica*. Poiché alcune

Tabella 12: Principali unità di misura tipografiche riconosciute da \LaTeX (*m-width* generalmente approssima la larghezza della *M* nel font in uso)

Unità	Codice	Valore
centimetro	cm	
millimetro	mm	
punto	pt	0,3514 mm
<i>x-height</i>	ex	Circa uguale all'altezza della <i>x</i> nel font in uso
<i>m-width</i>	em	Circa uguale al corpo del font in uso

di esse poco hanno a che fare con quelle più conosciute del sistema metrico decimale, nella tabella 12 si mostrano quelle effettivamente usate in questa guida.

Esistono inoltre i comandi di spaziatura `\quad` (o *quadrato*) e `\qquad` (o *quadrato*), che producono rispettivamente uno spazio di 1 e 2 em. Si noti che in un ambiente puramente testuale il loro uso è di regola fortemente sconsigliato e va limitato a casi particolarissimi (per spaziare sottooggetti mobili, per esempio, come si spiega nel paragrafo 8.5 a pagina 144).

3.14 DOCUMENTI DI GRANDI DIMENSIONI

Per scrivere senza sorprese un documento di grandi dimensioni come un libro o una tesi è importantissimo organizzarne razionalmente il materiale. Prendendo a esempio questa guida (ma i suggerimenti valgono anche per una tesi di laurea o un altro documento), si sono messi *tutti* i suoi file in una cartella *artelatex*, strutturata in sottocartelle come segue:

- La sottocartella *inizio*, con il materiale iniziale suddiviso nei corrispondenti file come *ringraziamenti.tex*, *introduzione.tex*, eccetera.
- La sottocartella *capitoli*, con il materiale principale suddiviso nei corrispondenti file come *basi.tex*, *testo.tex*, *tabellefigure.tex*, eccetera.
- La sottocartella *fine*, con il materiale finale suddiviso nei corrispondenti file come *acronimi.tex*, *sitiinternet.tex*, eccetera.
- La cartella *immagini*, con tutte le immagini incluse nella guida. Se sono molte, le si potrebbe distribuire in ulteriori sottocartelle corrispondenti ai diversi capitoli. Immaginando di chiamarle *grafici* e *foto*, basta scrivere nel file di impostazioni (si veda poco più sotto)

```
\graphicspath{{grafici/},{foto/}}
```

La cartella *artelatex* *deve* contenere anche altri due file:

- il file *principale* del documento, *artelatex.tex*, cioè quello che contiene dichiarazione di classe, preambolo, impostazioni generali e ambiente document;
- il file *arte-bibliografia.bib* (a meno di non volerlo sistemare come spiegato nel paragrafo 11.2 a pagina 219), cioè il database bibliografico del documento.

Se il documento non è troppo corposo, invece, se ne possono mettere tutti i file in una sola cartella e l'indicazione del percorso non serve più.

Infine, si sono raccolte definizioni di comandi e ambienti personali e impostazioni generali del documento in un pacchetto `impostazioni-arte.sty` (si scrive con l'editor in uso, si registra con estensione `.sty`, *non* richiede il preambolo e *non* va composto), caricato nel preambolo come un normale pacchetto immediatamente prima dell'inizio del documento. Questi piccoli accorgimenti “puliscono” il file principale semplificando notevolmente il proprio lavoro.

Nell'ambiente `document` vanno caricati i file `.tex` in cui si è suddiviso il documento con uno dei due metodi spiegati di seguito.

3.14.1 Documenti normali

Il primo metodo prevede di scrivere il nome del file *senza l'estensione* nell'argomento del comando `\input`, indicandone l'eventuale percorso, come nell'esempio seguente:

```
\begin{document}

...
\input{inizio/ringraziamenti}
\input{inizio/introduzione}
...
\input{capitoli/basi}
\input{capitoli/installare}
\input{capitoli/testo}
...
\input{fine/acronimi}
...

\end{document}
```

Questo accorgimento evidenzia molto chiaramente la struttura del documento e snellisce il file principale. In pratica, `\input` costruisce il documento “attaccando” semplicemente uno dopo l'altro i vari file. Si noti che:

- Questi ultimi *non* devono contenere alcun preambolo, ma solo comando di sezionamento e contenuto della sezione.
- `\input` permette l'annidamento (cioè ammette altri `\input` nel proprio argomento).
- Si abbia cura che l'eventuale percorso dei file inclusi (anche quelli con `\graphicspath`) sia *relativo* (un percorso assoluto verrebbe addirittura *rifiutato* da \LaTeX per motivi di sicurezza) e *non* contenga spazi.
- Per evitare problemi che poi sarebbe difficile risolvere, si raccomanda di nominare i file con *una sola* parola alfanumerica senza maiuscole, punti, spazi intermedi e caratteri particolari. Se fosse davvero necessario separare i due membri del nome del file, a parte `prima.tex` o `parte.prima.tex`, per esempio, si preferisca `parte-prima.tex`.

Il metodo appena spiegato è particolarmente utile per includere nel documento elementi come tabelle o grafici particolarmente complessi e possibile

fonte di errori difficilmente individuabili se composti direttamente nel sorgente. Lo si può usare con profitto anche per comporre una sezione del documento alla volta (commentando quelle che non servono) con notevole risparmio di tempo. In quest'ultimo caso, però, si noti che i contatori di pagina e di sezione vengono aggiornati a ogni composizione, e che gli eventuali riferimenti incrociati vanno perduti (in pratica, qualunque sezione si componga comincerà sempre a pagina 1 e i riferimenti incrociati alle altre verranno visualizzati con ??).

3.14.2 Documenti enormi

Risolve il problema la coppia di comandi `\include` e `\includeonly`, da usare così:

```
% nel preambolo
\includeonly{introduzione,%
             basi,%
             acronimi%
            }

\begin{document}

...
\include{ringraziamenti}
\include{introduzione}
\include{basi}
...
\include{acronimi}
...

\end{document}
```

A questo punto basterà comporre prima l'intero documento e poi solo il o i file che interessano (commentando quelli che non servono nell'argomento di `\includeonly`): numerazione delle pagine e riferimenti incrociati saranno a posto. I file da includere con `\include` devono avere le stesse caratteristiche di quelli appena visti per `\input`.

Si noti che `\include`:

- ordina a \LaTeX di cominciare a comporre il contenuto del file incluso sempre su una pagina nuova: perciò è più indicato per comporre interi capitoli e non è completamente intercambiabile con `\input`;
- non permette l'annidamento, ma nulla vieta di usare `\input` nel file presente nel suo argomento;
- prima di cominciare a comporre il nuovo file, ordina a \LaTeX di svuotare le eventuali code di oggetti mobili ancora in memoria.

Questo secondo metodo è utile per comporre le bozze di singoli capitoli di un lavoro particolarmente corposo (un manuale molto corposo o un'enciclopedia, per esempio).

Parte II

TESTO

4 | TESTO

4.1 STRUTTURA DEL TESTO

Lo scopo principale di chi scrive un testo è comunicare idee e conoscenze al lettore, che le comprenderà tanto più quanto meglio sono strutturate, e ne apprezzerà tanto più la struttura quanto più la forma tipografica del documento rispecchia la costruzione logica del suo contenuto.

Le suddivisioni elencate nella tabella 13 sono fondamentali per comprendere l'articolazione di un testo scritto, e vengono chiamate in generale *sezioni*.

A questo proposito si noti che:

- il sezionamento del documento è compito dell'utente, perché \LaTeX non lo fa automaticamente;
- è importante scandire il testo in capoversi, per chi scrive e per chi legge: le informazioni sono meglio articolate e più facilmente memorizzabili.

4.2 COMPORRE I CAPOVERSI

Spesso si sottovaluta l'importanza di scrivere un testo ben strutturato, e usando \LaTeX altrettanto frequentemente si comincia un nuovo capoverso senza nemmeno rendersene conto.

È facile commettere quest'ultimo errore se il testo contiene formule matematiche. Infatti l'abitudine, diffusa, di lasciare una riga vuota tra la fine di una formula e la prosecuzione del testo si sconta nel documento finito con altrettanti nuovi capoversi, anche laddove il flusso del discorso non li richiederebbe.

Cominciare un nuovo capoverso

Con l'ovvia eccezione del primo capoverso di una sezione, per cominciare un nuovo capoverso con \LaTeX si hanno due possibilità:

- si lascia una riga vuota nel sorgente (di solito si fa così);

Tabella 13: Lunghezza orientativa delle sezioni di un testo scritto

Sezione	Lunghezza orientativa
Parte	Imprecisabile
Capitolo	Da una decina a un centinaio di pagine
Paragrafo	Da mezza a una decina di pagine
Sottoparagrafo	Da poche righe a un paio di pagine
Capoverso	Da una a una ventina di righe
Enunciato	Da una parola a una decina di righe

- si dà il comando `\par`.

In ogni caso, non lo si faccia *mai* con `\\` (qualche esempio di questa guida lo ha richiesto per motivi di spazio).

Osservando gli esempi che seguono, si cerchi di capire perché a volte c'è la riga bianca e altre no. (Se non si comprendono ancora tutti i comandi, si leggano interamente questo capitolo e i primi paragrafi del capitolo 6 a pagina 77, e poi si ritorni su questo punto.)

```
\dots quando Einstein propose
l'equazione
\begin{equation}
E = mc^2
\end{equation}
che è la più nota e la meno
compresa formula della Fisica.
```

...quando Einstein propose l'equazione

$$E = mc^2 \quad (4.1)$$

che è la più nota e la meno compresa formula della Fisica.

```
\dots che, rispetto ai
precedenti, ha alcuni vantaggi.
```

...che, rispetto ai precedenti, ha alcuni vantaggi.

```
La formula
\begin{equation}
D=F-R
\end{equation}
definisce un modello molto
diverso di transistor.
```

La formula

$$D = F - R \quad (4.2)$$

definisce un modello molto diverso di transistor.

```
\dots da cui segue la legge di
Kirchhoff sulle correnti:
\begin{equation}
\sum_{k=1}^n I_k = 0
\end{equation}
```

...da cui segue la legge di Kirchhoff sulle correnti:

$$\sum_{k=1}^n I_k = 0 \quad (4.3)$$

```
La legge di Kirchhoff sulle
tensioni può essere ricavata\dots
```

La legge di Kirchhoff sulle tensioni può essere ricavata...

Terminare un capoverso con una formula in display (si veda il paragrafo 6.1 a pagina 77) come nel terzo degli esempi proposti è raro, ma comunque lecito. È invece *sempre* sconsigliabile cominciarlo con una formula matematica di qualunque tipo.

Capoversi ben composti e microtipografia

Un documento “ben composto” si riconosce da alcuni elementi: il testo è giustificato, le righe sono interamente riempite, le parole sono adeguatamente spaziate tra loro e sillabate a fine riga se proprio non ci stanno, i capoversi presentano la prima riga rientrata per facilitare la lettura. Di solito tutto questo si ottiene dando a mano le rispettive impostazioni; con \LaTeX , invece, non occorre nemmeno pensarci, perché il programma:

- giustifica il testo per impostazione predefinita;
- riempie la riga con un sofisticatissimo algoritmo di spaziatura tra le parole, sillabandole solo se *davvero* necessario;

- rientra automaticamente la prima riga di ogni capoverso tranne il primo (se per qualche motivo non si volesse il rientro, basta cominciare la riga interessata con `\noindent`);
- numera automaticamente le pagine del documento;
- non aggiunge spazio supplementare tra un capoverso e l'altro tranne quando non ha abbastanza materiale per riempire perfettamente la pagina.

Talvolta, invece, questo spazio supplementare potrebbe servire. Lo si può inserire con i seguenti comandi:

- `\bigskip`, `\medskip` e `\smallskip`, avendo cura di lasciare una riga bianca *prima*, inseriscono uno spazio verticale rispettivamente “grande”, “medio” e “piccolo” la cui ampiezza è in funzione del font utilizzato.
- `\vspace{⟨lunghezza⟩}` inserisce uno spazio verticale pari a `⟨lunghezza⟩` (che va perso se dopo la composizione viene a trovarsi all'inizio di una pagina: per mantenerlo basta usare la forma `\vspace*.`)

La tipografia anglosassone (predefinita in \LaTeX) non prevede il rientro della prima riga del primo capoverso di una sezione. Per ottenerlo, secondo una consuetudine spesso seguita in Italia, basta semplicemente caricare il pacchetto `indentfirst` nel modo consueto.

Il pacchetto `microtype` migliora il riempimento delle righe abilitando:

- *l'espansione dei font*, ovvero espande i caratteri per riempire la riga in modo ancora migliore;
- *la protrusione dei caratteri*, ovvero permette a certi caratteri di sporgere leggerissimamente a fine riga (di solito segni di punteggiatura e trattini).

Perciò si consiglia di caricarlo *sempre*, anche se la prima delle due funzionalità appena descritte ancora non funziona con \XeLaTeX (si veda il paragrafo 18.2.2 a pagina 297).

Interrompere una riga senza cominciare un nuovo capoverso

In casi particolari può essere necessario interrompere una riga. Per farlo si usano i comandi `\` o `\newline`, e se ne incomincia una nuova *senza iniziare un nuovo capoverso* (e senza rientro, dunque, come qui).

Si può inserire uno spazio aggiuntivo tra due linee dello stesso capoverso con il comando `\[⟨lunghezza⟩]`, in cui `⟨lunghezza⟩` può essere espressa in una qualunque delle unità di misura tipografiche accettate da \LaTeX , avendo cura di usare *il punto* come separatore decimale (si veda la tabella 12 a pagina 48).

Dividere le parole a fine riga

In generale, \LaTeX cerca di interrompere le righe sempre nel miglior punto possibile. Se, però, non riesce a farlo neppure secondo i propri severi

criteri, le lascia fuoriuscire dal margine destro e avverte l'utente con un messaggio di *Overfull hbox*. Non sempre è facile individuarle: nel capitolo 14 a pagina 257 si spiega come fare.

L'algoritmo di sillabazione di L^AT_EX funziona correttamente con quasi tutte le parole, ma in particolari circostanze si potrebbe volere una divisione diversa da quella automatica. Con nomi propri o tecnicismi come *nitro-idrossilamminico* o *macroistruzione*, per esempio, a volte si richiede la sillabazione etimologica anziché quella che L^AT_EX esegue di default: *nitro-idrossil-amminico* invece di *ni-troi-dros-si-lam-mi-ni-co* e *ma-cro-i-stru-zio-ne* invece di *ma-croi-stru-zio-ne*.

In questi casi basta scrivere le parole nell'argomento di `\hyphenation` (nel preambolo) già *sillabate*, separandole con uno spazio ed evitando caratteri speciali e simboli:

```
\hyphenation{nitro-idrossil-amminico ma-cro-istru-zio-ne}
```

Il comando precedente funziona anche al contrario. Una scrittura come la seguente

```
\hyphenation{nitro-idrossil-amminico FORTRAN}
```

sillaba *nitroidrossilamminico* e *Nitroidrossilamminico* come suggerito nell'argomento, ma non *FORTRAN*, *Fortran* e *fortran*. Si può usare `\hyphenation` per forzare qualunque cesura si desideri: se si vuole spezzare la parola *melograno* soltanto tra *melo* e *grano*, si scrive:

```
\hyphenation{melo-grano}
```

Se la parola in questione compare nel documento una sola volta, si può suggerirne la sillabazione direttamente nel testo. Il comando `\-` spezza la parola nel punto (o nei punti) in cui viene dato, e *in quel punto soltanto*.

La scoperta dell'acido
nitro\-idrossil\-amminico è
avvenuta nel lontano 1896.

La scoperta dell'acido nitro-
idrossilamminico è avvenuta nel
lontano 1896.

Si noti che anche gli interventi sulla sillabazione, come tutti quelli operati "a mano" sul documento, dovrebbero essere effettuati durante la revisione finale immediatamente precedente la stampa. La prima cura per un *Overfull hbox*, per esempio, dovrebbe consistere *sempre* nel riformulare l'enunciato piuttosto che nell'imporre una sillabazione particolare.

L'opzione `italian` di `babel` mette a disposizione il comando `"/`, utile per andare a capo dopo la barra nelle espressioni che comportano l'alternanza tra due possibilità. Per esempio, la scrittura `modulazione"/demodulazione` produce *modulazione/demodulazione* se si trova all'interno di una riga, *modulazione/demodulazione* se invece si trova alla fine.

Il comando

```
\mbox{\langle testo \rangle}
```

serve per mantenere unita una parola *senza* usare `\hyphenation`. Va usato all'occorrenza, magari perché in un certo punto del documento non va bene che la parola sia spezzata, ma altrove sì:

Entro l'anno avrò imparato il
Fortran. \[lex]
Entro l'anno avrò imparato il
\mbox{Fortran}.

Entro l'anno avrò imparato il For-
tran.
Entro l'anno avrò imparato il
Fortran.

Spazi interparola e punti fermi

Per giustificare i capoversi \LaTeX inserisce spazi interparola variabili e migliora la leggibilità del testo separando gli enunciati con uno spazio leggermente più ampio di quello inserito da un comune elaboratore di testo. Il programma interpreta diversamente il punto:

- un punto (fermo, interrogativo o esclamativo) dopo una *minuscola* indica la fine di un enunciato, e dopo di esso \LaTeX inserisce uno spazio supplementare;
- un punto dopo una *maiuscola* indica la fine di un'abbreviazione, e dopo di esso ci sarà uno spazio normale.

Le eccezioni alle regole generali appena esposte vanno specificate esplicitamente. I casi sono tre:

- immediatamente *dopo* un punto di fine abbreviazione dentro un enunciato (tranne se l'abbreviazione ne è l'ultima parola), si usa `_`;
- immediatamente *prima* di un punto di fine enunciato che segue una maiuscola (che per \LaTeX indica *comunque* un'abbreviazione) si usa `\@`;
- per tenere unite espressioni che non si vogliono o non possono *mai* essere spezzate da un fine riga si usa lo *spazio indivisibile* prodotto dalla tilde `~`.

L'esempio seguente mostra `\@` all'opera:

```
CEE. Poi CE. Ora UE. \\  
CEE\@. Poi CE\@. Ora UE\@.
```

```
CEE. Poi CE. Ora UE.  
CEE. Poi CE. Ora UE.
```

La spaziatura corretta è quella prodotta dalla seconda scrittura.

Si osservi come agisce la tilde nei due esempi seguenti:

```
Ho studiato il manuale del  
prof. Beccari. \\\[lex]  
Ho studiato il manuale del  
prof.~Beccari.
```

```
Ho studiato il manuale del prof.  
Beccari.
```

```
Ho studiato il manuale del  
prof. Beccari.
```

```
Il tutto è spiegato nel  
paragrafo \ref{sec:par}. \\\[lex]  
Il tutto è spiegato nel  
paragrafo~\ref{sec:par}.
```

```
Il tutto è spiegato nel paragrafo  
4.2.
```

```
Il tutto è spiegato nel paragra-  
fo 4.2.
```

Come si può notare, la seconda scrittura di ciascuna coppia, che è quella corretta, evita che le righe finiscano o comincino nel modo sbagliato.

Per disabilitare lo spazio supplementare dopo un punto *in tutto* il documento (come nella tipografia francese) anche se si è usato `\@`, basta dare nel preambolo la dichiarazione `\frenchspacing`.

Tabella 14: Virgolette, tratti e puntini di sospensione. Per evidenziare le differenze tra i vari segni, virgolette alte e apici sono composti con il font Computer Modern.

	Segno	Codice	Risultato
Virgolette	semplici alte	‘ ’	‘ ’
	doppie alte	“ ”	“ ”
		« »	« »
	doppie basse	<< >>	<< >>
		« »	« »
Tratti	trattino	-	-
	tratto	--	--
	lineetta	---	---
	meno	\$-\$	—
Puntini		\dots	...

4.3 CARATTERI PARTICOLARI E SIMBOLI

4.3.1 Virgolette, tratti e puntini di sospensione

Virgolette

In tipografia si usano comunemente tre tipi di virgolette: gli ‘apici’, le “virgolette inglesi” e le «virgolette caporali». I modi per ottenerle dipendono dalla codifica di input impostata, dall’editor in uso e dalle sequenze di tasti digitate: la tabella 14 raccoglie quelli più usati (scopra l’utente le poche alternative).

Gli esempi seguenti le mostrano all’opera:

Ora è chiaro il concetto di ‘composizione asincrona’.	Ora è chiaro il concetto di ‘compo- sizione asincrona’.
La Delta di Dirac è una “funzione impropria”.	La Delta di Dirac è una “funzione impropria”.
<<Se stai attento, Ermanno, capisci tutto anche tu.>>	«Se stai attento, Ermanno, capisci tutto anche tu.»

Si noti che nel font principale di questa guida virgolette inglesi e apici aperti e chiusi sono molto simili (ciò non accade con altri font).

Trattini, tratti e lineette

La tipografia distingue quattro tipi di tratto: tre (*trattino*, *tratto* e *lineetta*) corrispondono a un numero crescente di trattini consecutivi, mentre il quarto è il segno matematico *meno*. La tabella 14 mostra come ottenerli, e gli esempi seguenti ne illustrano alcuni possibili usi:

Stratford-on-Avon, e-mail \\ p.~13-67, 1921-28 \\ Ottica----Schema generale \\ ----Eccomi---- disse. \\ \$0\$, \$1\$ e \$-1\$	Stratford-on-Avon, e-mail p. 13-67, 1921-28 Ottica – Schema generale — Eccomi — disse. 0, 1 e –1
---	--

Tabella 15: Loghi frequenti nel mondo \LaTeX

Comando	Risultato	Pacchetto richiesto
<code>\TeX</code>	\TeX	
<code>\LaTeX</code>	\LaTeX	
<code>\LaTeXe</code>	$\text{\LaTeX}_{2\epsilon}$	
<code>\AmS</code>	$\mathcal{A}\mathcal{M}\mathcal{S}$	amsmath
<code>\MF</code>	METAFONT	mflogo
<code>\MP</code>	METAPOST	mflogo
<code>\GuIT, \GuIT*</code>	$\mathcal{G}\mathcal{U}\mathcal{I}\mathcal{T}$	guit
<code>\Ars</code>	$\mathcal{A}\mathcal{r}\mathcal{s}\mathcal{T}\mathcal{E}\mathcal{X}\mathcal{n}\mathcal{i}\mathcal{c}\mathcal{a}$	guit
<code>\BibTeX</code>	\BibTeX	dtklogos
<code>\MiKTeX</code>	\MiKTeX	dtklogos

Puntini di sospensione e segno di omissione

Se inseriti battendo tre punti consecutivi, i *puntini di sospensione* potrebbero compromettere la spaziatura tra le parole o la corretta interruzione di riga. \LaTeX risolve il problema definendo il comando `\dots`, che li produce correttamente spaziati e li tiene uniti *in ogni caso*:

Non così... ma così: `\dots`
Londra, Parigi`\dots` Berlino.

Non così... ma così:
Londra, Parigi... Berlino.

Per indicare l'omissione di una parola o una porzione di testo originali, si usa il segno di *omissione* [...] definendo nel preambolo un comando ad hoc `\omissis` (si veda il paragrafo 13.1 a pagina 245):

```
\newcommand{\omissis}{[\textellipsis\unlign]}
```

Si noti che entrambi i comandi appena visti “producono testo”: prima della parola successiva si metta uno spazio esplicito nei modi consueti.

4.3.2 Loghi, accenti, caratteri particolari, apici e pedici**Loghi, accenti e caratteri particolari**

La tabella 15 elenca i comandi che producono i loghi più comuni nel mondo \LaTeX (si veda la documentazione dei pacchetti dtklogos e hologo per ulteriori loghi).

\LaTeX permette di usare accenti e caratteri particolari di molte lingue (nella tabella 16 nella pagina successiva sono esemplificati per la lettera *o*, ma funzionano anche per tutte le altre lettere), come si può vedere nell'esempio seguente:

`H\^otel`, `na\~{i}f`, `Stra\ss e`,
`!~{S}e\~{n}orita!`, `élève`,
`Sm\o rrebr\o d`, `z\l otys`

Hôtel, naïf, Straße, ¡Señorita!,
élève, Smørrebrød, zlotys

Il simbolo ufficiale dell'euro (€), da mettere sempre *dopo* l'eventuale numero, si ottiene con il comando `\euro` del pacchetto eurosym.

Tabella 16: Accenti e caratteri particolari

<code>\‘o</code>	ò	<code>\u{o}</code>	ö	<code>\~o</code>	õ	<code>\.o</code>	ó
<code>\’o</code>	ó	<code>\t{oo}</code>	ôo	<code>\r{o}</code>	ô	<code>\d{o}</code>	ô
<code>\^o</code>	ô	<code>\"o</code>	ö	<code>\c{o}</code>	ç	<code>\=o</code>	ō
<code>\v{o}</code>	ö	<code>\H{o}</code>	ö	<code>\k{o}</code>	ç	<code>\b{o}</code>	ö
<code>\OE</code>	Œ	<code>\AE</code>	Æ	<code>\AA</code>	Å	<code>\O</code>	Ø
<code>\oe</code>	œ	<code>\ae</code>	æ	<code>\aa</code>	å	<code>\o</code>	ø
<code>\L</code>	Ł	<code>\DH</code>	Đ	<code>\DJ</code>	Đ	<code>\TH</code>	Þ
<code>\l</code>	ł	<code>\dh</code>	đ	<code>\dj</code>	đ	<code>\th</code>	þ

Apici e pedici

L’opzione `italian` di `babel` definisce una coppia di comandi che producono il proprio argomento in tondo *anche in modo matematico*. In modo testuale, inoltre, mantengono anche lo stile corrente, qualunque sia:

- `\ap{<testo>}` produce un apice come nelle abbreviazioni oggi in disuso *sig.^{ra}* o *f.^{lli}* (l’alternativa è `<testo>`);
- `\ped{<testo>}` produce un pedice, utile per qualche sostanza chimica come la vitamina B₁₂, per esempio.

Per l’elenco completo di *tutti* i simboli e i caratteri speciali di L^AT_EX (diverse migliaia), si veda [Pakin, 2015].

4.4 MODIFICARE STILE E CORPO DEL FONT

4.4.1 Modificare lo stile

I comandi elencati nella tabella 17 a fronte modificano lo *stile* del proprio argomento (e solo di quello), lasciando invariato il testo successivo:

La parola che segue è in
`\textit{corsivo}`.
Il resto del testo è normale.

La parola che segue è in *corsivo*. Il
resto del testo è normale.

I comandi si possono combinare, ma la combinazione richiesta potrebbe non esserci nel font in uso, come mostrano gli esempi seguenti:

L’espressione che segue
`\textit{\textbf{è in nero corsivo}}`, ma questa
`\textsc{\textit{non è in maiuscoletto corsivo}}`.

L’espressione che segue è *in nero corsivo*, ma questa non è in *maiuscoletto corsivo*.

A ciascun comando corrisponde una dichiarazione che si comporta come spiegato nel paragrafo 3.4.1 a pagina 24. Anche le dichiarazioni si possono combinare:

L’espressione che segue `\itshape`
è in `\bfseries` nero corsivo.

L’espressione che segue è *in nero corsivo*.

ma si consiglia di usarle, se proprio necessarie, per porzioni di testo consistenti e non per singole parole come qui.

Tabella 17: Comandi e dichiarazioni per modificare lo stile del font. Per evidenziare la differenza dal corsivo, lo stile inclinato è composto con il font Computer Modern.

Comando	Dichiarazione	Stile
<code>\emph</code>	<code>\em</code>	<i>Evidenziato</i>
<code>\textit</code>	<code>\itshape</code>	<i>Corsivo</i>
<code>\textsc</code>	<code>\scshape</code>	MAIUSCOLETTA
<code>\textbf</code>	<code>\bfseries</code>	Nero
<code>\textsl</code>	<code>\slshape</code>	<i>Inclinato</i>
<code>\textrm</code>	<code>\rmfamily</code>	Tondo
<code>\textsf</code>	<code>\sffamily</code>	Senza grazie
<code>\texttt</code>	<code>\ttfamily</code>	Macchina per scrivere

4.4.2 Modificare il corpo

L'effettivo corpo del font in un documento dipende da tre fattori:

- la classe di documento scelta;
- l'opzione di corpo (eventualmente) assegnata alla classe;
- le (eventuali) dichiarazioni per modificare il corpo del font date nel testo.

Le dichiarazioni elencate nella tabella 18 nella pagina seguente modificano il *corpo* del font. Anche il contenuto della tabella risente dei fattori appena elencati: in particolare, `\normalsize` è il corpo del testo principale di questa guida.

Lettere `{\Large}` grandi e
`{\scriptsize}` piccole}.

Lettere **grandi** e piccole.

Si noti che le dichiarazioni appena viste *modificano anche l'interlinea* del capoverso interessato, com'è giusto che sia, ma solo se esso termina entro il loro raggio d'azione. Nei due esempi seguenti, `\par` produce effetti differenti a seconda di dove lo si dà.

`{\large}` Socrate: «Platone mentirà
nella frase seguente».`\par`

Socrate: «Platone mentirà
nella frase seguente».

`{\large}` Platone: «Socrate
ha detto il vero
nella frase precedente».`\par`

Platone: «Socrate ha detto il
vero nella frase precedente».

Come si può osservare nel primo esempio, fuori dal gruppo `\par` non funziona più, con un risultato finale poco gradevole. La scrittura corretta è la seconda.

Questo viaggio tra stili e dimensioni si conclude con un simpatico consiglio, che starà all'utente seguire o meno:

Ricorda! Quanti Più corpi e stili scegli di usare in un documento,
tanto più LEGGIBILE e bello diventa.

Tabella 18: Dichiarazioni per modificare il corpo del font

Dichiarazione	Risultato
<code>\tiny</code>	Esempio
<code>\scriptsize</code>	Esempio
<code>\footnotesize</code>	Esempio
<code>\small</code>	Esempio
<code>\normalsize</code>	Esempio
<code>\large</code>	Esempio
<code>\Large</code>	Esempio
<code>\LARGE</code>	Esempio
<code>\huge</code>	Esempio
<code>\Huge</code>	Esempio

4.5 TITOLI E FRONTESPIZI

Titoli standard

Il comando

```
\maketitle
```

dato dopo `\begin{document}` produce il “titolo” del documento, un blocco di informazioni definite dai comandi

```
\title{⟨titolo⟩}
\author{⟨autore⟩}
\date{⟨data⟩}
```

Il loro funzionamento si spiega da sé, ma si osservi quanto segue:

- un `⟨titolo⟩` troppo lungo per stare su una sola riga si spezza con `\\` (ma lo si eviti il più possibile);
- i vari `⟨autore⟩` di un documento scritto a più mani si separano con `\and`;
- \LaTeX stampa la `⟨data⟩` della composizione anche se `\date` non viene dato, mentre la omette lasciandone vuoto l'argomento (`\date{}`).

Per inserire ringraziamenti veloci si usa il comando

```
\thanks{⟨ringraziamenti⟩}
```

che nelle classi standard produce il proprio argomento come una nota al piede con un simbolo a esponente. Lo si può dare *dentro* l'argomento di uno qualunque dei tre comandi appena esaminati. Ecco un esempio:

```
\author{Lorenzo Pantieri \and Tommaso Gordini%
\thanks{Ringraziamo i membri del \GuIT.}}
```

Nelle classi \mathcal{AMS} `\thanks` va invece dato in una riga a sé e *fuori* dai comandi.

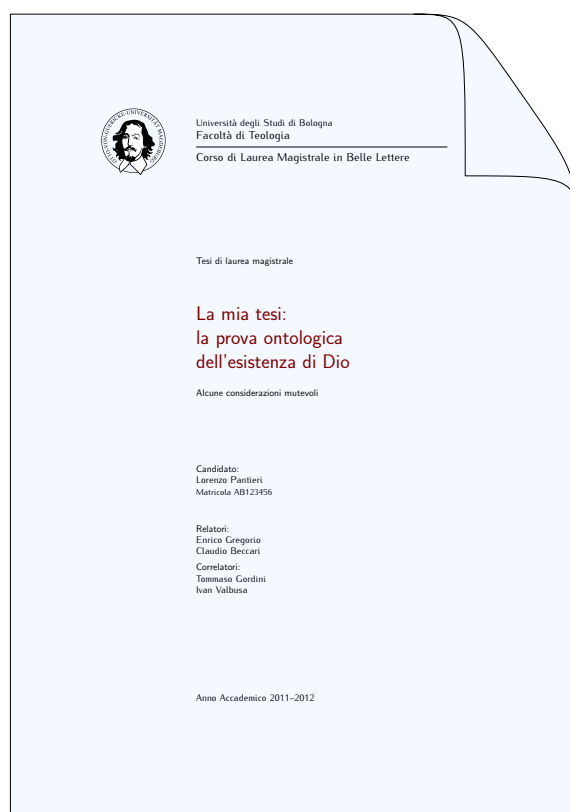


Figura 5: Esempio d'uso di frontespizio

Frontespizio

Il titolo generato dal comando `\maketitle`, si deve riconoscere, è piuttosto spartano, anche se si può accettare in articoli e relazioni. Si consiglia di comporre il frontespizio di una tesi di laurea o di dottorato con il pacchetto `frontespizio` (si veda la figura 5). Il pacchetto, personalizzabile, permette di inserire tutti i dati necessari, prevede opzioni per usare i diversi stili di carattere e inserire loghi universitari e immagini in filigrana. Se ne veda la documentazione (in italiano).

In alternativa si può usare la *suite* `ClassicThesis`, che comprende un modello di tesi pronto per l'uso completo di frontespizio (si veda il capitolo 22 a pagina 317).

Infine, se proprio nessuna delle soluzioni precedenti va bene, si può comporre un frontespizio personalizzato con l'ambiente `titlepage` (da aprire subito dopo `\begin{document}`) all'interno del quale si è completamente padroni dell'impaginazione.

4.6 NOTE A MARGINE E A PIÈ DI PAGINA

In linea generale si usino le note *con grande moderazione*: specie quelle al piede, infatti, interrompono la lettura e possono creare seri problemi d'impaginazione. Si tenga presente che una nota *deve* potersi omettere leggendo: se il suo contenuto si rivela essenziale alla comprensione del discorso, evidentemente va tolto dalla nota e messo nel corpo del testo.

Note a margine

Una nota a margine

Una nota di questo tipo si ottiene molto semplicemente con il comando

```
\marginpar{⟨testo della nota a margine⟩}
```

Nei documenti impostati per la stampa in fronte/retro le note vengono stampate nel margine destro nelle pagine dispari e nel margine sinistro in quelle pari. Nei documenti solo fronte, invece, saranno sempre nel margine destro. (Se una o più note dovessero comparire nel margine sbagliato, basta caricare il pacchetto mparhack.)

Note a piè di pagina

Il comando

```
\footnote{⟨testo della nota a piè di pagina⟩}
```

produce una nota in fondo alla pagina corrente con un riferimento nel testo costituito da un numero a esponente. Le note al piede dovrebbero essere messe, per quanto possibile, alla fine del relativo capoverso *dopo* il punto fermo.¹

Le note a piè di pagina sono
l’emblema della meticolosità.%
\footnote{Eccone un esempio.}

Le note a piè di pagina sono
l’emblema della meticolosità.^a

^a Eccone un esempio.

Si tenga presente che:

- la loro numerazione riprende a ogni \chapter o \section;
- se sono poche o pochissime, anziché il riferimento numerico predefinito se ne consiglia uno simbolico: basta scrivere nel preambolo

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
```

L^AT_EX produce note di alta qualità, ma quando circostanze particolari le richiedono in un formato diverso da quello predefinito basta caricare il pacchetto footmisc e configurarlo opportunamente (se ne veda la documentazione).

4.7 EVIDENZIARE LE PAROLE

Scrivendo a macchina, le parole importanti si evidenziano con una sottolineatura; in tipografia, invece, le parole si evidenziano mettendole *in corsivo*. Le possibilità sono due.

Per evidenziare una parola o una porzione di testo *indipendentemente dal contesto* in cui si trovano, L^AT_EX definisce il comando standard

```
\emph{⟨testo⟩}
```

che si vede all’opera nell’esempio seguente:

¹ Così.

```
\emph{All'interno di un testo
già evidenziato, \LaTeX{}
evidenzia ulteriormente con lo
\emph{stile tondo}.}
```

All'interno di un testo già evidenziato, \LaTeX evidenzia ulteriormente con lo stile tondo.

C'è anche un altro comando standard, `\textit`, che produce il proprio argomento in corsivo *in ogni caso*. Per cogliere la differenza logica tra corsivo ed evidenziato, si osservino le due scritture:

```
\emph{Fra \textit{un} minuto.} \\\
\textit{Fra \emph{un} minuto.}
```

Fra un minuto.
Fra un minuto.

4.8 AMBIENTI TESTUALI

4.8.1 Elenchi puntati, numerati e descrizioni

In un documento gli elenchi sono molto importanti. Infatti:

- fanno “respirare” il testo;
- ne migliorano la leggibilità;
- permettono di strutturare i pensieri.

L'elenco precedente è stato ottenuto con l'ambiente `itemize` come segue:

```
Gli elenchi:
\begin{itemize}
\item fanno “respirare” il testo;
\item ne migliorano
la leggibilità;
\item permettono di
strutturare i pensieri.
\end{itemize}
```

Gli elenchi:

- fanno “respirare” il testo;
- ne migliorano la leggibilità;
- permettono di strutturare i pensieri.

Il comando `\item` mette un pallino nero prima di ogni elemento dell'elenco.

L'ambiente `enumerate` si usa come `itemize`, ma qui a ogni elemento `\item` premette un numero puntato:

```
Ecco un elenco numerato:
\begin{enumerate}
\item Mane;
\item Tekel;
\item Fares.
\end{enumerate}
```

Ecco un elenco numerato:

1. Mane;
2. Tekel;
3. Fares.

È opportuno usare una lista numerata se in seguito ci si deve riferire a un suo elemento particolare (anche assegnandogli un'etichetta) o se per esempio si devono elencare le fasi di un procedimento. Altrimenti è più opportuna una lista puntata.

L'ambiente `description` si usa per le *descrizioni*, elenchi in cui il segno distintivo è una parola o un'espressione che si deve descrivere o spiegare, da scrivere nell'argomento facoltativo (*in questo caso*, però, obbligatorio) di `\item`:

```
E ora una descrizione:
\begin{description}
\item[itemize] Per gli
    elenchi puntati.
\item[enumerate] Per gli
    elenchi numerati.
\item[description] Per gli
    elenchi in cui ogni elemento
    comincia con un testo
    a piacere.
\end{description}
```

E ora una descrizione:

ITEMIZE Per gli elenchi puntati.

ENUMERATE Per gli elenchi numerati.

DESCRIPTION Per gli elenchi in cui ogni elemento comincia con un testo a piacere.

Come si può osservare, \LaTeX evidenzia automaticamente l'argomento del comando secondo le impostazioni generali della classe di documento in uso.

\LaTeX permette di annidare anche gli elenchi (si consiglia di non annidare mai più di una lista dentro l'altra, però), come mostra l'esempio seguente:

```
Gli elenchi:
\begin{itemize}
\item sono facili da usare;
\item rendono più chiaro il testo:
    \begin{itemize}
    \item articolandolo;
    \item facilitandone la lettura;
    \end{itemize}
\end{itemize}
\item permettono di strutturare
    meglio il pensiero.
\end{itemize}
```

Gli elenchi:

- sono facili da usare;
- rendono più chiaro il testo:
 - articolandolo;
 - facilitandone la lettura;
- permettono di strutturare meglio il pensiero.

Si noti che \LaTeX cambia automaticamente il contrassegno negli elenchi annidati, che si possono individuare più facilmente nel sorgente rientrandoli leggermente (può essere una buona abitudine, ma non è una regola).

Di seguito si riportano alcune convenzioni tipografiche comunemente seguite nella composizione delle liste (le stesse osservate in questa guida):

- ogni voce di un elenco *semplice* (i cui elementi sono costituiti da un solo enunciato) comincia con l'iniziale minuscola e termina con il punto e virgola tranne l'ultima, seguita dal punto fermo;
- ogni voce di un elenco *complesso* (in cui *almeno uno* degli elementi sia composto da più di un enunciato) comincia con l'iniziale maiuscola (anche dopo il segno di due punti) e termina con il punto fermo.

Non bisogna per forza uniformare *tutti* gli elenchi di un documento a criteri stabiliti a priori: l'importante è essere coerenti volta per volta.

4.8.2 Allineare e centrare i capoversi

\LaTeX definisce tre ambienti standard per allineare un capoverso a sinistra:

```
\begin{flushleft}
Questo testo è allineato a \
sinistra. \LaTeX{} non cerca di
creare righe di uguale lunghezza.
\end{flushleft}
```

Questo testo è allineato a sinistra. \LaTeX non cerca di creare righe di uguale lunghezza.

a destra:

```
\begin{flushright}
Questo testo è allineato a \\
destra. \LaTeX{} non cerca di
creare righe di uguale lunghezza.
\end{flushright}
```

Questo testo è allineato a
destra. \LaTeX non cerca di creare
righe di uguale lunghezza.

o per centrarlo sulla pagina:

```
\begin{center}
Al centro \\
dell'universo.
\end{center}
```

Al centro
dell'universo.

Come si può osservare, il testo va a capo automaticamente, a meno di un'interruzione esplicita con `\\`.

Dedica

Non esistono regole tipografiche vincolanti per produrre la dedica in una pubblicazione, se non il gusto e le esigenze dell'utente. In linea generale, se di dimensioni contenute si colloca orizzontalmente sulla pagina con uno degli ambienti appena considerati.

Altrettanto libero ne è il collocamento in verticale, che si può controllare con i comandi opportuni: qualche prova permette di ottenere il risultato desiderato.

4.8.3 Citazioni e intercitazioni

Esistono due modi per scrivere le citazioni con \LaTeX : “in linea” e “in display”.

Citazioni in linea

Una citazione “in linea” è un testo tra virgolette appartenente al flusso del discorso, come quando si cita il motto kantiano «il cielo stellato sopra di me, la legge morale dentro di me».

Una citazione in linea
è un'espressione appartenente
al flusso del discorso:
«il cielo stellato sopra di me,
la legge morale dentro di me».

Una citazione in linea è un'espres-
sione appartenente al flusso del di-
scorso: «il cielo stellato sopra di
me, la legge morale dentro di me».

Citazioni di questo tipo sono generalmente brevi.

Citazioni in display

Una citazione “in display” è un testo che va composto entro margini più ampi di quelli correnti e separandolo dal contesto con adeguati spazi bianchi, in modo da metterlo “in mostra” e bene in risalto sulla pagina.

I due ambienti standard definiti da \LaTeX allo scopo, `quote` e `quotation`, non sono del tutto soddisfacenti perché, per esempio, non riducono automaticamente il corpo del testo citato come richiedono le buone tradizioni

tipografiche. Per ottenere citazioni in display ben composte si consiglia il pacchetto `quoting`, da impostare come segue nel preambolo se si prevede di mantenere lo stesso stile in tutte le citazioni del documento (scelta consigliata):

```
\usepackage{quoting}
\quotingsetup{font=small}
```

Il pacchetto definisce l'omonimo ambiente `quoting` da usare così:

Una citazione in display è un testo che `\LaTeX{}` compone su linee a sé:

```
\begin{quoting}
Il cielo stellato sopra di me,
la legge morale dentro di me.
\end{quoting}
Come si può osservare, la
citazione è centrata e
separata dal resto del testo.
```

Una citazione in display è un testo che `\LaTeX` compone su linee a sé:

Il cielo stellato sopra
di me, la legge morale
dentro di me.

Come si può osservare, la citazione è centrata e separata dal resto del testo.

Si noti che la citazione può essere lunga a piacere e suddivisa in capoversi, se necessario. In particolare:

- se la citazione continua il discorso principale non deve avere la prima riga rientrata: in questo caso nel sorgente `quoting` seguirà immediatamente il testo che precede *senza righe bianche in mezzo*;
- se la citazione comincia un nuovo capoverso, invece, il rientro ci va: basta lasciare una riga bianca tra l'ambiente e il testo precedente.

In entrambi i casi (da intendere come suggerimenti più che come regole universalmente accettate), la prima riga dei capoversi successivi al primo sarà rientrata per impostazione predefinita.

Per ulteriori esigenze (citazioni in lingua straniera, indicazione della fonte, eccetera) si rimanda al paragrafo [13.2.2](#) a pagina [249](#) e al pacchetto `csquotes`.

Citazioni annidate

Per le *citazioni annidate* (cioè citazioni dentro altre citazioni) le convenzioni oscillano, perciò la cosa migliore è decidere per l'una o l'altra delle seguenti possibilità e mantenerla in tutto il documento. In generale:

- se è in linea, si può mettere la citazione tra virgolette diverse da quelle scelte per la citazione principale;
- se è in display, la si può mettere tra virgolette oppure in un altro ambiente `quoting` annidato in quello principale.

4.8.4 Poesie

Per scrivere poesie `\LaTeX` definisce l'ambiente standard `verse`, nel quale:

- i margini vengono aumentati come per le citazioni;
- ogni verso, tranne l'ultimo della strofa, *deve* finire con `\\`;

- le strofe vengono separate automaticamente tra loro con dello spazio bianco.

L'esempio seguente mostra l'ambiente all'opera:

```
La poesia \emph{Un'altra notte}
è di Giuseppe Ungaretti.
\begin{verse}
In quest'oscuo \\
colle mani \\
gelate \\
distinguo \\
il mio viso

Mi vedo abbandonato \\
nell'infinito
\end{verse}
```

La poesia *Un'altra notte* è di
Giuseppe Ungaretti.

In quest'oscuo
colle mani
gelate
distinguo
il mio viso

Mi vedo abbandonato
nell'infinito

Per esigenze poetiche più avanzate è utile il pacchetto `verse` (se ne veda la documentazione).

4.8.5 Codici e algoritmi

Talvolta capita di dover scrivere parole o frammenti di testo in modo *verbatim* ("alla lettera"), cioè *non* interpretando spazi, caratteri speciali, rientri, a capo, simboli e comandi, che potrebbero avere un'importanza particolare e devono rimanere tali. Questa modalità è utile per riportare esempi di codici informatici e linguaggi di programmazione, come si è fatto in questa guida tutte le volte che si è mostrata la sintassi dei comandi e degli ambienti di \LaTeX .

Per scrivere un frammento di testo *verbatim in linea* e che non debba andare a capo, \LaTeX definisce il comando standard

```
\verb!⟨testo verbatim⟩!
```

Il carattere `!` è solo uno dei possibili *caratteri delimitatori*, cioè con la sola funzione di indicare inizio e fine del `⟨testo verbatim⟩`: a questo scopo si può usare un carattere qualunque, *tranne* `*`, purché non compaia tra i caratteri da riprodurre. Se ne consiglia uno tra `!` `?` `|` `@`. Si noti che \LaTeX *non* sillaba il `⟨testo verbatim⟩`: se troppo lungo, infatti, sposterà nel margine destro.

Per scrivere testo *verbatim in display e su più righe*, invece, c'è l'ambiente standard `verbatim`, da usare come di consueto. A questo proposito si noti che:

- né `\verb` né `verbatim` possono comparire nell'argomento di un altro comando;
- `verbatim` può riprodurre tutto *tranne* `\end{verbatim}` (l'ambiente sarebbe chiuso due volte, infatti, e l'errore arriverebbe inesorabile).

Sia `\verb` sia `verbatim` prevedono una variante asterisco che riproduce lo spazio in modo visibile con il carattere `_`, come si può osservare negli esempi seguenti.

Il logo `"\LaTeX"` si ottiene
con il comando `\verb!\LaTeX!`.

Il logo `"\LaTeX"` si ottiene con il
comando `\LaTeX`.

```
\begin{verbatim}
Nell'ambiente verbatim
i comandi di \LaTeX,
gli
a capo, gli      spazi,
      i rientri e i
caratteri speciali (\{\}%$_&#^~)
non vengono interpretati.
\end{verbatim}
```

```
Nell'ambiente verbatim
i comandi di \LaTeX,
gli
a capo, gli      spazi,
      i rientri e i
caratteri speciali (\{\}%$_&#^~)
non vengono interpretati.
```

```
\begin{verbatim*}
Il comando \verb*
e l'ambiente verbatim*
mostrano gli spazi così.
\end{verbatim*}
```

```
Il_comando_\verb*
e_l'ambiente_verbatim*
mostrano_gli_spazi_così.
```

Questi strumenti generalmente riescono a soddisfare le esigenze più comuni, ma non si possono personalizzare in alcun modo (con colori e sfondi particolari, riquadri, possibilità di definire ambienti e linguaggi personali, per esempio). Si rimanda chi ne avesse bisogno al pacchetto `listings`.

4.9 ACRONIMI E GLOSSARI

In lavori particolarmente tecnici e complessi si consiglia di mettere nel *backmatter* un elenco degli acronimi menzionati nel testo e un *glossario*, cioè l'elenco alfabetico di una scelta di termini (di solito specialistici) presenti nel documento con la relativa definizione (ed eventualmente un simbolo).

Il pacchetto `acronym` (si veda il capitolo successivo) gestisce efficacemente gli acronimi, generando automaticamente anche i collegamenti ipertestuali tra acronimo nel testo e relativa spiegazione nell'elenco.

Per comporre un vero e proprio glossario c'è il pacchetto `glossaries` (utile anche per gli acronimi), che tramite il programma `MakeIndex` genera automaticamente i corrispondenti elenchi.

5 | ACRONIMI

Questo capitolo presenta il pacchetto `acronym`, che permette di gestire gli acronimi con \LaTeX .

ACRONIMI

\GUIT	Gruppo Utilizzatori Italiani di \TeX e \LaTeX È un'associazione che si prefigge di aumentare la diffusione di \TeX e \LaTeX in Italia.
WYSIWYM	What You See Is What You Mean L'acronimo ("ciò che vedi è ciò che intendi") indica un programma di scrittura dotato di una composizione asincrona.

5.1 DEFINIRE GLI ACRONIMI

Gli acronimi del documento vengono definiti mediante l'ambiente `acronym` (simile a `description`). Nell'ambiente `acronym`, ogni acronimo è definito con

```
\acro{<acronimo>}[<nome breve>]{<nome esteso>}
```

Dove:

- `<acronimo>` indica l'acronimo stesso (`WYSIWYM`, per esempio);
- `<nome esteso>` indica il nome per esteso dell'acronimo ("What You See Is What You Mean", per esempio);
- se l'acronimo richiede dei comandi di \LaTeX , il codice necessario si scrive nell'argomento facoltativo `<nome breve>`; in questo caso, `<acronimo>` rappresenta semplicemente un'etichetta per identificare l'acronimo.

Per esempio, in questo documento gli acronimi `WYSIWYM` e `\GUIT` sono stati introdotti con le istruzioni

```
\section*{Acronimi}
\begin{acronym}[WYSIWYM]
\acro{GUIT}{\GUIT}{Gruppo Utilizzatori Italiani di \TeX{} e \LaTeX{}}

{\small È un'associazione che si prefigge di aumentare la diffusione di
\TeX{} e \LaTeX{} in Italia.\par}

\acro{WYSIWYM}{What You See Is What You Mean}

{\small L'acronimo ("ciò che vedi è ciò che intendi") indica un
programma di scrittura dotato di una composizione asincrona.\par}
\end{acronym}
```

Il comando `\GuIT` richiede il pacchetto `guit`.

L'argomento facoltativo assegnato all'ambiente `acronym` ([WYSIWYM], nell'esempio proposto) adatta, nell'elenco degli acronimi, la larghezza della colonna degli acronimi alla larghezza del parametro dato (si specifica l'acronimo più lungo).

5.2 ACRONIMI NEL TESTO

Una volta definito l'acronimo, per scriverlo nel testo si usa il comando

```
\ac{<acronimo>}
```

La prima volta che si inserisce un acronimo viene stampato il suo nome per esteso, seguito dall'acronimo fra parentesi. Le volte successive viene stampato solo l'acronimo.

La prima volta che si scrive un acronimo viene stampato il suo nome per esteso, seguito dall'acronimo fra parentesi:
`\ac{GUIT}`, per esempio.

Le volte successive viene stampato solo l'acronimo:
`\ac{GUIT}`.

La prima volta che si scrive un acronimo viene stampato il suo nome per esteso, seguito dall'acronimo fra parentesi: Gruppo Utilizzatori Italiani di \TeX e \LaTeX (`\qJr`), per esempio.

Le volte successive viene stampato solo l'acronimo: `\qJr`.

Se si vuole inserire comunque il nome completo (*full name*) dell'acronimo (formato dal nome per esteso seguito dall'acronimo stesso), si usa il comando

```
\acf{<acronimo>}
```

Per inserirne la versione breve (*short name*), si usa il comando

```
\acs{<acronimo>}
```

mentre il comando

```
\acl{<acronimo>}
```

scrive solo il nome per esteso dell'acronimo (*long name*).

```
\acf{GUIT}
```

```
\acs{GUIT}
```

```
\acl{GUIT}
```

Gruppo Utilizzatori Italiani di \TeX e \LaTeX (`\qJr`)

`\qJr`

Gruppo Utilizzatori Italiani di \TeX e \LaTeX

Bisogna compilare due volte con \LaTeX . Il pacchetto `hyperref` produce i riferimenti ipertestuali tra gli acronimi nel testo e nel relativo elenco.

5.3 OPZIONI

Le principali opzioni del pacchetto `acronym` sono le seguenti:

- `footnote` stampa il nome esteso dell'acronimo in una nota a piè di pagina;
- `smaller` stampa gli acronimi in corpo più piccolo rispetto al testo, seguendo una diffusa convenzione tipografica;
- `printonlyused` stampa, nell'elenco degli acronimi, solo quelli effettivamente usati nel documento (senza questa opzione, l'elenco degli acronimi comprende tutti gli acronimi definiti, che siano usati o no);
- `withpage` nell'elenco degli acronimi, stampa accanto a ogni acronimo il numero di pagina dove esso viene menzionato la prima volta (deve essere specificata l'opzione `printonlyused`).

Parte III

MATEMATICA E CODICI

Questo capitolo esplora uno dei principali punti di forza di \LaTeX , anche se ne intacca solamente la superficie: la composizione di formule matematiche. Si spiegheranno strumenti sufficienti per la *maggior parte* delle esigenze, ma potrebbe darsi che *quella particolare* necessità non trovi risposta in queste pagine. Se così fosse, molto probabilmente la soluzione sta in una delle funzioni del pacchetto `amsmath` (che non potrà essere descritto per intero, dati i limiti di questo lavoro) o di qualche altro pacchetto dedicato.

La scrittura della matematica e delle altre scienze è regolata da norme dipendenti dalle tradizioni e dalla cerchia dei lettori cui lo scritto è destinato, e raccolte nel mondo dalla normativa ISO (*International Standard Organization*, “Organizzazione Internazionale per lo Standard”), in Italia dalla normativa UNI (Ente Nazionale Italiano di Unificazione). Le norme emesse dall’UNI hanno valore di legge (per cui se un documento per uso legale viene scritto in modo conforme a esse non perde tale valore) ma nella versione originale le si può consultare soltanto tramite un servizio a pagamento.

Di qui in avanti si danno per caricati i pacchetti `amsmath` e `amssymb`.

6.1 FORMULE IN LINEA E IN DISPLAY

In linea generale, nelle formule matematiche le variabili vengono rese in *corsivo matematico*, diverso dal *corsivo ordinario*.

Con \LaTeX si può scrivere la matematica in due modi: “in linea” e “in display”.

6.1.1 Formule in linea

Una formula “in linea” è incorporata nel testo: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$, per esempio. Come si può osservare, \LaTeX fa il possibile per comprimerla e modificare meno che può l’interlinea nel capoverso che la contiene. Non ci si preoccupi se con questa modalità di scrittura elementi che di solito si trovano sopra o sotto un simbolo gli compaiono accanto: è tipograficamente corretto. L’esempio seguente mostra come si scrive una formula di questo tipo:

Una formula in linea è incorporata nel testo:

```
\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2} =
\frac{\pi^2}{6}$. \LaTeX{}
```

modifica il meno possibile l’interlinea del capoverso.

Una formula in linea è incorporata nel testo: $\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$. \LaTeX modifica il meno possibile l’interlinea del capoverso.

Le formule in linea si scrivono tra dollari $\$ \dots \$$ (oppure, ma meno spesso, tra i comandi `\(\dots \)`) e si consiglia di usarle solo con espressioni di altezza contenuta come le seguenti:

Ci sono voluti secoli per dimostrare che quando $n > 2$ **\emph{non}** ci sono tre interi positivi a , b , c tali che $a^n + b^n = c^n$.

Ci sono voluti secoli per dimostrare che quando $n > 2$ *non* ci sono tre interi positivi a , b , c tali che $a^n + b^n = c^n$.

6.1.2 Formule in display

Una formula “in display”, invece, è un’espressione che \LaTeX compone su linee a sé, separate dal contesto con adeguati spazi bianchi per “metterla in mostra” e farla risaltare sulla pagina. L’esempio in linea del paragrafo precedente diventa, in display:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Come si può osservare, ora la formula è centrata, non compressa, e tutti i suoi elementi occupano il giusto spazio con un risultato finale di grande respiro.

L’unico modo corretto per scrivere queste formule è usare uno dei due ambienti matematici seguenti:

- `equation` per le formule numerate;
- `equation*` (di solito abbreviato in `\[...\]`) per quelle non numerate.

Degli altri modi esistenti per farlo, oggi *non devono essere più usati*:

- I dollari doppi `$$...$$`, che potrebbero compromettere la corretta spaziatura verticale delle formule o il funzionamento dell’opzione di classe `fleqn` [Fairbairns, 2014].
- Gli ambienti standard `eqnarray` e `eqnarray*` (per sistemi di formule numerate e non numerate rispettivamente), perché prima e dopo = inseriscono più spazio del dovuto. È un difetto conservato in $\text{\LaTeX}_{2\epsilon}$ per mantenerne la compatibilità con le vecchie versioni del programma.

Si considerino gli esempi seguenti:

Una formula in display
è un’espressione che `\LaTeX{}`
compone su linee a sé stanti:
`\[`
`\lim_{n \to \infty}`
`\sum_{k=1}^n \frac{1}{k^2} =`
`\frac{\pi^2}{6}`
`\]`

Una formula in display è un’espressione che \LaTeX compone su linee a sé stanti:

$$\lim_{n \rightarrow \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

Se f è continua e
`\[`
`F(x) = \int_a^x f(t) dt`
`\]`
allora
`\begin{equation}`
`F'(x) = f(x)`
`\end{equation}`

Se f è continua e

$$F(x) = \int_a^x f(t) dt$$

allora

$$F'(x) = f(x) \quad (6.1)$$

Come si può osservare, le formule non sono accompagnate da segni di punteggiatura.

Intorno alla punteggiatura nelle formule in display sono fiorite due scuole di pensiero. Alcuni, tra cui chi scrive, ritengono che non andrebbe *mai* usata, perché superflua e causa di possibili ambiguità nella lettura [Beccari, 2016]. Altri, ritenendo le formule in genere parte integrante dell'argomentazione, pensano che la punteggiatura aiuti chi legge e che quindi ci vada [Guiggiani e Mori, 2008]. Qualunque delle due scuole si scelga, l'importante è seguirla in *tutto* il documento.

Si scrivono in display espressioni complesse e di grandi dimensioni (troppo sacrificate tra le righe di un capoverso) e formule più contenute a cui si voglia dare un risalto particolare.

I comandi `\label` ed `\eqref` permettono i riferimenti incrociati alle formule (come già visto nel paragrafo 3.10 a pagina 43):

<pre>\begin{equation} \label{eqn:eulero} e^{i\pi}+1=0 \end{equation} Dalla formula~\eqref{eqn:eulero} si deduce che\dots</pre>	$e^{i\pi} + 1 = 0 \quad (6.2)$ <p>Dalla formula (6.2) si deduce che...</p>
--	--

6.1.3 Modo matematico e modo testuale

La modalità con cui si scrive la matematica (*modo matematico*) differisce per alcuni aspetti da quella con cui si scrive il testo (*modo testuale*). Ecco i principali.

- \LaTeX inserisce automaticamente gli spazi in base alla struttura della formula e ignora quelli che trova nel sorgente (interruzioni di riga comprese). Se serve, si possono inserire *a mano* ulteriori spazi con i comandi raccolti nella tabella 24 a pagina 87.
- Nella scrittura delle formule *non sono ammesse righe vuote*.
- \LaTeX mette in corsivo matematico *tutte* le lettere che trova in una formula, considerandole altrettante variabili. Per inserire in una formula in display un (breve) testo in tondo e spaziato normalmente si usa il comando `\text`, esplicitando la spaziatura prima e dopo.

I due esempi seguenti mostrano quanto si è appena descritto:

<pre>\$x+y+z=n\$ \\\ \$ x + y + z = n \$</pre>	$x + y + z = n$ $x + y + z = n$
<pre>\[z^2+1=0 \quad \text{per } z=\pm i \]</pre>	$z^2 + 1 = 0 \quad \text{per } z = \pm i$

Il comando `\pm` produce \pm (`\mp` produce \mp).

6.2 NOZIONI INTRODUTTIVE

Questo paragrafo descrive i comandi più usati per scrivere le formule matematiche. (Ulteriori comandi sono raccolti nelle tabelle contenute nelle prossime sezioni.)

6.2.1 Raggruppamenti

La maggior parte dei comandi matematici agisce soltanto sul carattere immediatamente successivo. Si evita questo comportamento racchiudendo il testo interessato in un gruppo di parentesi graffe:

```
\[
a^{x+y} \neq a^{x+y}
\]
```

$$a^x + y \neq a^{x+y}$$

6.2.2 Esponenti, indici e radici

Apici e pedici si scrivono dopo i caratteri `^` e `_` rispettivamente:

```
$a_1$ \quad $x^2$ \quad $e^{-\beta t}$ \quad $a_{ij}^3$
```

$$a_1 \quad x^2 \quad e^{-\beta t} \quad a_{ij}^3$$

Il comando `\qquad` produce uno spazio orizzontale di un *quadrato*.

Gli indici di secondo ordine vanno messi in un gruppo di graffe insieme a quelli di ordine superiore: una scrittura come `x_{n_k}` non ha senso.

```
Dalla successione $x_n$
estrarre $x_{n_k}$.
```

Dalla successione x_n estrarre x_{n_k} .

Il simbolo di radice quadrata si ottiene con `\sqrt`, quello di radice n -esima con

```
\sqrt[n]{\dots}
```

\LaTeX calcola automaticamente le dimensioni della radice:

```
\[
\sqrt{x} \quad \sqrt[3]{2} \quad \sqrt{\frac{a}{b}}
\]
```

$$\sqrt{x} \quad \sqrt[3]{2} \quad \sqrt{\frac{a}{b}}$$

6.2.3 Somme, prodotti e frazioni

Il simbolo di sommatoria è generato da `\sum` e quello di produttoria da `\prod`. Gli estremi si scrivono come indici.

```
Trova il massimo della funzione
\[
f(x_1, \dots, x_n) = \prod_{k=1}^n x_k
\]
sotto la condizione
\[
\sum_{k=1}^n x_k^2 = 1
\]
```

Trova il massimo della funzione

$$f(x_1, \dots, x_n) = \prod_{k=1}^n x_k$$

sotto la condizione

$$\sum_{k=1}^n x_k^2 = 1$$

Una frazione, anche complessa, si ottiene semplicemente con il comando

```
\frac{⟨numeratore⟩}{⟨denominatore⟩}
```

Per piccole quantità di materiale frazionario, a volte la forma n/m è più gradevole sulla pagina:

```
\[
\frac{1}{x^2+1} \quad x^{1/2}
\]
```

$$\frac{1}{x^2+1} \quad x^{1/2}$$

6.2.4 Limiti, derivate e integrali

Il comando

```
\lim_{⟨variabile⟩}\to⟨valore⟩
```

produce i limiti, e `\infty` produce ∞ .

```
\[
\lim_{x\to 0}
\frac{\sin x}{x}=1 \quad
\lim_{n\to +\infty} f_n=\delta
\]
```

$$\lim_{x \rightarrow 0} \frac{\sin x}{x} = 1 \quad \lim_{n \rightarrow +\infty} f_n = \delta$$

Le derivate si scrivono con il carattere `'`, che produce il segno di *primo*.

```
\[
y=x^2 \quad y'=2x \quad y''=2.
\]
```

$$y = x^2 \quad y' = 2x \quad y'' = 2.$$

Il comando `\int` produce il simbolo di integrale. Gli estremi di integrazione si scrivono come indici, e un indice formato da più di una lettera o una cifra va messo tra parentesi graffe.

```
\[
\int_a^T f(x)\,dx=
\int_0^T f(x)\,dx
\]
```

$$\int_a^{a+T} f(x) dx = \int_0^T f(x) dx$$

Come si può osservare, lo spazio sottile `\`, allontana dx da $f(x)$.

Per gli integrali multipli sono disponibili i comandi `\iint`, `\iiint`, `\iiiint` e `\idotsint`.

```
\[
\iint_D f(x,y)\,dx\,dy \quad
\iiint g\,dx\,dy\,dz
\]
```

$$\iint_D f(x,y) dx dy \quad \iiint g dx dy dz$$

Per gli integrali curvilinei lungo curve chiuse si usa `\oint`:

```
\[
\oint f(z)\,dz=2\pi i
\]
```

$$\oint f(z) dz = 2\pi i$$

Tabella 19: Simboli insiemistici (`\bigcup` e `\bigcap` sono operatori)

\subset	<code>\subset</code>	\supset	<code>\supset</code>	\cup	<code>\cup</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	\cap	<code>\cap</code>
\in	<code>\in</code>	\ni o \owns	<code>\ni</code> o <code>\owns</code>	\notin	<code>\notin</code>
\complement	<code>\complement</code>	\setminus	<code>\setminus</code>	\emptyset	<code>\emptyset</code>
\bigcup	<code>\bigcup</code>	\bigcap	<code>\bigcap</code>		

6.2.5 Insiemi numerici

I simboli degli insiemi numerici si ottengono con `\mathbb` (*blackboard bold*, “nero da lavagna”).

<pre>\[x\in\mathbb{R} \quad \text{\texttt{\code{quad}}} z\in\mathbb{C} \]</pre>	$x \in \mathbb{R} \quad z \in \mathbb{C}$
--	---

I simboli usati nell’esempio precedente sono raccolti insieme ad altri simboli insiemistici nella tabella 19.

Se si scrivono nel preambolo le definizioni seguenti (si veda il paragrafo 13.1 a pagina 245)

```
\newcommand{\numberset}{\mathbb}
\newcommand{\N}{\numberset{N}}
\newcommand{\R}{\numberset{R}}
```

per avere \mathbb{N} basta scrivere `\N`, e si può cambiare notazione con un’unica modifica.

6.2.6 Lettere greche

Le lettere greche minuscole e maiuscole si ottengono con i comandi elencati nella tabella 20 a fronte (per l’*omicron* minuscolo si usa il carattere latino *o*, e per le maiuscole che non vi compaiono si usano le corrispondenti maiuscole latine, identiche a quelle greche). Per le sei di esse prefissate con *var-* si noti quanto segue:

- `\varpi` e `\varsigma` non si usano praticamente mai;
- le altre quattro vanno usate in modo esclusivo: *o* la forma principale o la sua variante (ma si ricordi che a differenza di altri Paesi, in Europa si usa generalmente la seconda).

È per questo motivo che anche se nel font matematico in uso le diverse forme fossero uguali (come accade in questa guida) non ci sarebbe pericolo di confondersi o di commettere errori.

Diventa perciò conveniente ridefinire queste quattro varianti come caratteri normali, scrivendo nel preambolo (si veda il paragrafo 13.1 a pagina 245):

```
\renewcommand{\epsilon}{\varepsilon}
\renewcommand{\theta}{\vartheta}
\renewcommand{\rho}{\varrho}
\renewcommand{\phi}{\varphi}
```


Tabella 20: Lettere greche

α	<code>\alpha</code>	κ	<code>\kappa</code>	ς	<code>\varsigma</code>
β	<code>\beta</code>	λ	<code>\lambda</code>	τ	<code>\tau</code>
γ	<code>\gamma</code>	Λ	<code>\Lambda</code>	υ	<code>\upsilon</code>
Γ	<code>\Gamma</code>	μ	<code>\mu</code>	Υ	<code>\Upsilon</code>
δ	<code>\delta</code>	ν	<code>\nu</code>	ϕ	<code>\phi</code>
Δ	<code>\Delta</code>	ξ	<code>\xi</code>	Φ	<code>\Phi</code>
ϵ	<code>\epsilon</code>	Ξ	<code>\Xi</code>	φ	<code>\varphi</code>
ε	<code>\varepsilon</code>	π	<code>\pi</code>	χ	<code>\chi</code>
ζ	<code>\zeta</code>	Π	<code>\Pi</code>	ψ	<code>\psi</code>
η	<code>\eta</code>	ϖ	<code>\varpi</code>	Ψ	<code>\Psi</code>
θ	<code>\theta</code>	ρ	<code>\rho</code>	ω	<code>\omega</code>
Θ	<code>\Theta</code>	ϱ	<code>\varrho</code>	Ω	<code>\Omega</code>
ϑ	<code>\vartheta</code>	σ	<code>\sigma</code>		
ι	<code>\iota</code>	Σ	<code>\Sigma</code>		

6.2.7 Simboli che sormontano altri simboli

Il comando

```
\overset{⟨primo argomento⟩}{⟨secondo argomento⟩}
```

produce il simbolo indicato nel *⟨primo argomento⟩* rimpicciolito e sovrapposto a quello scritto nel *⟨secondo argomento⟩* (di solito un simbolo di relazione binaria), che rimane delle sue dimensioni e nella posizione abituale. Il comando `\underset` fa l'opposto.

Il simbolo

```
\[
\overset{R}{\sim}
\]
```

indica un'equivalenza.

Il simbolo

 $\overset{R}{\sim}$

indica un'equivalenza.

6.2.8 Barre e accenti

Il comando `\bar` pone un trattino sul proprio argomento: il simbolo \bar{x} indica un nome di variabile distinto da x (si veda la tabella 21 nella pagina seguente).

I comandi `\overline` e `\underline` (il secondo dei quali si usa piuttosto raramente) sopralineano e sottolineano rispettivamente tutto il proprio argomento: il simbolo \overline{x} indica un operatore applicato alla variabile x :

```
\bar{x} \quad \bar{X} \quad \overline{m+n}
```

 \bar{x}
 \bar{X}
 $\overline{m+n}$

I comandi `\vec` e `\overrightarrow` agiscono come `\bar` e `\overline`, ma producono frecce anziché barre orizzontali.

```
\vec{x} \quad \overrightarrow{AB}
```

 \vec{x}
 \overrightarrow{AB}

Esistono tre tipi di barra verticale, distinguibili dallo spazio richiesto prima e dopo:

Tabella 21: Accenti in modo matematico

\bar{a}	<code>\bar{a}</code>	\hat{a}	<code>\hat{a}</code>	\tilde{a}	<code>\tilde{a}</code>
\vec{a}	<code>\vec{a}</code>	\dot{a}	<code>\dot{a}</code>	\ddot{a}	<code>\ddot{a}</code>
\check{a}	<code>\check{a}</code>	\widehat{abc}	<code>\widehat{abc}</code>	\widetilde{abc}	<code>\widetilde{abc}</code>

- semplice | (ottenibile anche con `\vert`);
- delimitatore sinistro e destro (`\lvert` e `\rvert` rispettivamente);
- relazione binaria (`\mid`) per la divisibilità e il *tale che* negli insiemi.

`$F(x)|_{x=\gamma(t)}$ \quad`
`$\lvert x \rvert$ \quad`
`Se $p \mid n^2$, allora $p \mid n$.`

$F(x)|_{x=\gamma(t)} \quad |x|$
 Se $p \mid n^2$, allora $p \mid n$.

I comandi appena esaminati prevedono forme analoghe per barre verticali doppie: `\|` (o `\Vert`), `\lVert`, `\rVert` e `\parallel`.

La barra laterale è prevista talvolta nel calcolo degli integrali:

`\[`
`\int_a^b f(x)\,dx = F(x)\big|_a^b`
`\]`

$$\int_a^b f(x) dx = F(x) \Big|_a^b$$

La sua altezza va regolata a mano, premettendole uno dei comandi che verranno descritti nel paragrafo 6.4 a pagina 89.

La differenza tra due insiemi si realizza con il comando `\setminus`. Si confrontino le due scritture seguenti:

`$A\backslash B$ \quad`
`$A\setminus B$`

$A \backslash B$
 $A \setminus B$

Come si può osservare, il primo codice produce una spaziatura (leggermente) sbagliata.

Per il valore assoluto e la norma conviene caricare il pacchetto `mathtools` e definire nel preambolo due comandi ad hoc:

`\DeclarePairedDelimiter{\abs}{\lvert}{\rvert}`
`\DeclarePairedDelimiter{\norma}{\lVert}{\rVert}`

da usare nel modo seguente:

`\[`
`\abs{z} < 1 \quad`
`\norma{x} = \sqrt{x_1^2 + \dots + x_n^2}`
`\]`

$$|z| < 1 \quad \|x\| = \sqrt{x_1^2 + \dots + x_n^2}$$

Le varianti asterisco dei comandi appena definiti producono delimitatori ad altezza variabile:

`\[`
`\abs*{\sum_{i=1}^n x_i} \quad`
`\norma*{\frac{v}{\lambda}}`
`\]`

$$\left| \sum_{i=1}^n x_i \right| \quad \left\| \frac{v}{\lambda} \right\|$$

Per aggiungere alle variabili un accento matematico, come un cappello o una tilde, si possono usare i comandi della tabella 21. I comandi `\widehat` e `\widetilde` producono rispettivamente simboli di cappello e tilde che sormontano tutto il proprio argomento (di tre lettere al massimo).

Tabella 22: Frecce

\leftarrow	<code>\leftarrow</code> o <code>\gets</code>	\longleftarrow	<code>\longleftarrow</code>
\rightarrow	<code>\rightarrow</code> o <code>\to</code>	\longrightarrow	<code>\longrightarrow</code>
\leftrightarrow	<code>\leftrightarrow</code>	\longleftrightarrow	<code>\longleftrightarrow</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Longleftarrow	<code>\Longleftarrow</code>
\Rrightarrow	<code>\Rrightarrow</code>	\Longrightarrow	<code>\Longrightarrow</code>
\Leftrightarrow	<code>\Leftrightarrow</code>	\Longleftrightarrow	<code>\Longleftrightarrow</code>
\mapsto	<code>\mapsto</code>	\longmapsto	<code>\longmapsto</code>
\hookrightarrow	<code>\hookrightarrow</code>	\hookrightarrow	<code>\hookrightarrow</code>
\Uparrow	<code>\Uparrow</code>	\Uparrow	<code>\Uparrow</code>
\Downarrow	<code>\Downarrow</code>	\Downarrow	<code>\Downarrow</code>
\Updownarrow	<code>\Updownarrow</code>	\Updownarrow	<code>\Updownarrow</code>
\Lleftarrow	<code>\Lleftarrow</code>	\Rrightarrow	<code>\Rrightarrow</code>
\nearrow	<code>\nearrow</code>	\searrow	<code>\searrow</code>
\swarrow	<code>\swarrow</code>	\nwarrow	<code>\nwarrow</code>

6.2.9 Punti, frecce e simboli logici

In matematica esistono due tipi di due punti, distinguibili dal diverso spazio richiesto prima e dopo:

- semplice `:`, spaziato come in un'operazione binaria (divisione);
- `\colon`, spaziato come un segno di interpunzione.

Si confrontino le due scritture seguenti:

<code>\$f:\R\to\R\$ \\\</code>	$f : \mathbb{R} \rightarrow \mathbb{R}$
<code>\$f\colon\R\to\R\$</code>	$f : \mathbb{R} \rightarrow \mathbb{R}$

Come si può osservare, il primo codice produce una spaziatura (leggermente) sbagliata.

Per inserire punti ellittici in una formula si usa il comando `\dots`, che li mette automaticamente sulla linea di base del testo o li centra rispetto alla riga a seconda del contesto.

<code>\[</code>	
<code>x_1,\dots,x_n \quad \quad x_1+\dots+x_n</code>	$x_1, \dots, x_n \quad x_1 + \dots + x_n$
<code>\]</code>	

Per ulteriori esempi dell'uso di `\dots`, si rimanda al paragrafo 6.5 a pagina 90.

Oltre alla freccia semplice \rightarrow , che si ottiene con il comando `\to`, c'è anche quella con il trattino \mapsto , che si ottiene con `\mapsto`.

<code>\$f\colon\R\to\R\$ \\\</code>	$f : \mathbb{R} \rightarrow \mathbb{R}$
<code>\$x\mapsto x^2\$</code>	$x \mapsto x^2$

I comandi `\xleftarrow` e `\xrightarrow` producono frecce che si estendono automaticamente per accordarsi con indici di lunghezza non comune. Entrambi accettano un argomento facoltativo (il pedice) e uno obbligatorio (l'apice) che possono anche rimanere vuoti.

Tabella 23: Simboli logici

\vee	<code>\lor</code>	\wedge	<code>\land</code>	\neg	<code>\neg</code>
\exists	<code>\exists</code>	\nexists	<code>\nexists</code>	\forall	<code>\forall</code>
\Rightarrow	<code>\implies</code>	\Leftrightarrow	<code>\iff</code>	\models	<code>\models</code>

```
\[
\leftarrow{a} \quad \quad
\leftarrow[X]{a+b} \quad \quad
\rightarrow[X+Y+Z]{ }
\]
```

$$\xleftarrow{a} \quad \xleftarrow{\frac{a+b}{X}} \quad \xrightarrow{X+Y+Z}$$

La tabella 22 nella pagina precedente raccoglie i principali comandi per ottenere le frecce.

I simboli logici raccolti nella tabella 23 vanno usati *solo* in un lavoro sull'argomento (mentre in tutti gli altri contesti vanno senz'altro preferite le forme estese *se... allora*, *se e solo se*, eccetera).

6.2.10 Spazi in modo matematico

Può accadere, anche se di rado, che la spaziatura scelta da \LaTeX per le formule risulti insoddisfacente. Per modificarla si usano i comandi raccolti nella tabella 24 a fronte. Analogamente a quanto avviene in modo testuale (si veda il paragrafo 3.13), i comandi `\quad` (*quadrato*) e `\qquad` (*quadratone*) producono rispettivamente uno spazio di 1 e 2 em.

Spazi troppo stretti

Lo spazio sottile prodotto dal comando `\,` è molto utile in alcune formule. Si confronti

```
\[
\int_a^b f(x) dx \quad \sqrt{2} a \quad \sqrt{\log x}
\]
```

$$\int_a^b f(x) dx \quad \sqrt{2} a \quad \sqrt{\log x}$$

con

```
\[
\int_a^b f(x)\,,dx \quad \sqrt{2}\,,a \quad \sqrt{\log\,,x}
\]
```

$$\int_a^b f(x) dx \quad \sqrt{2} a \quad \sqrt{\log x}$$

Come si può osservare, il primo codice produce una spaziatura (leggermente) insufficiente.

Spazi troppo larghi

Il comando `\!` può migliorare le spaziature eccessive prodotte, per esempio, da simboli inclinati (barre di frazione, radicali) preceduti o seguiti da simboli particolari (esponenti, apici o operatori). Si confronti

```
\[
x^2/2 \quad a/\sin b
\]
```

$$x^2/2 \quad a/\sin b$$

Tabella 24: Spazi in modo matematico

Comando	Tipo di spazio
<code>\,</code>	Spazio sottile positivo
<code>\!</code>	Spazio sottile negativo
<code>\quad</code>	Spazio di un <i>quadrato</i>
<code>\qqquad</code>	Spazio di un <i>quadrato</i> ne

con

<code>\[</code> <code>x^2\!/2 \quad a\!\sin b</code> <code>\]</code>	$x^2/2 \quad a/\sin b$
--	------------------------

Come si può osservare, il primo codice produce una spaziatura (leggermente) esagerata.

Si ricorda che i due spazi appena descritti vanno usati per migliorare ulteriormente la qualità (già alta) del documento *se e solo se* la spaziatura predefinita non fosse adeguata ai simboli che compaiono nelle formule.

6.3 OPERATORI

6.3.1 Caratteristiche generali

In \LaTeX , le funzioni come `\sin`, `\cos` e `\log` presentano le seguenti caratteristiche:

- per essere più visibili sulla pagina (in accordo con le norme ISO-UNI) vengono rese in tondo normale e non in corsivo matematico come le variabili;
- richiedono una particolare spaziatura prima e dopo, che il programma inserisce automaticamente;
- i comandi che le producono, come `\sin`, `\cos` e `\log`, sono detti *operatori*.

Li si vede all’opera negli esempi seguenti:

<code>\[</code> <code>\cos2x \quad \log\log x \quad \log(x+y)</code> <code>\]</code>	$\cos 2x \quad \log \log x \quad \log(x + y)$
--	---

Si osservi che:

- nella prima formula, fra `\cos` e `2` c’è più spazio che fra `2` e `x`;
- nella seconda i tre elementi sono separati da uno spazio sottile;
- nella terza non c’è alcuno spazio tra `\log` e la parentesi.

Tabella 25: Operatori predefiniti

<code>\min</code>	<code>\max</code>	<code>\inf</code>	<code>\sup</code>	<code>\gcd</code>	<code>\arg</code>
<code>\sin</code>	<code>\cos</code>	<code>\tan</code>	<code>\cot</code>	<code>\sec</code>	<code>\csc</code>
<code>\sinh</code>	<code>\cosh</code>	<code>\tanh</code>	<code>\coth</code>	<code>\exp</code>	<code>\lim</code>
<code>\arcsin</code>	<code>\arccos</code>	<code>\arctan</code>	<code>\log</code>	<code>\lg</code>	<code>\ln</code>
<code>\liminf</code>	<code>\limsup</code>	<code>\deg</code>	<code>\det</code>	<code>\dim</code>	<code>\hom</code>
<code>\ker</code>	<code>\Pr</code>				

L'unico modo corretto di scriverli è quello appena mostrato: omettendo la barra rovescia si otterrebbe “ $\cos 2x$ ”, tutto in corsivo matematico e senza alcuna spaziatura, che in matematica non significa nulla. Soltanto scrivendo gli operatori come si è appena mostrato \LaTeX si comporta nel giusto modo e assegna loro font e spazi corretti.

I seguenti sono alcuni operatori predefiniti (la tabella 25 ne riporta l'elenco completo):

<code>\sin</code> x , <code>\exp</code> x , <code>\log</code> x , <code>\det</code> A , <code>\min_{x \in A}</code> $f(x)$	$\sin x$, $\exp x$, $\log x$, $\det A$, $\min_{x \in A} f(x)$
---	--

I due comandi `\bmod` e `\pmod` riguardano la relazione di congruenza modulo m :

<code>a\bmod b</code> <code>\qquad</code> <code>a\equiv b \pmod{m}</code>	$a \bmod b$ $a \equiv b \pmod{m}$
--	-----------------------------------

6.3.2 Definire nuovi operatori

Le pubblicazioni specialistiche introducono continuamente nuove funzioni che devono poter essere definite, non essendo contemplate né da \LaTeX né dai pacchetti dedicati. Risolve il problema il comando `\DeclareMathOperator`.

Per esempio, per definire la funzione matematica “sgn” che denoti il segno di un numero reale (funzione non prevista né da \LaTeX né da `amsmath`), si scrive nel preambolo

```
\DeclareMathOperator{\sgn}{sgn}
```

Nel documento, poi, si darà `\sgn` per ottenere “sgn” nel font corretto e adeguatamente spaziato su entrambi i lati.

L'operatore che denota la parte reale di un numero complesso è `\Re`, che produce il simbolo \Re . Lo si può ridefinire perché produca il simbolo in tondo anziché in gotico dando nel preambolo i comandi

```
\DeclareMathOperator{\Realpart}{Re}  
\renewcommand{\Re}{\Realpart}
```

Se il nuovo operatore presenta indici scritti come nei limiti (lim, sup o max), si usa la variante asterisco di `\DeclareMathOperator`:

```
\DeclareMathOperator*{\argmax}{arg\,max}
```

6.4 PARENTESI

\LaTeX e amsmath definiscono numerosi simboli per parentesi e altri delimitatori. Le parentesi tonde e quadre si scrivono con i corrispondenti caratteri da tastiera, mentre quelle graffe *anche in modo matematico* devono essere precedute da \backslash . Tutti gli altri delimitatori vengono generati da comandi dedicati.

$\backslash[\{a,b,c\} \backslash ne \{a,b,c\} \backslash]$	$a,b,c \neq \{a,b,c\}$
---	------------------------

Talvolta bisogna aggiustarne a mano le dimensioni: lo si può fare prefissandoli con i comandi $\backslash big$, $\backslash Big$, $\backslash bigg$ e $\backslash Bigg$. I comandi $\backslash bigl$ (*big left*) e $\backslash bigr$ (*big right*) ingrandiscono lievemente le parentesi:

$\backslash[\backslash bigl ((x-y)+(x+y) \backslash bigr) \backslash]$	$((x-y) + (x+y))$
---	-------------------

I comandi $\backslash Bigl$ e $\backslash Bigr$ producono parentesi ancora più grandi:

$\backslash[\backslash Bigl (1+\backslash frac{1}{n} \backslash Bigr)^n \backslash]$	$\left(1 + \frac{1}{n}\right)^n$
--	----------------------------------

I comandi $\backslash biggl$ e $\backslash biggr$ ne generano di più grandi ancora:

$\backslash[\backslash biggl (\backslash sum_n x_n^2 \backslash biggr)^{\{1/2\}} \backslash]$	$\left(\sum_n x_n^2\right)^{1/2}$
--	-----------------------------------

Se non basta, ci sono anche $\backslash Biggl$ e $\backslash Biggr$. Si noti che alcuni visualizzatori di PDF potrebbero mostrare non correttamente *a schermo* delimitatori particolarmente grandi. Il problema non si presenta, invece, a stampa.

\LaTeX può determinare le dimensioni dei delimitatori anche automaticamente, premettendo $\backslash left$ a quello di apertura e $\backslash right$ al corrispondente delimitatore di chiusura (se si vuole aprire l'espressione e non chiuderla, si userà $\backslash left$ con il punto finale, e $\backslash right$ in caso contrario). Questi comandi, che funzionano solo se usati in coppia e sulla stessa riga, quasi sempre inseriscono spaziature indesiderate e parentesi più grandi del necessario: perciò va preferito loro il metodo manuale appena descritto.

I comandi $\backslash overbrace$ e $\backslash underbrace$ creano lunghe graffe orizzontali sopra o sotto un'espressione:

$\backslash[\backslash underbrace{1+2+\backslash dots+n}_{\{ \}} = \backslash frac{n(n+1)}{2} + (n+1) \backslash]$	$\underbrace{1+2+\cdots+n}_{=\frac{n(n+1)}{2}} + (n+1)$
---	---

Per scrivere coefficienti binomiali si usa il comando $\backslash binom$:

```
\[
(a+b)^n=
\sum_{\substack{k\in\mathbb{N} \\ 0\leq k\leq n}}
\binom{n}{k}a^{n-k} b^k
\]
```

$$(a+b)^n = \sum_{\substack{k \in \mathbb{N} \\ 0 \leq k \leq n}} \binom{n}{k} a^{n-k} b^k$$

Il comando

```
\substack{\langle sopra \rangle \langle sotto \rangle}
```

produce un indice su più righe.

Per i sistemi di equazioni si può usare l'ambiente `cases` (si veda il paragrafo 6.7.4 a pagina 94):

```
\[
\begin{cases}
x+y=2 \\
x-y=0
\end{cases}
\]
```

$$\begin{cases} x+y=2 \\ x-y=0 \end{cases}$$

Per gli insiemi è utile il pacchetto `braket`, che definisce l'apposito comando `\Set`:

```
\[
\Set{\frac{1}{n^2} \mid n \in \mathbb{N}}
\]
```

$$\left\{ \frac{1}{n^2} \mid n \in \mathbb{N} \right\}$$

e, per le parentesi angolate, `\Bra`, `\Ket` e `\Braket`:

```
\[
\Bra{\psi_n} \quad \Ket{\psi}
\quad c_n = \Braket{\psi_n \mid \psi}
\]
```

$$\langle \psi_n | \quad | \psi \rangle \quad c_n = \langle \psi_n | \psi \rangle$$

6.5 VETTORI E MATRICI

I vettori si scrivono di solito in tondo nero (corsivo, secondo le norme ISO-UNI) oppure in semplice corsivo matematico; talvolta, soprattutto nei testi di fisica, sono sormontati da una freccia. Nel primo caso si può usare il comando `\mathbf`; nel secondo il comando `\boldsymbol`; nel terzo il comando `\vec`. Può essere conveniente ridefinire nel preambolo quest'ultimo comando (si veda il paragrafo 13.1 a pagina 245):

```
\renewcommand{\vec}{\boldsymbol}
```

In questo modo basta scrivere `\vec{v}` per ottenere \boldsymbol{v} e si può cambiare notazione con un'unica modifica (si veda anche il paragrafo 6.8 a pagina 95).

Le matrici si scrivono negli ambienti `pmatrix`, `bmatrix`, `Bmatrix`, `vmatrix` e `Vmatrix`, che hanno come delimitatori rispettivamente parentesi tonde, quadre (*braces*), graffe (*curly braces*), barre verticali e doppie barre verticali. C'è anche l'ambiente `matrix` senza delimitatori.

Gli elementi della matrice vengono centrati automaticamente, e righe e colonne si scrivono come una normale tabella `tabular`, ricordando che gli spazi espliciti sono ignorati.

```
\[
\begin{pmatrix}
1 & 2 \\
3 & 4
\end{pmatrix}
\end{pmatrix}
\]
```

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
\[
\begin{bmatrix}
1 & 2 \\
3 & 4
\end{bmatrix}
\end{bmatrix}
\]
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Si possono scrivere matrici con punti ellittici, come nell'esempio seguente:

```
\[
A=
\begin{bmatrix}
x_{11} & x_{12} & \dots \\
x_{21} & x_{22} & \dots \\
\vdots & \vdots & \ddots
\end{bmatrix}
\end{bmatrix}
\]
```

$$A = \begin{bmatrix} x_{11} & x_{12} & \dots \\ x_{21} & x_{22} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Come si può osservare, il comando `\vdots` produce tre punti ellittici verticali e `\ddots` tre in diagonale.

Il comando

```
\hdotsfor{<n>}
```

riempie di punti la riga della matrice per $\langle n \rangle$ colonne:

```
\[
\begin{bmatrix}
a_{11} & \dots & a_{1n} \\
a_{21} & \dots & a_{2n} \\
\hdotsfor{3} \\
a_{n1} & \dots & a_{nn}
\end{bmatrix}
\end{bmatrix}
\]
```

$$\begin{bmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$$

Una piccola matrice in linea si scrive nell'ambiente `smallmatrix`:

```
Sia $A=\bigl(
\begin{smallmatrix}
a & b \\
c & d
\end{smallmatrix}
\bigr)$
una matrice invertibile.
```

Sia $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ una matrice invertibile.

In questo caso le parentesi vanno aggiunte a mano.

La tabella 26 nella pagina successiva raccoglie altri simboli matematici di uso comune. Per un loro elenco completo si veda [Pakin, 2015].

Tabella 26: Simboli misti (`\bigodot`, `\bigoplus` e `\bigotimes` sono operatori)

\vee	<code>\vee</code>	\wedge	<code>\wedge</code>	\div	<code>\div</code>
\Re	<code>\Re</code>	\Im	<code>\Im</code>	\aleph	<code>\aleph</code>
∂	<code>\partial</code>	∇	<code>\nabla</code>	\cdot	<code>\cdot</code>
\hbar	<code>\hbar</code>	\hslash	<code>\hslash</code>	\circ	<code>\circ</code>
\imath	<code>\imath</code>	\jmath	<code>\jmath</code>	\bullet	<code>\bullet</code>
ℓ	<code>\ell</code>	\wp	<code>\wp</code>	$\sqrt{}$	<code>\sqrt</code>
\dagger	<code>\dagger</code>	\ddagger	<code>\ddagger</code>	$*$	<code>*</code>
\vdash	<code>\vdash</code>	\dashv	<code>\dashv</code>	\angle	<code>\angle</code>
\triangleleft	<code>\triangleleft</code>	\triangleright	<code>\triangleright</code>	\triangle	<code>\triangle</code>
\square	<code>\square</code>	\blacksquare	<code>\blacksquare</code>	\diamond	<code>\diamond</code>
\odot	<code>\odot</code>	\oplus	<code>\oplus</code>	\otimes	<code>\otimes</code>
\bigodot	<code>\bigodot</code>	\bigoplus	<code>\bigoplus</code>	\bigotimes	<code>\bigotimes</code>

6.6 SPEZZARE FORMULE LUNGHE

L^AT_EX non spezza automaticamente una formula più lunga d'una riga. Solo chi l'ha scritta, infatti, ne conosce il ritmo di lettura e sa dov'è più opportuno andare a capo e se allinearne o meno le varie parti. In generale, tuttavia, vale la regola per cui si può andare a capo:

- dopo i simboli di relazione e, subordinatamente, dopo i simboli di relazione e di operazione binaria (si veda la tabella 27 a fronte) nelle formule in linea;
- prima dei simboli di relazione e dopo i simboli binari nelle formule in display;
- *mai* dopo gli operatori funzionali (si veda la tabella 25 a pagina 88), i grandi operatori e i delimitatori di apertura.

Per spezzare e raggruppare formule in display, il pacchetto `amsmath` definisce (fra gli altri) gli ambienti `multline`, `split`, `gather` e `align`, che si descrivono di seguito.

6.6.1 Spezzare formule senza incolonnarle

Per spezzare una formula in più righe non incolonnate si usa l'ambiente `multline`.

```
\begin{multline}
f=a+b+c \\
+i+j+k+l \\
+x+y+z \\
\end{multline}
```

$$\begin{aligned}
 f &= a + b + c \\
 &\quad + i + j + k + l \\
 &\quad + x + y + z \quad (6.3)
 \end{aligned}$$

Si noti che:

- la prima riga viene allineata a sinistra e l'ultima a destra;
- le rimanenti vengono centrate (a meno che non sia attiva l'opzione `fleqn`, che allinea comunque le formule a sinistra rispetto a un margine rientrato);

Tabella 27: Simboli di relazione

\leq	<code>\le</code>	\geq	<code>\ge</code>	\sim	<code>\sim</code>
\ll	<code>\ll</code>	\gg	<code>\gg</code>	\simeq	<code>\simeq</code>
\prec	<code>\prec</code>	\succ	<code>\succ</code>	\approx	<code>\approx</code>
\preceq	<code>\preceq</code>	\succeq	<code>\succeq</code>	\asymp	<code>\asymp</code>
\parallel	<code>\parallel</code>	\perp	<code>\perp</code>	\cong	<code>\cong</code>
\mid	<code>\mid</code>	\propto	<code>\propto</code>	\equiv	<code>\equiv</code>
\neq	<code>\ne</code>				

- il numero progressivo della formula viene messo nel margine destro in corrispondenza dell'ultima riga.

La variante asterisco `multline*` produce formule dello stesso tipo *non numerate*.

6.6.2 Spezzare formule incolonnandole

Per spezzare una formula in più righe incolonnate si usa l'ambiente `split`.

<code>\begin{equation}</code>	
<code>\begin{split}</code>	
<code>a &= b+c-d \\</code>	$a = b + c - d$
<code>&= e-f \\</code>	$= e - f$
<code>&= g</code>	$= g$
<code>\end{split}</code>	
<code>\end{equation}</code>	

(6.4)

Come si può osservare:

- il carattere `&` incolonna le righe della formula a partire dal punto in cui viene dato (di solito subito prima di un `=`);
- `split` va *necessariamente* usato dentro un altro ambiente per la matematica in display, responsabile della numerazione della formula (ma si può ottenere una formula di questo tipo anche non numerata).

6.7 RAGGRUPPARE PIÙ FORMULE

Per raggruppare più formule in display, il pacchetto `amsmath` definisce (fra gli altri) gli ambienti `gather` e `align`, descritti di seguito.

6.7.1 Raggruppare formule senza incolonnarle

L'ambiente `gather` raggruppa più formule centrandole e numerando autonomamente ciascuna su una riga a sé e, se necessario, assegnandole un'etichetta.

<code>\begin{gather}</code>	
<code>a=b+c \\</code>	$a = b + c$
<code>V+F-S=2</code>	$V + F - S = 2$
<code>\end{gather}</code>	

(6.5)

(6.6)

Si noti che si avrebbe un risultato “simile” scrivendo ogni formula in un ambiente `equation`, ma poi lo spazio tra di esse sarebbe esagerato. La variante asterisco `gather*` produce formule dello stesso tipo non numerate.

6.7.2 Raggruppare formule incolonnandole

L’ambiente `align` incolonna gruppi di due o più formule mettendo e numerando ciascuna su una riga a sé, come mostra l’esempio seguente:

<pre>\begin{align} a &= b+c+d \\ e &= f \notag \\ x-1 &= y+z \\ \end{align}</pre>	$a = b + c + d \quad (6.7)$ $e = f$ $x - 1 = y + z \quad (6.8)$
---	---

La variante asterisco `align*` produce formule dello stesso tipo non numerate. (Si ottiene lo stesso risultato dando `\notag` alla fine della formula interessata).

L’ambiente è utile anche per incolonnare più righe di formule autonome. In tal caso, `&` assume due significati diversi a seconda della posizione in cui si trova sulla riga:

- se occupa un posto dispari, indica a \LaTeX i punti da incolonnare;
- se occupa un posto pari, è un separatore come in `tabular`.

<pre>\begin{align} a &= b & c &= d & e &= f \\ u &= v & w &= x & y &= z \\ \end{align}</pre>	$a = b \quad c = d \quad e = f \quad (6.9)$ $u = v \quad w = x \quad y = z \quad (6.10)$
--	--

6.7.3 Raggruppare formule con una parentesi

Per raggruppare più formule con una parentesi, magari da integrare con del testo esplicativo, \LaTeX definisce i due ambienti `gathered` e `aligned`, che trattano le espressioni al proprio interno come i due ambienti corrispondenti appena esaminati.

<pre>\[\left. \begin{aligned} a &= b+1 \\ c &= d \end{aligned} \right\} \text{due equazioni}</pre>	$\left. \begin{array}{l} a = b + 1 \\ c = d \end{array} \right\} \text{ due equazioni}$
---	---

6.7.4 Casi e sottonumerazioni

L’ambiente `cases` serve per le definizioni fatte per casi. Graffa e allineamento sono automatici; il testo nella seconda colonna va nell’argomento di `\text`.

Tabella 28: Stili dei font matematici (`\mathscr` richiede il pacchetto `mathrsfs`). Si noti che alcuni di essi non hanno effetto su minuscole e numeri.

Stile	Codice	Risultato
Tondo	<code>\mathrm{ABCdef123}</code>	ABCdef123
Corsivo	<code>\mathit{ABCdef123}</code>	<i>ABCdef123</i>
Nero	<code>\mathbf{ABCdef123}</code>	ABCdef123
Dattilografico	<code>\mathtt{ABCdef123}</code>	ABCdef123
Senza grazie	<code>\mathsf{ABCdef123}</code>	ABCdef123
Gotico	<code>\mathfrak{ABCdef123}</code>	$\mathfrak{ABCdef123}$
Nero da lavagna	<code>\mathbb{ABC}</code>	\mathbb{ABC}
Calligrafico	<code>\mathcal{ABC}</code>	\mathcal{ABC}
Manoscritto	<code>\mathscr{ABC}</code>	\mathscr{ABC}

```
\[
n!=
\begin{cases}
1 & \& \text{se } n=0 \\
n(n-1)! & \& \text{se } n \geq 1
\end{cases}
\]
```

$$n! = \begin{cases} 1 & \text{se } n = 0 \\ n(n-1)! & \text{se } n \geq 1 \end{cases}$$

L'ambiente `subequations` produce sottonumerazioni:

```
\begin{subequations}
\label{eqn:schema}
\begin{align}
a &= b+c \\
c &= d \label{eqn:sub} \\
e &= f+g
\end{align}
\end{subequations}
Le formule~\eqref{eqn:schema},
e in particolare
la~\eqref{eqn:sub}, \dots
```

$$a = b + c \quad (6.11a)$$

$$c = d \quad (6.11b)$$

$$e = f + g \quad (6.11c)$$

Le formule (6.11), e in particolare la (6.11b), ...

6.8 MODIFICARE STILE E CORPO DEL FONT

In modo matematico, \LaTeX armonizza stile e corpo del font con il contesto in cui le formule si trovano. A volte, però, può essere necessario modificare a mano questi due parametri: nei prossimi paragrafi si spiega come farlo.

Modificare lo stile

Gli esempi seguenti mostrano come i comandi per cambiare lo stile del font (raccolti anche nella tabella 28) agiscano su lettere e numeri, ma *non* sui segni di operazione.

<code>\$x+y+2^n M\cos t\$ \\</code>	$x + y + 2^n M \cos t$
<code>\$\mathit{x+y+2^n M\cos t}\$ \\</code>	$x + y + 2^n M \cos t$
<code>\$\mathbf{x+y+2^n M\cos t}\$ \\</code>	$\mathbf{x + y + 2^n M \cos t}$
<code>\$\mathrm{x+y+2^n M\cos t}\$ \\</code>	$x + y + 2^n M \cos t$
<code>\$\mathtt{x+y+2^n M\cos t}\$ \\</code>	$x + y + 2^n M \cos t$
<code>\$\mathsf{x+y+2^n M\cos t}\$ \\</code>	$x + y + 2^n M \cos t$

Spesso i principianti abusano dei simboli in nero, tipograficamente piuttosto “pesanti”. Questi si ottengono con il comando `\mathbf`, il cui comportamento dipende in larga parte dalla serie di font matematici in uso e di solito ha effetto solo sulle lettere. Si osservi l’esempio seguente:

<code>\[</code> <code>\mu, M \quad</code> <code>\mathbf{\mu}, \mathbf{M}</code> <code>\]</code>	$\mu, M \quad \mathbf{\mu}, \mathbf{M}$
--	---

Come si può osservare, il comando non ha effetto sulla lettera μ e produce lettere in tondo anziché in corsivo matematico.

Per comporre simboli matematici in nero corsivo si consiglia il comando `\boldsymbol`.

<code>\[</code> <code>\mu, M \quad</code> <code>\boldsymbol{\mu}, \boldsymbol{M}</code> <code>\]</code>	$\mu, M \quad \boldsymbol{\mu}, \boldsymbol{M}$
--	---

Gli indici letterali vanno scritti in corsivo matematico se rappresentano quantità variabili (cioè se sono *simboli*), in tondo se rappresentano apposizioni di una grandezza fisica (cioè se sono semplice *testo*). In quest’ultimo caso si usa il comando `\textup`.

<code>\[</code> <code>V_{\textup{eff}} \quad</code> <code>\psi_{\textup{incidente}}</code> <code>\]</code>	$V_{\textup{eff}} \quad \psi_{\textup{incidente}}$
---	--

Si confrontino le due scritture seguenti:

<code>\$V_{\textup{eff}}\$ \\</code>	$V_{\textup{eff}}$
<code>\$V_{\text{eff}}\$ \\</code>	V_{eff}

Come si può osservare, \LaTeX interpreta il pedice nel primo codice come tre variabili da moltiplicare, rendendole in corsivo matematico e spaziandole di conseguenza. Il codice corretto è il secondo.

Modificare il corpo

In modo matematico si possono impostare a mano la dimensione del font servendosi delle dichiarazioni `\displaystyle`, `\textstyle`, `\scriptstyle` e `\scriptscriptstyle`. Si considerino gli esempi seguenti:

```
\[
\sum_{k=0}^n z^k \quad
\textstyle\sum_{k=0}^n z^k
\]
$\displaystyle\sum_{k=0}^n z^k$
\quad $\sum_{k=0}^n z^k$ \quad
$\scriptstyle\sum_{k=0}^n z^k$
```

$$\sum_{k=0}^n z^k \quad \sum_{k=0}^n z^k \quad \sum_{k=0}^n z^k$$

```
\[
x_G=
\frac{\displaystyle
\sum_{i=1}^n m_i x_i}
{\displaystyle\sum_{i=1}^n m_i}
\]
```

$$x_G = \frac{\sum_{i=1}^n m_i x_i}{\sum_{i=1}^n m_i}$$

Come si può osservare, il cambiamento del corpo influisce anche sul modo in cui vengono resi gli indici.

Si possono usare i due comandi `\dfrac` e `\tfrac` come abbreviazioni di `{\displaystyle\frac{...}}` e `{\textstyle\frac{...}}`:

```
$\frac{1}{n}\log x$ \quad
$\dfrac{1}{n}\log x$
\[
\frac{1}{n}\log x \quad
\tfrac{1}{n}\log x
\]
```

$$\frac{1}{n} \log x \quad \frac{1}{n} \log x$$

$$\frac{1}{n} \log x \quad \frac{1}{n} \log x$$

6.9 ENUNCIATI E DIMOSTRAZIONI

Di qui in avanti si dà per caricato *anche* il pacchetto `amsthm`.

6.9.1 Enunciati

Nella scrittura della matematica è utile poter disporre di un metodo per introdurre e numerare definizioni, teoremi e strutture simili. I tipi di enunciato non sono predefiniti, ma vanno dichiarati dall'utente, chiamato a prendere alcune decisioni globali:

- stabilire il tipo di enunciato da inserire (definizioni e teoremi, per esempio);
- assegnare un nome a ogni ambiente (definizione e teorema, per esempio; non si può usare `def`, perché è già un comando di base di \LaTeX);
- titolare gli enunciati (con *Definizione* e *Teorema*, per esempio).

Il comando `\newtheorem`, dato *nel preambolo*, permette di fare le relative dichiarazioni globali e prevede due forme di definizione:

```
\newtheorem{<nome dell'enunciato>}{<titolo>}[<sezione>]
```

oppure, in alternativa,

```
\newtheorem{<nome dell'enunciato>}{<numerato come>}{<titolo>}
```

dove:

- $\langle \text{nome dell'enunciato} \rangle$ è una parola chiave che identifica l'enunciato;
- $\langle \text{titolo} \rangle$ specifica il titolo dell'enunciato che comparirà nel documento;
- $\langle \text{sezione} \rangle$ specifica il livello di sezionamento (chapter o section, di regola) a cui collegare la numerazione dell'enunciato;
- in $\langle \text{numerato come} \rangle$ si scrive il nome di un enunciato dichiarato in precedenza, in modo che quello nuovo ne prosegua la numerazione.

La variante asterisco `\newtheorem*` produce enunciati non numerati.

Il testo dell'enunciato va messo nel corrispondente ambiente, e un'eventuale specificazione (fra parentesi tonde, nel documento finito) si scrive nell'argomento facoltativo immediatamente *dopo* il comando d'apertura, così:

```
\begin{⟨nome dell'enunciato⟩}[⟨eventuale specificazione⟩]
...
\end{⟨nome dell'enunciato⟩}
```

Il pacchetto `amsthm` mette a disposizione tre stili predefiniti per gli enunciati (`plain`, `definition` e `remark`) i cui dettagli tipografici dipendono dalla classe di documento in uso, anche se in linea di massima il primo dei tre produce il proprio contenuto in corsivo, mentre gli altri due lo lasciano in tondo. Di seguito si riportano le principali categorie di enunciato, ciascuna associata al proprio stile più tipico:

plain Per teoremi, lemmi, corollari, proposizioni, congetture, criteri, leggi e algoritmi.

definition Per definizioni, condizioni, problemi ed esempi.

remark Per osservazioni e annotazioni.

Si possono scrivere gli enunciati con stili diversi dal predefinito `plain` dividendoli in gruppi per tipo di enunciato e premettendo a ciascun gruppo il comando `\theoremstyle`. (Si può personalizzare lo stile dell'enunciato con il comando `\newtheoremstyle`.)

A questo punto le nozioni teoriche dovrebbero bastare. Alcuni esempi mostreranno quanto si è appena esaminato. Scrivendo nel preambolo

```
\theoremstyle{definition}
\newtheorem{definizione}{Definizione}
```

e

```
\theoremstyle{plain}
\newtheorem{teorema}{Teorema}
```

gli ambienti `definizione` e `teorema` si usano così:

```
\begin{definizione}[di Gauss]
Si dice \emph{matematico} colui
per il quale è ovvio che
 $\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$ .
\end{definizione}
\begin{teorema}
I matematici, se ce ne sono,
sono molto rari.
\end{teorema}
```

Definizione 1 (di Gauss). Si dice *matematico* colui per il quale è ovvio che $\int_{-\infty}^{+\infty} e^{-x^2} dx = \sqrt{\pi}$.

Teorema 1. I matematici, se ce ne sono, sono molto rari.

Il seguente teorema è a tutti ben noto.
`\begin{teorema}[di Pitagora]`
 In un triangolo rettangolo, la somma dei quadrati costruiti sui cateti è uguale al quadrato costruito sull'ipotenusa.
`\end{teorema}`
 La dimostrazione è lasciata per esercizio.

Il seguente teorema è a tutti ben noto.

Teorema 2 (di Pitagora). *In un triangolo rettangolo, la somma dei quadrati costruiti sui cateti è uguale al quadrato costruito sull'ipotenusa.*

La dimostrazione è lasciata per esercizio.

Come si può osservare, \LaTeX :

- produce etichetta e numero (automatico) dell'enunciato in tondo nero e la conclude con un punto fermo;
- separa ogni enunciato dal resto del testo senza rientrarlo;
- mette le definizioni in tondo e i teoremi in corsivo.

L'esempio seguente riguarda la numerazione di tre enunciati consecutivi, il primo e il terzo dei quali sono dello stesso tipo, ma il secondo no. Scrivendo nel preambolo

```
\theoremstyle{plain}
\newtheorem{legge}{Legge}
\newtheorem{decreto}[legge]{Decreto}
```

gli ambienti `legge` e `decreto` si usano come segue:

```
\begin{legge}
\label{lex:capo}
Il capo ha ragione.
\end{legge}
\begin{decreto}[Aggiornamento
alla legge~\ref{lex:capo}]
Il capo ha \emph{sempre} ragione.
\end{decreto}
\begin{legge}
Se il capo ha torto, vedere la
legge~\ref{lex:capo}.
\end{legge}
```

Legge 1. *Il capo ha ragione.*

Decreto 2 (Aggiornamento alla legge 1). *Il capo ha sempre ragione.*

Legge 3. *Se il capo ha torto, vedere la legge 1.*

Si osservi che il numero assegnato a *Decreto* prosegue la numerazione di *Legge* anziché cominciarne una nuova, perché l'argomento facoltativo [`legge`] nella definizione del teorema assegna entrambi gli enunciati allo stesso contatore. Come di consueto, `\label` permette riferimenti incrociati anche a enunciati matematici.

Se si desidera introdurre un enunciato *Murphy*, per esempio, la cui numerazione sia collegata al paragrafo corrente, basta specificare nell'argomento facoltativo l'opzione `section`, in questo modo:

```
\newtheorem{murphy}{Murphy}[section]
```

L'ambiente `murphy` così definito si usa come al solito:

```
\begin{murphy}
Se qualcosa può andar male,
lo farà.
\end{murphy}
```

Murphy 6.9.1. *Se qualcosa può andar male, lo farà.*

6.g.2 Dimostrazioni

L'ambiente `proof` permette di scrivere una dimostrazione, che nel documento finito sarà chiusa da un quadratino. Si osservi l'esempio seguente:

```
\begin{teorema}[di Euclide]
I numeri primi sono infiniti.
\end{teorema}
\begin{proof}
Per assurdo, sia dato l'elenco
finito di tutti i primi. Siano
 $q$  il loro prodotto e  $p$  un
primo che divide  $q+1$ . Allora
 $p$  non sta nell'elenco, perché
altrimenti dividerebbe  $1$ .
\end{proof}
```

Teorema 3 (di Euclide). *I numeri primi sono infiniti.*

Dimostrazione. Per assurdo, sia dato l'elenco finito di tutti i primi. Siano q il loro prodotto e p un primo che divide $q+1$. Allora p non sta nell'elenco, perché altrimenti dividerebbe 1 . \square

Per sostituire la scritta *Dimostrazione* con un'altra, per esempio *Soluzione*, basta scrivere

```
\begin{proof}[Soluzione]
```

Il comando `\qedhere` colloca correttamente il simbolo di fine dimostrazione anche se questa termina con una formula in display. Si confronti

```
\begin{proof}
Basta usare la formula
\[
E=mc^2
\]
\end{proof}
```

Dimostrazione. Basta usare la formula

$$E = mc^2$$

\square

con

```
\begin{proof}
Basta usare la formula
\[
E=mc^2\qedhere
\]
\end{proof}
```

Dimostrazione. Basta usare la formula

$$E = mc^2$$

\square

Come si può osservare, nel primo esempio il quadratino è collocato in modo (leggermente) sbagliato: la scrittura corretta è la seconda.

6.10 ALTRE SCIENZE

Le unità di misura del Sistema Internazionale s'inseriscono con i comandi del pacchetto `siunitx` (se ne veda la documentazione), che permette di regolarne molto finemente il formato e di cambiare il risultato nel documento finito operando un'unica modifica nel preambolo anziché agire a mano su ciascuna unità di misura.

Dato che le convenzioni tipografiche italiane prevedono la virgola e non il punto (predefinito dal pacchetto) come separatore decimale, il pacchetto va caricato almeno con l'opzione seguente:

```
\usepackage[output-decimal-marker={,}]{siunitx}
```

Ecco qualche esempio:

<code>\SI{23.4}{kg.m.s^{-2}} \\\</code>	$23,4 \text{ kg m s}^{-2}$
<code>\$r=\SI{0.8768(11)e-15}{m}\$ \\\</code>	$r = 0,8768(11) \times 10^{-15} \text{ m}$
<code>\si{\joule\per\mole\per\kelvin}\\\</code>	$\text{J mol}^{-1} \text{ K}^{-1}$
<code>\si{j.mol^{-1}.K^{-1}}</code>	$\text{J mol}^{-1} \text{ K}^{-1}$

<code>\SI{100}{\celsius} \\\</code>	100°C
<code>\ang{1;2;3}</code>	$1^{\circ}2'3''$

Tutti i precedenti comandi funzionano sia in modo testuale sia in modo matematico. Si tenga presente, tuttavia, che se le unità di misura da inserire sono poche, si possono scrivere più semplicemente così:

<code>L'auto viaggia a ~\$65\$\,km/h.</code>	L'auto viaggia a 65 km/h.
--	---------------------------

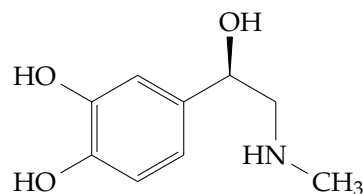
Se si prevede di inserire i simboli dei gradi Celsius o di quelli sessagesimali, si consiglia di caricare *anche* il pacchetto `textcomp`, che abilita la ricerca di $^{\circ}$ nel documento finito.

Per comporre semplici formule chimiche è utile il pacchetto `chemformula`.

<code>\ch{H2O} \quad \quad</code>	H_2O
<code>\ch{^{227}_{90}\text{Th}+} \quad \quad</code>	$^{227}_{90}\text{Th}^{+}$
<code>\ch{C6H5-CHO} \quad \quad \ll[2ex]</code>	$\text{C}_6\text{H}_5\text{-CHO}$
<code>\ch{SO4^{2-} + Ba^{2+} -> BaSO4 v}</code>	$\text{SO}_4^{2-} + \text{Ba}^{2+} \longrightarrow \text{BaSO}_4\downarrow$

Il pacchetto `chemfig` compone strutture molecolari, formule di reazione e tutta la chimica di cui si può aver bisogno.

```
\chemname{\chemfig{*6((-HO)-=%
-(-(<[:60]OH)-[:60]-%
[:60,,,2]HN-[:60]CH_3)=%
-(-HO=))}}{Adrenalina}
```



Adrenalina

Chi si occupa di informatica ha bisogno delle parentesi angolate $\langle \dots \rangle$, che si ottengono con i comandi `\langle` e `\rangle` in modo matematico.

Per la scrittura di tutte le altre discipline scientifiche si rimanda al catalogo tematico dei pacchetti disponibile su [TEXCATALOGUE](#).

7 | CODICI

Questo capitolo presenta il pacchetto `listings`, che permette di inserire codici all'interno di un documento, con un'elevata flessibilità.

7.1 INTRODUZIONE

Ci sono due tipi di codici:

UN CODICE IN LINEA è un frammento di codice scritto in linea con il testo, come per esempio `var i : integer`.

UN CODICE IN DISPLAY è formato da uno o più capoversi staccati dal testo precedente e seguente mediante spazi di ampiezza adeguata.

Per inserire codici in linea e in display \LaTeX definisce rispettivamente il comando `\verb` e l'ambiente `verbatim`, che stampano il codice alla lettera, come se fosse battuto a macchina, con tutti gli spazi e le interruzioni di riga, senza che vengano interpretati i comandi e i caratteri speciali. Il pacchetto `listings` fornisce invece il comando `\lstinline` (per i codici in linea) e l'ambiente `lstlisting` (per i codici in display).

Codici in linea

Per esempio, il frammento di codice Pascal `var i : integer` si ottiene con l'istruzione `\lstinline!var i : integer!`. Come delimitatore, al posto del simbolo `!`, si può usare qualsiasi carattere tranne le lettere o lo spazio: `\lstinline$var i : integer$` dà lo stesso risultato.

Codici in display

Il seguente è un esempio di codice in display, scritto in Pascal, ottenuto inserendo il codice all'interno dell'ambiente `lstlisting`.

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
5 write('Benvenuto in Pascal');
```

Il nome `lstlisting` è stato preferito al semplice `listing` poiché altri pacchetti definiscono ambienti con quel nome. Per essere compatibile con questi pacchetti, i comandi e gli ambienti di `listings` cominciano con il prefisso `lst`.

Di seguito si ripropone l'esempio precedente: a sinistra c'è il sorgente e a destra il risultato.

```
\begin{lstlisting}
for i:=maxint to 0 do
begin
    { non far nulla }
end;
write('Benvenuto in Pascal');
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
5 write('Benvenuto in Pascal');
```

L'ambiente `lstlisting` e il comando `\lstinline` hanno un argomento facoltativo, costituito da un elenco di opzioni, separate da virgole. Un'opzione può essere costituita da un'unica parola o da una voce $\langle \text{chiave} \rangle = \langle \text{valore} \rangle$ (i valori `=true` possono essere omessi). Per esempio, per stampare solo le linee comprese tra la 2 e la 5 si usa il codice:

```
\begin{lstlisting}%
[firstline=2,lastline=5]
for i:=maxint to 0 do
begin
    { non far nulla }
end;

write('Benvenuto in Pascal');
\end{lstlisting}
```

```
1 begin
2     { non far nulla }
3 end;
```

L'esempio precedente mostra che le righe vuote alla fine del codice non vengono stampate (la riga 5, nel caso considerato); se si desidera modificare questo comportamento, basta specificare l'opzione `showlines=true`.

Il comando `\lstinputlisting` stampa in display il codice contenuto di un file a sé stante, senza bisogno di copiarlo nel documento. Il comando ha come argomento il nome del file. Di seguito sono riportate le prime righe del file `listings.sty`.

```
\lstinputlisting[lastline=4]%
{listings.sty}
```

```
1 %%
2 %% This is file 'listings.sty',
3 %% generated with the docstrip utility.
4 %%
```

È opportuno che il percorso (*path*) dei file inclusi mediante il comando `\lstinputlisting` non contenga spazi.

Codici con didascalia

Il pacchetto `listings` permette di trasformare un codice in display in un oggetto mobile. Per esempio, il codice 1 è stato prodotto con le seguenti istruzioni:

Codice 1: Un codice mobile

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
5 write('Benvenuto in Pascal');
```

```

\begin{lstlisting}[float=tb,
                  caption={Un esempio di codice mobile},
                  label=lst:esempio]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
write('Benvenuto in Pascal');
\end{lstlisting}

```

Naturalmente, nel documento ci si può riferire al codice con l'istruzione `\ref{lst:esempio}`.

Il valore della chiave `float` è un sottoinsieme dell'elenco di caratteri `tbph` e indica a \LaTeX dove posizionare l'oggetto mobile: in cima (*top*) o in fondo (*bottom*) alla pagina (la corrente o una successiva), su una pagina di soli oggetti mobili (*page of floats*) oppure nel punto esatto in cui è situato il relativo ambiente (*here*); la sintassi completa è `float=[*]{< sottoinsieme di tbph>}`, dove l'asterisco opzionale permette di avere un oggetto mobile esteso su tutta la pagina in un documento a più colonne.

Si può anche scegliere se posizionare la didascalia sopra (*top*) o sotto (*bottom*) al codice. A tal fine si usa la chiave `captionpos`, che ha come valore una fra le iniziali `t` e `b`.

Il funzionamento delle chiavi `caption` e `label` è simile a quello dei comandi standard `\caption` e `\label`, con la differenza che `listings` permette di avere codici con didascalia ed etichetta anche se i codici non sono mobili:

```

\begin{lstlisting}%
[caption={Un codice fisso},
 label=lst:fisso]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
write('Benvenuto in Pascal');
\end{lstlisting}

```

Codice 2: Un codice fisso

```

1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
5 write('Benvenuto in Pascal');

```

Il comando `\lstlistoflistings` stampa l'elenco dei codici (è analogo a `\listoffigures` e `\listoftables`). Si può specificare una didascalia breve (che verrà stampata nell'elenco dei codici) nel modo seguente:

```
caption={[<didascalia breve>]<didascalia>}
```

Si notino le graffe di raggruppamento. L'opzione `no1ol` fa sì che il relativo codice non compaia nell'elenco dei codici.

Se si desidera sostituire le intestazioni (predefinite) *Listing* e *List of Listings* con *Codice* ed *Elenco dei codici* rispettivamente, basta scrivere (nel preambolo, dopo aver caricato `babel` con l'opzione `italian`):

```

\addto\captionsitalian{%
  \renewcommand{\lstlistingname}{Codice}
  \renewcommand{\lstlistlistingname}{Elenco dei codici}}

```

Per avere una semplice didascalia senza intestazione e senza etichetta, si usa la chiave `title`:

```
\begin{lstlisting}[title={Una
  semplice didascalia}]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
\end{lstlisting}
```

Una semplice didascalia

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
```

7.2 IMPOSTAZIONI

Come al solito, il pacchetto si carica con

```
\usepackage[opzioni]{listings}
```

Il pacchetto ha alcune opzioni che permettono di compilare documenti scritti per vecchie versioni di listings o di risparmiare un po' di memoria in fase di correzione delle bozze. Di regola non c'è bisogno di preoccuparsene, per cui daremo per caricato il pacchetto senza alcuna opzione.

Per specificare le impostazioni, il pacchetto fornisce il comando `\lstset` (di regola nel preambolo), che modifica i valori *per tutti i codici successivi* nello stesso ambiente o gruppo. Il comando ha come argomento un elenco di coppie $\langle \text{chiave} \rangle = \langle \text{valore} \rangle$, separate da virgole.

```
\lstset{\langle \text{chiave} \rangle = \langle \text{valore} \rangle, \dots}
```

In alternativa, si può usare l'argomento facoltativo dei comandi `\lstinline`, `\lstinputlisting` e dell'ambiente `lstlisting`: in questo modo le opzioni hanno effetto *solo per il relativo codice*.

Per esempio, scrivendo nel preambolo

```
\lstset{basicstyle=\small\ttfamily}
```

tutti i codici del proprio documento verranno stampati in caratteri dal corpo piccolo (scelta consigliabile per i codici in display) nel font a spaziatura fissa predefinito, a meno che non si specifichi diversamente per qualche codice.

Il comando `\lstinline` eredita tutte le impostazioni di `\lstset` che lo precedono. Per esempio, se nel preambolo del documento si dà l'opzione globale `basicstyle=\small\ttfamily`, anche `\lstinline` produrrà codici in caratteri dal corpo piccolo. Per far sì che i codici inseriti con `\lstinline` siano composti usando il corpo corrente evitando di esplicitare ogni volta `basicstyle=\normalsize`, basta introdurre tutti i codici in display con gli ambienti personalizzati descritti nel paragrafo 7.4 a pagina 111.

Il pacchetto listings riconosce molti linguaggi di programmazione; la tabella 29 nella pagina successiva ne mostra alcuni, con i loro dialetti. Ogni linguaggio è della forma $[\langle \text{dialetto} \rangle] \langle \text{linguaggio} \rangle$ e si sceglie con la chiave `language`. Senza l'argomento facoltativo $[\langle \text{dialetto} \rangle]$, si carica il dialetto predefinito. Per esempio,

```
\lstset{language=[77]Fortran}
```

seleziona il Fortran 77 e

```
\lstset{language=Pascal}
```

imposta il Pascal standard.

Tabella 29: Alcuni linguaggi riconosciuti (i dialetti sottolineati sono predefiniti)

Algol (60, <u>68</u>)	Ada (<u>2005</u> , 83, 95)
Basic (Visual)	C (<u>ANSI</u> , Objective)
C++ (<u>ANSI</u> , GNU, <u>ISO</u> , Visual)	Cobol (1974, <u>1985</u>)
Delphi	Fortran (77, 90, <u>95</u>)
HTML	Java
Mathematica (1.0, 3.0, 5.2)	Modula-2
Pascal (Borland6, <u>Standard</u> , XSC)	PHP
Prolog	SQL
TeX (LaTeX, <u>plain</u>)	XML

7.3 PERSONALIZZAZIONI

Il pacchetto listings permette di avere un controllo molto preciso del formato del codice. Negli esempi di codice Pascal proposti fin qui, le parole chiave del linguaggio sono stampate in blu, i commenti in verde e le stringhe in rosso. Nell'esempio seguente, invece, le parole chiave sono in neretto, i commenti in grigio e le stringhe in nero; il font usato è quello a spaziatura fissa.

```
\begin{lstlisting}
for i:=maxint to 0 do
begin
    { non far nulla }
end;
write('Benvenuto in Pascal');
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
5 write('Benvenuto_in_Pascal');
```

Il codice precedente è stato ottenuto con le seguenti impostazioni:

```
\lstset{basicstyle=\small\ttfamily,
keywordstyle=\color{black}\bfseries,
commentstyle=\color{darkgray},
stringstyle=\color{black},
showstringspaces=true}
```

L'opzione showstringspaces=true fa sì che gli spazi nelle stringhe siano resi come _ e non come caratteri bianchi.

In generale, si consiglia di scegliere degli stili sobri per i propri codici. Non si possono fornire indicazioni precise, poiché lo stile adottato dipende dal documento che si sta scrivendo: una presentazione, per esempio, richiede spesso uno stile di maggiore impatto di quello impiegato in un libro. Si tratta di trovare la giusta misura.

7.3.1 Numerare le righe

Si possono numerare le righe dei codici in display. Per esempio, si possono avere numeri in corpo minuscolo sulla sinistra, una riga sì e una no.

```

\lstset{numbers=left,
        numberstyle=\tiny,
        stepnumber=2}

\begin{lstlisting}
for i:=maxint to 0 do
begin
    { non far nulla }
end;
write('Benvenuto in Pascal');
\end{lstlisting}

```

```

1  for i:=maxint to 0 do
    begin
3      { non far nulla }
    end;
5  write('Benvenuto in Pascal');

```

L'ambiente `lstlisting` permette di interrompere un codice e di riprenderlo successivamente, mantenendo la numerazione corretta.

```

\begin{lstlisting}%
[firstnumber=100]
for i:=maxint to 0 do
begin
    { non far nulla }
end;

\end{lstlisting}
Riprendiamo il codice:
\begin{lstlisting}%
[firstnumber=last]
write('Benvenuto in Pascal');
\end{lstlisting}

```

```

100 for i:=maxint to 0 do
101 begin
102     { non far nulla }
103 end;

```

Riprendiamo il codice:

```

105 write('Benvenuto in Pascal');

```

In questo esempio la chiave `firstnumber` è inizialmente impostata a 100; nell'ultima riga della prima parte del codice il suo valore è `last` (104, in questo caso), che permette di continuare la numerazione delle righe nella seconda parte del codice. La riga vuota alla fine della prima parte non è stampata, ma conta per la numerazione. Si può scrivere nel preambolo `\lstset{firstnumber=last}` una volta per tutte, così da avere le righe di tutti i propri codici numerate consecutivamente. In alternativa, si può dare un nome al codice (bisogna distinguere tra lettere maiuscole e minuscole): parti diverse dello stesso codice condividono lo stesso contatore di riga.

```

\begin{lstlisting}[name=Test]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
\end{lstlisting}
Riprendiamo il codice:
\begin{lstlisting}[name=Test]
write('Benvenuto in Pascal');
\end{lstlisting}

```

```

1  for i:=maxint to 0 do
2  begin
3      { non far nulla }
4  end;

```

Riprendiamo il codice:

```

5  write('Benvenuto in Pascal');

```

Il successivo codice `Test` comincerà con il numero di riga 6, indipendentemente dalla presenza di eventuali altri codici in mezzo.

7.3.2 Rientri

I rientri (tabulazioni o spazi) rendono il codice più leggibile. Il pacchetto assume che le tabulazioni si arrestino alle colonne 9, 17, 25, 33, e così via.

Questi valori dipendono dalla chiave `tabsize`, il cui valore predefinito è 8. Se si modifica questo valore e lo si pone pari a n , le interruzioni si arresteranno alle colonne $n + 1$, $2n + 1$, $3n + 1$, e così via.

```
\lstset{tabsize=2}
```

```
\begin{lstlisting}
123456789
      { una tabulazione }
          { due tabulazioni }
123      { 123 + due tab }
\end{lstlisting}
```

```
1 123456789
2   { una tabulazione }
3   { due tabulazioni }
4 123   { 123 + due tab }
```

Il codice a destra usa `tabsize=2`, mentre il codice sorgente \LaTeX è impostato con `tabsize=4`.

Rendere visibili le tabulazioni e gli spazi

Si possono rendere visibili i segni di tabulazione e gli spazi. A tal fine, si usano le chiavi `showtabs` e `showspaces`.

```
\lstset{showspaces=true,
      tabsize=8,showtabs=true,
      tab=\rightarrowfill}
```

```
\begin{lstlisting}
for i:=maxint to 0 do
begin
  { non far nulla }
end;
\end{lstlisting}
```

```
1  for i:=maxint to 0 do
2  begin
3  → { non far nulla }
4  end;
```

Se si seleziona l'opzione `showspaces` ma non `showtabs` le tabulazioni sono trasformate in spazi.

7.3.3 Riquadri

Per dotare i propri codici di un riquadro si usa la chiave `frame`, che può avere come valori `leftline` (che stampa una linea a sinistra del codice), `topline` (che stampa una linea sopra al codice), `bottomline` (stampa una linea sotto), `lines` (una linea sopra e una sotto), `single` (per riquadri singoli) o `shadowbox` (per riquadri ombreggiati); c'è anche `none`, che non stampa nulla.

```
\begin{lstlisting}%
[frame=lines]
for i:=maxint to 0 do
begin
  { non far nulla }
end;
\end{lstlisting}
```

```
1  for i:=maxint to 0 do
2  begin
3    { non far nulla }
4  end;
```

In alternativa, si possono impostare direttamente le linee in cima (*top*), a destra (*right*), in basso (*bottom*), e a sinistra (*left*) usando rispettivamente le quattro iniziali `t`, `r`, `b` e `l` per ottenere una linea singola e le loro versioni maiuscole per ottenere una linea doppia.

```
\begin{lstlisting}[frame=trBL]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
```

Si possono avere gli angoli arrotondati (t) o dritti (f) usando la chiave `frameround`, che ha come valore un elenco di quattro caratteri, scelti fra le lettere t o f. Il primo carattere è relativo all'angolo in alto a destra, e si continua in senso orario.

```
\lstset{frameround=fttt}

\begin{lstlisting}[frame=trBL]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
```

7.3.4 Sfondi colorati

Per inserire uno sfondo colorato si usa la chiave `backgroundcolor`, che prende come valore `\color{<colore>}`. Serve il pacchetto `xcolor`.

```
\lstset{%
backgroundcolor=\color{pink}}

\begin{lstlisting}%
[frame=single]
for i:=maxint to 0 do
begin
    { non far nulla }
end;
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     { non far nulla }
4 end;
```

7.3.5 Evidenziare parole

Di regola, le parole chiave sono evidenziate automaticamente da listings. Se però si vogliono evidenziare manualmente alcune parole, si possono usare le chiavi `emph` ed `emphstyle`.

```
\lstset{emph={square,root},
emphstyle=\bfseries}

\begin{lstlisting}
for i:=maxint to 0 do
begin
    j:=square(root(i));
end;
\end{lstlisting}
```

```
1 for i:=maxint to 0 do
2 begin
3     j:=square(root(i));
4 end;
```

Queste chiavi hanno un argomento facoltativo che permette di evidenziare parole diverse con stili diversi:

```

\lstset{emph={square},
  emphstyle=\color{green},
  emph={{[2]root}},
  emphstyle={{[2]\color{red}}}}

\begin{lstlisting}
for i:=maxint to 0 do
begin
    j:=square(root(i));
end;
\end{lstlisting}

```

```

1 for i:=maxint to 0 do
2 begin
3     j:=square(root(i));
4 end;

```

7.3.6 Inserire una parola nell'indice analitico

Per inserire una parola nell'indice analitico si usa la chiave `index`:

```

\lstset{index={square,root}}

\begin{lstlisting}
for i:=maxint to 0 do
begin
    j:=square(root(i));
end;
\end{lstlisting}

```

```

1 for i:=maxint to 0 do
2 begin
3     j:=square(root(i));
4 end;

```

In questo modo, le voci `square` e `root` vengono inserite nell'indice analitico.

7.3.7 Allineamento delle colonne

L'allineamento fra le colonne è uno degli aspetti tipografici più importanti nella composizione dei codici, oltre ai colori, i rientri e i riquadri.

Di regola, nella composizione dei codici si desidera mantenere l'allineamento delle colonne: bisogna allora usare un font a spaziatura fissa. Se però l'allineamento non ha funzioni sintattiche, si può usare il font che si vuole. In entrambi i casi è conveniente impostare l'opzione `columns=fullflexible`, che fa sì che tutti i caratteri siano composti con la loro larghezza "naturale". (L'opzione `columns=fixed`, sconsigliata, cerca invece di mantenere l'allineamento tra le colonne modificando la spaziatura tra i font, con risultati tipografici discutibili.)

Si consiglia inoltre di scegliere sempre l'opzione `keepspaces=true`, che impedisce che vengano modificati gli spazi tra le parole nel tentativo di mantenere l'allineamento tra le colonne.

7.4 TECNICHE AVANZATE

7.4.1 Ambienti personalizzati

Il pacchetto `listings` permette di definire ambienti personalizzati con cui scrivere i propri codici. A tal fine si usa il comando `\lstnewenvironment`, la cui sintassi è analoga a quella del comando standard `\newenvironment`.

```
\lstnewenvironment{<nome>}[<numero di argomenti>][<predefinito>]
  {<prima>}{<dopo>}
```

Quando \LaTeX incontra `\begin{<nome>}`, compone un codice in display con le impostazioni specificate nell'argomento `<prima>`, e quando incontra `\end{<nome>}` inserisce quanto specificato nell'argomento `<dopo>`. L'ambiente così definito può ricevere (in apertura) un certo `<numero di argomenti>`, il primo dei quali può essere facoltativo, se viene espressa anche la seconda opzione di `\lstnewenvironment`. Se in un nuovo ambiente che accetta un argomento facoltativo quest'ultimo non viene specificato, esso assume il valore `<predefinito>`.

Se, per esempio, scriviamo nel preambolo

```
\lstnewenvironment{pascal}{\lstset{language=pascal}}{}
```

l'ambiente `pascal` si usa nel modo seguente:

<pre>\begin{pascal} for i:=maxint to 0 do begin { non far nulla } end; write('Benvenuto in Pascal'); \end{pascal}</pre>	<pre>1 for i:=maxint to 0 do 2 begin 3 { non far nulla } 4 end; 5 write('Benvenuto in Pascal');</pre>
---	---

Gli ambienti con un argomento facoltativo permettono di specificare delle impostazioni differenti per ciascun codice:

```
\lstnewenvironment{pascalx}[1][
  {\lstset{language=pascal,numbers=left,float=tb,#1}}{}
```

7.4.2 Inserire comandi \LaTeX all'interno di un codice

Normalmente, il codice contenuto all'interno dei comandi e degli ambienti di listings viene stampato alla lettera, cioè senza che \LaTeX interpreti i comandi e i caratteri speciali.

Tuttavia, a volte si possono voler inserire comandi \LaTeX all'interno di un codice in display, in modo che essi vengano interpretati. A tal fine bisogna specificare una sequenza di caratteri che permetta di “passare a \LaTeX ” e successivamente di “ritornare al codice”. Per esempio:

```
\lstset{escapeinside={f!}{!f}}
```

All'interno di un codice, con `f!` si passa a \LaTeX , mentre con `!f` si ritorna al codice.

Si noti che ogni passaggio a \LaTeX interferisce con l'allineamento delle colonne. Il seguente esempio è in C++:

<pre>\lstset{language=C++, commentstyle=\color{black}} \begin{lstlisting} // Calcoliamo f!\$a_{ij}\$!f A[i][j] = A[j][j]/A[i][j]; \end{lstlisting}</pre>	<pre>1 // Calcoliamo a_{ij} 2 A[i][j] = A[j][j]/A[i][j];</pre>
---	--

7.4.3 Definire nuovi linguaggi

Il pacchetto permette di definire linguaggi personalizzati, specificando il formato delle parole chiave, dei commenti e delle stringhe.

```
\lstdefinlanguage{rock}
  {morekeywords=%
    {one,two,three,four,five,six,seven,eight,nine,ten,eleven,
     twelve,o,clock,rock,around,the,tonight},
   morecomment=[l][\color{green}]{//},
   morecomment=[s][\color{green}]{/*}{*/},
   morestring=[b][\color{red}]"}
```

La chiave `morekeywords` permette di aggiungere quante parole chiave si vuole a un linguaggio.

Per quanto riguarda i commenti e le stringhe, si usano le parole chiave `morecomment` e `morestring`, rispettivamente. Il loro valore comincia con un parametro [*<tipo>*], che indica il tipo di commento o di stringa, seguito da un altro parametro [*<stile>*], che indica lo stile del commento o della stringa, e dal numero variabile di delimitatori di apertura e di chiusura.

I tipi più comuni per i commenti sono `l` e `s`, rispettivamente con uno e due delimitatori. Un commento di tipo `l` inizia in un punto qualunque della riga e si estende fino alla fine della riga stessa, mentre un commento di tipo `s` inizia con il primo delimitatore e si conclude con il secondo delimitatore.

Il tipo più comune per le stringhe è `b`, con un delimitatore che inizia e conclude la stringa; se dentro una stringa c'è bisogno di scrivere il delimitatore (caso raro, ma possibile), si può usare il carattere di barra rovescia (*backslash*) per scrivere il delimitatore senza che venga conclusa la stringa.

```
\begin{lstlisting}
Parole chiave: one, two, ...
"Una str\"inga" Del testo
// Una riga di commento
Un commento Del testo
\end{lstlisting}
```

```
1 Parole chiave: one, two, ...
2 "Una str\"inga" Del testo
3 // Una riga di commento
4 Un commento Del testo
```

Per l'elenco completo dei tipi di commento e di stringa definiti da listings si rimanda alla documentazione del pacchetto.

Si possono infine definire delimitatori più generali con la chiave `moredelim`. I tipi possibili sono `l` e `s`, che possono essere preceduti dalla lettera `i` (*invisible*), che elimina i delimitatori dal documento finale. Marcatori di questo tipo consentono di impostare stili a piacere. Per esempio, se si scrive nel preambolo

```
\lstset{moredelim=[is][\color{orange}]{|*}{*|}}
```

i delimitatori `|* e *|` si usano nel modo seguente:

```
\begin{lstlisting}
Una parola |*arancione*|.
\end{lstlisting}
```

```
1 Una parola arancione.
```


Parte IV

TABELLE, FIGURE E DISEGNI

Tabelle e figure sono tra gli “oggetti” più usati nei documenti e tra i più problematici, perché di regola non si possono spezzare su più pagine. In questo capitolo, che spiega come servirsene senza sorprese, si danno per caricati i pacchetti `booktabs` e `caption` per le tabelle, e `graphicx` per le figure.

8.1 STRUMENTI FONDAMENTALI

Per inserire tabelle e figure in un documento da comporre con \LaTeX esistono tre strumenti essenzialmente:

- l’ambiente standard `tabular`, per tabelle che contengono prevalentemente testo;
- l’ambiente standard `array`, per tabelle che contengono prevalentemente matematica;
- il comando `\includegraphics` definito dal pacchetto `graphicx`, per includere nel documento le figure quando sono file *esterni*.

Li si vede all’opera nei tre esempi seguenti.

```
La tabella
\begin{center}
\begin{tabular}{ll}
\toprule
Alcaloide & Origine \\
\midrule
atropina & belladonna \\
morfina & papavero \\
nicotina & tabacco \\
\bottomrule
\end{tabular}
\end{center}
mostra l’origine di tre alcaloidi.
```

La tabella

Alcaloide	Origine
atropina	belladonna
morfina	papavero
nicotina	tabacco

mostra l’origine di tre alcaloidi.

```
La tabella
\[
\begin{array}{ll}
\toprule
f(x) & f'(x) \\
\midrule
x^n & nx^{n-1} \\
e^x & e^x \\
\sin x & \cos x \\
\bottomrule
\end{array}
\]
mostra le derivate di alcune
funzioni elementari.
```

La tabella

$f(x)$	$f'(x)$
x^n	nx^{n-1}
e^x	e^x
$\sin x$	$\cos x$

mostra le derivate di alcune funzioni elementari.

```

La figura
\begin{center}
\includegraphics[width=
0.5\textwidth]{Rettili}
\end{center}
riproduce
l'incisione su legno
\emph{Tassellazione
del piano con rettili}
di M.~Escher.

```

La figura



riproduce l'incisione su legno *Tassellazione del piano con rettili* di M. Escher.

Si noti che:

- Tutti e tre *non* cominciano un nuovo capoverso, ma producono un'unità tipografica indivisibile che il programma tratta come se fosse un unico carattere, e che nel caso di `tabular` e `\includegraphics` va centrata rispetto alla giustezza del testo mettendola nell'ambiente `center`.
- Una tabella `array` va racchiusa a propria volta tra comandi matematici: se in testo, di solito si usano `\[...\]`, che sostituiscono l'ambiente `center` (ma se l'opzione di classe `fleqn` è attiva, la tabella non risulterà più centrata); se `mobile`, invece, si usano `$. .$.`.
- Quando richiesta, si consiglia di assegnare all'oggetto una larghezza *relativa* espressa con una frazione della giustezza stabilita dalla classe in uso (`\textwidth`, ma si veda anche il paragrafo 13.3.4 a pagina 251). Una larghezza *assoluta* causerebbe ovvi inconvenienti cambiando classe di documento o aumentando le colonne di composizione.
- L'ambiente `center` si omette anche quando si vuole in linea una figura particolarmente piccola, come mostra l'esempio seguente:

```

La mela morsicata
\includegraphics[width=
0.10\textwidth]{apple}
è il logo di Apple.

```

La mela morsicata  è il logo di Apple.

8.2 OGGETTI IN TESTO E FUORI TESTO

In tipografia esistono due tipi di oggetto: “in testo” e “fuori testo”.

8.2.1 Tabelle e figure in testo

Osservando con attenzione gli esempi del paragrafo precedente, si possono notare le caratteristiche degli oggetti “in testo”, i quali:

- appartengono al flusso del discorso e non possono esserne scorporati senza comprometterne la comprensione;
- non prevedono didascalia (proprio perché la loro funzione è spiegata nel contesto) né riferimenti incrociati a sé stessi;
- devono essere, perciò, quanto mai chiari e intuitivi.

Apparentemente innocui, oggetti di questo tipo possono comportare in realtà problemi di impaginazione a volte irrisolvibili. S’immagini, per esempio, di essere arrivati quasi alla fine della pagina, e di dover inserire *proprio lì*, perché richiesto dal discorso, una figura alta cinque centimetri avendone però soltanto tre a disposizione: va da sé che *lì* la figura *non* ci può stare in nessun modo: se lo spazio fisico non c’è, non lo si può inventare! Questa situazione, si badi bene, non si verifica solo con \LaTeX , ma si dà indipendentemente dal programma in uso per scrivere. Come fare?

Una prima soluzione ingenua (e da evitare) potrebbe essere quella di cominciare una nuova pagina ogni volta che un oggetto non può stare in quella corrente; in questo modo, però, le pagine interrotte rimarrebbero parzialmente bianche, con un risultato tipografico insoddisfacente.

Ecco perché gli oggetti in testo devono essere *eccezionali* (tornano utili per mettere un logo “proprio lì” e in pochissime altre circostanze) e di piccole o piccolissime dimensioni.

8.2.2 Tabelle e figure fuori testo

Si risolve il problema rendendo gli oggetti “fuori testo” (o “mobili”; in inglese *floating*, “galleggianti”) e lasciando fare a \LaTeX che, nell’esatto ordine in cui oggetti dello stesso tipo sono definiti nel sorgente, li metterà nel punto *per lui* migliore (sulla pagina corrente se ci stanno, oppure in pagine successive a quella in cui finirebbero) riempiendo lo spazio rimanente con l’altro materiale a disposizione. Questa soluzione è molto vantaggiosa, perché garantisce l’ottimale riempimento della pagina tipico di \LaTeX .

A differenza di quelli in testo, gli oggetti fuori testo:

- non appartengono al flusso del discorso e per esigenze tipografiche possono essere spostati altrove dal punto esatto in cui stanno nel sorgente;
- devono avere *obbligatoriamente* un’etichetta, un numero progressivo per gli eventuali riferimenti incrociati e una didascalia che ne descriva il contenuto.

Indipendentemente dalle dimensioni del documento e degli oggetti, perciò, si raccomanda di includerli *sempre* così (come si è fatto in tutta questa guida), vincendo quanto prima le iniziali e comprensibili perplessità derivanti dal vederli molto spesso in un punto diverso da quello in cui li si è definiti, come si spiegherà tra poco.

Nei prossimi paragrafi si spiegano gli ambienti mobili e come comportarsi durante la stesura del documento; nel paragrafo 14.2 a pagina 258 si mostra che cosa (eventualmente) si può fare durante la revisione per risolvere collocazioni poco gradite e migliorarlo ulteriormente.

Ambienti standard per gli oggetti mobili

Per rendere mobile un oggetto basta inserirne il relativo codice nell’ambiente standard `table`

```
\begin{table}[⟨preferenze di collocazione⟩]
...
\end{table}
```

se è una tabella, oppure in quello figure

```
\begin{figure}[⟨preferenze di collocazione⟩]
...
\end{figure}
```

se è una figura. Come si può notare, entrambi accettano l'argomento facoltativo *⟨preferenze di collocazione⟩*, il cui funzionamento verrà spiegato nel paragrafo 8.2.2 a pagina 122. (Se si sta componendo un documento a due colonne si possono usare le varianti asterisco dei due ambienti, che mettono l'oggetto sull'intera pagina e non sulla singola colonna di composizione.) I due ambienti appena esaminati richiedono alcuni comandi importanti, descritti di seguito.

Il comando

```
\caption{⟨didascalia⟩}
```

produce, nell'ordine, l'intestazione *Tabella* o *Figura*, il numero progressivo dell'oggetto e la sua *⟨didascalia⟩*. C'è anche la variante

```
\caption*{⟨didascalia⟩}
```

che produce didascalie prive di intestazione e numero, utili per esempio in tavole illustrate fuori testo o in un libro d'arte.

I due comandi

```
\listoftables
\listoffigures
```

producono *nel punto in cui li si dà* rispettivamente la sezione contenente l'elenco delle tabelle o delle figure (esattamente come fa `\tableofcontents` per l'indice generale) con relativi titolo e testatina. Le loro voci sono le stesse didascalie degli oggetti o, se troppo lunghe, una versione ridotta da scrivere nell'argomento facoltativo di `\caption`:

```
\caption[⟨didascalia breve⟩]{⟨didascalia normale⟩}
```

Si badi bene a mettere i due indici appena visti solo se il numero degli oggetti è rilevante o è importante per il lettore poterli ritrovare agevolmente: un elenco di sole due figure, per esempio, non avrebbe alcuna utilità. Anche questi due indici richiedono *due* composizioni successive.

Il comando `\label`, da dare sempre *dopo* il corrispondente `\caption`, assegna all'oggetto un'etichetta per i riferimenti incrociati (si veda il paragrafo 3.10 a pagina 43).

Codici tipo

Il modo migliore per introdurre un oggetto mobile nel sorgente è scriverne il relativo ambiente preceduto e seguito da una riga vuota. Ecco un esempio tipico per una tabella tabular (in modo del tutto analogo si compone una tabella array, ricordandosi di racchiuderla tra dollari) con il relativo richiamo:

```
\dots qui finisce un capoverso.

\begin{table}
\caption{⟨...⟩}
\label{tab:esempio}
```

```
\centering
\begin{tabular}{\langle...\rangle}
...
\end{tabular}
\end{table}
```

La tabella~\ref{tab:esempio} è un esempio di tabella mobile.

E uno per una figura:

\dots qui finisce un capoverso.

```
\begin{figure}
\centering
\includegraphics[width=0.5\textwidth]{\langle...\rangle}
\caption{\langle...\rangle}
\label{fig:esempio}
\end{figure}
```

La figura~\ref{fig:esempio} è un esempio di figura mobile.

Come si può osservare:

- per centrare un oggetto mobile sulla pagina si usa `\centering`, perché l'ambiente `center` lascia tra testo e oggetto uno spazio verticale eccessivo (ma adeguato per un oggetto in testo);
- la corretta posizione della didascalia varia a seconda dell'oggetto cui è apposta: la tradizione italiana la vuole prima di una tabella e dopo una figura, e in queste posizioni *deve* essere messa anche nel sorgente.

Le classi standard prevedono la didascalia sempre *sotto* l'oggetto, perciò \LaTeX non la "riconosce" se messa sopra e la restituisce in modo non appropriato mettendogliela troppo a ridosso. Risolve il problema il pacchetto `caption`, da impostare come segue:

```
\usepackage{caption}
\captionsetup{tableposition=top,figureposition=bottom,font=small}
```

dove:

- `tableposition=top` ordina al programma di inserire uno spazio adeguato tra didascalia e tabella;
- `figureposition=bottom` è l'opzione analoga alla precedente per le figure;
- `font=small` produce didascalie in corpo più piccolo, secondo l'uso italiano.

Quando le opzioni sono numerose o la loro forma lo richiede, si possono scrivere con il comando `\captionsetup`.

Si noti che molte classi non standard modificano il rapporto tra ambiente mobile e didascalia per questioni di stile. Il risultato finale, perciò, *potrebbe* non corrispondere a quello appena descritto: si faccia qualche prova. Infine, si tenga presente che `caption` aumenta lo spazio tra didascalia e oggetto anche solo caricandolo.

Tabella 30: Preferenze di collocazione per gli oggetti mobili

Preferenza	Chiede a \LaTeX di mettere l'oggetto
h	Qui (<i>here</i>), se possibile
t	In cima (<i>top</i>) alla pagina
b	In fondo (<i>bottom</i>) alla pagina
p	In una pagina di soli oggetti mobili (<i>page of floats</i>)
!	Dove vorrebbe l'utente per quanto possibile

Personalizzare la didascalia

Il pacchetto `caption` permette anche di personalizzare finemente le didascalie in *ogni* loro aspetto. Quelle di questa guida, per esempio, sono state composte aggiungendo le seguenti opzioni a quelle appena viste:

```
\captionsetup{format=hang,labelfont={sf,bf}}
```

dove:

- `format=hang` allinea (*hang*) alla prima riga quelle successive (\LaTeX centra automaticamente le didascalie che occupano una sola riga);
- `labelfont={sf,bf}` imposta l'etichetta della didascalia in caratteri senza grazie neri.

Che cosa fare durante la stesura

Durante la stesura del documento, si sa, bisogna concentrarsi sul contenuto del lavoro, lasciando fare a \LaTeX tutto il resto. La gestione degli oggetti non si sottrae a questa regola, per cui in prima battuta si consiglia di inserirli tutti *senza specificare alcuna preferenza di collocazione* (si veda poco sotto): il risultato è generalmente ottimo. In linea di massima si troveranno gli oggetti “abbastanza” vicini al punto in cui stanno nel sorgente, e in particolare:

- quasi sempre in cima alla pagina (corrente o una delle successive);
- raramente in basso;
- se sono grandi (secondo i parametri di \LaTeX), in un pagina di soli oggetti mobili, sempre successiva al punto in cui li si è messi nel sorgente.

Basta sfogliare un qualunque libro ben composto per verificare che le cose stanno proprio così.

Non ci si lamenti per non vederli esattamente dove li si è inseriti nel sorgente! Non è un difetto di \LaTeX , questo, ma un vantaggio, proprio come lo è il non doversi preoccupare di numerare a mano sezioni e pagine o il non dover pensare a quanto spazio ci va tra un titolo e il testo successivo.

I risultati automatici sono *quasi sempre* migliori di quelli che si potrebbero ottenere cercando di collocare gli oggetti a mano. Quando i gusti dell'utente non coincidono con quelli del programma, tuttavia, si possono usare le *preferenze di collocazione* raccolte nella tabella 30, che “suggeriscono” a \LaTeX come si vorrebbero vedere gli oggetti sulle pagine del proprio documento. \LaTeX seguirà i suggerimenti nell'ordine in cui li trova (fa eccezione la preferenza `p`, come si spiega nel paragrafo 14.2 a pagina 258). Di seguito si propongono un paio delle possibilità più usate:

- `tp` se non si vuole nessun oggetto in fondo alla pagina;
- `htp` se si vuole che \LaTeX cerchi come prima cosa di mettere l'oggetto esattamente lì dove lo si è inserito (se è sufficientemente piccolo, di solito si viene accontentati).

Ecco ora le opzioni da evitare *sempre*:

- `h o`, peggio, `h!` *possono* funzionare solo con oggetti molto piccoli; in caso contrario, l'oggetto viene messo alla fine del capitolo (o del documento) portandosi dietro tutti gli altri inseriti successivamente (si tenga *ben* presente questo comportamento);
- `t e b da sole`, perché è buona regola dare al programma almeno un paio di possibilità (ma è ammessa la sola `p`).

In casi estremi, e solo per ottenere effetti particolari, si può forzare l'oggetto nella posizione desiderata con la preferenza `H` del pacchetto `float` (da usare *sempre* da sola).

```
\dots
```

qui finisce un capoverso.

```
\begin{figure}[H]
\centering
\includegraphics[width=0.5\textwidth]{Formica}
\caption{Figura collocata a mano}
\label{fig:float}
\end{figure}
```

La figura~\vref{fig:float} è un esempio di figura mobile collocata a mano.

...qui finisce un capoverso.



Figura 6: Figura collocata a mano

La figura 6 è un esempio di figura mobile collocata a mano.

8.3 TABELLE

Comporre tabelle di altissima qualità è una delle specialità di \LaTeX , ma i suoi comandi standard sono piuttosto limitati. Numerosi pacchetti, però, ne definiscono di nuovi e più avanzati con cui si può personalizzare finemente il proprio lavoro. Questo paragrafo, basato su [Mori, 2006], cui si rimanda per gli approfondimenti, spiega come usare gli uni e gli altri e affronta gli aspetti principali dell'argomento.

8.3.1 Indicazioni generali

Regole generali di composizione

Le regole seguenti permettono di ottenere una tabella ben composta:

- non usare *mai* filetti verticali;
- evitare filetti doppi;

Tabella 31: Tabella che non rispetta le regole generali

D	P	u	β	G
.500 m	269.8 kg	.000674 m	1.79	.04089 Pa
1.50 m	421.0 kg	.001035 m	3.59	"
10.0 m	640.2 kg	.001565 m	7.18	"

Tabella 32: Tabella che rispetta le regole generali

D (m)	P (kg)	u (m)	β	G (Pa)
0,500	269,8	0,000 674	1,79	0,040 89
1,50	421,0	0,001 035	3,59	0,040 89
10,0	640,2	0,001 565	7,18	0,040 89

- scrivere *sempre* le unità di misura nell'intestazione della colonna e non nel corpo della tabella;
- non usare *mai* le virgolette per ripetere il contenuto di una cella;
- far precedere *sempre* il separatore decimale da almeno una cifra.

A queste si possono aggiungere le seguenti:

- incolonnare i numeri al separatore decimale, se presente, che nei documenti in italiano è la virgola;
- usare i numeri maiuscoli nelle tabelle numeriche (si veda il paragrafo [A.4.2](#) a pagina [342](#)).

Così vuole la tradizione tipografica, in contrasto con la cattiva abitudine, purtroppo oggi molto diffusa, di comporre le tabelle come se fossero parti di un foglio elettronico. Per capire quanto sia importante rispettare queste regole, si confrontino le tabelle [31](#) e [32](#).

Separare le celle e chiudere le righe

Le celle di una tabella vanno separate tra loro con il carattere separatore & e le righe *devono* terminare con il comando `\`, pena un errore. Si noti che se una riga ha meno celle piene di quante sono le colonne, può essere chiusa dopo l'ultima cella riempita.

Filetti professionali

Filetti migliori di quelli che si ottengono con il comando standard `\hline`, dalla resa tipografica insoddisfacente per via dello spazio troppo risicato che risulta tra filetti e testo nelle celle, sono prodotti da tre comandi definiti dal pacchetto `booktabs`. Questi comandi non vogliono `\` dopo di sé, producono filetti di spessore differente (le righe prodotte da `\midrule`, infatti, sono più sottili delle altre) e vanno dati secondo un ordine rigoroso:

1. `\toprule` produce il primo filetto della tabella;
2. `\midrule` produce il filetto interno (o, ripetendolo, i filetti, ma non se ne abusi);

Tabella 33: Descrittori standard delle colonne

Descrittore	Spiegazione
<code>l</code>	Allinea il contenuto della cella a sinistra (<i>left</i>)
<code>c</code>	Centra il contenuto della cella (<i>center</i>)
<code>r</code>	Allinea il contenuto della cella a destra (<i>right</i>)
<code>p</code>	Giustifica un testo lungo entro una $\langle larghezza \rangle$
<code>*</code>	Ripete i descrittori

3. `\bottomrule` produce l'ultimo filetto.

Il comando

```
\cmidrule{\langle n \rangle - \langle m \rangle}
```

invece, disegna un filetto orizzontale dalla sinistra della colonna $\langle n \rangle$ -esima fino alla destra della colonna $\langle m \rangle$ -esima. Per migliorare leggermente la resa della tabella, si può specificare subito dopo `\cmidrule` un argomento facoltativo $\langle \text{troncamento} \rangle$ fra parentesi tonde:

```
\cmidrule(\langle \text{troncamento} \rangle){\langle n \rangle - \langle m \rangle}
```

che accetta tre possibilità: `r`, `l` o `rl`. Si sta dicendo a \LaTeX che il filetto va “rasato” a destra (`r`), a sinistra (`l`) o a entrambe le estremità (`rl`).

Si possono rasare anche i filetti prodotti dai primi tre comandi visti semplicemente scrivendo `@{}` all’inizio e alla fine del preambolo della tabella (si veda il paragrafo 8.3.2): potrebbe servire se lo spazio è poco o semplicemente come tocco di stile.

Codice ordinato

Anche se \LaTeX non richiede di incolonnare le celle nel sorgente, si consiglia di farlo ugualmente: un codice ordinato facilita eventuali modifiche, diminuisce la probabilità di commettere errori e aumenta quella di scovarli.

8.3.2 Tabelle standard

In linea generale, una tabella che contiene prevalentemente testo va composta dentro l’ambiente `tabular`; una tabella che contiene prevalentemente matematica va composta dentro l’ambiente `array`. Entrambi si comportano in modo molto simile, come si può osservare negli esempi seguenti:

```

\begin{tabular}{lcr}
\toprule
Grandezza & Simbolo & Unità \\
\midrule
forza      &  $F$       & newton \\
energia    &  $E$       & joule \\
tensione   &  $V$       & volt \\
\bottomrule
\end{tabular>

```

Grandezza	Simbolo	Unità
forza	F	newton
energia	E	joule
tensione	V	volt

```

\begin{array}{cc}
\toprule
f(x) & \text{Una primitiva} \\
\midrule
e^x & e^x \\
\cos x & \sin x \\
\sin x & -\cos x \\
\bottomrule
\end{array}

```

$f(x)$	Una primitiva
e^x	e^x
$\cos x$	$\sin x$
$\sin x$	$-\cos x$

Alle osservazioni del paragrafo 8.1 a pagina 117 si aggiungano le seguenti:

- `tabular` e `array` richiedono un argomento, detto *preambolo della tabella*, formato da un certo numero di *descrittori*, ciascuno dei quali definisce il comportamento di un *tipo di colonna* come spiegato nella tabella 33 nella pagina precedente;
- in `tabular`, eventuali formule matematiche si scrivono con i comandi per le formule in linea, per esempio fra dollari $\$ \dots$;
- in `array`, un eventuale testo si scrive nell'argomento del comando `\text` del pacchetto `amsmath`, che va dunque caricato.

Si noti che per “alleggerire” il codice, nel preambolo di una tabella si può sempre usare la scrittura

```
*{\langle n \rangle}{\langle descrittore \rangle}
```

equivalente a $\langle n \rangle$ colonne consecutive del tipo indicato dal $\langle descrittore \rangle$.

8.3.3 Celle con testo troppo lungo

Le tabelle migliori si ottengono lasciando loro la propria larghezza naturale. È ciò che fanno i tre descrittori `l c r`, allargando automaticamente la cella in base al contenuto. Se quest'ultimo è costituito da un testo troppo lungo, però, la tabella eccede la giustezza della riga e \LaTeX lo notifica con il relativo avviso. Per queste celle non si possono più usare le colonne appena viste, ma bisogna ricorrere ad altri strumenti:

- a un descrittore `p{\langle larghezza \rangle}`, che permette di stabilire a priori la larghezza di una sola colonna;
- al pacchetto `tabularx`, che permette di stabilire a priori la larghezza dell'intera tabella.

Si noti che in entrambi i casi le eventuali intestazioni di colonna vengono allineate a sinistra per impostazione predefinita: per cambiare questo risultato si usi il comando `\multicolumn` spiegato nel paragrafo 8.3.5 a pagina 131.

Colonne di larghezza prefissata

Il codice seguente, che mostra all'opera il descrittore `p{\langle larghezza \rangle}`, produce la tabella 34 nella pagina successiva:

Tabella 34: Tabella con colonna p

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

```

\begin{tabular}{lp{0.5\textwidth}}
\toprule
\textbf{Forza} & Una forza è una grandezza fisica che si manifesta
nell'interazione di due o più corpi materiali, che cambia lo stato
di quiete o di moto dei corpi stessi. \\
\midrule
\textbf{Momento polare} & Il momento polare di una forza rispetto a
una determinata origine è definito come il prodotto vettoriale tra
il vettore posizione (rispetto alla stessa origine) e la forza. \\
\bottomrule
\end{tabular}

```

Si noti che:

- `\textbf` produce il proprio argomento in nero;
- per impostazione predefinita, il contenuto di una colonna p viene giustificato e sillabato automaticamente (per forzare l'andata a capo in una cella si usa `\newline`);
- le eventuali colonne l, c e r rimangono della propria larghezza naturale;
- la cella è allineata alla riga a cui appartiene rispetto alla linea di base superiore.

Tabelle di larghezza prefissata

Il pacchetto `tabularx`, che carica il pacchetto `array` (si veda il paragrafo 8.3.8 a pagina 134), definisce l'omonimo ambiente `tabularx` e un nuovo tipo di colonna X, che alle caratteristiche delle colonne p appena esaminate aggiunge un vantaggio: è \LaTeX a calcolarne automaticamente la larghezza in base alla larghezza complessiva assegnata *all'intera tabella*. Infatti `tabularx` richiede *obbligatoriamente* un secondo argomento in cui indicarla (nei prossimi esempi è pari a `\textwidth`, ma sono ammessi anche altri valori, da impostare come già spiegato nel paragrafo 8.1 a pagina 117). Si osserva il pacchetto all'opera nella tabella 35 nella pagina seguente, identica alla 34 nel contenuto, ottenuta con il codice seguente:

```

\begin{tabularx}{\textwidth}{lX}
\toprule

```

Tabella 35: Tabella di larghezza prefissata ottenuta con `tabularx`

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

Tabella 36: Tabella con due colonne della stessa larghezza ottenuta con `tabularx`

Periodo	Fenomeni geologici	Biosfera
Giurassico	Periodo caratterizzato da variazioni del livello del mare; prevalenza delle terre emerse in America, Asia, Australia.	Fauna: compaiono i primi marsupiali; dominano i grandi rettili (dinosauri). Flora: predominano le conifere.
Triassico	Intensa l'erosione dei continenti; profonde fratture da cui escono lave che originano altopiani estesi.	Fauna: si diffondono i rettili; nei mari prosperano pesci e invertebrati. Flora: si sviluppano alghe caratteristiche.

```

\textbf{Forza} & Una forza è una grandezza fisica che si manifesta
nell'interazione di due o più corpi materiali, che cambia lo stato
di quiete o di moto dei corpi stessi. \\
\midrule
\textbf{Momento polare} & Il momento polare di una forza rispetto a
una determinata origine è definito come il prodotto vettoriale tra
il vettore posizione (rispetto alla stessa origine) e la forza. \\
\bottomrule
\end{tabularx}

```

La tabella 36, prodotta con il codice seguente, mostra come a *tutte* le colonne X , se più d'una, \LaTeX assegna la stessa larghezza *indipendentemente* dalle altre colonne presenti.

```

\begin{tabularx}{\textwidth}{lXX}
\toprule
Periodo & Fenomeni geologici & Biosfera \\
\midrule
\textbf{Giurassico} & Periodo caratterizzato da variazioni del
livello del mare; prevalenza delle terre emerse in America, Asia,
Australia. & Fauna: compaiono i primi marsupiali; dominano i grandi
rettili (dinosauri). Flora: predominano le conifere. \\
\midrule
\textbf{Triassico} & Intensa l'erosione dei continenti; profonde
fratture da cui escono lave che originano altopiani estesi.
& Fauna: si diffondono i rettili; nei mari prosperano pesci e
invertebrati. Flora: si sviluppano alghe caratteristiche. \\
\bottomrule
\end{tabularx}

```

Tabella 37: Tabella con colonna S

Espressione	Valore
π	3,1416
π^π	36,46
π^{π^2}	80 662,7

8.3.4 Colonne di soli numeri

Il pacchetto `siunitx` gestisce in modo molto potente e flessibile anche la resa tipografica dei numeri nelle tabelle, definendo un nuovo tipo di colonna S *specificata per dati numerici* che si comporta come segue:

- nei numeri di cinque o più cifre separa automaticamente le cifre a gruppi di tre per migliorarne la leggibilità (si veda il paragrafo A.4.2 a pagina 342);
- per impostazione predefinita, mette il separatore decimale al centro della colonna, di cui espande flessibilmente entrambi i margini in base alla lunghezza del numero;
- spazia correttamente il separatore decimale (se c'è) tra parte intera e decimale del numero (si veda il paragrafo A.4.2 a pagina 342).

Lo si vede all'opera nella tabella 37, ottenuta con il seguente codice:

```
\usepackage{siunitx}
\sisetup{output-decimal-marker={,}}

\begin{tabular}{cS}
\toprule
Espressione      & {Valore} \\
\midrule
 $\pi$           & 3.1416 \\
 $\pi^\pi$        & 36.46 \\
 $\pi^{\pi^2}$     & 80662.7 \\
\bottomrule
\end{tabular}
```

Si noti che:

- si possono scrivere le opzioni di `siunitx` (specie se numerose) anche con il comando `\sisetup` immediatamente dopo aver caricato il pacchetto;
- dell'eventuale testo in una colonna S (di solito nell'intestazione) si scrive tra parentesi graffe per non "confondere" il pacchetto, che lì si aspetterebbe dei numeri.

Tra le numerose opzioni che il pacchetto definisce per trattare i numeri nelle tabelle, se ne segnalano qui un paio, utili per risolvere alcune situazioni in cui il comportamento predefinito delle colonne S risulta inadeguato. Nell'esempio seguente si nota uno spazio tra le colonne decisamente esagerato:

Tabella 38: Tabella con cella multicolonna

Nome		Carica
Particella	Antiparticella	(e)
elettrone	positrone	∓ 1
protone	antiprotone	± 1
neutrone	antineutrone	0

```
\begin{tabular}{SS}  
\toprule  
{P$} & {u$} \\  
{(kg)} & {(m)} \\  
\midrule  
269.8 & 0.000674 \\  
421.0 & 0.001035 \\  
640.2 & 0.001565 \\  
\bottomrule  
\end{tabular}
```

P (kg)	u (m)
269,8	0,000 674
421,0	0,001 035
640,2	0,001 565

Risolve il problema l’opzione `table-format=<valore>`, da assegnare a ciascuna colonna S nel preambolo della tabella. La sintassi completa dell’opzione è:

```
table-format=<prima>.<dopo>
```

dove:

- `<prima>` è il numero di cifre intere del numero che nella colonna ha la parte intera più lunga;
- `’.’` è il separatore decimale, che nel documento finito sarà reso con la virgola;
- `<dopo>` è il numero di cifre decimali del numero che nella colonna ha la parte decimale più lunga.

La tabella precedente diventa:

```
\begin{tabular}%  
{S[table-format=3.1]%  
S[table-format=1.6]}  
\toprule  
{P$} & {u$} \\  
{(kg)} & {(m)} \\  
\midrule  
269.8 & 0.000674 \\  
421.0 & 0.001035 \\  
640.2 & 0.001565 \\  
\bottomrule  
\end{tabular}
```

P (kg)	u (m)
269,8	0,000 674
421,0	0,001 035
640,2	0,001 565

Nello stesso modo è stata prodotta la tabella 32 a pagina 124.

Quando c’è poco spazio nella tabella ma non s’intende rinunciare alle colonne S, risolve il problema l’opzione `table-parse-only`, che in pratica si comporta come un descrittore standard c. Eccola all’opera:

Tabella 39: Tabella con celle multiriga

Famiglia	Particella	Simbolo
leptoni	elettrone	e
	muone	μ
	tau	τ
	neutrino	ν
barioni	protone	p
	neutrone	n

```
\begin{tabular}%  
  {SS[table-parse-only]}  
\toprule  
{Allineati} & {Centrati} \\  
\midrule  
3.1416      & 3.1416 \\  
36.46       & 36.46 \\  
80662.7     & 80662.7 \\  
\bottomrule  
\end{tabular}
```

Allineati	Centrati
3,1416	3,1416
36,46	36,46
80 662,7	80 662,7

Se in un documento le tabelle numeriche sono poche o pochissime, siunitx potrebbe rivelarsi troppo laborioso: si abbia cura allora di allineare i numeri a destra e di assegnare loro lo stesso numero di cifre decimali.

8.3.5 Celle multicolonna

Il comando

```
\multicolumn{<n>}{<descrittore>}{<testo>}
```

sostituisce a $\langle n \rangle$ celle successive un'unica cella, il cui $\langle testo \rangle$ viene organizzato nei modi specificati con il $\langle descrittore \rangle$. Lo si vede all'opera nel prossimo esempio, che produce la tabella 38 a fronte:

```
\begin{tabular}{llc}  
\toprule  
\multicolumn{2}{c}{Nome} & Carica \\  
Particella & Antiparticella & (e) \\  
\midrule  
elettrone & positrone &  $\mp 1$  \\  
protone & antiprotone &  $\pm 1$  \\  
neutrone & antineutrone &  $0$  \\  
\bottomrule  
\end{tabular}
```

Si può usare il comando anche per *una sola* cella, come spesso accade nell'intestazione della tabella: in questo caso, a $\langle n \rangle$ si sostituisce 1.

8.3.6 Celle multiriga

Il comando

```
\multirow{<n>}{<testo>}
```

Tabella 40: Tabella con celle multiriga e multicolonna

Elemento	Strati		
	K	L	M
idrogeno	1		
litio	2	1	
sodio	2	8	1

(che richiede il pacchetto omonimo) crea una cella alta $\langle n \rangle$ righe, il cui $\langle testo \rangle$ verrà centrato verticalmente. Il codice

```
\begin{tabular}{clc}
\toprule
Famiglia & Particella & Simbolo \\
\midrule
\multirow{4}{*}{leptoni} & elettrone &  $e^-$  \\
& muone &  $\mu^-$  \\
& tau &  $\tau^-$  \\
& neutrino &  $\nu$  \\
\midrule
\multirow{2}{*}{barioni} & protone &  $p$  \\
& neutrone &  $n$  \\
\bottomrule
\end{tabular}
```

produce la tabella 39 nella pagina precedente. Si noti, però, che le celle multiriga:

- *non sempre* saranno perfettamente centrate rispetto alle righe vicine;
- sono incompatibili con le colonne X.

I due comandi si possono combinare. Il codice

```
\begin{tabular}{lccc}
\toprule
\multirow{2}{*}{Elemento} & \multicolumn{3}{c}{Strati} \\
\cmidrule{lr}{2-4}
& K & L & M \\
\midrule
idrogeno & 1 & & \\
litio & 2 & 1 & \\
sodio & 2 & 8 & 1 \\
\bottomrule
\end{tabular}
```

produce la tabella 40.

8.3.7 Spaziare a mano righe e colonne

Quando (molto di rado) i risultati di \LaTeX non soddisfano completamente, si può migliorare la resa tipografica della tabella con piccoli aggiustamenti manuali. I casi, di solito, sono due:

1. le righe della tabella appaiono troppo ravvicinate;
2. le colonne della tabella appaiono troppo ravvicinate.

Risolve il primo caso il comando

```
\[⟨altezza⟩]
```

che, sostituito al delimitatore di riga standard `\` *nel corpo della tabella*, abbassa la riga immediatamente successiva (e solo quella) di uno spazio verticale pari ad $\langle\text{altezza}\rangle$. Di solito s’inserisce questo spazio supplementare per rimediare a lievi sovrapposizioni (specialmente di formule matematiche in display) oppure per dare maggiore “respiro” alla tabella come si è fatto nella tabella 44 a pagina 138, dove si sono allontanati leggermente i blocchi di testo altrimenti troppo vicini.

Si osservi come la spaziatura automatica nella tabella seguente è troppo risicata

```
\[
\begin{array}{cc}
\toprule
f(x)      & f'(x) \\
\midrule
\log x    & \dfrac{1}{x} \\
\arctan x & \dfrac{1}{1+x^2} \\
\bottomrule
\end{array}
\]
```

$f(x)$	$f'(x)$
$\log x$	$\frac{1}{x}$
$\arctan x$	$\frac{1}{1+x^2}$

e come la si può correggere molto semplicemente con

```
\[
\begin{array}{cc}
\toprule
f(x)      & f'(x) \\
\midrule
\log x    & \dfrac{1}{x} \\
\arctan x & \dfrac{1}{1+x^2} \\
\bottomrule
\end{array}
\]
```

$f(x)$	$f'(x)$
$\log x$	$\frac{1}{x}$
$\arctan x$	$\frac{1}{1+x^2}$

Il comando `\dfrac` richiede il pacchetto `amsmath`.

Non accade praticamente *mai* di dover aumentare la distanza tra due colonne. Se proprio ce ne fosse bisogno, si può usare il separatore

```
@{⟨larghezza⟩}
```

che, dato *nel preambolo della tabella* tra i descrittori delle due colonne che si desidera allontanare, inserisce uno spazio orizzontale pari a $\langle\text{larghezza}\rangle$. Si noti che *in questo caso* la $\langle\text{larghezza}\rangle$ va indicata:

- mettendo le consuete espressioni (valore e unità di misura) nell’argomento di `\hspace{⟨...⟩}`;
- con comandi di spaziatura fissa (`\quad` e `\qquad`, per esempio).

Le due istruzioni appena esaminate si possono eseguire insieme e anche più di una volta nella stessa tabella, là dove serve.

8.3.8 Personalizzare le colonne: array

I diversi tipi di colonna esaminati fino a qui (gli standard `l`, `c`, `r` e `p` e gli “speciali” `X` e `S`) riescono a soddisfare la maggior parte delle esigenze, ma non si possono personalizzare. S’immagini, per esempio, di dover scrivere il contenuto di un’intera colonna in nero: può essere più vantaggioso, soprattutto se le celle sono numerose, definire questo stile una volta per tutte anziché farlo in ognuna.

Risolve il problema il pacchetto `array` (da non confondere con l’omonimo ambiente standard), che per personalizzare le colonne definisce le due istruzioni `>\{<dichiarazione iniziale\}` e `<\{<dichiarazione finale\}`, la cui sintassi completa è

```
>\{<dichiarazione iniziale\}<descrittore><\{<dichiarazione finale\}
```

dove:

- `>\{<dichiarazione iniziale\}` va data prima del `<descrittore>` per dire a \LaTeX di eseguirla *prima* del contenuto della colonna;
- `<descrittore>` è un descrittore di colonna, di quelli già esaminati o eventualmente definito con `\newcolumnntype` (si veda più sotto);
- `<\{<dichiarazione finale\}` va data dopo il `<descrittore>` per dire a \LaTeX di eseguirla *dopo* il contenuto della colonna.

Le istruzioni appena viste si possono usare nel preambolo della tabella se servono ogni tanto; se invece le tabelle con colonne personalizzate sono molte, è più conveniente definirle una volta per tutte nel preambolo del documento con il comando `\newcolumnntype`, la cui sintassi completa è

```
\newcolumnntype{<carattere>}{>\{<dichiarazione>}<descrittore><\{<dichiarazione>}}
```

dove `<carattere>` è una lettera, diversa da tutti i descrittori di colonna già in uso nel documento, che identifica il nuovo descrittore, da usare come al solito nel preambolo della tabella.

Nei prossimi paragrafi si vedono le dichiarazioni all’opera.

Modificare lo stile del font di una colonna

Le caratteristiche di `array` tornano utili per modificare lo stile del font di una colonna: basta sostituire a `<dichiarazione iniziale>` una delle dichiarazioni elencate nella tabella 17 a pagina 61. Il codice

```
\begin{tabular}{>\{\bfseries\}lp{0.5\textwidth}}
\toprule
Forza & Una forza è una grandezza fisica che si manifesta
nell’interazione di due o più corpi materiali, che cambia lo stato
di quiete o di moto dei corpi stessi. \\\
\midrule
Momento polare & Il momento polare di una forza rispetto a una
determinata origine è definito come il prodotto vettoriale tra il
vettore posizione (rispetto alla stessa origine) e la forza. \\\
\bottomrule
\end{tabular}
```

produce la tabella 34 a pagina 127, prodotta in quella sede specificando a mano lo stile della prima colonna.

Tabella 41: Tabelle con testo diversamente allineato in una colonna p

(a) Testo allineato a destra.		(b) Testo allineato a sinistra.	
Grandezza	Descrizione	Grandezza	Descrizione
Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.	Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.	Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

Modificare l'allineamento del testo in una colonna

Le istruzioni appena viste possono servire anche quando una colonna p o X fosse troppo stretta per giustificare il testo in modo soddisfacente. Basta sostituire a *⟨dichiarazione iniziale⟩* una delle tre dichiarazioni seguenti:

- `\raggedright` (“disordinato a destra”) per avere il contenuto delle celle allineato *a sinistra*;
- `\centering` per averlo centrato;
- `\raggedleft` (“disordinato a sinistra”) per averlo allineato *a destra*.

Si confrontino le tabelle 41a e 41b, la seconda delle quali è stata ottenuta con il codice seguente:

```
\begin{tabular}{>{\bfseries}l>{\raggedright\arraybackslash}%
                p{0.2\columnwidth}}
\toprule
Grandezza & Descrizione \\
\midrule
Forza & Una forza è una grandezza fisica che si manifesta
nell'interazione di due o più corpi materiali, che cambia lo
stato di quiete o di moto dei corpi stessi. \\
\midrule
Momento polare & Il momento polare di una forza rispetto a
una determinata origine è definito come il prodotto vettoriale tra
il vettore posizione (rispetto alla stessa origine) e la forza. \\
\bottomrule
\end{tabular}
```

Si noti che:

- Diversamente da quanto accade in una colonna p standard, l'istestazione di colonna segue l'allineamento dichiarato.
- Se nell'ultima colonna della tabella c'è una delle tre dichiarazioni appena viste, la riga della tabella va terminata con `\tabularnewline`, mentre si

Tabella 42: Tabella ottenuta con array

$\int \cos x \, dx$	$\sin x + c$
$\int e^x \, dx$	$e^x + c$
$\int \sec^2 x \, dx$	$\tan x + c$

userà `\\` per andare a capo *nella stessa cella*. Si può usare il consueto `\\` solo dando anche `\arraybackslash` come mostrato.

- Il testo *non è sillabato*. Per sillabarlo, basta caricare l'apposito pacchetto `ragged2e` e sostituire a quelli usati i comandi `\RaggedRight` e `\RaggedLeft`.

Una colonna `X` si comporta in modo analogo.

Colonne di sola matematica

I comandi seguenti

```
\newcolumnntype{L}{>{\$}l<{\$}}
\newcolumnntype{C}{>{\$}c<{\$}}
\newcolumnntype{R}{>{\$}r<{\$}}
```

definiscono tre nuove colonne `L`, `C` e `R` nelle quali le formule matematiche vengono rispettivamente allineate a sinistra, centrate e allineate a destra. Se si preferisce il formato in `display`, basta aggiungere il relativo comando alla definizione della nuova colonna. Il codice

```
\newcolumnntype{L}{>{\$ \displaystyle}l<{\$}}
\newcolumnntype{C}{>{\$}c<{\$}}

\begin{tabular}{LC}
\toprule
\int \cos x \, dx & \sin x + c \\
\midrule
\int e^x \, dx & e^x + c \\
\midrule
\int \sec^2 x \, dx & \tan x + c \\
\bottomrule
\end{tabular}
```

produce la tabella 42.

Ulteriori allineamenti

Il pacchetto `array` definisce ulteriori due descrittori, `m` e `b`. Con entrambi il contenuto delle celle è composto come in una colonna `p`, ma:

- con `m` la cella è centrata rispetto alla riga a cui appartiene;
- con `b` la cella è allineata rispetto alla riga a cui appartiene rispetto alla linea di base inferiore.

Per ulteriori dettagli su `array` si consiglia la lettura della sua documentazione e di [Gregorio, 2010].

Tabella 43: Tabella con corpo di carattere inferiore (`footnotesize`) a quello del testo principale (`normalsize`)

Forza	Una forza è una grandezza fisica che si manifesta nell'interazione di due o più corpi materiali, che cambia lo stato di quiete o di moto dei corpi stessi.
Momento polare	Il momento polare di una forza rispetto a una determinata origine è definito come il prodotto vettoriale tra il vettore posizione (rispetto alla stessa origine) e la forza.

8.3.9 Tabelle con note

Il comando `\footnote`, che in un testo produce le note al piede, non funziona nell'ambiente `tabular` e, come per la maggior parte delle cose in `LATEX`, questa limitazione ha ottime ragioni per esserci. Come si sa, una tabella dovrebbe essere *sempre* inserita fuori testo in un documento: una nota al piede la vincolerebbe alla pagina in cui si trova la nota. Il luogo più adatto per mettere eventuali annotazioni è la didascalia.

8.3.10 Tabelle grandi

Se le dimensioni della tabella finita eccedono quelle della gabbia del testo in lunghezza, in larghezza o in entrambe, si prospettano soluzioni diverse a seconda della dimensione in eccesso. Se *troppo lunga* o *troppo larga*, si può spezzarla su più pagine, ridurre il corpo del font o ruotarla (ma quest'ultima soluzione si applica solo se troppo larga). A ogni caso si possono applicare più soluzioni contemporaneamente.

Ridurre il corpo del font

Per ridurre il corpo del font in una tabella (ma non nell'eventuale didascalia) si usano le stesse dichiarazioni elencate nella tabella 18 a pagina 62, con l'avvertenza di darle:

- *fuori* da `tabular` se la tabella è *in testo* (subito dopo la tabella una nuova dichiarazione ripristinerà il font corrente);
- *dentro* `table` se la tabella è *mobile* (si veda il paragrafo 8.2 a pagina 118).

Il codice

```
\begin{table}[tb]
\footnotesize
\caption{⟨...⟩}
\label{⟨...⟩}
\centering
\begin{tabular}{lp{0.5\textwidth}}
...
\end{tabular}
\end{table}
```

produce la tabella 43. (Se ne confronti il risultato con la tabella 34 a pagina 127.)

Ruotare una tabella

Per ruotare di 90° una tabella è utile il pacchetto `rotating`, che definisce l'ambiente `sidewaystable` da usare nel modo seguente:

```
\begin{sidewaystable}
\caption{...}
\label{...}
\centering
\begin{tabular}
...
\end{tabular}
\end{sidewaystable}
```

Si noti che una tabella di questo tipo è mobile e occupa *sempre* una pagina a sé, come mostra la tabella 8 a pagina 37. Per ruotare le immagini il pacchetto definisce l'analogo ambiente `sidewaysfigure`.

Tabelle su più pagine

Una tabella *deve* sempre poter stare in una sola pagina: se è più lunga, L^AT_EX taglia le parti in eccesso. Il pacchetto `longtable` può risolvere il problema e ripartire una tabella su più pagine, anche se oggetti di questo tipo, si ricordi, andrebbero usati *soltanto se inevitabili*.

A differenza di altri pacchetti simili (come `supertabular` e `xtab`), `longtable` di solito riesce a usare su ogni pagina la stessa larghezza di riga, corrispondente a quella della riga più lunga della tabella: il risultato finale potrebbe richiedere alcune composizioni successive. Si noti che anche se può avere una didascalia, una tabella `longtable` *non* è mobile: se ne valutino sempre con attenzione opportunità e posizione, dunque, e se dovesse andare a pagina nuova molto prima della fine di quella corrente si sia pronti a riformulare il testo nei dintorni per bilanciare il risultato.

La tabella 44, che mostra il pacchetto all'opera, riporta *tutte* le parti di una `longtable` nell'ordine in cui vanno scritte, ma si ricordi che non sono obbligatorie né si deve abusarne: la didascalia non sempre è necessaria, e anche intestazioni e piedi sono facoltativi, ma si consiglia di scrivere almeno uno dei due per segnalare al lettore che la stessa tabella occupa più di una pagina.

Tabella 44: Esempio di tabella ripartita su più pagine

Comando	Effetto
<code>\begin{longtable}</code>	Comincia la tabella. L'ambiente si comporta come <code>tabular</code> , ma in più, dopo aver composto ogni riga controlla l'altezza complessiva della tabella: se supera quella della pagina, vengono inseriti automaticamente il contenuto del piede (<i>foot</i>) e il comando <code>\end{tabular}</code> , e la tabella continua su una nuova pagina con l'intestazione scelta (<i>head</i>).

Continua nella prossima pagina

Continua dalla pagina precedente

Comando	Effetto
[$\langle\text{carattere}\rangle$]	Va eventualmente specificato tra l'apertura dell'ambiente e il preambolo della tabella per impostarne la posizione sulla pagina (c se al centro, l se a sinistra e r se a destra). Non indicandolo, la tabella viene centrata per impostazione predefinita.
{ $\langle\text{preambolo}\rangle$ }	È il consueto preambolo nel quale indicare i descrittori già esaminati nei paragrafi precedenti.
<code>\caption{$\langle\text{didascalia}\rangle$}</code> \\	Se specificato, assegna alla tabella una didascalia, anche nella versione [$\langle\text{didascalia breve}\rangle$] (si veda il paragrafo 8.2.2 a pagina 119).
<code>\label{$\langle\text{etichetta}\rangle$}</code> \\	Assegna alla tabella un'etichetta, utile per i riferimenti incrociati (si veda lo stesso paragrafo).
$\langle\text{intestazione iniziale}\rangle$ \\ <code>\endfirsthead</code>	Specifica l' $\langle\text{intestazione iniziale}\rangle$, cioè l'intestazione della tabella nella prima pagina in cui compare.
$\langle\text{intestazione normale}\rangle$ \\ <code>\endhead</code>	Specifica l' $\langle\text{intestazione normale}\rangle$, cioè l'intestazione della tabella dalla seconda pagina in poi (Continua dalla pagina precedente, per esempio).
$\langle\text{piede normale}\rangle$ \\ <code>\endfoot</code>	Specifica il $\langle\text{piede normale}\rangle$, cioè il testo che deve comparire alla fine di ogni pagina (Continua nella prossima pagina).
$\langle\text{piede finale}\rangle$ \\ <code>\endlastfoot</code>	Specifica il $\langle\text{piede finale}\rangle$, cioè il testo che deve apparire subito dopo l'ultima riga della tabella (Si conclude dalla pagina precedente).
$\langle\text{corpo della tabella}\rangle$	Qui si mette il contenuto della tabella, separando le colonne e terminando le righe come al solito.
<code>\end{longtable}</code>	Termina la tabella.

Si conclude dalla pagina precedente

Il codice seguente ha prodotto la tabella 44 nella pagina precedente:

```
\begin{longtable}{lp{0.48\textwidth}}
% intestazione iniziale
\caption{Esempio di tabella ripartita su più pagine}
```

```

\label{tab:longtable} \\
\toprule
Comando & Effetto \\
\midrule
\endfirsthead
% intestazione normale
\multicolumn{2}{l}{\footnotesize\itshape
Continua dalla pagina precedente} \\
\toprule
Comando & Effetto \\
\midrule
\endhead
% piede normale
\midrule
\multicolumn{2}{r}{\footnotesize\itshape
Continua nella prossima pagina} \\
\endfoot
% piede finale
\bottomrule
\multicolumn{2}{r}{\footnotesize\itshape
Si conclude dalla pagina precedente} \\
\endlastfoot
% corpo della tabella
... & ... \\
...
... & ... \\
\end{longtable}

```

Per evitare errori nella composizione, si ricordi di terminare con `\\` le righe di intestazioni, piedi e didascalia (o etichetta, se presente). Si noti, infine, che in casi particolarissimi si può ruotare anche una `longtable` mettendola nell'ambiente `landscape` del pacchetto `pdflscape` (se ne veda la documentazione), ma come al solito si raccomanda di non abusare di questa possibilità.

Per gli approfondimenti si consiglia la lettura di [Gregorio, 2010].

8.4 FIGURE

Le figure rientrano tra gli argomenti più studiati dalle guide a \LaTeX , tanto che ne esistono di specifiche, cui si rimanda per gli approfondimenti.

Questi oggetti presentano almeno due problemi diversi, riguardanti:

- il *tipo di file* da introdurre nel documento (verrà trattato in questo paragrafo);
- la *collocazione* della figura sulla pagina (verrà trattato nel paragrafo successivo).

Di qui in avanti si dà per caricato il pacchetto `graphicx`.

8.4.1 Immagini vettoriali e bitmap

Si possono dividere le figure in due grandi classi: le immagini *vettoriali* e le immagini *bitmap* [Mori, 2007].

Immagini vettoriali

Le immagini vettoriali sono descritte da forme, possono essere scalate e deformate senza perdere in definizione e sono adatte soprattutto per schemi e grafici, argomento non considerato in questa guida per via della sua complessità (gli strumenti più diffusi per disegnare con \LaTeX sono i pacchetti *TikZ*, che produce disegni d'ogni tipo, e *pgfplots*, che fa grafici di funzioni). Molto più semplicemente, le si può aggiungere al documento dopo averle preparate a parte con programmi specifici. I formati vettoriali più noti e diffusi sono il *PDF*, il *PS* e il suo parente stretto *EPS* e l'*SVG*, usato specialmente per le applicazioni Web.

Il paragrafo 8.4.4 nella pagina seguente raccoglie alcuni programmi per la grafica vettoriale.

Immagini bitmap

Le immagini bitmap sono matrici di pixel colorati, di solito perdono in definizione se ingrandite o rimpicciolite e sono più adatte a fotografie e icone. I formati bitmap sono numerosissimi e comprendono il *JPEG*, molto diffuso in ambito fotografico e nella grafica con colori morbidi, il *PNG*, adatto per grafica con colori decisi, il *GIF*, il *TIFF*. Alcuni di essi sono compressi per sfruttare al meglio la ridondanza d'informazione.

8.4.2 Convertire i formati

Prima ancora di includere le immagini nel documento bisogna produrle nel formato più adatto al proprio scopo. È inutile registrare una figura come *JPEG* per poi convertirla in *PDF*, perché la conversione include semplicemente il file bitmap in una “cornice” *PDF* senza migliorarne in alcun modo la qualità. È sbagliato anche fare l'opposto, perché così si perdono le informazioni sulla geometria della figura, abbassandone la qualità. Nonostante questo, si potrebbero avere a disposizione soltanto immagini in formati *non* adatti a \LaTeX , e allora la conversione sarebbe davvero necessaria.

A questo proposito si ricordi bene che \LaTeX accetta immagini *PDF*, *JPEG*, *PNG* ed *EPS*. Il pacchetto *bmpsize*, infine, permette di inserire anche immagini *BMP*, *GIF* e *TIFF*.

Il paragrafo 8.4.4 nella pagina successiva descrive alcuni programmi per convertire i diversi formati.

8.4.3 Ritagliare le immagini

Uno dei parametri più importanti di una figura è l'informazione sulle dimensioni del rettangolo circoscritto a essa (*bounding box*). Questo contorno determina la grandezza effettiva dell'immagine e serve a \LaTeX per calcolare lo spazio da riservarle sulla pagina. Idealmente, il contorno dovrebbe coincidere con il limite dell'immagine, ma talvolta le figure sono circondate da un invisibile bordo bianco più o meno ampio, fonte di non pochi problemi estetici: la figura appare sulla pagina troppo piccola o non centrata o circondata da eccessivi margini verticali, per esempio, anche se \LaTeX la sta trattando nel modo corretto.

La primissima cosa da verificare, quindi, è che le dimensioni della *bounding box* siano corrette, aprendo la figura con un programma opportuno

Tabella 45: Alcuni programmi utili per lavorare con \LaTeX (le lettere G, C, R e V indicano rispettivamente le funzioni di grafica vettoriale, conversione dei formati, ritaglio immagini e visualizzazione; il simbolo • indica che la funzione è disponibile)

Programma	G	C	R	V
Adobe Acrobat			•	•
Adobe Reader				•
Anteprima		•	•	•
Asymptote	•			
Ghostview e GSview		•		•
GIMP		•	•	•
Gnuplot	•			
ImageMagick		•		
Inkscape	•	•		•
Mathematica	•			
OmniGraffle	•			
Xfig e WinFIG	•			

(come Adobe Reader o GIMP) e attivando la visualizzazione del contorno che, se scorretto, va ridimensionato. Se il problema riguarda poche figure si può risolvere a mano, ma se i file da ottimizzare sono molti, vanno corretti all’origine (magari configurando ad hoc il programma usato per produrli).

8.4.4 Alcuni programmi utili

La composizione asincrona di \LaTeX ha anche un vantaggio da non trascurare: permette di usare sempre il prodotto migliore. Per ciascuna operazione sul documento, infatti, l’utente può usare un programma specializzato, ciò che non sarebbe permesso con un software “tuttofare” come un editor di testi tradizionale. La tabella 45 raccoglie, senza pretese di completezza, alcuni programmi utili per lavorare con \LaTeX (ma preziosi anche in molte altre occasioni), specificandone le funzioni principali. Una veloce ricerca in Rete permette di recuperarli.

8.4.5 Includere le immagini nel documento

Il pacchetto `graphicx`, che in genere non richiede opzioni, gestisce il trattamento delle immagini con \LaTeX . Il comando `\includegraphics`, la cui sintassi completa è

```
\includegraphics[⟨chiave⟩=⟨valore⟩,⟨...⟩]{⟨immagine⟩}
```

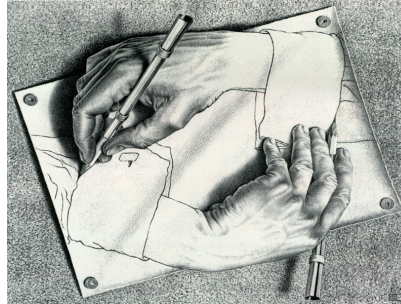
le include nel documento. Si osservi che:

- nell’argomento facoltativo ci vanno le opzioni che regolano l’aspetto della figura sulla pagina nella forma `⟨chiave⟩=⟨valore⟩` (si veda la tabella 46 a pagina 144);
- nell’argomento obbligatorio ci va il nome dell’immagine *senza* specificarne l’estensione.

Gli esempi seguenti mostrano il pacchetto all’opera.

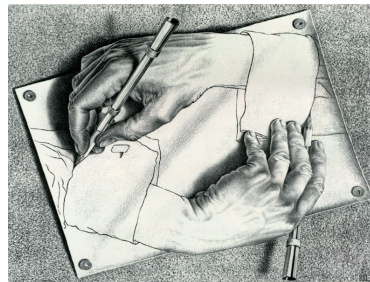
L'utente può assegnare all'immagine una larghezza (width)

```
\includegraphics[width=%  
\textwidth]{mani}
```



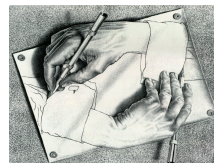
o un'altezza (height) a piacere:

```
\includegraphics[height=%  
0.15\textheight]{mani}
```



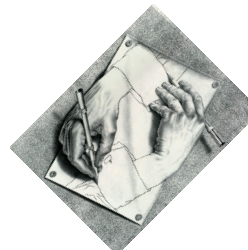
Ancora, la può ridimensionare nel suo complesso

```
\includegraphics[scale=0.10]{mani}
```



o ruotare di un certo angolo in entrambi i sensi:

```
\includegraphics[width=0.5%  
\textwidth,angle=45]{mani}
```



Si noti che:

- per i motivi già spiegati nel paragrafo 8.1 a pagina 117, le figure devono avere dimensioni *relative*, cioè essere larghe una frazione di `\textwidth` e alte una frazione di `\textheight` (l'altezza della gabbia del testo);
- il valore di scalatura si esprime con un numero decimale;
- l'angolo di rotazione si esprime con un numero (negativo, se la rotazione è oraria) nell'intervallo 0-360;
- se s'intende assegnare all'immagine sia una larghezza sia un'altezza determinate, si ricordi di specificare anche la chiave `keepaspectratio` per evitare di distorcerla.

Tabella 46: Principali chiavi di `graphicx`

Chiave	Agisce su
<code>width</code>	Larghezza
<code>height</code>	Altezza
<code>scale</code>	Larghezza e altezza
<code>angle</code>	Orientamento

8.5 DISPOSIZIONI PARTICOLARI

Questo paragrafo presenta alcune possibilità per organizzare la disposizione degli oggetti mobili sulla pagina in modi diversi da quelli predefiniti. Quanto si dirà vale sia per le tabelle sia per le figure.

8.5.1 Didascalie laterali

Il pacchetto `sidecap` produce la didascalia *accanto* all'oggetto (di solito una figura, più raramente una tabella) anziché sopra o sotto. Le opzioni fondamentali del pacchetto sono le seguenti:

outercaption La didascalia è posta nel margine esterno della pagina (a sinistra in quelle pari e a destra in quelle dispari). È l'opzione predefinita.

innercaption La didascalia è posta nel margine interno della pagina (a destra in quelle pari e a sinistra in quelle dispari).

leftcaption, rightcaption La didascalia è sempre posta a sinistra o a destra dell'oggetto, rispettivamente.

ragged, raggedright, raggedleft Gestiscono l'allineamento delle didascalie brevi.

Il pacchetto definisce due nuovi ambienti `SCfigure` e `SCtable` (analoghi agli ambienti `figure` e `table`) che prevedono due argomenti facoltativi,

```
\begin{SCfigure}[⟨larghezza relativa⟩][⟨collocazione⟩]
```

e

```
\begin{SCtable}[⟨larghezza relativa⟩][⟨collocazione⟩]
```

dove:

- *⟨larghezza relativa⟩* indica il rapporto fra larghezza della didascalia e larghezza della figura (o della tabella) corrispondente. Un valore elevato di *⟨larghezza relativa⟩* (50, per esempio) assegna alla didascalia la massima larghezza possibile.
- *⟨collocazione⟩* indica le preferenze di collocazione degli ambienti mobili, da usare come al solito. Il valore predefinito è `tbp`.

Esistono anche gli ambienti `SCfigure*` e `SCtable*` (la cui sintassi è la stessa di `SCfigure` e `SCtable`) che in un documento a più colonne permettono di ottenere rispettivamente una figura o una tabella mobile (con didascalia laterale) estesa su tutta la pagina.

La figura 7 nella pagina successiva è stata inclusa nel documento con un codice del tipo:

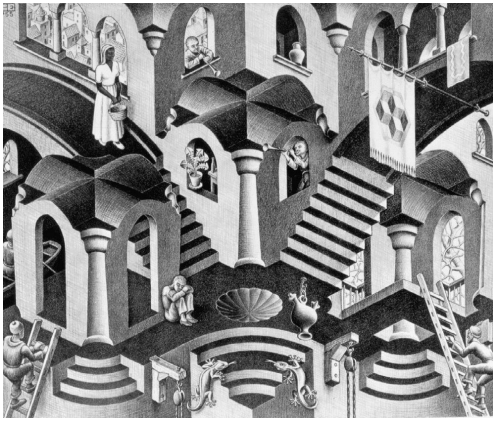


Figura 7: Figura con didascalia laterale (l'immagine riproduce la litografia *Concavo e convesso* di M. Escher)

```
\begin{SCfigure}[]
\centering
\includegraphics[width=0.5\textwidth]{ConcavoConvesso}
\caption{Figura con didascalia laterale}
\label{fig:sidecap}
\end{SCfigure}
```

8.5.2 Oggetti multipli

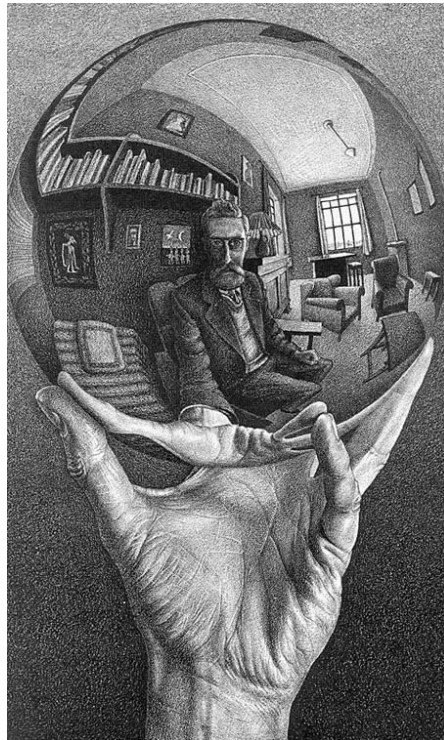
Il pacchetto subfig (che richiede caption) gestisce con `\subfloat` più (sotto)-figure (come la figura 8 nella pagina seguente) o (sotto)tabelle in un unico ambiente mobile e ne regola le didascalie molto finemente.

La figura 8 nella pagina successiva è stata ottenuta con un codice del tipo:

```
\begin{figure}
\centering
\subfloat[][\emph{Mano con sfera riflettente}]
{\includegraphics[width=.45\textwidth]{Sfera}} \quad
\subfloat[][\emph{Belvedere}]
{\includegraphics[width=.45\textwidth]{Belvedere}} \quad
\subfloat[][\emph{Cascata}]
{\includegraphics[width=.45\textwidth]{Cascata}} \quad
\subfloat[][\emph{Salita e discesa}]
{\includegraphics[width=.45\textwidth]{SalitaDiscesa}}
\caption{Figura composta da più sottofigure}
\label{fig:subfig}
\end{figure}
```

Si noti che:

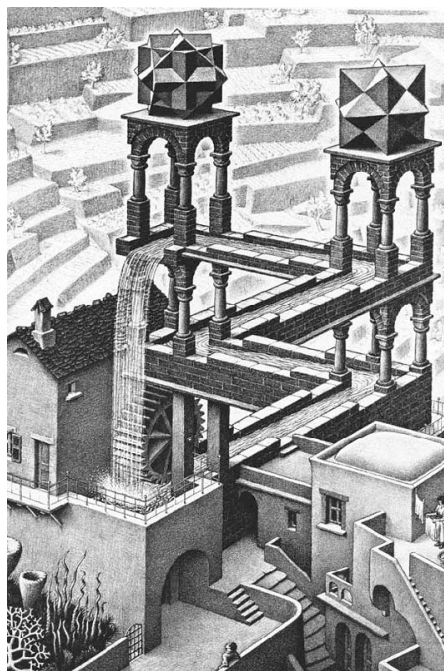
- nel primo argomento facoltativo di `\subfloat`, se usato, si mette la didascalia breve da mandare nel relativo indice (`\listoffigures` o `\listoftables`), purché si impostino adeguatamente i contatori *lofdepth* e *lotdepth* (si veda la documentazione di subfig);
- nel secondo ci va la didascalia che comparirà effettivamente sulla pagina;
- per riferirsi a un sottooggetto in particolare da altre parti del documento, `\label` va dato *dentro* il secondo argomento facoltativo immediatamente dopo la sottodidascalia.



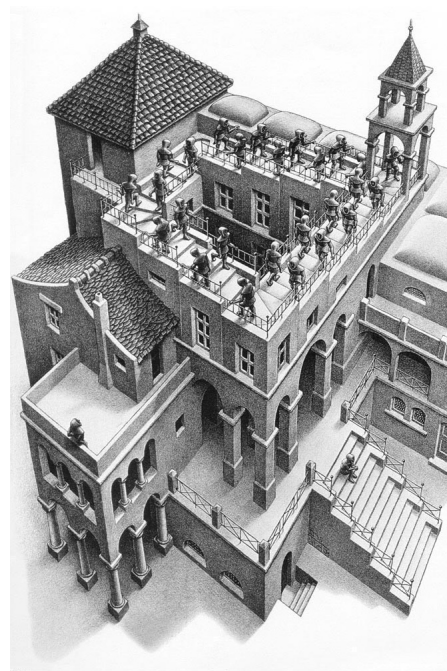
(a) *Mano con sfera riflettente*



(b) *Belvedere*



(c) *Cascata*



(d) *Salita e discesa*

Figura 8: Figura composta da più sottofigure (le immagini riproducono alcune litografie di M. Escher)

8.5.3 Oggetti immersi nel testo

In alcune circostanze può essere desiderabile “avvolgere” un oggetto con del testo, magari anche solo per movimentare la pagina. Risolve il problema il pacchetto `wrapfig`, particolarmente indicato perché interagisce correttamente con `caption` per personalizzare la didascalia.

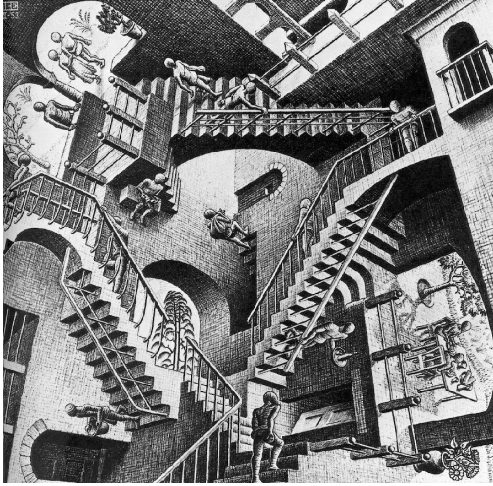


Figura 9: Figura immersa nel testo (l’immagine riproduce la litografia *Relatività* di M. C. Escher)

Il pacchetto definisce l’apposito ambiente `wrapfloat`, nel quale mettere l’oggetto con i comandi consueti. Ragioni estetiche impongono di circondarlo soltanto con testo continuo (come qui), rimandando più oltre eventuali altri oggetti o ambienti particolari. Tuttavia, anche operando correttamente il pacchetto non garantisce un risultato ottimale già alla prima composizione: potrebbero essere necessari numerosi aggiustamenti manuali della pagina, più o meno consistenti.

Come si può notare, una “wrapfig” correttamente ottenuta dà un risultato altamente professionale, ma si raccomanda di sfruttare questa possibilità soltanto in circostanze

davvero eccezionali, cioè praticamente *mai*. La buona riuscita dell’operazione richiede condizioni particolari: composizione a piena pagina e gabbia sufficientemente ampia da poter accogliere una figura larga la sua metà: in caso contrario, la colonna di testo accanto all’immagine risulterà troppo stretta e inevitabilmente solcata da ruscelli. La regola da seguire rimane la stessa: includere tutti gli oggetti fuori testo e lasciar fare a \LaTeX . Soltanto a lavoro ultimato, quando il risultato finale è davvero soddisfacente, sarà possibile ricollocarne qualcuno secondo il proprio gusto.

La figura 9 è stata ottenuta con un codice del tipo:

Talvolta si può voler “avvolgere” un oggetto con del testo.

```
\begin{wrapfloat}{figure}{I}{0pt}
\includegraphics[width=0.5\textwidth]{Relativo}
\caption{Figura immersa nel testo}
\end{wrapfloat}
```

Per avere un buon risultato, possono servire alcuni aggiustamenti manuali.

Come si vede nell’esempio precedente, l’ambiente `wrapfloat` richiede tre argomenti obbligatori:

```
\begin{wrapfloat}{<oggetto>}{<collocazione>}{<larghezza>}
```

Dove:

- `<oggetto>` indica il tipo di *oggetto* da includere (`figure` o `table`, da non confondere con gli ambienti omonimi);

Tabella 47: Preferenze di collocazione dell'ambiente `wrapfloat`

<code>r, R</code>	Sul lato destro del testo (<i>right</i>)
<code>l, L</code>	Sul lato sinistro del testo (<i>left</i>)
<code>i, I</code>	Sul margine interno (<i>inner</i>)
<code>o, O</code>	Sul margine esterno (<i>outer</i>)

- `<collocazione>`, che dice a `LATEX` dove mettere l'oggetto sulla pagina, accetta *una sola* delle otto opzioni della tabella 47, in minuscolo o in maiuscolo a seconda che si voglia mettere l'oggetto "esattamente qui nel testo" o si voglia creare un oggetto mobile, rispettivamente;
- `<larghezza>` specifica la larghezza dell'oggetto che, se nulla (0pt), equivale all'opzione assegnata a `\includegraphics`.

8.6 NUOVI OGGETTI MOBILI

In alcuni documenti accade talvolta che certi oggetti (approfondimenti, note storiche o biografiche, esercizi), considerati non appartenenti al flusso del testo principale, vengano separati da questo anche visivamente. Di solito essi sono inseriti in un riquadro munito di una didascalia.

Il pacchetto `float` gestisce facilmente oggetti di questo tipo. L'unica condizione è che essi non occupino più di una pagina. Il pacchetto permette di definire nuovi tipi di oggetti, analoghi alle figure e alle tabelle mobili ottenute con gli ambienti standard `figure` e `table`.

Per definire nuovi oggetti mobili `float` offre il comando `\newfloat` (analogo a `\newtheorem`). Il comando, da dare nel preambolo, ha tre argomenti obbligatori e uno facoltativo:

```
\newfloat{<nome>}{<posizionamento>}{<estensione>}[<sezione>]
```

Dove:

- `<nome>` è il nome della nuova classe di oggetti;
- `<posizionamento>` indica il posizionamento predefinito dei nuovi oggetti; come per le figure e le tabelle, un `<posizionamento>` è costituito da una sequenza di preferenze di posizionamento (di solito si dà `tbp`).
- `<estensione>` specifica l'estensione del file ausiliario che serve per produrre l'elenco dei nuovi oggetti (è analogo a `lof` e `lot` per l'elenco delle figure e delle tabelle);
- `<sezione>` specifica a quale tipo di unità di sezionamento (di regola `chapter` o `section`) collegare la numerazione degli oggetti.

Per esempio, le note biografiche contenute in questo articolo sono state definite con l'istruzione

```
\newfloat{nota}{tbp}{lon}
```

dove `lon` sta per *list of notes* (elenco delle note biografiche).

Ci sono i seguenti stili predefiniti per gli oggetti:

Donald Ervin Knuth è nato a Milwaukee nel 1938. Dopo la laurea in matematica e il dottorato, dal 1968 è membro della Stanford University. Ha creato T_EX e METAFONT. Nel 1992 si è ritirato dall'insegnamento regolare, anche se tiene tuttora sporadiche lezioni informali, da lui chiamate *Computer Musings* ("meditazioni assortite sui calcolatori"), presso la Stanford University. Knuth è noto per il suo umorismo. Paga 2,56 dollari per ogni errore scoperto nei suoi libri, perché «256 penny sono un dollaro esadecimale». I numeri di versione di T_EX e METAFONT convergono a π e a e , rispettivamente.

Nota biografica 1: Un esempio di nuovo oggetto mobile

- `plain` È lo stile delle figure e delle tabelle standard, con la differenza che la didascalia viene sempre posta *sotto* l'oggetto.
- `plaintop` È simile a `plain`, a parte il fatto che la didascalia viene sempre posta *sopra* l'oggetto.
- `boxed` Stampa l'oggetto dentro un riquadro e scrive la didascalia sotto il riquadro.
- `ruled` Riproduce lo stile delle tabelle dell'opera *Concrete Mathematics* di Donald Knuth: la didascalia è stampata in cima all'oggetto, delimitata da linee; un'altra linea è posta sotto l'oggetto.

Per impostare lo stile di una classe di oggetti si usa `\floatstyle`, che ha come argomento il nome di uno degli stili precedenti. Per esempio, lo stile delle note biografiche di questo articolo è stato impostato con

```
\floatstyle{boxed}
```

Lo stile specificato viene usato per tutte le classi di oggetti definite dopo di esso, fino a che non venga dato un altro `\floatstyle`.

Il comando `\floatname` specifica l'intestazione della didascalia dei nuovi oggetti mobili (analogo di *Figura* e *Tabella*). Per esempio:

```
\floatname{nota}{Nota biografica}
```

Ricapitolando, le note biografiche di questo articolo sono state definite scrivendo nel preambolo le istruzioni:

```
\floatstyle{boxed}
\newfloat{nota}{tbp}{lon}
\floatname{nota}{Nota biografica}
```

Dopo di che, nel documento si usa l'ambiente `nota` al solito modo. Per esempio, la nota biografica 1 è stata prodotta con il codice

```
\begin{nota}[tb]
\textbf{Donald Ervin Knuth} è nato a Milwaukee nel~1938.
\dots
\caption{Un esempio di nuovo oggetto mobile}
\end{nota}
```

C'è anche il comando `\listof` che stampa l'elenco di tutti gli oggetti di un dato tipo (analogo a `\listoffigures` e `\listoftables`):

```
\listof{<nome>}{<titolo>}
```

Dove:

Leslie Lamport, nato nel 1941 a New York, è l'autore di \LaTeX . Laureatosi in matematica al MIT, ha ottenuto un master e un dottorato in matematica alla Brandeis University. Lamport ha lavorato come informatico al Massachusetts Computer Associates, alla Digital Equipment Corporation e alla Compaq. Nel 2001 si è unito al gruppo di ricerche di Microsoft, a Mountain View. Lamport è accreditato come autore del seguente aforisma: «Capisci di avere un sistema distribuito quando il blocco di un calcolatore di cui non avevi mai sentito parlare ti impedisce di concludere il tuo lavoro».

Nota biografica 2: Un altro esempio di nuovo oggetto mobile

- $\langle nome \rangle$ è il nome della classe di oggetti mobili definita con `\newfloat`;
- $\langle titolo \rangle$ indica l'intestazione usata nell'elenco degli oggetti (e anche nelle testatine, se la relativa pagina le contiene).

Per esempio, per ottenere l'elenco delle note biografiche di questo articolo si scrive

```
\listof{nota}{Elenco delle note biografiche}
```

nel punto in cui si desidera che compaia l'elenco.

Se si vuole che tutti gli oggetti mobili del proprio documento abbiano lo stesso stile tipografico, c'è il comando

```
\restylefloat{\langle nome \rangle}
```

che ridisegna la classe di oggetti mobili indicata con $\langle nome \rangle$ secondo lo stile valido in quel momento (dichiarato con `\floatstyle`). Il comando `\restylefloat` permette di modificare lo stile degli oggetti mobili definiti con gli ambienti standard `figure` e `table`:

```
\floatstyle{ruled}
\restylefloat{figure}
```

cioè si possono applicare gli stili di float anche agli oggetti non definiti con `\newfloat`.

Non è raro che un documento composto con \LaTeX richieda di realizzare un disegno. Se è molto complesso o se si ha poco tempo a disposizione per imparare un nuovo linguaggio, la soluzione più semplice e veloce è produrlo con un software di grafica vettoriale esterno, registrarlo in formato PDF e includerlo nel documento come al solito. In questo modo, però, viene meno la perfetta integrazione di testo e figure tipica di \LaTeX e garantita invece da alcuni pacchetti specializzati nel disegno e compresi nelle principali distribuzioni.

I più importanti di essi sono *TikZ* e *PSTricks*, entrambi potentissimi e sostanzialmente equivalenti nell'uso comune. Il primo è direttamente compatibile con \PDF\LaTeX , mentre il secondo, basato sul linguaggio PostScript, lo è solo adottando qualche espediente, e anche per questo motivo, nonostante le maggiori attitudini al calcolo e al disegno tridimensionale, gli utenti preferiscono generalmente il suo diretto concorrente.

Questo capitolo, basato essenzialmente su [Tantau, 2013], introduce alle nozioni fondamentali necessarie per produrre semplici disegni con *TikZ*.

9.1 INTRODUZIONE

Un'immagine *vettoriale* è un disegno i cui elementi sono memorizzati mediante una descrizione sintattica delle loro caratteristiche geometriche e delle loro proprietà (colore, spessore delle linee, eccetera), anziché mediante una conversione in matrice di punti colorati, come avviene invece in un'immagine *raster*. Ciò permette da un lato di ridurre le dimensioni del file corrispondente e dall'altro di ottenere un'elevata qualità indipendentemente dalla risoluzione.

TikZ è un pacchetto per produrre disegni vettoriali a partire da una descrizione geometrica, che fornisce comandi di alto livello basandosi a propria volta su PGF, un linguaggio di livello più basso (un po' come \LaTeX fa rispetto a \TeX , per capirci). PGF è l'acronimo di *Portable Graphics Format* ("formato di disegni portabile"), mentre *TikZ* è l'acronimo ricorsivo di *TikZ ist kein Zeichenprogramm* ("*TikZ non è un programma di disegno*").

L'idea di *programmare un disegno* potrebbe sembrare a prima vista esoterica ma, a pensarci bene, non lo è molto di più dell'idea di *programmare un documento* alla base di \LaTeX . Si tratta di vincere una certa (e comprensibile) perplessità iniziale e di investire energie per imparare come fare, ma la fatica sarà ampiamente ripagata: se si desidera realizzare un lavoro di qualità senza compromessi, in cui testo e grafica si integrino perfettamente, *TikZ* è irrinunciabile e, non meno importante, permette di disegnare "nello spirito di \LaTeX ".

Ricorrere o meno a *TikZ* dipende dalle proprie esigenze di qualità tipografica e dal tipo di disegni da realizzare, ma prima di tutto dal *tempo* a disposizione. Per *padroneggiare* *TikZ* ci vuole *molto* tempo, e non è detto

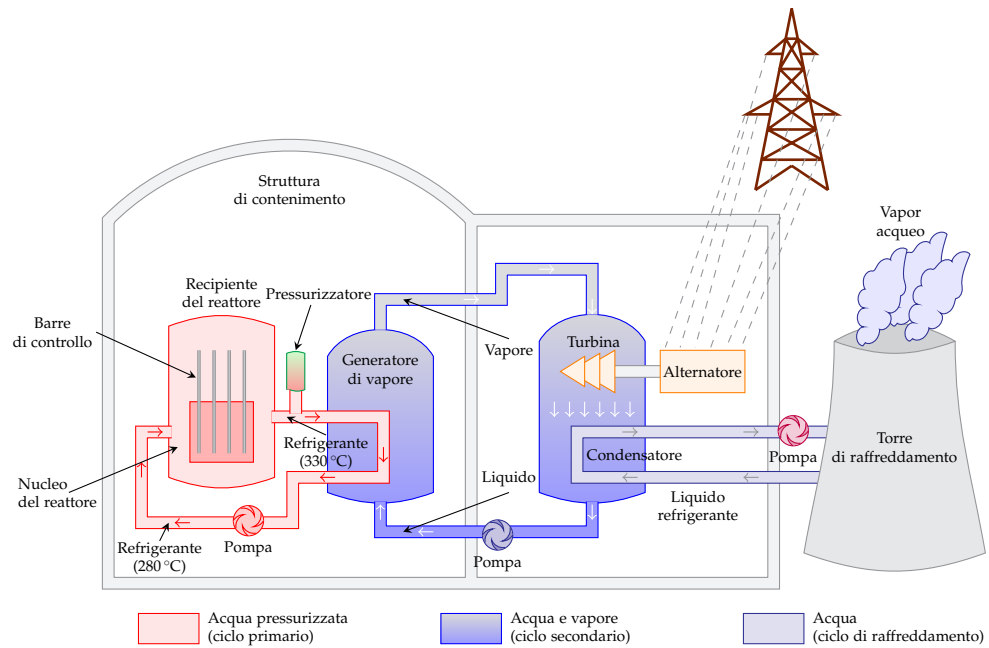


Figura 10: Schema del funzionamento di una centrale nucleare (l'esempio è tratto da [TEXAMPLE](#))

che l'utente ce l'abbia. In tal caso, appoggiarsi a un programma esterno per produrre ciò che serve e includerlo poi nel documento con \LaTeX è un compromesso onorevole.

Se invece si opta per `TikZ`, si sappia che schemi semplici si realizzano rapidamente (anche grazie agli esempi contenuti nella documentazione e a quelli disponibili in Rete su [TEXAMPLE](#), per esempio), ma per disegni complessi come lo schema del funzionamento di una centrale nucleare riportato nella figura 10, tanto per fare un esempio all'estremo opposto, ci vuole molto più tempo: si può fare anche quello, perché con `TikZ` si può disegnare virtualmente *qualunque* cosa.

Per il momento, le potenzialità di `TikZ` nel disegno tridimensionale sono piuttosto limitate, perciò ci occuperemo esclusivamente di disegni in due dimensioni.

9.2 FERRI DEL MESTIERE

9.2.1 Caricamento, librerie e ambienti

`TikZ` si carica nel solito modo:

```
\usepackage{tikz}
```

e a propria volta invoca automaticamente i seguenti pacchetti: `graphicx`, `keyval` e `xcolor`. Se fosse necessario caricarli con qualche opzione, dunque, si ricordi di farlo *prima*.

Data la sua versatilità, per organizzare meglio il codice e ridurre i tempi di composizione del documento l'autore gli ha dato struttura modulare basata sul concetto di libreria. Una *libreria* è una porzione di codice del pacchetto specializzata nell'eseguire una particolare funzione, da caricare solo se il

disegno da realizzare lo richiede effettivamente. Le librerie si caricano *nel preambolo dopo TikZ*, scrivendone il $\langle nome \rangle$ nell'argomento di

```
\usetikzlibrary{\langle nome \rangle}
```

che, come al solito, permette di caricarne più d'una, separandole con la virgola. Le librerie di TikZ sono numerose e si distinguono in *interne*, cioè già presenti nel pacchetto (se ne veda la documentazione) ed *esterne*, cioè veri e propri pacchetti che si caricano come si è appena visto. Di seguito si descrivono brevemente alcune librerie del primo tipo per scopi generici (altre, più specializzate, sono descritte nel paragrafo 9.10.1 a pagina 186):

- `calc` permette di eseguire calcoli sulle coordinate;
- `graphs` semplifica la sintassi per disegnare i grafi;
- `intersections` permette di determinare i punti di intersezione di due percorsi;
- `matrix` permette di realizzare matrici di nodi;
- `patterns` permette di riempire i percorsi con motivi di fantasia;
- `plotmarks` mette a disposizione ulteriori marcatori oltre a quelli predefiniti;
- `shadows` permette di aggiungere le ombre agli elementi del disegno;
- la famiglia `decorations` permette di aggiungere elementi di decorazione al disegno;
- la famiglia `shapes` permette di inserire forme già pronte per l'uso.

L'ambiente principale di TikZ è `tikzpicture`, che presenta la seguente sintassi:

```
\begin{tikzpicture} [ \langle opzioni globali \rangle ]
\langle istruzioni di TikZ \rangle
\end{tikzpicture}
```

dove:

- le $\langle opzioni globali \rangle$, da separare con la virgola se più d'una, agiscono su *tutte* le istruzioni presenti nell'ambiente, se non ridefinite localmente;
- $\langle istruzioni di TikZ \rangle$ sono le istruzioni necessarie per realizzare il disegno.

Si noti inoltre che:

- TikZ sa riconoscere e ignorare gli spazi *non* significativi, che pertanto si possono aggiungere per aumentare la leggibilità del codice;
- le opzioni si applicano *rispettando l'ordine di scrittura*, perciò quando si dichiarano più opzioni incompatibili tra loro (come due colori diversi a uno stesso elemento del disegno) quella effettivamente assegnata è l'opzione dichiarata *per ultima*.

L'alternativa all'ambiente `tikzpicture` è il comando `\tikz`, con la sintassi:

```
\tikz {⟨istruzioni di TikZ⟩}
```

Si può trattare in modo particolare una determinata porzione di codice mettendola nell'ambiente `scope` (qui tradotto con *ambito*), da annidare a propria volta in un ambiente `tikzpicture` insieme alle altre istruzioni. Le opzioni di `scope` saranno *locali*, cioè valide *solo al suo interno*:

```
\begin{tikzpicture} [⟨opzioni globali⟩]
...
\begin{scope} [⟨opzioni locali⟩]
⟨istruzioni di TikZ⟩
\end{scope}
...
\end{tikzpicture}
```

Poiché permettono di agire in una volta sola su porzioni anche ampie di codice, gli *ambiti* sono molto utili per trattare come un'unità indivisibile un elemento complesso del disegno, cioè un particolare realizzato a propria volta tramite un insieme di istruzioni.

I disegni realizzati con TikZ sono oggetti in linea con il testo. Si rendono mobili inserendo l'ambiente `tikzpicture` o il comando `\tikz` in un normale ambiente `figure` come al solito:

```
\begin{figure}
\centering
\begin{tikzpicture} [⟨opzioni globali⟩]
⟨istruzioni di TikZ⟩
\end{tikzpicture}
\caption{⟨didascalia⟩}
\label{fig:⟨etichetta⟩}
\end{figure}
```

Documenti di soli disegni

Qualche circostanza particolare potrebbe richiedere un documento costituito esclusivamente da uno o più disegni. In tal caso si consiglia di ricorrere alla classe `standalone`, che si occupa di scontornare automaticamente le immagini (se ne consulti la documentazione).

Altre volte può essere utile generare per ogni disegno del documento un file di output inserito automaticamente nel punto giusto (per ridurre i tempi di composizione, per esempio, oppure quando si vogliono riutilizzare i disegni in un poster). La libreria interna `external` permette di farlo automaticamente.

9.2.2 Sistemi di riferimento

TikZ non richiede di specificare *a priori* un sistema di riferimento: infatti individua la posizione dei punti sulla *tela*, cioè la porzione della pagina riservata al disegno, tramite le loro coordinate e di volta in volta determina il sistema di riferimento adottato a partire dalla sintassi delle singole istruzioni. In questo modo, l'utente può usare i diversi sistemi di riferimento riconosciuti dal pacchetto in uno stesso disegno e addirittura in una stessa istruzione. Quelli più comunemente utilizzati sono il sistema di riferimento *cartesiano* e quello *polare*.

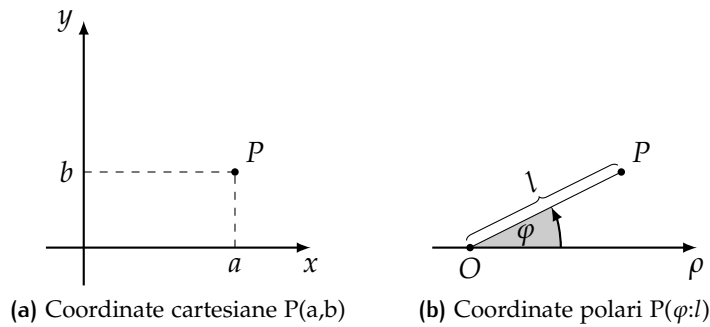


Figura 11: Coordinate cartesiane e polari

Si noti che TikZ accetta anche altri sistemi di riferimento (qui ignorati per semplicità) descritti nella documentazione del pacchetto.

La sintassi per individuare un punto del piano in coordinate cartesiane è

$(\langle x \rangle, \langle y \rangle)$

dove $\langle x \rangle$ e $\langle y \rangle$, separate con la virgola e racchiuse tra parentesi tonde, sono rispettivamente l'ascissa e l'ordinata del punto, cioè le proiezioni del punto sugli assi coordinati (si veda la figura 11a).

La sintassi per individuare un punto del piano in coordinate polari è

$(\langle \varphi \rangle : \langle \rho \rangle)$

dove $\langle \varphi \rangle$ e $\langle \rho \rangle$, separate con i due punti e racchiuse tra parentesi tonde, sono rispettivamente la coordinata angolare in gradi e la coordinata radiale, come mostra la figura 11b.

9.2.3 Espressioni coordinate

Oltre che nei modi appena visti, si possono indicare le coordinate di un punto (una sola o entrambe) anche tramite semplici espressioni matematiche, ricordandosi di racchiuderle tra graffe se contengono a propria volta parentesi tonde.

TikZ riconosce gli operatori più comuni e i seguenti degni di nota:

- l'operatore \wedge di elevamento a potenza;
- l'operatore postfixato r che converte in gradi una misura di ampiezza in radianti;
- gli operatori di confronto per uguaglianza $==$ e disuguaglianza $!=$;
- gli operatori logici di congiunzione $\&\&$ e disgiunzione inclusiva $||$.

Inoltre il pacchetto riconosce le costanti matematiche pigreco π e numero di Nepero e .

La tabella 48 nella pagina successiva elenca alcune funzioni predefinite di uso più frequente, tra cui si segnalano le funzioni circolari e le loro inverse, che richiedono e producono angoli in gradi e la funzione `deg`, che converte in gradi una misura in radianti. Gli argomenti delle funzioni vanno racchiusi tra parentesi tonde e, se più d'uno, separati con la virgola. Si noti che il valore di una funzione matematica è sempre un numero perciò, per trasformarlo in una dimensione, occorre moltiplicarlo per una lunghezza unitaria.

Tabella 48: Alcune funzioni matematiche predefinite in TikZ

Funzione	Effetto	Funzione	Effetto
abs	valore assoluto	frac	parte decimale
acos	arcocoseno	int	parte intera
asin	arcoseno	ln	logaritmo naturale
atan	arcotangente	log10	logaritmo decimale
atan2	arcotangente in due variabili	max	massimo
ceil	parte intera superiore	min	minimo
cos	coseno	round	arrotonda
cot	cotangente	sin	seno
deg	radianti → gradi	sqrt	radice quadrata
floor	parte intera inferiore	tan	tangente

Ecco due esempi di espressioni, ciascuna delle quali può rappresentare una singola coordinata di un punto:

```
{sqrt(2)/2-sin(pi/4 r)}
```

oppure

```
{1cm*(sqrt(2)/2-sin(45))}
```

Si possono anche sommare lunghezze espresse in unità di misura differenti. In tal caso, i numeri adimensionali vengono automaticamente moltiplicati per la lunghezza di 1 pt, cioè $-5+1\text{cm}+2\text{pt}$ equivale a $1\text{cm}-3\text{pt}$.

9.2.4 Percorsi

In TikZ un *percorso* è una successione di linee (segmenti, spezzate, curve aperte o chiuse) non necessariamente contigue, come quello mostrato nella figura 12 a fronte. Il comando generale per definire un percorso è `\path`, che presenta la seguente sintassi:

```
\path [opzioni] istruzioni;
```

Si noti che:

- le *opzioni*, da separare con la virgola se più d'una, agiscono solo sulle *istruzioni* che le seguono;
- ogni istruzione, che può svilupparsi anche su più righe per esigenze di spazio o per maggiore ordine nel codice, corrisponde a una diversa *linea*, che viene realizzata rispettando l'ordine di scrittura nel file sorgente;
- il punto e virgola deve terminare *obbligatoriamente* la sequenza di istruzioni.

L'istruzione più semplice corrisponde allo spostamento dall'ultima posizione nel percorso, detta *posizione corrente*, in un altro punto eventualmente differente, che diventa la posizione corrente per l'istruzione successiva, e si realizza semplicemente dichiarando le coordinate della destinazione nel percorso. Per esempio, il comando `\path (1,0);` indica di spostarsi nel punto di coordinate (1,0). Ogni percorso comincia normalmente con un'istruzione

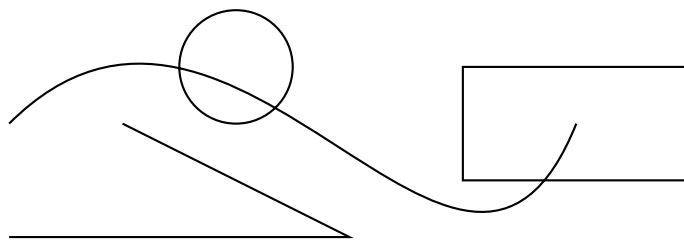


Figura 12: Esempio di percorso elaborato in TikZ

di spostamento, che di fatto definisce la prima posizione corrente. Di qui in avanti, se non diversamente specificato, si assume che la posizione corrente all'inizio di un'istruzione sia il punto finale della linea precedente.

Il comando `\path` definisce il percorso e aggiorna dimensioni e posizione della tela, ma *non* realizza il disegno a meno che non lo si specifichi tra le opzioni, perciò di fatto si usa raramente. Le opzioni per disegnare, riempire e sfumare un percorso sono rispettivamente `draw`, `fill` e `shade`. Per ciascuna di esse TikZ definisce una scorciatoia di più agile utilizzo.

9.3 DISEGNARE IL PERCORSO

Il comando per disegnare un percorso è `\draw`, scorciatoia di `\path [draw]`, dal quale eredita la sintassi:

```
\draw [<opzioni>] <istruzioni>;
```

Esso permette di tracciare direttamente segmenti, rettangoli, circonferenze, ellissi e loro archi, curve di Bézier, grafici di funzioni e, utilizzando le librerie, perfino forme geometriche più elaborate.

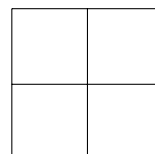
Si noti che, se non ridefinite attraverso le `<opzioni>`, le caratteristiche del tratto (colore, spessore, eccetera) sono quelle predefinite. Proprio in virtù di tali caratteristiche, la differenza che passa tra due disegni realizzati l'uno con un comando diverso per ogni istruzione e l'altro con un unico comando per tutte le istruzioni è la stessa che corre tra due schizzi a mano libera eseguiti l'uno con un gesto differente per ogni elemento e l'altro con un unico gesto. Perciò, specie in presenza di linee contigue, si consiglia di definire un singolo percorso ogni volta che è possibile.

A lavoro in corso, può essere molto utile, ma non obbligatorio, aiutarsi con una griglia, che di solito si rimuove a disegno completato. L'istruzione

```
grid [<opzioni>] (<vertice finale>)
```

definisce la griglia rettangolare di vertici opposti coincidenti con la posizione corrente e il `<vertice finale>`, individuato dalle sue coordinate con uno dei metodi esposti nel paragrafo 9.2.2 a pagina 154. Eccone un esempio:

```
\begin{tikzpicture}
\draw (0,0) grid (2,2);
\end{tikzpicture}
```



Per ottenere linee di colore grigio anziché nero si può utilizzare l'opzione `help lines`, come si è fatto in tutto questo capitolo.

Il *passo*, cioè la larghezza delle maglie della griglia, è pari a 1 cm per impostazione predefinita, ma lo si può modificare con la chiave `step`, da dichiarare indifferentemente tra le opzioni della griglia o del percorso, che presenta la sintassi seguente:

```
step=<valore>
```

dove *<valore>* può essere espresso come multiplo o frazione dell'unità di misura degli assi coordinati. Poiché la griglia serve a individuare più facilmente i punti notevoli del disegno, il passo dovrebbe essere scelto in base alle dimensioni del disegno stesso e alla posizione dei suoi elementi.

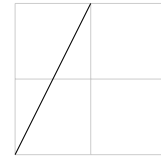
9.3.1 Segmenti

L'istruzione

```
-- (<punto finale>)
```

nella quale l'operatore `--` è costituito da due trattini in successione, definisce il *segmento* che unisce la posizione corrente con il *<punto finale>*. Eccola all'opera:

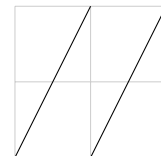
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) -- (1,2);
\end{tikzpicture}
```



Si osservi che la griglia è tracciata *sotto* il segmento, perché le istruzioni vengono eseguite seguendo l'ordine di scrittura.

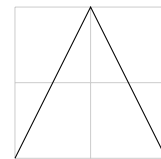
Un percorso appena più elaborato è costituito da più istruzioni all'interno dello stesso comando. Così, per disegnare due segmenti paralleli si può scrivere:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) -- (1,2) (1,0) -- (2,2);
\end{tikzpicture}
```



e per disegnare una spezzata:

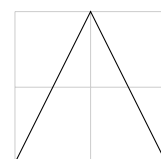
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) -- (1,2) -- (2,0);
\end{tikzpicture}
```

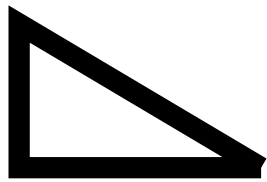


Si noti che i due segmenti vengono raccordati come se le linee fossero tracciate con un unico gesto.

Per disegnare un percorso chiuso, anziché ripetere le coordinate del punto iniziale va usata l'istruzione `-- cycle`, che garantisce la perfetta chiusura della linea, come mostra la figura 13 a fronte. Per esempio, un triangolo si disegna in questo modo:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) -- (1,2) -- (2,0) -- cycle;
\end{tikzpicture}
```





(a) Linea chiusa ottenuta ripetendo le coordinate iniziali

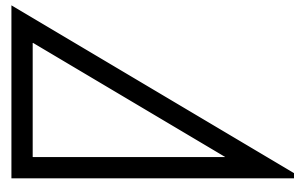
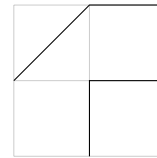
(b) Linea chiusa ottenuta con l'operatore `-- cycle`

Figura 13: Disegnare un percorso chiuso

9.3.2 Coordinate relative

Come si è accennato nel paragrafo 9.2.2 a pagina 154, oltre che rispetto all'origine del sistema di riferimento si può individuare la posizione di un punto anche rispetto a un altro punto del percorso già definito, operando una traslazione *temporanea* del sistema di riferimento. Un esempio chiarirà le idee. Il codice

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (1,0) -- ++(0,1) -- ++(1,0)
      ++(90:1) -- (1,2) -- (0,1);
\end{tikzpicture}
```



va letto così:

1. `(1,0) -- ++(0,1)` disegna il segmento da $(1,0)$ a $(1,1)$;
2. `++(0,1) -- ++(1,0)` disegna il segmento da $(1,1)$ a $(2,1)$;
3. `++(1,0) ++(90:1)` si sposta da $(2,1)$ in $(2,2)$;
4. `++(90:1) -- (1,2)` disegna il segmento da $(2,2)$ a $(1,2)$ (coordinate assolute);
5. `(1,2) -- (0,1)` disegna il segmento da $(1,2)$ a $(0,1)$.

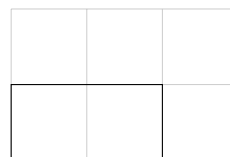
9.3.3 Rettangoli

L'istruzione

```
rectangle (<vertice finale>)
```

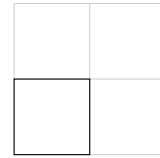
definisce il rettangolo con i lati paralleli agli assi coordinati, con un vertice nella posizione corrente e quello opposto nel *<vertice finale>*, che diventa la posizione corrente dell'istruzione successiva. La scelta dei vertici è indifferente, purché siano opposti. Eccola all'opera:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (0,0) rectangle (2,1);
\end{tikzpicture}
```



Con la stessa istruzione si può disegnare anche un quadrato:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,1) rectangle (1,0);
\end{tikzpicture}
```



Si noti che non è necessario dichiarare le unità di misura.

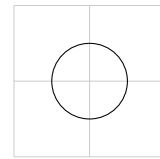
9.3.4 Circonferenze ed ellissi

L'istruzione

```
circle (<raggio>)
```

definisce la circonferenza di $\langle \text{raggio} \rangle$ assegnato con centro nella posizione corrente. Eccola all'opera per disegnare una circonferenza con centro nel punto (1,1) e raggio di 5 mm:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (1,1) circle (5mm);
\end{tikzpicture}
```

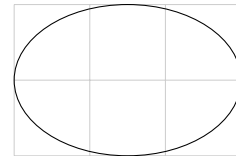


L'istruzione

```
ellipse (<semiasse x> and <semiasse y>)
```

definisce l'ellisse con gli assi paralleli agli assi coordinati e centro nella posizione corrente, della quale sono assegnate le lunghezze dei due semiassi. In particolare, il $\langle \text{semiasse } x \rangle$ è la lunghezza del semiasse parallelo all'asse delle ascisse e il $\langle \text{semiasse } y \rangle$ quella del semiasse parallelo all'asse delle ordinate. Eccola all'opera per disegnare un'ellisse con centro nel punto (1,5;1) e semiassi orizzontale e verticale di 1,5 cm e 1 cm rispettivamente:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (1.5,1) ellipse (1.5 and 1);
\end{tikzpicture}
```



Anche in questo caso la posizione corrente non varia dopo l'istruzione. Si noti che *circle* ed *ellipse* sono sinonimi, perciò permettono di disegnare indifferentemente circonferenze o ellissi a seconda della sintassi usata.

9.3.5 Archi

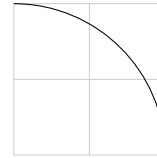
Archi di circonferenza e di ellisse

L'istruzione

```
arc (<angolo di partenza> : <angolo d'arrivo> : <raggio>)
```

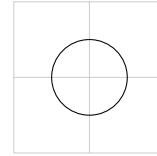
definisce l'arco di circonferenza di $\langle \text{raggio} \rangle$ assegnato che parte dalla posizione corrente, per il quale gli angoli di partenza e d'arrivo sono le coordinate angolari dei due estremi rispetto al *centro di curvatura*. L'esempio seguente traccia l'arco di 2 cm di raggio, che parte da (2,0) a est (0°) del centro di curvatura e termina a nord (90°) di esso:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (2,0) arc (0:90:2);
\end{tikzpicture}
```



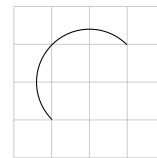
In questo caso, angoli che differiscono per un multiplo intero di 360° *non* sono considerati equivalenti, perciò il codice seguente disegna un'intera circonferenza:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (1.5,1) arc (0:360:0.5);
\end{tikzpicture}
```



Quando è noto il centro di curvatura anziché il punto di partenza, le coordinate relative si rivelano particolarmente utili. L'esempio seguente traccia l'arco con centro di curvatura nel punto di coordinate (1, 1), raggio di 7 mm e angoli di partenza e d'arrivo di 45° e 225° rispettivamente:

```
\begin{tikzpicture}
\draw [help lines]
(0,0) grid [step=0.5] (2,2);
\draw (1,1) ++(45:7mm) arc (45:225:7mm);
\end{tikzpicture}
```

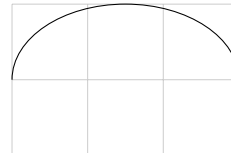


Analogamente, l'istruzione

```
arc (<angolo di partenza>:<angolo d'arrivo>:<semiasse x> and <semiasse y>)
```

definisce l'arco che parte dalla posizione corrente, appartenente all'ellisse con le caratteristiche geometriche descritte nel paragrafo precedente. Eccone un esempio:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (3,1) arc (0:180:1.5 and 1);
\end{tikzpicture}
```



Anche qui, angoli che differiscono per un multiplo intero di 360° *non* sono considerati equivalenti.

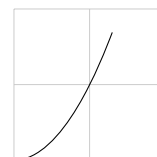
Archi di parabola

L'istruzione

```
parabola (<punto finale>)
```

definisce l'arco di parabola che parte dalla posizione corrente e termina nel *<punto finale>*, con vertice nel punto iniziale e asse di simmetria parallelo all'asse delle ordinate. Eccola all'opera:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) parabola (1.3, 1.3^2);
\end{tikzpicture}
```



L'istruzione più completa

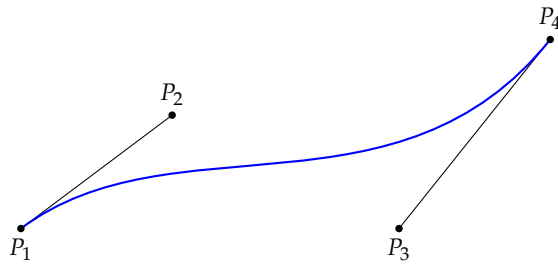
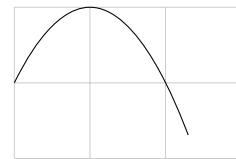


Figura 14: Curva di Bézier cubica

```
parabola bend (<vertice>) (<punto finale>)
```

definisce, purché esista, l'arco di parabola di *<vertice>* e *<punto finale>* assegnati, con asse parallelo all'asse delle ordinate e punto iniziale nella posizione corrente:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (0,1) parabola bend (1,2)
      (2.3, 2-1.3^2);
\end{tikzpicture}
```



9.3.6 Curve di Bézier

Una *curva di Bézier cubica* è una particolare curva piana definita univocamente da quattro punti (si veda la figura 14): i due estremi della curva P_1 e P_4 e i due *punti di controllo* P_2 e P_3 , che godono della proprietà che i segmenti P_1P_2 e P_3P_4 sono tangenti alla curva nei suoi estremi. TikZ permette di disegnare curve di Bézier in diversi modi, descritti nelle prossime sezioni.

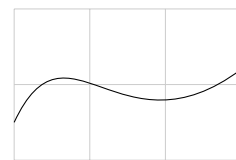
Curve definite dalla posizione dei punti notevoli

Il più completo di essi richiede di dichiarare i quattro punti notevoli nell'istruzione

```
.. controls (<punto di controllo1>) and (<punto di controllo2>) .. (<punto finale>)
```

con ovvio significato dei parametri, nella quale il punto iniziale, non espresso, coincide con la posizione corrente. Eccone un esempio:

```
\begin{tikzpicture}
\draw [help lines] (0,-1) grid (3,1);
\draw (0,-0.5) .. controls (0.7,1) and
      (1.4,-1) .. (3,0.2);
\end{tikzpicture}
```



Combinando più linee in un percorso, si possono costruire curve più complesse:

Curve definite tramite le proprietà geometriche

In alternativa al metodo appena esaminato, si può definire la curva più intuitivamente dichiarandone le proprietà geometriche con una serie di opzioni, attraverso l'istruzione

to [*opzioni*] (*punto finale*)

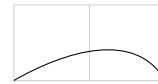
dove il punto iniziale coincide con la posizione corrente e l'operatore to sostituisce --. Ogni opzione non dichiarata è sostituita con il proprio valore predefinito, perciò più opzioni compatibili si dichiarano, più accuratamente si potrà controllare la forma della curva. Di seguito si descrivono le opzioni più importanti (per il loro elenco completo si rimanda il lettore alla documentazione del pacchetto).

Le opzioni

out=*angolo di partenza*, in=*angolo d'arrivo*

definiscono le direzioni delle due semirette tangenti alla curva nei suoi estremi, espresse in forma di coordinata angolare rispetto a essi. I valori predefiniti, utilizzati *solo se c'è* almeno un parametro che definisce la curva, sono 45° e 135° rispettivamente. Il codice

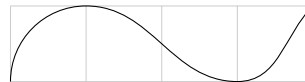
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,1);
\draw (0,0) to [out=30, in=120] (2,0);
\end{tikzpicture}
```



disegna una curva che parte da (0,0) con un *angolo di partenza* (out) di 30° e termina in (2,0) con un *angolo d'arrivo* (in) di 120°.

Come al solito, combinando più linee in un percorso si possono costruire curve più complesse:

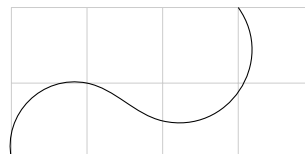
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,1);
\draw (0,0) to [out=90, in=180] (1,1)
to [out=0, in=180] (3,0)
to [out=0, in=-135] (4,1);
\end{tikzpicture}
```



Curve composte mediante l'algoritmo di Hobby

Infine, un ulteriore metodo consiste nell'appoggiarsi alla libreria esterna hobby, che per determinare la curva di Bézier composta passante per i punti assegnati mette a disposizione l'algoritmo di Hobby. Ogni coppia di punti consecutivi individua un differente tratto della curva che ha la proprietà di condividere le direzioni delle tangenti negli estremi con i due tratti adiacenti. Per attivare l'algoritmo, dopo averne caricato la libreria come al solito basta dichiarare l'opzione use Hobby shortcut e sostituire l'operatore -- con .., come mostra l'esempio seguente:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw [use Hobby shortcut]
(0,0) .. (1,1) .. (2,0.5) .. (3,2);
\end{tikzpicture}
```



La libreria permette di modificare singolarmente i parametri della curva.

9.3.7 Grafici di funzione

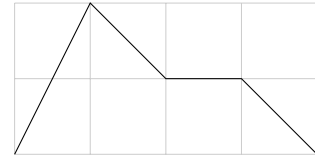
Mediante l'istruzione `plot`, TikZ permette di disegnare anche grafici di funzione, dove i punti iniziale e finale della linea corrispondono al primo e all'ultimo punto del grafico rispettivamente.

L'istruzione

```
plot [<opzioni>] coordinates {(<punto1>) (<punto2>) (<...>) (<ultimo punto>)}
```

definisce la poligonale sui punti assegnati. Eccone un esempio:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw plot coordinates {(0,0) (1,2) (2,1)
                       (3,1) (4,0)};
\end{tikzpicture}
```



L'istruzione

```
plot [<opzioni>] (<espressioni coordinate>)
```

definisce la poligonale sui punti individuati da una coppia di *<espressioni coordinate>* della forma descritta nel paragrafo 9.2.3 a pagina 155 e da esprimere in funzione di un parametro (per impostazione predefinita la macro `\x`) che rappresenta la variabile indipendente. Si può personalizzare il grafico mediante opzioni specifiche da indicare *dopo* `plot`. Di seguito se ne descrivono le principali.

Opzioni di `plot`

L'opzione

```
domain=<valore iniziale>:<valore finale>
```

limita la variabilità del parametro tra il *<valore iniziale>* e il *<valore finale>* (il valore predefinito è `domain=-5:5`).

L'opzione

```
samples=<numero>
```

imposta il *<numero>* di valori da assegnare al parametro, uniformemente distribuiti nel dominio (il valore predefinito è `samples=25`).

L'opzione

```
samples at={<elenco>}
```

attribuisce direttamente al parametro i valori presi ordinatamente da un *<elenco>* di numeri separati dalla virgola e racchiusi tra parentesi graffe. Si può comprimere l'elenco mediante i puntini di sospensione, come mostrano le due scritture seguenti, del tutto equivalenti:

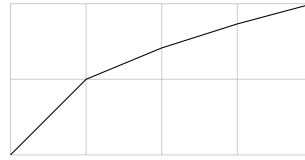
```
{1,1.5,...,3}
{1, 1.5, 2, 2.5, 3}
```

L'opzione

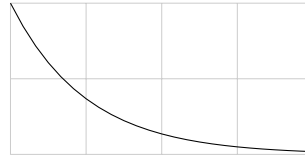
```
variable=<macro>
```

imposta il nome della macro (`\x` per impostazione predefinita). Gli esempi seguenti mostrano all'opera le opzioni appena descritte.

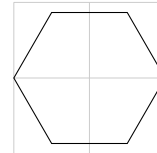
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw plot [domain=0:4, samples=5]
(\x, {sqrt(\x)});
\end{tikzpicture}
```



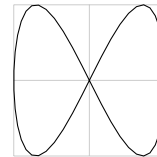
```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw plot [domain=0:4] (\x, {2*exp(-\x)});
\end{tikzpicture}
```



```
\begin{tikzpicture}
\draw [help lines] (-1,-1) grid (1,1);
\draw plot [samples at={0,60,...,360}]
(\x:1);
\end{tikzpicture}
```



```
\begin{tikzpicture}
\draw [help lines] (-1,-1) grid (1,1);
\draw plot [variable=\t, domain=0:360,
samples=50]
({cos(\t)}, {sin(2*\t)});
\end{tikzpicture}
```



Si noti che `plot` riconosce le coordinate polari e che permette di disegnare curve parametriche.

Infine, si descrivono altre due opzioni accettate da entrambe le forme dell'istruzione. L'opzione

`mark=<simbolo>`

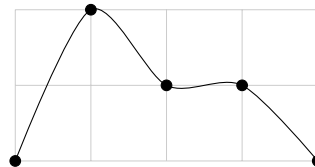
inserisce un marcatore in ogni punto assegnato del grafico. I simboli predefiniti sono `*`, `+` e `x` per ottenere un circoletto pieno, una croce greca e una x rispettivamente. La libreria `plotmarks` (descritta nella documentazione del pacchetto) ne fornisce molti altri.

L'opzione

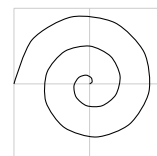
`smooth`

anziché con segmenti, raccorda i punti del grafico mediante curve di Bézier cubiche che condividono a coppie consecutive la direzione della tangente negli estremi. Di fatto, si tratta di un ulteriore metodo per disegnare curve di Bézier. Eccone due esempi:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw plot [mark=*, smooth] coordinates
{(0,0) (1,2) (2,1) (3,1) (4,0)};
\end{tikzpicture}
```

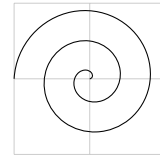


```
\begin{tikzpicture}
\draw [help lines] (-1,-1) grid (1,1);
\draw plot [variable=\t, domain=0:900,
smooth]
(\t: \t/900);
\end{tikzpicture}
```



Una buona alternativa a `smooth`, il cui algoritmo non funziona molto bene con angoli di curvatura superiori a 30° , è il più efficiente algoritmo di Hobby, attivabile con l'opzione `hobby` dell'omonima libreria. Si confronti l'ultimo esempio con il seguente:

```
\begin{tikzpicture}
\draw [help lines] (-1,-1) grid (1,1);
\draw plot [variable=\t, domain=0:900,hobby]
(\t: \t/900);
\end{tikzpicture}
```



Si ricordi che TikZ può contare su potenza di calcolo e numero di funzioni matematiche predefinite relativamente limitati, il che potrebbe costituire un problema quando il grafico richieda calcoli particolarmente onerosi. In tal caso, l'istruzione `plot` può avvalersi efficacemente del motore di calcolo del programma `gnuplot` (da installare a parte), come descritto nella documentazione del pacchetto, che spiega anche come tracciare un grafico a partire da una sequenza di dati raccolti in un file esterno.

9.4 NODI

Un *nodo* è una forma (di solito un cerchio o un rettangolo) che può contenere un testo al proprio interno: la sua funzione primaria, infatti, è quella di aggiungere scritte al disegno. Nella sua forma più semplice, la sintassi di un nodo è

```
\node [opzioni] (<nome>) at (<punto>) {\testo};
```

dove:

- `\node` è l'abbreviazione di `\path [node]`;
- `<nome>` è un'etichetta di riferimento *facoltativa* che può contenere lettere, numeri e spazi ma *non* segni d'interpunzione;
- `<punto>` è per impostazione predefinita la posizione del centro del nodo;
- il `<testo>` va racchiuso *obbligatoriamente* tra parentesi graffe *anche se vuoto* e può consistere di testo puro (si veda anche il paragrafo 9.9.2 a pagina 185) o di espressioni matematiche.

Si noti ancora che, per impostazione predefinita:

- un nodo ha forma rettangolare, ma si possono ottenere nodi circolari con l'opzione `circle`;
- un nodo è invisibile, a meno che non si specifichi di volerne disegnare il contorno o di riempirlo con le opzioni opportune;
- la dimensione del nodo si adatta automaticamente al `<testo>` che contiene.

Ecco due esempi che mostrano quanto si è appena descritto. Un nodo invisibile intorno al proprio contenuto (una parola):

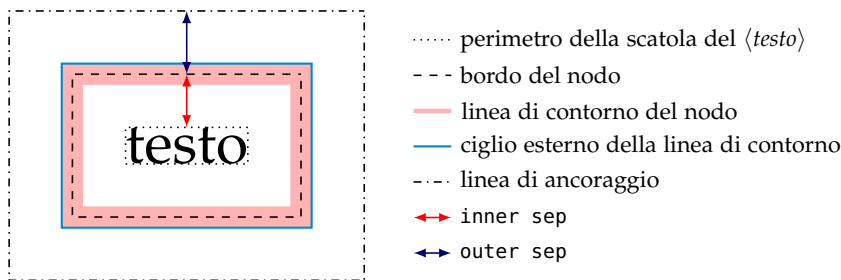
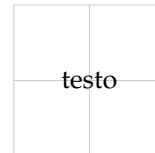


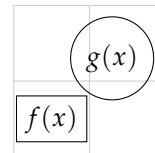
Figura 15: Elementi di un nodo

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\node at (1,1) {testo};
\end{tikzpicture}
```



e due nodi di forme diverse (con dentro espressioni matematiche):

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\node [draw] at (0.5,0.5) {$f(x)$};
\node [draw, circle] at (1.3,1.3) {$g(x)$};
\end{tikzpicture}
```



La figura 15 illustra in dettaglio gli elementi di un nodo. Si possono riconoscere:

- la scatola del $\langle testo \rangle$, cioè il rettangolo più interno che racchiude la parola *testo*;
- il *bordo* del nodo, sul quale viene tracciata la *linea di contorno*, qui ingrossata per maggiore chiarezza;
- la *linea di ancoraggio* ideale (spiegata nelle prossime sezioni), che per impostazione predefinita coincide con il ciglio esterno della linea di contorno (in celeste) e qui distanziata per maggiore chiarezza.

9.4.1 Opzioni dei nodi

Il comando `\node` accetta diverse opzioni, raccolte nella documentazione del pacchetto, a cui si rimanda il lettore. Di seguito se ne descrivono le più importanti.

L'opzione

```
font= $\langle$ dichiarazioni $\rangle$ 
```

permette di personalizzare lo stile del font in uso mediante le opportune \langle dichiarazioni \rangle standard di \LaTeX , eventualmente racchiudendole tra parentesi graffe. L'opzione non influenza le eventuali dimensioni degli elementi del nodo espresse in rapporto alle dimensioni del font usato (em e ex), al contrario dell'opzione `node font`, che ne condivide la finalità.

L'opzione

```
align= $\langle$ tipo di allineamento $\rangle$ 
```

specifica il tipo di allineamento per il $\langle testo \rangle$ e accetta i seguenti valori:

- `left`, `right`, `center` allineano il $\langle testo \rangle$ a sinistra, a destra e al centro rispettivamente, eventualmente sillabandolo;
- `flush left`, `flush right` e `flush center` agiscono come i valori precedenti ma senza sillabazione;
- `justify` giustifica il testo.

Si noti che si può mandare a capo il testo in un punto qualunque mediante il solito `\\` solo *dopo* aver dichiarato esplicitamente il tipo di allineamento desiderato. Inoltre, si può aumentare (o ridurre) l'avanzamento tra la riga terminata da `\\` e quella successiva specificando tra parentesi quadre la misura positiva (o negativa) desiderata. Ecco un esempio:

```
\begin{tikzpicture} [font=\itshape]
\draw [help lines] (0,0) grid (3,5);
\draw (0,0) -- (0,5)
      (-0.1,0) -- (0.1,0)
      (-0.1,1) -- (0.1,1)
      (-0.1,2.5) -- (0.1,2.5)
      (-0.1,4) -- (0.1,4)
      (-0.1,5) -- (0.1,5);
\node [align=right] at (1.5,4)
      {Testo allineato\\ a destra};
\node [align=center] at (1.5,2.5)
      {Testo\\ centrato};
\node [align=left] at (1.5,1)
      {Testo allineato\\ a sinistra};
\end{tikzpicture}
```

	Testo allineato a destra	
	Testo centrato	
	Testo allineato a sinistra	

Si osservi che la chiave `font` è stata assegnata all'ambiente `tikzpicture` nel suo complesso, quindi agisce su *tutti* gli elementi dell'ambiente sui quali può aver effetto, cioè su tutti i nodi.

L'opzione

```
text width= $\langle larghezza \rangle$ 
```

imposta a una dimensione fissa la larghezza della scatola del $\langle testo \rangle$ all'interno del nodo. A meno che non si intervenga sulla chiave `align`, il testo è allineato a sinistra, come nell'esempio seguente:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\node [font=\footnotesize, text width=15mm]
      at (1,1) {Questo testo è lungo};
\end{tikzpicture}
```

Questo testo è lungo	
----------------------------	--

Quando si assegna esplicitamente una larghezza alla scatola del testo, si può usare l'operatore `\\` per forzare un'interruzione di linea.

9.4.2 Ancore

Un'*ancora* è un punto sulla linea di ancoraggio o all'interno del nodo, che si può far corrispondere al $\langle punto \rangle$ assegnando l'opportuno valore alla chiave `anchor` con la sintassi:

```
anchor= $\langle direzione \rangle$ 
```

Tabella 49: Ancore notevoli di un nodo espresse come valori della chiave anchor (colonne dispari) e come opzioni (colonne pari)

Valore di anchor	Scorciatoia	Valore di anchor	Scorciatoia
east	left	west	right
north east	below left	south west	above right
north	below	south	above
north west	below right	south east	above left
center		base	

dove *<direzione>* può essere:

- un numero, e in tal caso rappresenta la coordinata angolare che individua un punto della linea di ancoraggio rispetto al centro del nodo;
- un'espressione analoga a quelle mostrate nella tabella 49 e spiegate qui di seguito.

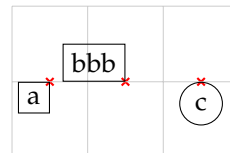
La maggior parte dei nodi possiede almeno dieci ancore notevoli alle quali sono associati altrettanti valori della chiave anchor:

- Un'ancora centrale (predefinita), a cui è associato il valore center.
- Un'ancora all'intersezione della linea di base del testo con la sua perpendicolare per il centro del nodo, a cui è associato il valore base.
- Otto ancore sulla linea di ancoraggio, corrispondenti alle direzioni della rosa dei venti rispetto al centro del nodo, a cui sono associati i rimanenti valori della tabella 49. Per ognuno di essi è disponibile anche una scorciatoia in forma di *opzione* (si veda la tabella appena menzionata) che indica intuitivamente la posizione del centro del nodo rispetto al *<punto>*. Quindi left, equivalente a anchor=east o a anchor=0, indica che il centro del nodo si trova a sinistra del *<punto>*.

Ecco un esempio in cui si sono evidenziate le ancore per chiarezza:

```
\begin{tikzpicture}
\draw [red, thick] plot [mark=x, only marks]
coordinates {(0.5,1) (1.5,1) (2.5,1)};

\draw [help lines] (0,0) grid (3,2);
\node [draw, anchor=45] at (0.5,1) {a};
\node [draw, anchor=south east]
at (1.5,1) {bbb};
\node [circle, draw, below] at (2.5,1) {c};
\end{tikzpicture}
```

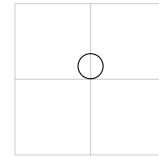


Se usate come *chiavi*, le scorciatoie appena viste accettano un valore facoltativo che quantifica lo spostamento nella direzione indicata con la sintassi

<scorciatoia>=<distanza>

Per esempio, above=1cm (o above=-1cm) ancora il nodo in modo che il lato inferiore della linea di ancoraggio si trovi 1 cm sopra (o sotto) al *<punto>*:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\node [circle, draw, above=1cm] (a)
    at (1,0) {};
\end{tikzpicture}
```



L'opzione

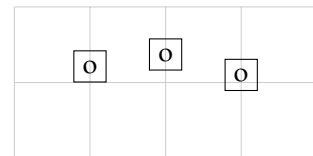
`outer sep=<distanza>`

imposta la *<distanza>* tra il bordo del nodo e la linea di ancoraggio. Si noti che:

- il valore predefinito fa coincidere la linea di ancoraggio con il ciglio esterno della linea di contorno;
- una *<distanza>* pari a zero può essere utile per collocare correttamente le ancore in un nodo riempito ma non tracciato;
- un valore negativo fa rientrare le ancore nel nodo.

Il prossimo esempio mostra all'opera quanto si è appena descritto:

```
\begin{tikzpicture} [above]
\draw [help lines] (0,0) grid (4,2);
\node [draw] at (1,1) {o};
\node [draw, outer sep=1ex] at (2,1) {o};
\node [draw, outer sep=-0.3em] at (3,1) {o};
\end{tikzpicture}
```

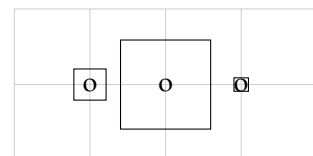


L'opzione

`inner sep=<distanza>`

imposta la *<distanza>* tra il perimetro della scatola del *<testo>* e il bordo del nodo, permettendo di aumentare le dimensioni del nodo rispetto a quelle del suo contenuto:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\node [draw] at (1,1) {o};
\node [draw, inner sep=5mm] at (2,1) {o};
\node [draw, inner sep=0pt] at (3,1) {o};
\end{tikzpicture}
```



9.4.3 Nodi coordinata, etichette e forme

Un nodo particolare è il *nodo coordinata*, che si può intendere come un nodo senza dimensione (un punto geometrico, per capirci). Serve principalmente come *alias* delle coordinate di un punto da richiamare successivamente nel disegno. La sua sintassi è:

```
\coordinate [opzioni] (<nome>) at (<punto>);
```

dove:

- `\coordinate` è l'abbreviazione di `\path [coordinate];`

- il $\langle nome \rangle$ della coordinata è un’etichetta di riferimento *obbligatoria* che può contenere lettere, numeri e spazi ma *non* segni d’interpunzione.

Si noti che, non avendo dimensione, un nodo coordinata è anche privo del $\langle testo \rangle$ e perciò tutte le opzioni per i nodi fino a qui descritte non hanno alcun effetto su di esso.

Si può assegnare un’etichetta testuale a un nodo, e quindi a un nodo coordinata in particolare, mediante l’opzione `label`, che nella sua forma completa presenta la sintassi:

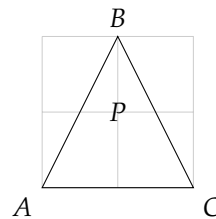
```
label={ [ $\langle opzioni \rangle$ ]  $\langle direzione \rangle$ :  $\langle testo \rangle$  }
```

dove:

- le $\langle opzioni \rangle$ agiscono solo sul $\langle testo \rangle$ dell’etichetta;
- la $\langle direzione \rangle$, *facoltativa* (compresi i due punti), può essere una delle scorciatoie elencate nella tabella 49 a pagina 169 (il valore predefinito è `above`) oppure `center` o ancora un angolo rispetto al $\langle punto \rangle$;
- le parentesi graffe si possono omettere se non ci sono $\langle opzioni \rangle$ e se il $\langle testo \rangle$ non contiene virgole e segni di uguaglianza.

L’esempio seguente mostra all’opera quanto si è appena descritto:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\coordinate [label=below left:$A$]
(A) at (0,0);
\coordinate [label=$B$] (B) at (1,2);
\coordinate [label=-45:$C$] (C) at (2,0);
\coordinate [label={ [font=\small]
center:$P$}] (P) at (1,1);
\draw (A) -- (B) -- (C) -- cycle;
\end{tikzpicture}
```



Si osservi che nel percorso il $\langle nome \rangle$ sostituisce il $\langle punto \rangle$ corrispondente. La dichiarazione di più opzioni `label` per uno stesso nodo permette di assegnargli più di un’etichetta di testo.

Le librerie interne della famiglia `shapes` mettono a disposizione molte altre forme di nodo.

Sistema di riferimento dei nodi

Quando un nodo è definito mediante un $\langle nome \rangle$, ci si può riferire alle sue ancore direttamente nelle coordinate di un percorso mediante la sintassi

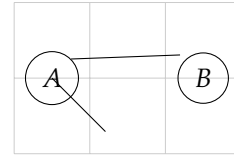
```
(  $\langle nome \rangle$  .  $\langle direzione \rangle$  )
```

dove $\langle direzione \rangle$, separata dal $\langle nome \rangle$ con il punto (`.`), può essere uno dei valori accettati dalla chiave `anchor` (se un angolo, deve essere espresso in gradi interi). Ecco un esempio:

```

\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\node [circle, draw] (A) at (0.5,1) {$A$};
\node [circle, draw, outer sep=1mm]
      (B) at (2.5,1) {$B$};
\draw (A.north east) -- (B.135)
      (A.center) -- +(-45:1);
\end{tikzpicture}

```



Si noti che la sintassi (*<nome>.<direzione>*) costituisce un ulteriore metodo per specificare le coordinate di un punto, che si aggiunge di fatto a quelli già esposti nel paragrafo 9.2.2 a pagina 154.

9.4.4 Nodi definiti direttamente in un percorso

Si può aggiungere un nodo direttamente nella posizione corrente di un percorso mediante l'istruzione

```
node [<opzioni>] (<nome>) {<contenuto>}
```

Lo stesso si può fare con un nodo coordinata tramite l'istruzione

```
coordinate [<opzioni>] (<nome>)
```

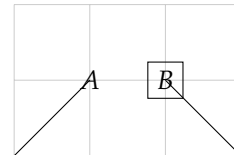
In tal caso, quanto detto finora riferito al *<punto>* si intende riferito alla posizione corrente.

Si noti che i nodi definiti in un percorso *non* ne fanno parte, ma normalmente gli sono aggiunti solo *dopo* averlo tracciato, perciò si possono sovrapporre alle altre linee. Eccone un esempio:

```

\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,2);
\draw (0,0) -- (1,1) node {$A$}
      (3,0) -- (2,1) node [draw] {$B$};
\end{tikzpicture}

```



Si osservi che anche se aggiunto a un percorso tracciato, un nodo rimane invisibile a meno di non esplicitarne il disegno (con *draw*, per esempio) tra le sue opzioni.

L'opzione

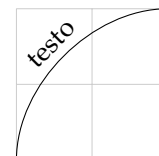
```
sloped
```

ruota automaticamente il nodo in modo tale che il *<testo>* sia parallelo alla tangente alla linea nel punto in cui il nodo è aggiunto (si veda anche il paragrafo 9.8.3 a pagina 183), come mostra l'esempio seguente:

```

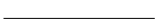






\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw (0,0) .. controls (0,1) and
      (1,2) .. (2,2)
      node [pos=0.5, sloped, above] {testo};
\end{tikzpicture}

```



Le operazioni che si possono compiere sui nodi sono davvero tante, e molte di esse esulano dagli obiettivi di questo capitolo. Perciò, al lettore che

Tabella 50: Spessori di linea predefiniti da TikZ

Opzione	Risultato	Opzione	Risultato
ultra thin		thick	
very thin		very thick	
thin		ultra thick	
semithick			

abbia acquisito una discreta padronanza degli argomenti qui trattati e con esigenze che non vi trovino immediato riscontro, si consiglia di consultare la documentazione del pacchetto.

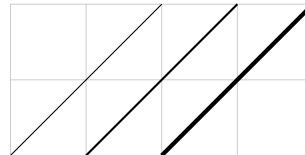
9.5 PERSONALIZZARE IL TRATTO

Le istruzioni per disegnare il percorso descritte finora non hanno contemplato la possibilità di personalizzare il disegno. Questa sezione mostra come sia possibile modificare le caratteristiche del tratto tramite le opzioni dei comandi.

9.5.1 Spessori di linea

Lo spessore di linea utilizzato da TikZ per impostazione predefinita corrisponde all'opzione `thin` mostrata nella tabella 50, ma lo si può cambiare scegliendo una delle altre opzioni lì raccolte. Il prossimo esempio ne mostra tre all'opera:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw [thin] (0,0) -- (2,2);
\draw [thick] (1,0) -- (3,2);
\draw [ultra thick] (2,0) -- (4,2);
\end{tikzpicture}
```

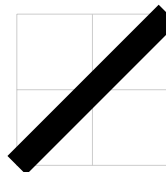


Se ancora non dovessero bastare, si può definire uno spessore a piacere con l'opzione

```
line width=<groszezza>
```














dove `<groszezza>` è una misura da esprimere con una delle abbreviazioni riconosciute da L^AT_EX. Eccone un esempio:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw [line width=10pt] (0,0) -- (2,2);
\end{tikzpicture}
```



Si osservi che le linee vengono sempre tracciate a cavallo della linea ideale di spessore nullo definita nel percorso.

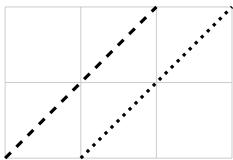
Tabella 51: Tipi di linea predefiniti da TikZ

Opzione	Risultato	Opzione	Risultato
solid		dash dot	
dotted		densely dash dot	
densely dotted		loosely dash dot	
loosely dotted		dash dot dot	
dashed		densely dash dot dot	
densely dashed		loosely dash dot dot	
loosely dashed			

9.5.2 Tipi di linea

Per impostazione predefinita, TikZ traccia il disegno con una linea continua, che si può cambiare scegliendo tra quelle raccolte nella tabella 51. Il prossimo esempio ne mostra un paio all’opera:

```
\begin{tikzpicture} [very thick]
\draw [help lines] (0,0) grid (3,2);
\draw [dashed] (0,0) -- (2,2);
\draw [dotted] (1,0) -- ++(2,2);
\end{tikzpicture}
```



Si noti che l’opzione `very thick` è assegnata all’ambiente `tikzpicture` nel suo complesso e perciò agisce su *tutti* gli elementi del disegno. TikZ permette di personalizzare anche la sequenza di tratti e punti mediante la chiave `dash pattern` (si veda la documentazione del pacchetto).

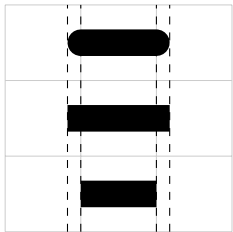
9.5.3 Estremità e raccordi delle linee

L’opzione

```
line cap=<stile>
```

permette di specificare lo *<stile>* delle estremità delle linee e accetta uno dei seguenti tre valori: `round` (“arrotondato”), `rect` (“rettangolare”) e `butt` (“mozzato”, stile predefinito). Il prossimo esempio la mostra all’opera:

```
\begin{tikzpicture} [line width=10pt]
\draw [help lines] (0,0) grid (3,3);
\draw [thin, dashed]
  (1,0) -- (1,3)
  ++(-5pt,-3) -- ++(90:3)
  (2,0) -- (2,3)
  ++( 5pt,-3) -- ++(90:3);
\draw [line cap=round] (1,2.5) -- (2,2.5);
\draw [line cap=rect] (1,1.5) -- (2,1.5);
\draw [line cap=butt] (1,0.5) -- (2,0.5);
\end{tikzpicture}
```



Si noti che:

- gli stili `round` e `rect` allungano la linea di metà *<groszza>* da entrambe le estremità;

- l'opzione `thin` data al secondo percorso scavalca localmente quella data all'ambiente `tikzpicture` per impostare lo spessore delle linee del disegno.

Invece l'opzione

`line join=<stile>`

determina lo `<stile>` del punto di raccordo tra le linee contigue del percorso e accetta uno dei seguenti tre valori: `round` ("arrotondato"), `bevel` ("smussato") e `miter` ("spigoloso", stile predefinito). Eccola in azione:

```
\begin{tikzpicture} [line width=8pt]
\draw [help lines] (0,0) grid (4,1);
\draw [line join=round]
  (0,1) -- (0.5,0) -- (1,1);
\draw [line join=bevel]
  (1.5,0) -- (2,1) -- (2.5,0);
\draw [line join=miter]
  (3,1) -- (3.5,0) -- (4,1);
\end{tikzpicture}
```

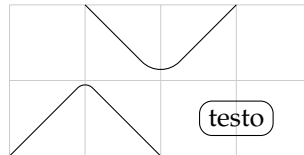


L'opzione

`rounded corners=<misura>`

dove `<misura>` è pari a 4 punti per impostazione predefinita e può essere omesso, permette di raccordare dolcemente le linee contigue di un percorso. La `<misura>` è direttamente collegata al raggio di curvatura del raccordo, come mostra l'esempio seguente:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw [rounded corners]
  (0,0) -- (1,1) -- (2,0);
\node [rounded corners=1ex, draw]
  at (3,0.5) {testo};
\draw [rounded corners=10pt]
  (1,2) -- (2,1) -- (3,2);
\end{tikzpicture}
```



9.5.4 Frecce

L'opzione generale

`<stile iniziale> - <stile finale>`

disegna due punte alle estremità dell'ultima linea di un percorso aperto. Si noti che:

- se uno dei due `<stili>` è omesso, la punta verrà disegnata solo dall'altra parte;
- gli stili predefiniti più utilizzati sono `To`, `latex`, `stealth` e `|` (linea verticale);
- lo stile `>` (`<`, se iniziale) è un *alias* per `To`, a meno che non venga ridefinito dall'utente;

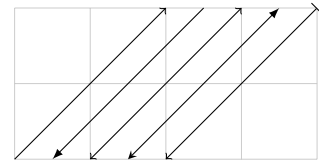
Tabella 52: Colori predefiniti da TikZ

 black	 darkgray	 lime	 pink	 violet
 blue	 gray	 magenta	 purple	 white
 brown	 green	 olive	 red	 yellow
 cyan	 lightgray	 orange	 teal	

- le frecce predefinite non sporgono oltre l'estremità della linea, per non alterarne la lunghezza;
- la dimensione della punta si accorda automaticamente con lo spessore della linea per impostazione predefinita.

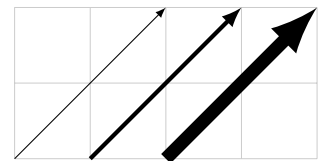
Ecco le opzioni appena descritte all'opera:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\draw [->] (0,0) -- (2,2);
\draw [latex-] (0.5,0) -- +(2,2);
\draw [->] (1,0) -- +(2,2);
\draw [stealth-latex] (1.5,0) -- +(2,2);
\draw [-|] (2,0) -- +(2,2);
\end{tikzpicture}
```



Nell'esempio seguente si è ridefinito l'alias > :

```
\begin{tikzpicture} [>=latex]
\draw [help lines] (0,0) grid (4,2);
\draw [->] (0,0) -- (2,2);
\draw [->, ultra thick] (1,0) -- (3,2);
\draw [->, line width=5pt] (2,0) -- (4,2);
\end{tikzpicture}
```

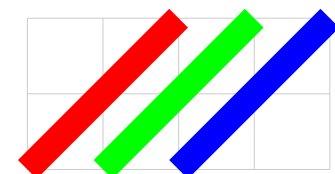


Esistono altri stili predefiniti (qui omessi per brevità) e numerosi altri ne offre la libreria `arrows.meta`, che permette di personalizzare l'aspetto delle punte delle frecce regolandone finemente dimensioni, proporzioni, forma e colore.

9.5.5 Colori

TikZ carica il pacchetto `xcolor` *senza opzioni*, perciò ne mette a disposizione solo i comandi e i colori di base mostrati nella tabella 52. Per colorare un percorso basta scrivere il nome del colore scelto tra le sue opzioni. Il prossimo esempio mostra tre spesse linee colorate con colori puri:

```
\begin{tikzpicture} [line width=10pt]
\draw [help lines] (0,0) grid (4,2);
\draw [red] (0,0) -- (2,2);
\draw [green] (1,0) -- (3,2);
\draw [blue] (2,0) -- (4,2);
\end{tikzpicture}
```



Si possono mescolare due colori per ottenerne altri in questo modo:

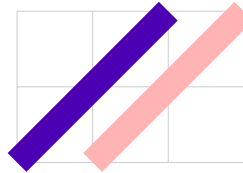
$$\langle \text{colore}_1 \rangle ! \langle \text{valore} \rangle ! \langle \text{colore}_2 \rangle$$

dove $\langle \text{valore} \rangle$ è la percentuale di $\langle \text{colore}_1 \rangle$ da mescolare con la rimanente percentuale di $\langle \text{colore}_2 \rangle$ per raggiungere il 100 %, oppure così:

$$\langle \text{colore} \rangle ! \langle \text{valore} \rangle$$

dove $\langle \text{valore} \rangle$ è la percentuale di $\langle \text{colore} \rangle$ da mescolare con la rimanente percentuale di bianco per raggiungere il 100 %. Il prossimo esempio mostra all'opera quanto si è appena descritto:

```
\begin{tikzpicture} [line width=10pt]
\draw [help lines] (0,0) grid (3,2);
\draw [red!30!blue] (0,0) -- (2,2);
\draw [red!30] (1,0) -- (3,2);
\end{tikzpicture}
```



dove:

- `red!30!blue` mescola il 30 % di rosso con il 70 % di blu;
- `red!30` mescola il 30 % di rosso con il 70 % di bianco.

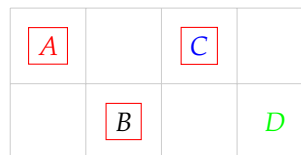
Si noti che l'eventuale colore dichiarato tra le opzioni di un percorso viene applicato su *tutte* le azioni eseguite sul percorso stesso, se più d'una (per esempio, la tracciatura del contorno, il riempimento con un colore, l'aggiunta di un testo). Si può assegnare il colore *solo a un'azione* dichiarandola nella forma $\langle \text{chiave} \rangle = \langle \text{valore} \rangle$ ed esprimendo il colore come $\langle \text{valore} \rangle$. Per esempio, per colorare di rosso solo il tratto in un percorso tracciato e riempito, l'opzione è `draw=red` (si veda anche il paragrafo 9.6.1 a pagina 179).

Analogamente accade con i nodi: l'eventuale colore dichiarato tra le opzioni colora *tutti* gli elementi del nodo (testo, bordo e riempimento). Per colorare solo il testo, per esempio, l'opzione è

$$\text{text} = \langle \text{colore} \rangle$$

Il colore dell'etichetta di un nodo coordinata, invece, va specificato tra le opzioni dell'etichetta stessa:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\node [draw, red] at (0.5,1.5) {$A$};
\node [draw=red] at (1.5,0.5) {$B$};
\node [draw, red, text=blue]
  at (2.5,1.5) {$C$};
\coordinate [label={\textcolor{green}{center:$D$}}]
  (C) at (3.5,0.5);
\end{tikzpicture}
```



Per estendere il supporto ai comandi e ai colori di base, si carichi il pacchetto `xcolor` con le opzioni opportune *prima* del pacchetto `TikZ`.

9.5.6 Definire uno stile personale

Uno *stile* è un insieme di opzioni grafiche identificato da un nome. `TikZ` permette di ridefinire gli stili esistenti e di definirne di nuovi mediante la sintassi

 $\langle nome \rangle/.style=\{\langle opzioni \rangle\}$

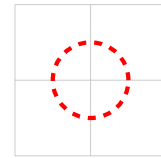
dove

- $\langle nome \rangle$ si spiega da sé;
- le parentesi graffe sono necessarie se si dichiarano più opzioni separate da virgole, oppure quando si dichiarano delle opzioni nella forma $\langle chiave \rangle=\langle valore \rangle$;
- si faccia attenzione a scrivere correttamente l'operatore $/.$ (con il punto finale) che precede `style`.

Applicare lo stile $\langle nome \rangle$ equivale ad applicare in una volta sola tutte le $\langle opzioni \rangle$ dichiarate nella sua definizione. Per esempio, l'opzione `help lines` data all'elemento `grid` è in realtà uno stile che imposta colore e spessore delle linee della griglia.

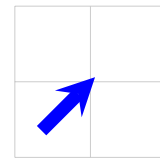
Se lo stile è definito come opzione di un ambiente, è disponibile *localmente*:

```
\begin{tikzpicture} [%
  linea/.style={ultra thick, dashed, red}]
\draw [help lines] (0,0) grid (2,2);
\draw [linea] (1,1) circle (0.5);
\end{tikzpicture}
```



In alternativa, si può definire uno stile a livello globale con il comando `\tikzset`, che si può dare nel preambolo, preferibilmente, oppure nel corpo del documento: si noti che in quest'ultimo caso lo stile sarà disponibile solo per i disegni *successivi* al punto in cui il comando compare:

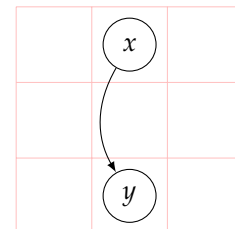
```
\tikzset{freccia/.style={%
  -stealth, blue, line width=5pt}}
\begin{tikzpicture}
\draw [help lines] (0,0) grid (2,2);
\draw [freccia] (45:0.5) -- (45:1.5);
\end{tikzpicture}
```



Si badi che uno stile definito mediante il comando `\tikzset` dato in un ambiente è disponibile soltanto al suo interno.

Se infine si volesse modificare uno stile esistente aggiungendogli opzioni, basta usare `append style` al posto di `style`. Nell'esempio seguente lo si è fatto per ottenere la griglia colorata di rosa:

```
\begin{tikzpicture}
\tikzset{every node/.style={%
  draw, circle, minimum size=7mm},
  help lines/.append style=pink}
\draw [help lines] (0,0) grid (3,3);
\node (x) at (1.5,2.5) {$x$};
\node (y) at (1.5,0.5) {$y$};
\draw [-latex, bend right] (x) to (y);
\end{tikzpicture}
```



Nell'esempio appena mostrato si è personalizzato lo stile predefinito `every node` (inizialmente vuoto), molto utile per applicare in una sola volta tutte le opzioni in esso dichiarate a *tutti* i nodi del disegno.

9.6 RIEMPIMENTI

TikZ permette di riempire i percorsi con un colore uniforme, una sfumatura o un motivo e di applicare trasparenze uniformi o graduali. Si possono usare sfumature e motivi predefiniti oppure crearne di personali con le indicazioni contenute nella documentazione del pacchetto.

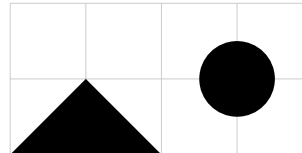
9.6.1 Campiture

Per riempire un percorso con un colore uniforme si usa il comando `\fill`, scorcio di `\path [fill]`, dal quale eredita la sintassi:

```
\fill [<opzioni>] <istruzioni>;
```

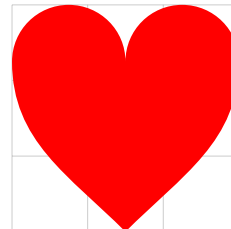
Le *<istruzioni>* permesse sono le stesse accettate dal comando `\draw`, ma questa volta le figure sono riempite senza tracciarne i bordi. Tutte le linee aperte vengono prima chiuse collegandone i due estremi con un segmento e poi riempite. Per le figure intrecciate, un meccanismo interno stabilisce quale parte debba essere riempita. Ecco il comando all'opera:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\fill (0,0) -- (1,1) -- (2,0)
      (3,1) circle (0.5);
\end{tikzpicture}
```



Il colore predefinito è il nero, ma può essere modificato come indicato nel paragrafo 9.5.5 a pagina 176. Ecco un esempio:

```
\begin{tikzpicture}
\draw [help lines] (0,0) grid (3,3);
\fill [red] (1.5,0)
  to [out=45, in=270] (3,2.25)
  to [out=90, in=0] (2.25,3)
  to [out=180, in=90] (1.5,2.25)
  to [out=90, in=0] (0.75,3)
  to [out=180, in=90] (0,2.25)
  to [out=270, in=135] (1.5,0);
\end{tikzpicture}
```



Si può dichiarare un riempimento anche dentro un percorso tracciato con l'opzione

```
fill=<colore>
```

che applica l'eventuale colore specificato solo al momento di riempire il percorso. Analogamente, si può dichiarare una tracciatura anche dentro un percorso riempito con l'opzione

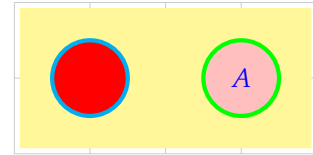
```
draw=<colore>
```

che applica l'eventuale colore dichiarato solo al tratto. In entrambi i casi, se il *<colore>* è omissso, il percorso verrà riempito (o tracciato) con lo stesso colore applicato dal comando `\draw` (o `\fill`). Ecco all'opera quanto si è appena descritto:

```

\begin{tikzpicture} [ultra thick]
\draw [help lines] (0,0) grid (4,2);
\fill [yellow!50, draw]
      (0.1,0.1) rectangle (3.9,1.9);
\fill [red, draw=cyan] (1,1) circle (0.5);
\draw [green, fill=pink] (3,1) circle (0.5)
      node [text=blue] {$A$};
\end{tikzpicture}

```



Si noti che la tracciatura viene eseguita sempre *dopo* il riempimento. Quando si sovrappongono più percorsi opachi colorati, come nell'esempio precedente, il colore delle aree comuni sarà quello del percorso disegnato per ultimo. Talvolta, però, si potrebbe volere al suo posto il colore risultante dalla fusione dei colori di partenza. La documentazione del pacchetto spiega come ottenere questo effetto mediante i *metodi di fusione*, applicabili anche alle sfumature.

9.6.2 Sfumature

Per riempire un percorso con una sfumatura di colore si usa il comando `\shade`, scorciatoia di `\path [shade]`, dal quale eredita la sintassi:

```
\shade [<opzioni>] <istruzioni>;
```

Il comando si comporta come `\fill`, ma questa volta il riempimento consiste in una sfumatura di colore definita attraverso le *<opzioni>*, le più importanti delle quali si descrivono qui di seguito.

Le opzioni

```
top color=<colore1>, bottom color=<colore2>
```

definiscono una sfumatura *verticale* tra il *<colore₁>* (in alto) e il *<colore₂>* (in basso).

Le opzioni

```
left color=<colore1>, right color=<colore2>
```

definiscono una sfumatura *orizzontale* tra il *<colore₁>* (a sinistra) e il *<colore₂>* (a destra). In entrambi i casi c'è anche l'opzione `middle color=<colore3>` per impostare il colore intermedio.













Le opzioni

```
inner color=<colore1>, outer color=<colore2>
```

definiscono una sfumatura *radiale* tra il *<colore₁>* (all'interno) e il *<colore₂>* (all'esterno).

Si noti che `\shade` disegna solo la sfumatura. Se si volesse tracciare anche il bordo, basta dare al suo posto `\draw` con almeno una delle opzioni appena viste. Ecco all'opera quanto si è appena descritto:

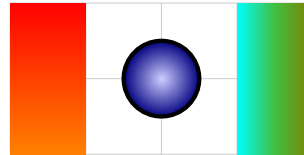
Tabella 53: Alcuni motivi di riempimento forniti dalla libreria patterns

	horizontal lines		grid		fivepointed stars
	vertical lines		crosshatch		sixpointed stars
	north east lines		dots		bricks
	north west lines		crosshatch dots		checkerboard

```

\begin{tikzpicture} [ultra thick]
\draw [help lines] (0,0) grid (4,2);
\shade [top color=red, bottom color=orange]
(0,0) rectangle (1,2);
\draw [inner color=blue!20, outer color=
blue!50!black] (2,1) circle (0.5);
\shade [left color=cyan, right color=olive]
(3,0) rectangle (4,2);
\end{tikzpicture}

```



9.6.3 Motivi di riempimento

La libreria `patterns` di TikZ mette a disposizione numerosi motivi di riempimento predefiniti. Per riempire un percorso con un motivo si usa il comando `\pattern`, scorciatoia di `\path [pattern]`, dal quale eredita la sintassi:

```
\pattern [pattern=<motivo>, pattern color=<colore>] <istruzioni>;
```

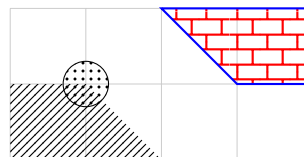
Il comando si comporta come `\fill`, ma questa volta il riempimento è il `<motivo>` indicato da scegliere tra quelli disponibili, alcuni dei quali sono raccolti nella tabella 53.

Anche in questo caso, `\pattern` disegna solo il motivo. Se si volesse tracciarne anche il bordo basta dare al suo posto `\draw` con le stesse opzioni. Eccone un esempio:

```

\begin{tikzpicture}
\draw [help lines] (0,0) grid (4,2);
\pattern [pattern=north east lines]
(0,0) -- (0,1) -- (1,1) -- (2,0);
\draw [pattern=dots] (1,1) circle (3mm);
\draw [draw=blue, thick, pattern=bricks,
pattern color=red] (4,2) -- (2,2)
-- (3,1) -- (4,1) -- cycle;
\end{tikzpicture}

```



9.7 TRASPARENZE

L'opzione

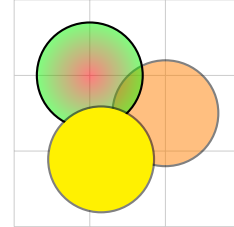
```
opacity=<valore>
```

permette di regolare il grado di opacità di un percorso mediante un `<valore>` compreso tra 0 (completamente trasparente) e 1 (totalmente opaco). Le opzioni analoghe

```
draw opacity=<valore>, fill opacity=<valore>, text opacity=<valore>
```

regolano il livello di opacità dei tracciati, dei riempimenti e del testo rispettivamente (si noti che i motivi al tratto sono considerati tracciati). Eccole all'opera:

```
\begin{tikzpicture} [thick]
\draw [help lines] (0,0) grid (3,3);
\draw [fill=orange, opacity=0.5]
      (2,1.5) circle (0.7);
\draw [inner color=red, outer color=green,
      fill opacity=0.5] (1,2) circle (0.7);
\draw [fill=yellow, draw opacity=0.5]
      (1.15,0.89) circle (0.7);
\end{tikzpicture}
```



La libreria *fadings* (se ne veda la documentazione) permette di definire una trasparenza graduale appoggiandosi alla sintassi delle sfumature.

L'argomento "sfumature e trasparenze" è piuttosto articolato e va ben oltre la semplice introduzione appena esposta, perciò si rimanda il lettore alla documentazione del pacchetto.

9.8 TRASFORMAZIONI

Le *trasformazioni* più comuni effettuabili su un percorso o su un intero ambiente sono tre: *scalatura*, *traslazione* e *rotazione*. Esse agiscono solo sulle coordinate dei punti del percorso e non sulle altre sue caratteristiche come lo spessore di linea o la distanza del tratteggio.

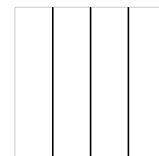
9.8.1 Scalatura

La *scalatura* è l'operazione che moltiplica le coordinate per un fattore di proporzionalità e si realizza con l'opzione

```
scale=<fattore>
```

Eccola all'opera:

```
\begin{tikzpicture} [scale=2, semithick]
\draw [help lines] (0,0) grid (1,1);
\draw (0.25,0) -- (0.25,1)
      (0.5cm,0cm) -- (0.5cm,1cm)
      (0.75,0) -- +(90:1cm);
\end{tikzpicture}
```



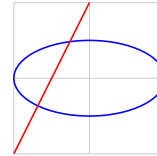
Si osservi che la scalatura agisce anche sulle coordinate espresse come misure di lunghezza.

In alternativa, si può scalare nella direzione di uno solo degli assi coordinati oppure applicare contemporaneamente due differenti fattori di scala con le varianti

```
xscale=<fattore>, yscale=<fattore>
```

il cui valore predefinito è 1. Eccone un esempio:

```
\begin{tikzpicture} [semithick]
\draw [help lines] (0,0) grid (2,2);
\draw [xscale=2, blue] (0.5,1) circle (0.5);
\draw [xscale=1, yscale=2, red]
      (0,0) -- (1,1);
\end{tikzpicture}
```



Si noti che la forma delle linee può risultare alterata quando si applica una scalatura dimetrica.

L'opzione `xscale=-1` (o `yscale=-1`) produce un ribaltamento rispetto all'asse delle ordinate (o asse delle ascisse).

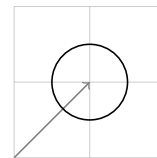
9.8.2 Traslazione

La *traslazione* è l'operazione che sposta l'origine del sistema di riferimento in un altro punto e si realizza con l'opzione

```
shift={(\langle punto \rangle)}
```

il cui effetto è di sommare le coordinate del $\langle punto \rangle$ a tutte le coordinate del percorso, come si vede nell'esempio seguente:

```
\begin{tikzpicture} [semithick]
\draw [help lines] (0,0) grid (2,2);
\draw [->, gray]      (0,0) -- (1,1);
\draw [shift={\langle 1,1 \rangle}] (0,0) circle (0.5);
\end{tikzpicture}
```



Le due varianti

```
xshift=\langle lunghezza \rangle, yshift=\langle lunghezza \rangle
```

permettono di traslare il sistema di riferimento singolarmente nelle due direzioni di un valore pari alla $\langle lunghezza \rangle$ assegnata.

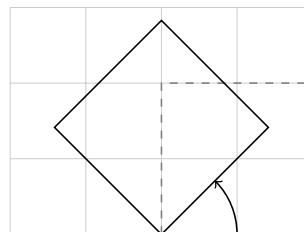
9.8.3 Rotazione

La *rotazione* è l'operazione che permette di ruotare il sistema di riferimento intorno all'origine e si realizza con l'opzione

```
rotate=\langle angolo \rangle
```

dove l' $\langle angolo \rangle$ è una coordinata angolare espressa in gradi:

```
\begin{tikzpicture} [semithick]
\draw [help lines] (-2,0) grid (2,3);
\draw [dashed, gray] (0,0) rectangle (2,2);
\draw [rotate=45]    (0,0) rectangle (2,2);
\draw [->] (1,0) arc (0:45:1);
\end{tikzpicture}
```



L'opzione

```
rotate around={\langle angolo \rangle : (\langle punto \rangle)}
```

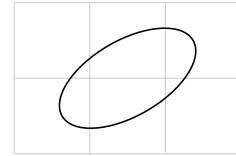
invece, permette di ruotare il sistema di riferimento di un certo $\langle angolo \rangle$ intorno al $\langle punto \rangle$ specificato:

```

\begin{tikzpicture} [semithick]

\draw [help lines] (0,0) grid (3,2);
\draw [rotate around={30:(1.5,1)}]
(1.5,1) ellipse (1 and 0.5);
\end{tikzpicture}

```



Si noti che le trasformazioni appena descritte vengono applicate solo agli *elementi* di un nodo, se dichiarate tra le sue opzioni. In tutti gli altri casi agiscono solo sulle *coordinate* della posizione corrente del nodo, a meno di non specificare anche l'opzione `transform shape`, che estende le trasformazioni anche agli elementi del nodo.

L'opzione `sloped` può produrre un risultato inatteso se allo stesso nodo si applica *anche* una rotazione, pertanto si sconsiglia di usare contemporaneamente le due funzioni.

9.9 APPROFONDIMENTI

Le nozioni introdotte in queste pagine non pretendono certo di esaurire le funzioni di TikZ, che oltre alle possibilità offerte dalle varie librerie di cui è corredato mette a disposizione numerosissime altre istruzioni, qui omesse per dovere di brevità, per soddisfare praticamente *qualunque* esigenza grafica. Questa sezione getta un rapido sguardo su poche caratteristiche avanzate, rimandando ancora una volta il lettore alla documentazione del pacchetto per eventuali approfondimenti.

9.9.1 Ripetere le azioni

Non è raro aver bisogno di ripetere più volte la stessa azione variandone solo uno o pochi parametri, effettuando un *ciclo*, nel gergo di TikZ. A questo scopo si usa il comando

```
\foreach <parametro> [ <opzioni> ] in { <elenco> } { <comandi> }
```

dove:

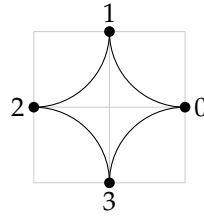
- `\foreach` è il comando che ripete le azioni descritte dai `<comandi>`;
- `<parametro>`, in pratica un comando in stile L^AT_EX (`\i`, per esempio), è il parametro che varia tra i valori dichiarati nell'`<elenco>`;
- le `<opzioni>` si spiegano da sé;
- `<elenco>` è un insieme di valori separati dalla virgola e racchiusi tra parentesi graffe o, in alternativa, il nome di una macro che contiene tali valori;
- i `<comandi>`, da racchiudere tra parentesi graffe se più d'uno, vanno espressi in funzione del `<parametro>` dichiarato in modo che a ogni passo tale `<parametro>` sia sostituito da un valore preso ordinatamente nell'`<elenco>`.

In altre parole, `\foreach` ripete le azioni descritte dai `<comandi>` variando il `<parametro>` tra i valori dichiarati nell'`<elenco>`. Eccolo all'opera:

```

\begin{tikzpicture}
\draw [help lines] (-1,-1) grid (1,1);
\foreach \i in {0,1,2,3} {%
  \draw (90*\i:1) coordinate
    [label=90*\i:$\i$]
    arc (270+90*\i:180+90*\i:1);
  \fill (90*\i:1) circle (2pt);}
\end{tikzpicture}

```



Si osservi che il $\langle parametro \rangle$ può comparire anche nelle opzioni delle istruzioni e nel testo dei nodi.

Si noti che per comprimere il contenuto dell' $\langle elenco \rangle$ si possono usare i puntini di sospensione: per esempio, $\{0, \dots, 3\}$ equivale a $\{0, 1, 2, 3\}$ e $\{9, 7, \dots, 2\}$ a $\{9, 7, 5, 3\}$. La documentazione del pacchetto descrive tutte le altre alternative disponibili.

9.9.2 Annotare le immagini

In genere un nodo contiene del testo, ma può contenere anche dell'altro. Con il comando `\includegraphics`, per esempio, al disegno si possono aggiungere immagini trattandole come semplici nodi rettangolari, oppure annotarle con scritte e linee, come si mostra nell'esempio seguente per il file `orologio.jpg`:

```

\tikzset{immagine/.style={%
  above right, inner sep=0pt, outer sep=0pt},
testo/.style={fill=white, align=center,
  fill opacity=0.6, text opacity=1, below,
  font=\sffamily\bfseries\footnotesize}}
\begin{tikzpicture} [>=latex, red,
  ultra thick]
\node [immagine] at (0,0)
  {\includegraphics[width=4cm]{orologio}};
\draw [->] (2,1) node [testo]
  {Posizione del sole\\
  rispetto all'eclittica} -- (1.57,2);
\end{tikzpicture}

```



L'immagine è stata inserita in modo tale che il suo vertice inferiore sinistro coincida con l'origine (stile `immagine`). Si noti che, avendo specificato una trasparenza per il riempimento dell'etichetta, è stato necessario dichiarare anche il livello di opacità del testo, altrimenti uguale a quello dell'etichetta.

Per ricavare le coordinate dei particolari dell'immagine, durante la realizzazione del disegno può essere molto utile sovrapporre una griglia con passo fine (qui omessa per maggiore chiarezza).

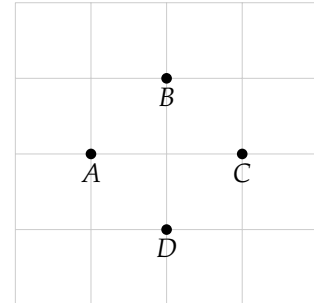
9.9.3 Calcolare le coordinate

La libreria `calc` permette di dichiarare le coordinate di un punto mediante espressioni matematiche nelle quali compaiono a propria volta delle coordinate. La sintassi generale delle coordinate calcolate è

$(\langle espressione \rangle)$

dove i dollari delimitano l'«espressione» da valutare, il cui contenuto è una successione di addendi, ciascuno dei quali è costituito da una coordinata moltiplicata per un fattore numerico che a propria volta è il risultato di un'espressione matematica del tipo illustrato nel paragrafo 9.2.3 a pagina 155, eventualmente delimitato da parentesi graffe per maggiore chiarezza:

```
\begin{tikzpicture}
\draw [help lines] (0,-1) grid (4,3);
\coordinate [label=below:$A$] (A) at (1,1);
% Punti determinati a partire da A
\coordinate [label=below:$B$]
(B) at ({cos(60)+1.5}*(A));
\coordinate [label=below:$C$]
(C) at ($(A)+2*(0:1)$);
\coordinate [label=below:$D$]
(D) at ($(C)-(A)$);
\fill (A) circle (2pt) (B) circle (2pt)
(C) circle (2pt) (D) circle (2pt);
\end{tikzpicture}
```



La sintassi

$(\$ \langle punto_1 \rangle ! \langle valore \rangle ! \langle angolo \rangle : \langle punto_2 \rangle \$)$

nella quale $\langle angolo \rangle$: (compresi i due punti) è facoltativo, svolge due funzioni differenti a seconda del formato del $\langle valore \rangle$. In particolare:

- se $\langle valore \rangle$ è un numero, individua un punto che dista dal $\langle punto_1 \rangle$ una lunghezza pari a $\langle valore \rangle$ volte la distanza che intercorre tra il $\langle punto_2 \rangle$ e il $\langle punto_1 \rangle$;
- se $\langle valore \rangle$ è una lunghezza, individua un punto che dista $\langle valore \rangle$ dal $\langle punto_1 \rangle$.

L'«angolo» è la coordinata angolare del punto individuato rispetto all'asse polare diretto dal $\langle punto_1 \rangle$ al $\langle punto_2 \rangle$. Un angolo di 0° (valore predefinito) individua il punto sulla retta che congiunge il $\langle punto_1 \rangle$ al $\langle punto_2 \rangle$.

9.10 UNIVERSO GRAFICO

Con i comandi di TikZ si può disegnare virtualmente qualsiasi cosa; tuttavia, per esigenze particolari, sono disponibili librerie e pacchetti dedicati a settori specifici (biologia, chimica, ingegneria, matematica e altre discipline) che permettono di realizzare disegni anche complessi con una sintassi semplificata. Questo paragrafo ne elenca alcuni, suddivisi per discipline, e offre una galleria d'esempi (senza il codice sorgente, figure 16 a pagina 188 e 17 a pagina 189) realizzati con alcuni tra i più riusciti pacchetti basati su TikZ.

9.10.1 Librerie

automata Per disegnare macchine a stati finiti e di Turing (figura 16a).

lindenmayersystems Per disegnare frattali bidimensionali e profili di arbusti (figura 16b).

`mindmap` Per realizzare mappe mentali o concettuali (figura 16c).

`spy` Per ingrandire un particolare del disegno in una zona della figura (figura 16b).

9.10.2 Pacchetti

Chimica e biologia

`chemfig` Per disegnare la struttura di qualunque tipo di molecola chimica (figura 16d).

`chemmacros` *Suite* di tre pacchetti: `chemmacros` è un insieme di macro predefinite per scrivere la chimica; `chemformula` (alternativo a `mhchem`) permette di scrivere qualunque tipo di formula chimica; `ghsystem` permette di inserire *indicazioni di pericolo* e *consigli di prudenza*

`modiagram` Crea facilmente diagrammi dei livelli energetici degli orbitali molecolari (figura 16f).

Matematica e fisica

`hf-tikz` Per evidenziare formule matematiche o loro parti. Funziona anche nelle presentazioni `beamer` (figura 17e).

`pgfplots` Per disegnare grafici di funzione nel piano e nello spazio e altri tipi di diagrammi per la rappresentazione grafica dei dati (figura 16g).

`tikz-cd` Per disegnare diagrammi commutativi (figura 16h).

`tkz-euclide` Per il disegno geometrico nel piano cartesiano (figura 17a).

`tkz-graph` Per creare grafi (figura 17b).

Informatica ed elettronica

`bodegraph` Per disegnare diagrammi di Bode, Nichols e Nyquist (figura 17c).
Richiede `gnuplot`.

`circuitikz` Per disegnare reti elettriche, logiche ed elettroniche (figura 17d).

`schemabloc` Per disegnare schemi a blocchi (figura 17e).

Grafica e geometria della pagina

`pgf-blur` Per realizzare ombre sfocate da aggiungere agli elementi del percorso, nodi compresi (figure 16a, 16c e 17c).

`tcolorbox` Per realizzare box di testo colorati e incorniciati, con un titolo e personalizzabili in ogni aspetto (figura 17f).

Varie

`bchart` Per disegnare semplici grafici a barre orizzontali (figura 17g).

`forest` Per disegnare alberi linguistici e d'altro tipo (figura 17h).

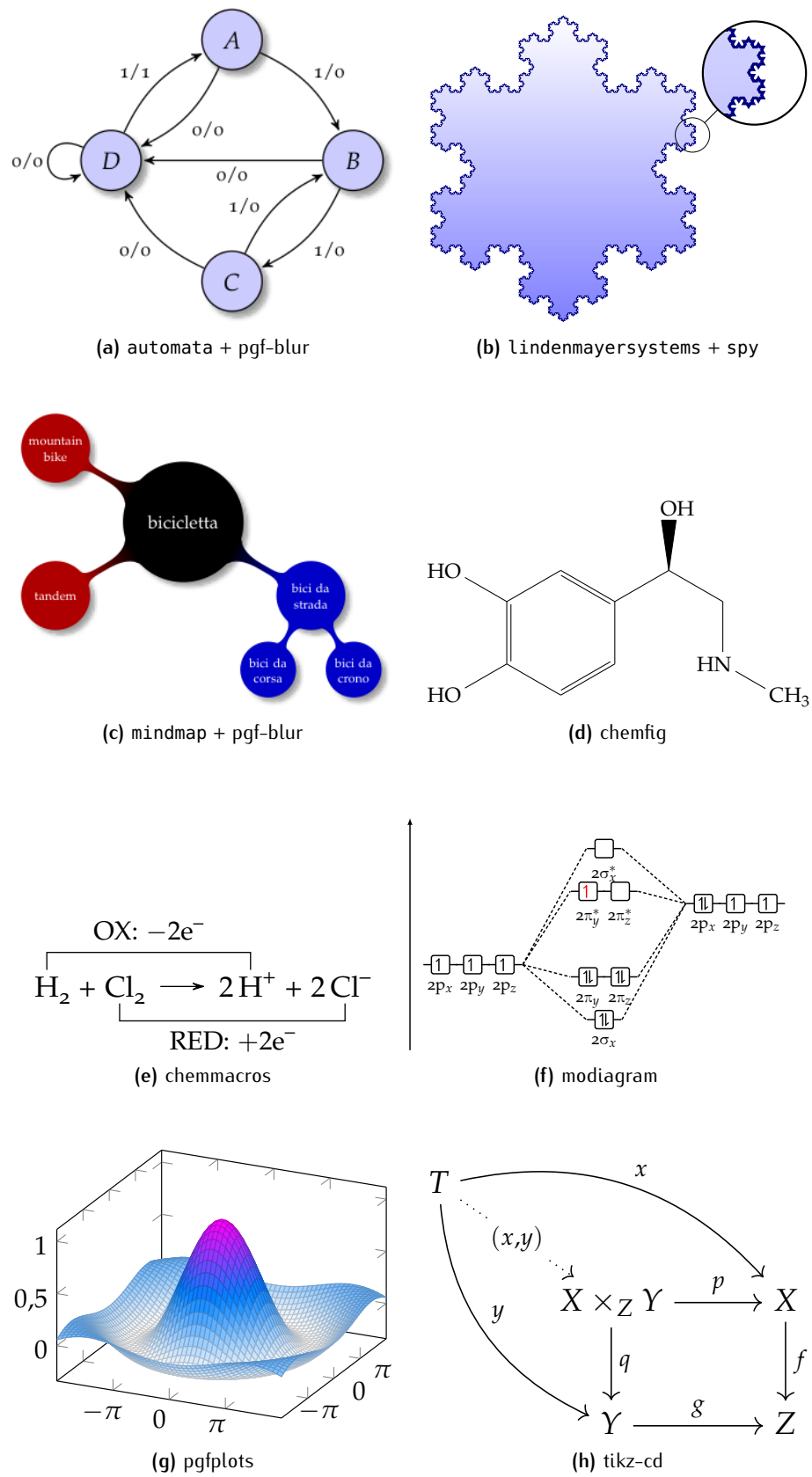
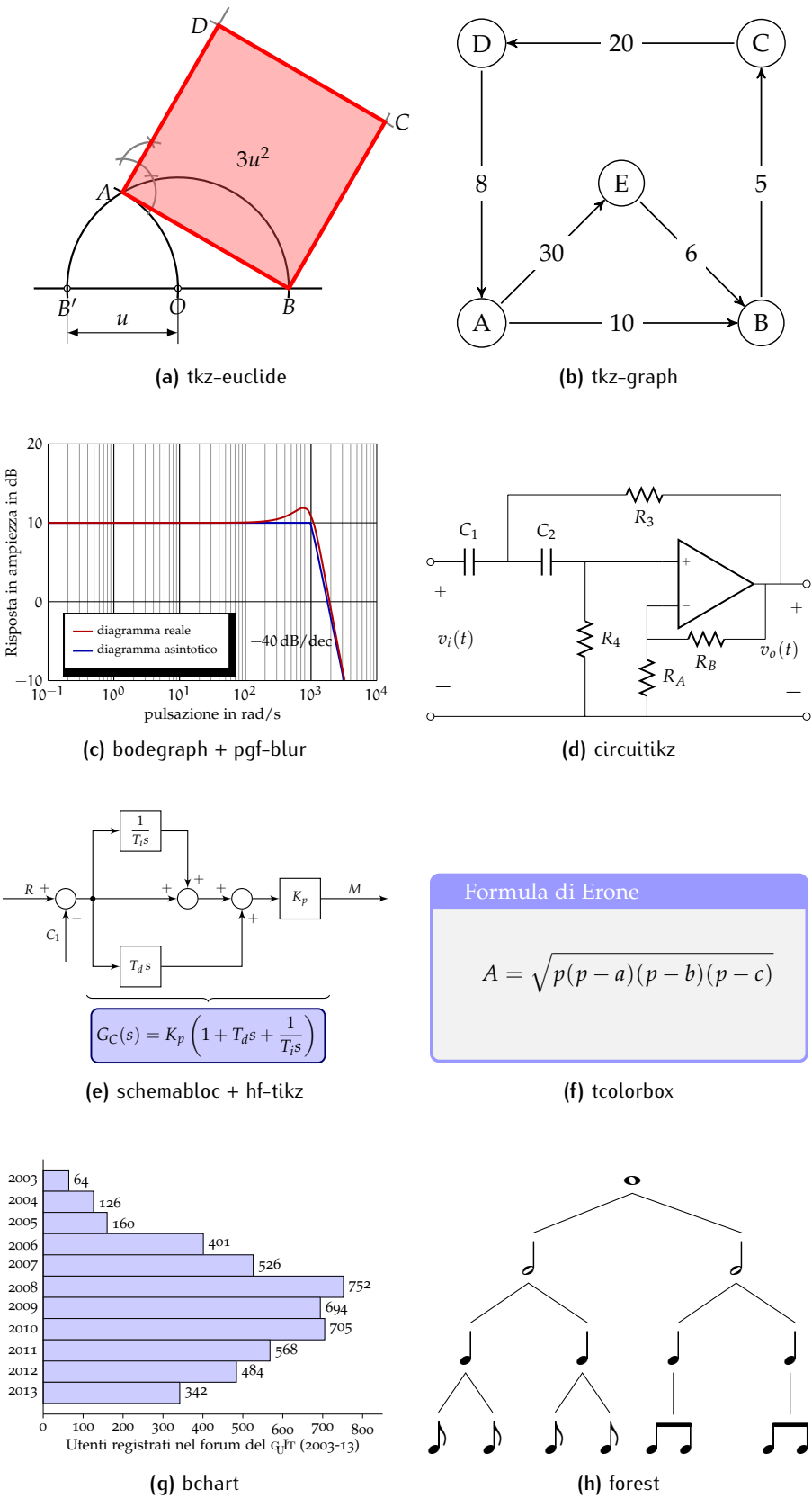


Figura 16: Galleria d'esempi grafici/1



Anno	Utenti registrati
2003	64
2004	126
2005	160
2006	401
2007	526
2008	752
2009	694
2010	705
2011	568
2012	484
2013	342

Utenti registrati nel forum del QIT (2003-13)

(g) bchart



(h) forest

Figura 17: Galleria d'esempi grafici/2

Questo capitolo, basato su [De Marco e Giacomelli, 2011], cui si rimanda per ogni approfondimento, presenta `pgfplots`, un pacchetto per la rappresentazione grafica di dati derivato da `TikZ`. Si mostrerà come impostare un sistema di riferimento, come scrivere le istruzioni per creare grafici di funzione e d'altro tipo, come realizzare diagrammi a barre e come personalizzare l'aspetto dei vari elementi del disegno.

10.1 GRAFICI E TIPOGRAFIA

La rappresentazione di dati numerici tramite grafici di vario tipo è una parte consistente e indispensabile della comunicazione tecnico-scientifica, perché molto spesso permette di esporre concetti matematici e fenomeni fisici in modo molto più semplice e intuitivo di quanto farebbero formule da sole o elenchi di numeri o tabelle.

Come fare per inserire un grafico in un documento \LaTeX ? Una via potrebbe essere quella di includerlo come file esterno prodotto con un programma specializzato: Mathematica, MATLAB e Octave, per citarne alcuni, generano grafici molto sofisticati e li possono esportare nel formato PDF accettato da \LaTeX .

Gli inconvenienti, però, non tardano a presentarsi, perché un disegno importato, in generale:

- usa font diversi da quelli del proprio documento;
- contiene simboli matematici ed elementi grafici che male si adattano allo stile scelto;
- peggio: le formule matematiche, se presenti, appaiono completamente diverse;
- presenta linee o troppo grosse o troppo sottili.

10.2 FERRI DEL MESTIERE

Il pacchetto `pgfplots`, compreso in tutte le distribuzioni complete di \LaTeX , permette di comporre grafici coerenti con le impostazioni tipografiche del documento in lavorazione scrivendone le istruzioni *direttamente* nel testo sorgente e assicurando la più alta qualità tipica di \LaTeX .

Con `pgfplots` si possono tracciare curve e superfici di qualunque tipo, in due e tre dimensioni, creare diagrammi a barre e altri grafici particolari, aggiungervi etichette, legende, titoli e personalizzare ogni elemento del disegno. Inoltre, `pgfplots` può eseguire i calcoli necessari sfruttando le stesse capacità di \LaTeX senza appoggiarsi a strumenti esterni.

Tabella 54: Sistemi di riferimento disponibili in pgfplots e librerie richieste

Sistema di riferimento	Ambiente	Libreria richiesta
Cartesiano	axis	
Cartesiano logaritmico	loglogaxis	
Ascissa logaritmica	semilogxaxis	
Ordinata logaritmica	semilogyaxis	
Coordinate polari	polaraxis	polar
Diagramma ternario	ternaryaxis	ternary
Carta di Smith	smithchart	smithchart

Il pacchetto carica automaticamente TikZ (permettendo eventualmente di usarne i comandi) e xcolor, *dopo* il quale (se già presente nel preambolo) va sempre caricato. Eventuali opzioni valide per tutti i grafici del documento si possono mettere nell'argomento di `\pgfplotsset` nel preambolo:

```
\usepackage{pgfplots}
\pgfplotsset{/pgf/number_format/use_comma,compat=newest,<altre opzioni>}
```

Si raccomanda di scrivere le opzioni *rispettando sempre gli eventuali spazi* (qui evidenziati con `_`), e in particolare le due nell'esempio precedente:

- la prima imposta la virgola come separatore decimale (le istruzioni nel sorgente, invece, richiedono il punto);
- la seconda assicura che si usino le caratteristiche della versione più recente del pacchetto.

Si noti, infine, che i grafici prodotti da pgfplots sono oggetti *in testo*, con tutti i possibili inconvenienti del caso. Nulla vieta però di renderli *mobili* semplicemente scrivendone il codice in un ambiente figure come si mostra di seguito:

```
\begin{figure}
\centering
\begin{tikzpicture}
...
\end{tikzpicture}
\caption{...}
\label{fig:...}
\end{figure}
```

10.2.1 Impostare il sistema di riferimento

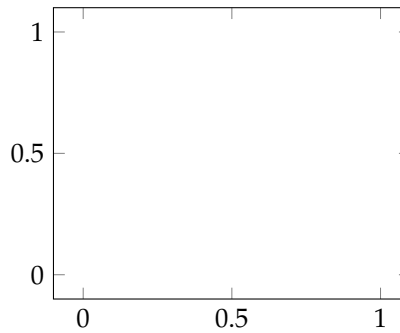
L'ambiente fondamentale di pgfplots è axis, da inserire a propria volta nell'ambiente tikzpicture (definito da TikZ) con la sintassi:

```
\begin{tikzpicture}
\begin{axis}[<opzioni>]
<istruzioni di pgfplots o TikZ>
\end{axis}
\end{tikzpicture}
```

Si noti che le *<opzioni>*, se presenti, agiranno su *tutti* i grafici inseriti in quell'ambiente axis (e solo su quelli).

Eccone un esempio davvero minimo:

```
\begin{tikzpicture}
\begin{axis}
\end{axis}
\end{tikzpicture}
```



Si noti che:

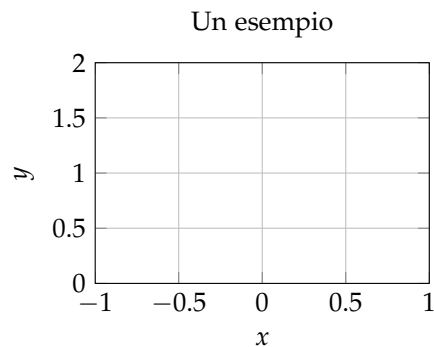
- `axis` definisce un sistema di riferimento cartesiano ortogonale visualizzandone parte del primo quadrante (e una piccola porzione degli altri tre) come un riquadro;
- i lati del riquadro portano tacche di marcatura distanti tra loro un certo passo, calcolato automaticamente in base a parametri interni;
- per esigenze tipografiche i disegni di questo capitolo hanno larghezza fissa: riproducendoli, si potrebbero avere risultati diversi (ma non scorretti).

Gli altri ambienti elencati nella tabella 54 a fronte producono i sistemi di riferimento mostrati nella figura 18 nella pagina successiva e nel paragrafo 10.6 a pagina 210, nel quale si spiega anche come caricare le librerie richieste.

Si può personalizzare il risultato predefinito passando ad `axis` opportune opzioni nella notazione $\langle chiave \rangle = \langle valore \rangle$ o anche solo $\langle chiave \rangle$ (i valori `=true` si possono omettere).

Il prossimo è un esempio con qualche opzione:

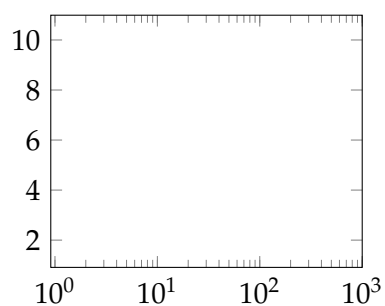
```
\begin{tikzpicture}
\begin{axis} [xmin=-1,xmax=1,
ymin=0,ymax=2,grid=major,
xlabel=$x$,ylabel=$y$,
title={Un esempio},
width=6cm,height=4.5cm]
\end{axis}
\end{tikzpicture}
```



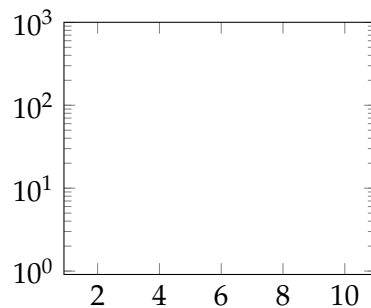
Si noti quanto segue.

- `xmin` e `xmax` fissano rispettivamente il valore minimo e massimo delle ascisse; analoghe chiavi si useranno per le ordinate.
- `grid=major` visualizza una griglia agganciata alle tacche di marcatura degli assi per leggere più facilmente il grafico.
- `xlabel` e `ylabel` producono le etichette degli assi (non obbligatorie), qui x e y . Quest'ultima per impostazione predefinita è ruotata di 90° in senso antiorario e centrata verticalmente: per averla diritta, come mostra l'esempio seguente, basta scrivere tra le opzioni di `axis`:

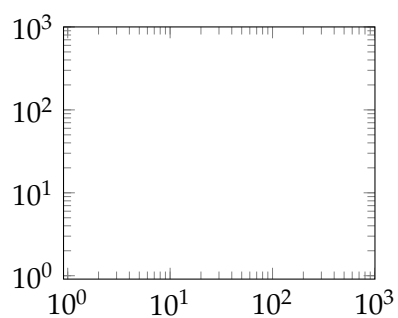
```
 $\langle nome\ dell'etichetta \rangle label\ style={rotate=-90}$ 
```



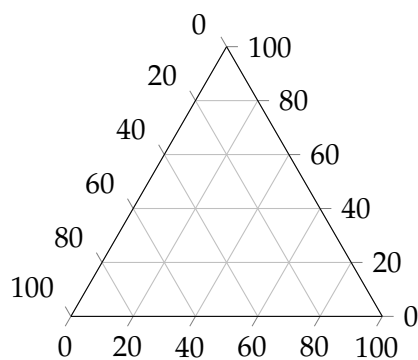
(a) Piano con ascissa logaritmica



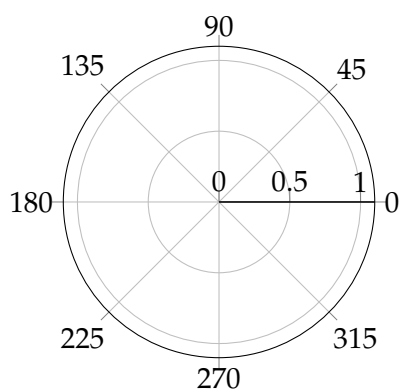
(b) Piano con ordinata logaritmica



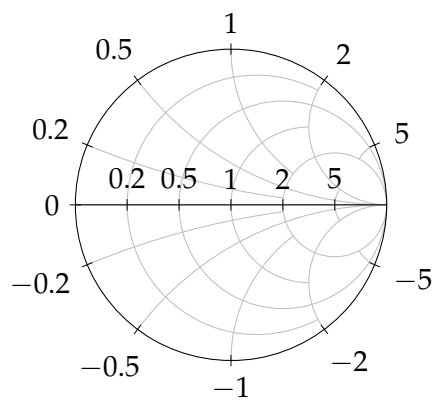
(c) Piano logaritmico



(d) Diagramma ternario



(e) Sistema di coordinate polari



(f) Carta di Smith

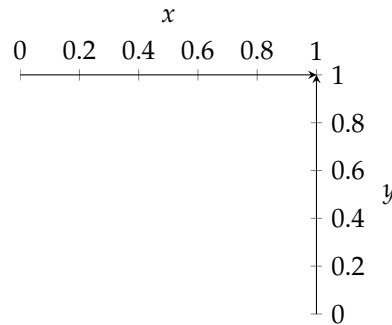
Figura 18: Sistemi di riferimento predefiniti di pgfplots (tranne quello prodotto da axis)

Se il valore di un'etichetta contiene caratteri particolari, va racchiuso tra parentesi graffe.

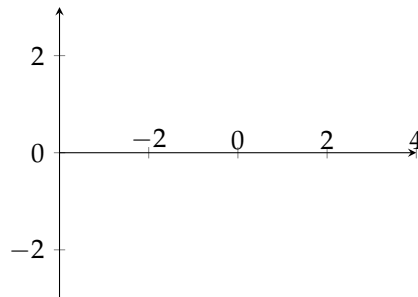
- `title` produce un titolo sopra il disegno (ma se il grafico è mobile, lo si metta nell'argomento di `\caption`).
- `width` e `height`, esprimibili come al solito, impostano rispettivamente larghezza e altezza dell'intero disegno, etichette e titoli *compresi*. Per riferire le dimensioni al solo riquadro, invece, si aggiunga l'opzione `scale only axis`.

Di seguito si mostrano alcuni esempi in cui si è modificata la posizione degli assi. Si noti che dichiararli esplicitamente attiva lo stile tradizionale ed elimina il riquadro.

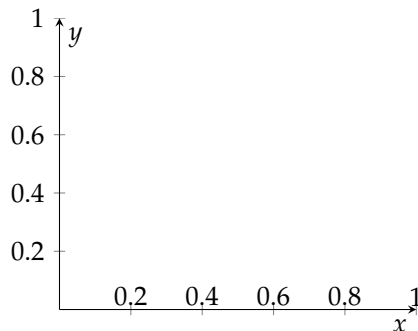
```
\begin{tikzpicture}
\begin{axis}
[axis x line=top,
axis y line=right,
xlabel=$x$,ylabel=$y$,
ylabel style={rotate=-90}]
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}
[xmin=-4,xmax=4,ymin=-3,ymax=3,
axis x line=middle,
axis y line=left]
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}
[axis lines=middle,
xlabel=$x$,ylabel=$y$]
\end{axis}
\end{tikzpicture}
```



Si noti quanto segue.

- `axis x line` regola la posizione delle ascisse: accetta i valori `bottom`, `middle` e `top`, che ne impongono il passaggio rispettivamente per $y = y_{\min}$, $y = 0$ e $y = y_{\max}$. (Quando però il grafico è tutto sotto o sopra l'asse x , il valore `middle` corrisponde a `top` o `bottom` rispettivamente.)
- `axis y line` regola la posizione delle ordinate: accetta i valori `left`, `middle` e `right`, che ne impongono il passaggio per $x = x_{\min}$, $x = 0$

e $x = x_{\max}$, rispettivamente. (Quando però il grafico è tutto a sinistra o a destra dell'asse y , il valore `middle` corrisponde a `right` o `left` rispettivamente.)

- `axis lines` imposta contemporaneamente entrambi gli assi con il valore scelto, di solito `middle` (se non li si desidera visualizzare, il valore da dare è `none`) e raddrizza automaticamente l'etichetta delle ordinate.

La documentazione del pacchetto spiega come personalizzare la posizione delle etichette.

10.2.2 Disegnare il grafico

Il comando `\addplot`, dato in uno degli ambienti di `pgfplots` e ripetibile per ogni grafico da aggiungere nel sistema di riferimento, disegna un grafico nel piano. La sintassi generale è la seguente:

```
\addplot [opzioni]  
  superistruzione  
  {istruzioni};
```

Si noti che:

- *opzioni*, da esprimere come al solito, sono le opzioni che definiscono l'aspetto del grafico;
- la *superistruzione* (`coordinates`, `file` o `table`) va indicata nei casi spiegati tra poco;
- *istruzioni* sono le istruzioni per ottenere il grafico vero e proprio;
- il punto e virgola immediatamente dopo l'argomento di `\addplot` è *obbligatorio*;
- si possono separare i vari elementi del codice uno spazio bianco per rendere il sorgente più leggibile.

Le *istruzioni* possono essere dei tre tipi descritti di seguito.

- Un'espressione matematica, che verrà valutata per un opportuno numero di punti del dominio di definizione, con il codice

```
\begin{axis}[...]  
  \addplot [...]  
    {espressione matematica};  
  \end{axis}
```

- Una sequenza di coppie di valori (corrispondenti a coordinate di punti del piano), con il codice

```
\begin{axis}[...]  
  \addplot [...] coordinates  
    {(x1, y1) (x2, y2)  
     ... (xn, yn)};  
  \end{axis}
```

dove `coordinates` ordina a `pgfplots` di disegnare il grafico usando le coordinate scritte nell'argomento.

- Una sequenza di coppie di valori separati da almeno uno spazio e disposte su più righe, contenuta in un file esterno prodotto con uno dei programmi nominati nel paragrafo 10.1 a pagina 191 e sistemato nella cartella di lavoro. Il codice generale è il seguente:

```
\begin{axis}[\langle...\rangle]
\addplot [\langle...\rangle]
file {\langlenome del file di dati con l'estensione\rangle};
\end{axis}
```

dove `file` ordina a pgfplots di usare il file indicato (registrato con una delle estensioni accettate dal pacchetto). In alternativa si può usare `table`, una variante di `file` personalizzabile (si veda la documentazione del pacchetto).

Con la stessa sintassi, il comando `\addplot3` permette di disegnare un grafico nello spazio. Nei casi in cui l'istruzione non sia un'espressione matematica, le coordinate vanno espresse come terne anziché coppie di valori.

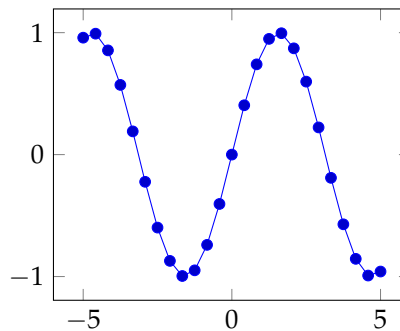
10.3 FUNZIONI ESPRESSE ANALITICAMENTE

La tabella 48 a pagina 156 mostra le più importanti funzioni e costanti matematiche predefinite da pgfplots. Di seguito se ne presentano alcune realizzazioni.

10.3.1 Funzioni reali d'una variabile reale

Un semplice esempio senza opzioni per cominciare.

```
\begin{tikzpicture}
\begin{axis}
\addplot {sin(deg(x))};
\end{axis}
\end{tikzpicture}
```



Si noti che:

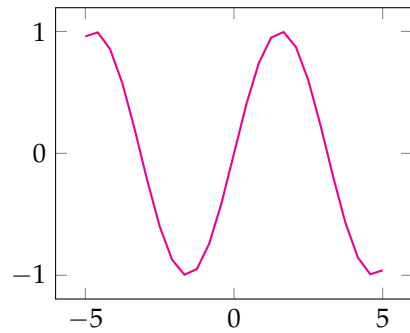
- il risultato predefinito è una curva blu contrassegnata da marcatori circolari (per avere solo i marcatori, ma in nero, si passi a `\addplot` la chiave `only marks`);
- `deg` trasforma in gradi il proprio argomento, da mettere fra parentesi *tonde* (pgfplots assume che l'argomento delle funzioni trigonometriche sia espresso in gradi e non in radianti).

Ora un esempio con qualche personalizzazione:

```

\begin{tikzpicture}
\begin{axis}
\addplot [thick,color=magenta]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}

```



Si noti che:

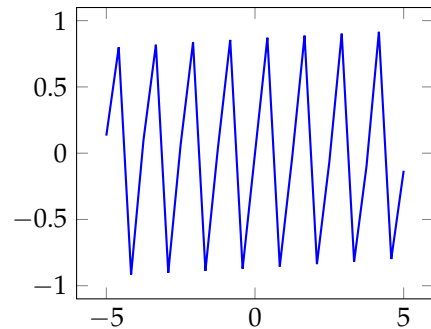
- non ci sono marcatori, perché le opzioni di `\addplot` sostituiscono le impostazioni predefinite (per mantenerle aggiungendovi nuove opzioni o per ridefinire localmente un'opzione, il comando è `\addplot+`; in questo caso, per eliminare del tutto i marcatori la chiave è `no marks`);
- `thick` imposta lo spessore della curva, da scegliere tra quelli mostrati nella tabella 50 a pagina 173;
- `color=magenta` (ma basta scrivere `magenta`) imposta il colore desiderato per la curva, da scegliere tra quelli elencati nella tabella 52 a pagina 176 (se non bastassero, si possono usare le tavolozze predefinite di `xcolor` o definirne una personale regolandone a mano le componenti).

La curva dell'esempio precedente è leggermente spigolosa, perché `pgfplots` disegna le curve approssimandole alla spezzata che unisce un opportuno campione di punti. Se questi sono sufficientemente vicini, si percepisce la spezzata come una curva "regolare", tracciata con precisione e qualità; ma se sono insufficienti, il risultato è inaccettabile, come nell'esempio seguente:

```

\begin{tikzpicture}
\begin{axis}
\addplot [thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}

```

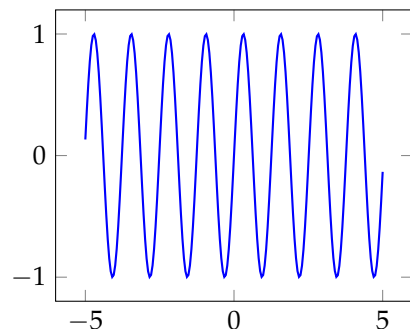


Aumentando il valore della chiave `samples`, che gestisce il numero di punti campionati (il suo valore predefinito è 25), si risolve il problema:

```

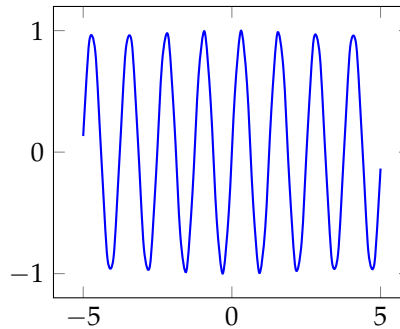
\begin{tikzpicture}
\begin{axis}
\addplot
[samples=200,thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}

```



Si può anche usare la chiave `smooth`, da sola o insieme alla precedente: essa ordina al programma di tracciare il grafico con curve di Bézier (cubiche raccordate con continuità di tangente e concavità) anziché con segmenti.

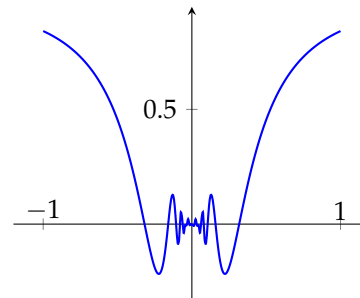
```
\begin{tikzpicture}
\begin{axis}
\addplot
[samples=50,smooth,thick,blue]
{sin(5*deg(x))};
\end{axis}
\end{tikzpicture}
```



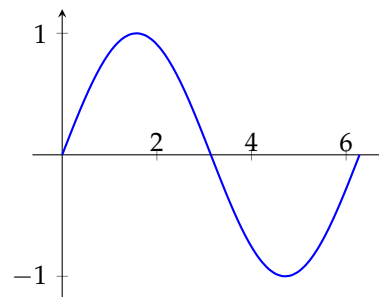
Permettendo (spesso, ma non sempre) di diminuire il valore di `samples`, `smooth` può ridurre o evitare del tutto le probabilità di eccedere la memoria di calcolo del programma (che reagisce arrestando la composizione), ciò che troppi grafici ad altissima risoluzione in uno stesso documento potrebbero causare: la documentazione del pacchetto spiega come risolvere questi problemi.

Altri due esempi:

```
\begin{tikzpicture}
\begin{axis}
[axis lines=middle,
enlargelimits]
\addplot
[domain=-1:1,samples=200,smooth,
thick,blue]
{x*sin(deg(1/x))};
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}[axis lines=middle,
enlargelimits]
\addplot
[domain=0:2*pi,samples=40,smooth,
thick,blue]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}
```

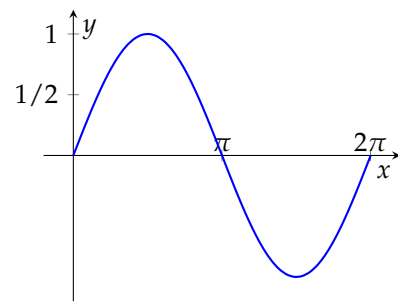


Si noti quanto segue.

- La chiave `enlargelimits`, da mettere *dopo* `axis lines`, aumenta di poco la lunghezza di *entrambi* gli assi oltre i limiti stabiliti da `xmax` e `ymax` (`enlarge x limits` e `enlarge y limits` permettono di farlo separatamente), evitando che la curva oltrepassi i limiti del sistema di riferimento o finisca proprio sulla freccia di uno degli assi, oppure ancora solo per dare respiro al grafico.
- La chiave `domain`, per impostazione predefinita pari a $[-5, 5]$, imposta il dominio, nel primo esempio pari a $[-1, 1]$ e nel secondo a $[0, 2\pi]$.

Ora si ripropone l'esempio precedente con qualche variante:

```
\begin{tikzpicture}
\begin{axis} [axis lines=middle,
enlargelimits,
xtick={3.14,6.28},ytick={0.5,1},
xticklabels={\pi, 2\pi},
yticklabels={1/2, 1},
xlabel=$x$,ylabel=$y$]
\addplot [domain=0:2*pi,
samples=40,smooth,thick,blue]
{sin(deg(x))};
\end{axis}
\end{tikzpicture}
```

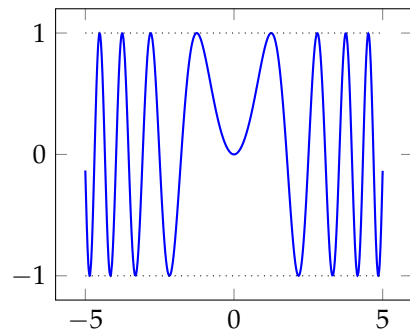


Si noti che:

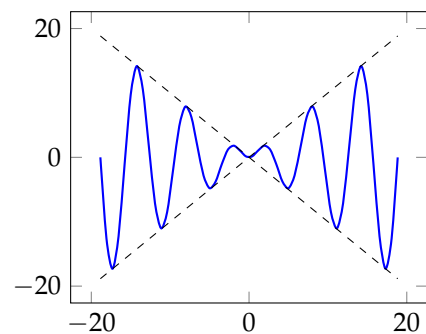
- con `xtick` e `ytick` si sceglie il valore delle tacche sugli assi x e y ;
- `xticklabels` e `yticklabels` producono le corrispondenti etichette se quelle automatiche non sono soddisfacenti.

Ecco un paio d'esempi con più `\addplot` consecutivi:

```
\begin{tikzpicture}
\begin{axis}
\addplot [samples=200,thick,
smooth,blue] {sin(deg(x^2))};
\addplot [dotted] {1};
\addplot [dotted] {-1};
\end{axis}
\end{tikzpicture}
```



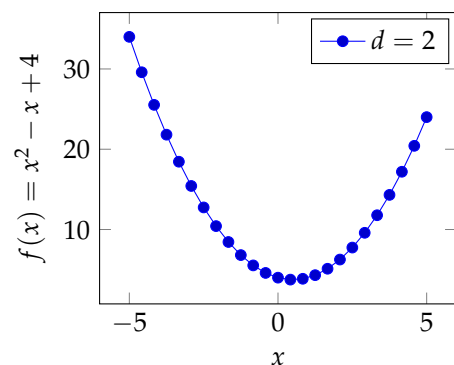
```
\begin{tikzpicture}
\begin{axis} [domain=-6*pi:6*pi]
\addplot [samples=50,smooth,
thick,blue] {x*sin(deg(x))};
\addplot [dashed] {x};
\addplot [dashed] {-x};
\end{axis}
\end{tikzpicture}
```



dove `dotted` disegna una linea punteggiata e `dashed` una linea tratteggiata (i tratti disponibili sono mostrati nella tabella 51 a pagina 174).

Il comando `\legend` produce una legenda:

```
\begin{tikzpicture}
\begin{axis}
[xlabel=$x$,
ylabel={$f(x)=x^2-x+4$}]
\addplot {x^2-x+4};
\legend{$d=2$}
\end{axis}
\end{tikzpicture}
```

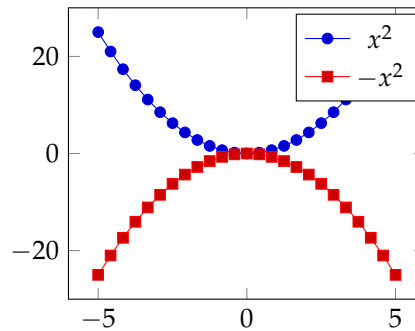


Si noti che:

- aspetto (riquadrata) e posizione (in alto a destra) della legenda sono predefiniti ma personalizzabili;
- se contiene caratteri particolari come = o la virgola, l'etichetta di un asse va racchiusa tra parentesi graffe;
- se è una formula matematica lunga, l'etichetta dell'asse y può rimanere nella posizione predefinita (in alternativa, si metta la formula nell'argomento di `title`).

Una legenda può contenere una voce per ciascun grafico tracciato:

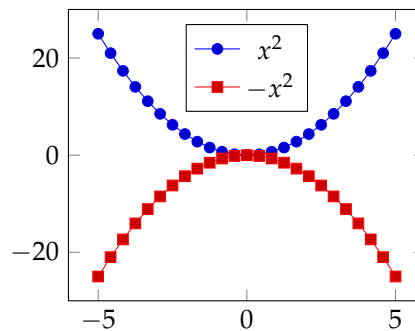
```
\begin{tikzpicture}
\begin{axis}
\addplot {x^2};
\addplot {-x^2};
\legend{$x^2$, $-x^2$}
\end{axis}
\end{tikzpicture}
```



Si noti che se in uno stesso sistema di riferimento ci sono più grafici, `pgplots` assegna automaticamente a ogni curva colore e marcatore distinti, scegliendoli secondo un ordine interno.

Come si vede, però, per impostazione predefinita la legenda compare sempre *dentro* il sistema di riferimento (se cartesiano) e può sovrapporsi al grafico nascondendone una parte. Si risolve il problema spostandola (la si può mettere in qualunque posizione dentro e fuori il sistema), come si mostra nell'esempio seguente:

```
\begin{tikzpicture}
\begin{axis}
[legend style={anchor=north,
at={(0.5,0.95)}}]
\addplot {x^2};
\addplot {-x^2};
\legend{$x^2$, $-x^2$}
\end{axis}
\end{tikzpicture}
```



Qualche prova e la lettura della documentazione del pacchetto permetteranno d'ottenere il risultato desiderato. Si noti che:

- `legend style` modifica lo stile predefinito della legenda;
- `anchor` specifica uno dei punti d'ancoraggio predefiniti per il riquadro della legenda (coincidenti essenzialmente con le direzioni d'una rosa dei venti a otto punte più un punto per il centro): qui `north` indica il punto medio del lato superiore;
- `at` definisce le coordinate del punto d'ancoraggio.

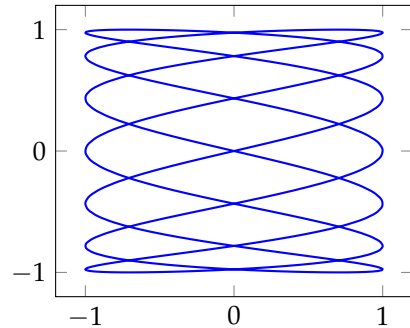
10.3.2 Curve in forma parametrica

Curve nel piano

Di seguito si mostrano alcuni esempi di curve nel piano.

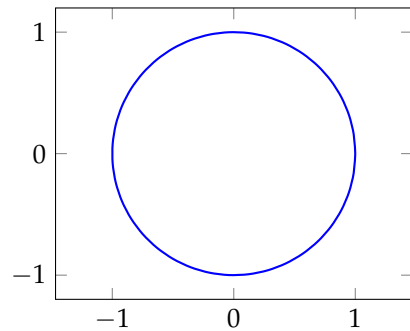
```
\begin{tikzpicture}
\begin{axis}
[title={Figura di Lissajous}]
\addplot
[domain=0:360,variable=\t,
samples=200,smooth,thick,blue]
({sin(7*t)},{sin(2*t)});
\end{axis}
\end{tikzpicture}
```

Figura di Lissajous



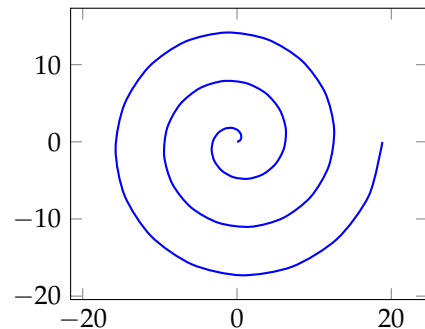
```
\begin{tikzpicture}
\begin{axis} [axis equal,
title={Circonferenza}]
\addplot
[domain=0:360,variable=\t,
samples=40,smooth,thick,blue]
({cos(t)},{sin(t)});
\end{axis}
\end{tikzpicture}
```

Circonferenza



```
\begin{tikzpicture}
\begin{axis} [axis equal,
title={Spirale di Archimede}]
\addplot
[domain=0:6*pi,variable=\t,
samples=50,smooth,thick,blue]
({t*cos(deg(t))},
{t*sin(deg(t))});
\end{axis}
\end{tikzpicture}
```

Spirale di Archimede



Si noti che:

- `axis equal` imposta la stessa unità di misura su entrambi gli assi;
- le istruzioni per le curve parametriche nel piano vanno date nella forma

```
{x}, {y});
```

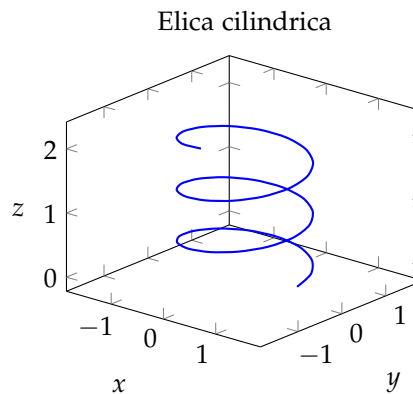
dove x e y sono funzioni del parametro;

- `variable` imposta il parametro (t , nei casi considerati), che nella propria definizione va preceduto da una barra rovescia.

Curve nello spazio

Ecco un esempio di curva nello spazio tridimensionale:

```
\begin{tikzpicture}
\begin{axis}
[view={40}{20},axis equal,
xlabel=$x$,ylabel=$y$,
zlabel=$z$,
zlabel style={rotate=-90},
title={Elica cilindrica}]
\addplot3
[domain=0:5.5*pi,variable=\t,
samples=40,samples y=0,
smooth,thick,blue]
({cos(deg(t))},{sin(deg(t))},
{2*t/(5*pi)});
\end{axis}
\end{tikzpicture}
```



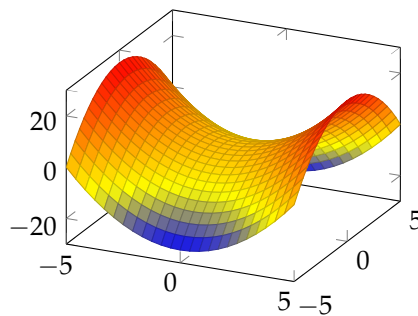
Si noti quanto segue.

- Il comando per tracciare grafici nello spazio è `\addplot3`, che richiede la sintassi spiegata nel paragrafo 10.2.2 a pagina 196.
- La chiave `view` imposta il punto di vista dell'osservatore (per una sua descrizione più completa si veda il paragrafo successivo).
- La chiave `zlabel` produce l'etichetta dell'asse z , ruotata con l'opzione descritta nel paragrafo 10.2.1 a pagina 192 per una maggiore leggibilità (in un grafico di questo tipo, in genere l'etichetta dell'asse y è già dritta).
- L'opzione `samples y=0` dichiara che si tratta di una curva e non di una superficie.
- Le istruzioni per le curve parametriche nello spazio richiedono la stessa sintassi già vista per quelle nel piano.

10.3.3 Funzioni reali di due variabili reali

Un paraboloide iperbolico per cominciare:

```
\begin{tikzpicture}
\begin{axis}
\addplot3 [surf]
{x^2-y^2};
\end{axis}
\end{tikzpicture}
```



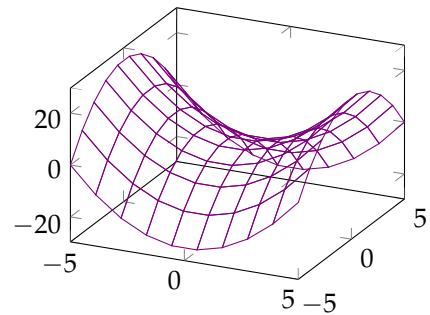
Si noti che `surf` disegna una superficie, visualizzata con colori predefiniti.

La chiave `mesh` produce grafici "a rete":

```

\begin{tikzpicture}
\begin{axis}
\addplot3
[mesh,samples=10,violet]
{x^2-y^2};
\end{axis}
\end{tikzpicture}

```

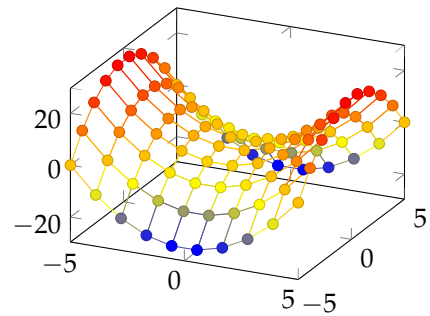


Aggiungendole scatter i nodi sono evidenziati con marcatori:

```

\begin{tikzpicture}
\begin{axis}
\addplot3
[mesh,scatter,samples=10]
{x^2-y^2};
\end{axis}
\end{tikzpicture}

```

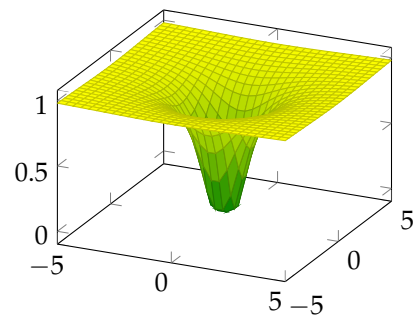


Ecco un altro esempio:

```

\begin{tikzpicture}
\begin{axis}
\addplot3 [samples=30,surf,
colormap/greenyellow]
{exp(-1/(x^2+y^2))};
\end{axis}
\end{tikzpicture}

```



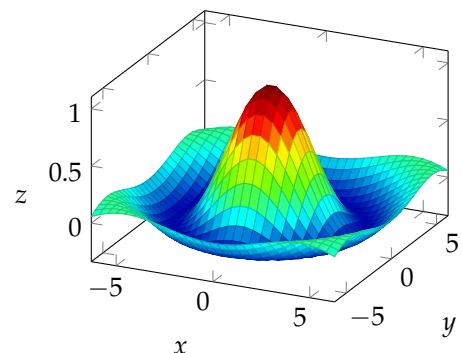
Si noti che colormap/⟨colorazione⟩ personalizza i colori predefiniti (si veda la documentazione del pacchetto per le possibilità disponibili, alcune delle quali verranno mostrate nei prossimi esempi).

Ora due varianti di uno stesso grafico ottenute modificando i valori di domain, che imposta il dominio della funzione. Nel primo:

```

\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [domain=-2*pi:2*pi,
samples=30,surf,
colormap/bluered]
{sin(deg(sqrt(x^2+y^2)))/%
sqrt(x^2+y^2)};
\end{axis}
\end{tikzpicture}

```



il dominio della funzione è $[-2\pi, 2\pi] \times [-2\pi, 2\pi]$: specificando solo domain, con domain=A il dominio della funzione è il quadrato $A \times A$.

Nel secondo:

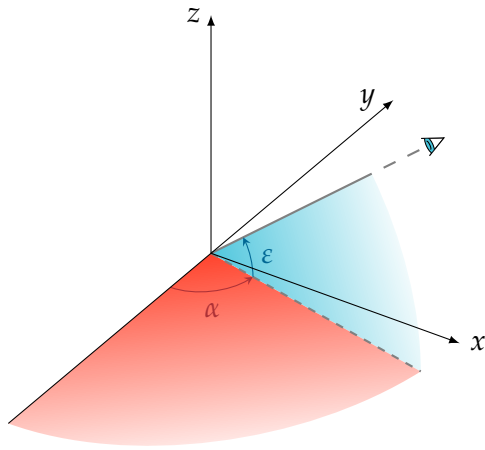
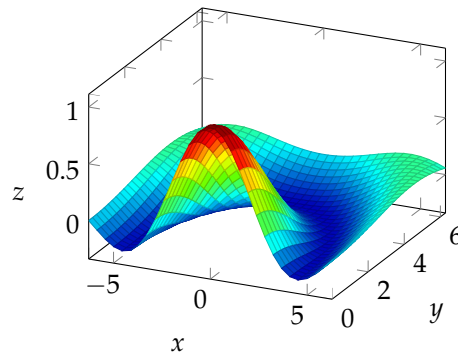


Figura 19: Definizione di azimuth (α) ed elevazione (ϵ) secondo la convenzione adottata da pgfplots. Nel caso mostrato: $\alpha = 70^\circ$ e $\epsilon = 35^\circ$.

```
\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [domain=-2*pi:2*pi,
y domain=0:2*pi,samples=30,
surf,colormap/bluered]
{sin(deg(sqrt(x^2+y^2)))/%
sqrt(x^2+y^2)};
\end{axis}
\end{tikzpicture}
```



il dominio della funzione è $[-2\pi, 2\pi] \times [0, 2\pi]$: specificando anche `y domain`, con `domain=A` e `y domain=B` il dominio della funzione è il rettangolo $A \times B$.

Punto di vista

La chiave `view` imposta il punto di vista dell'osservatore e richiede la seguente sintassi generale:

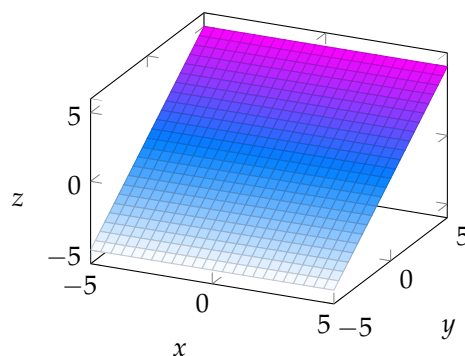
```
view={\langle azimuth \rangle}{\langle elevazione \rangle}
```

dove i due valori richiesti (pari a 25 e 30 per impostazione predefinita) indicano rispettivamente azimuth ed elevazione nel sistema di coordinate sferiche. La figura 19 mostra come pgfplots gestisce il punto di vista.

Seguono ora alcune varianti notevoli di uno stesso piano ottenute modificando i valori di `view`.

```
\begin{tikzpicture}
\begin{axis} [view={25}{30},
title={Punto di vista
predefinito},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}
```

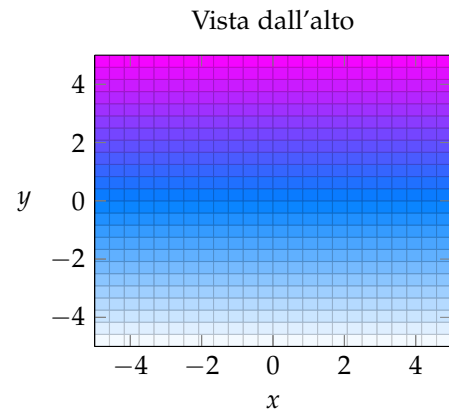
Punto di vista predefinito



```

\begin{tikzpicture}
\begin{axis} [view={0}{90},title=
{Vista dall'alto},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
ylabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}

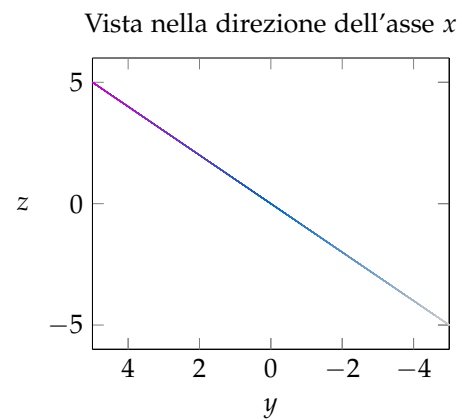
```



```

\begin{tikzpicture}
\begin{axis} [view={270}{0},
title={Vista nella direzione
dell'asse $x$},xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf,colormap/cool]
{y};
\end{axis}
\end{tikzpicture}

```



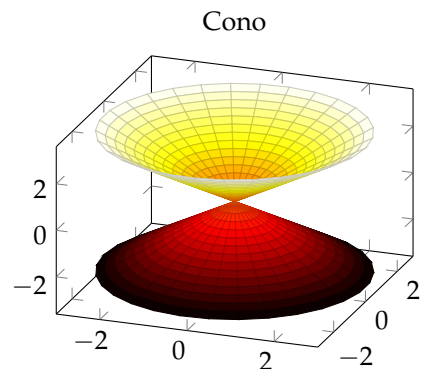
10.3.4 Superfici in forma parametrica

Un esempio:

```

\begin{tikzpicture}
\begin{axis}
[view={20}{30},title={Cono}]
\addplot3
[domain=-3:3,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/hot2]
({u*cos(v)}, {u*sin(v)}, u);
\end{axis}
\end{tikzpicture}

```



Si noti che:

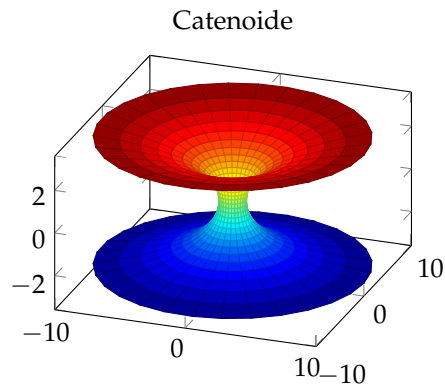
- variable y imposta il secondo parametro coordinato (v , in questo caso);
- la chiave `z buffer` suggerisce a `pgfplots` i criteri da seguire per proiettare i punti dello spazio tridimensionale sul quadro di proiezione (in questo caso, `sort` traccia per primi i segmenti più distanti dal punto di osservazione).

Di seguito si mostra una galleria d'esempi.

```

\begin{tikzpicture}
\begin{axis}
[view={20}{35},title={Catenoide}]
\addplot3
[domain=0:360,y domain=-3:3,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/jet]
({cos(u)*cosh(v)},
{sin(u)*cosh(v)},v);
\end{axis}
\end{tikzpicture}

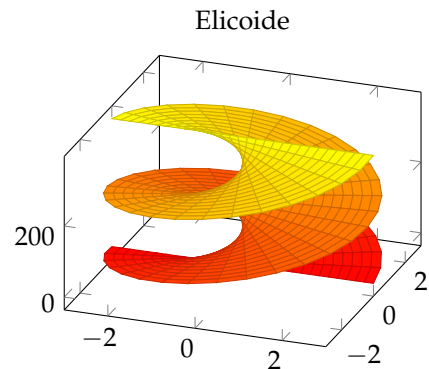
```



```

\begin{tikzpicture}
\begin{axis}
[view={20}{35},title={Elicoide}]
\addplot3
[domain=-3:3,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/rediyellow]
({u*cos(v)}, {u*sin(v)}, v);
\end{axis}
\end{tikzpicture}

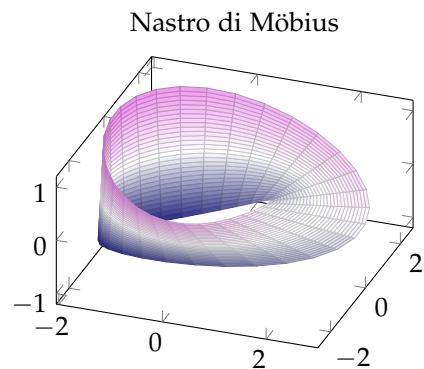
```



```

\begin{tikzpicture}
\begin{axis}
[view={20}{45},
title={Nastro di Möbius}]
\addplot3
[domain=0:360,y domain=-1:1,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/violet]
({2*cos(u)+v*cos(u)*cos(u/2)},
{2*sin(u)+v*sin(u)*sin(u/2)},
{v*sin(u/2)});
\end{axis}
\end{tikzpicture}

```



```

\begin{tikzpicture}
\begin{axis}
[view={20}{55},title={Toro}]
\addplot3
[domain=0:360,y domain=0:360,
variable=\u,variable y=\v,
samples=30,z buffer=sort,
surf,colormap/blackwhite]
(({3+cos(u))*cos(v)},
(({3+cos(u))*sin(v)},
{sin(u)});
\end{axis}
\end{tikzpicture}

```

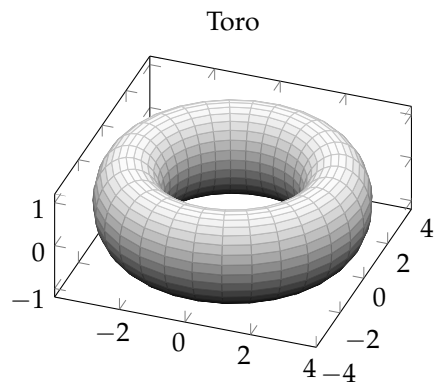


















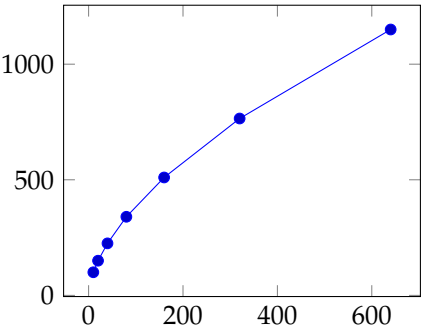
Tabella 55: Alcuni marcatori disponibili in pgfplots

Chiave	Risultato	Chiave	Risultato
*		square	
o		square*	
x		halfsquare*	
+		triangle	
asterisk		triangle*	
star		diamond	
oplus		diamond*	
otimes		halfdiamond*	

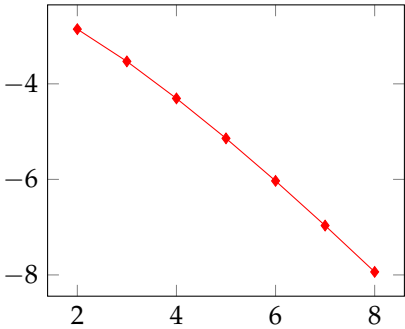
10.4 CURVE E SUPERFICI DATE PER COORDINATE

I prossimi sono due esempi di curve date per coordinate:

```
\begin{tikzpicture}
\begin{axis}
\addplot coordinates
{(10, 100) (20, 150)
(40, 225) (80, 340)
(160, 510) (320, 765)
(640, 1150)};
\end{axis}
\end{tikzpicture}
```



```
\begin{tikzpicture}
\begin{axis}
\addplot [red,mark=diamond*]
coordinates
{(2, -2.855) (3, -3.530)
(4, -4.305) (5, -5.141)
(6, -6.032) (7, -6.967)
(8, -7.937)};
\end{axis}
\end{tikzpicture}
```



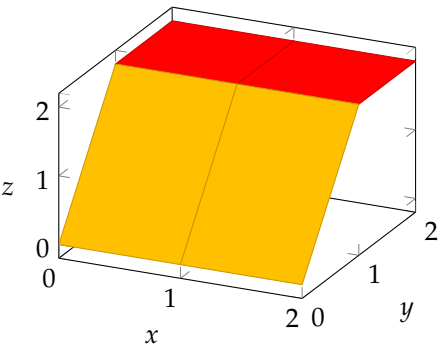
Si noti che mark imposta il tipo di marcatore, nel secondo esempio un rombo. In alternativa, se ne può scegliere un altro tra quelli mostrati nella tabella 55 o definirne uno personale.

Infine un esempio di superficie data per coordinate:

```
\begin{tikzpicture}
\begin{axis} [xlabel=$x$,
ylabel=$y$,zlabel=$z$,
zlabel style={rotate=-90}]
\addplot3 [surf] coordinates
{(0,0,0) (1,0,0) (2,0,0)

(0,1,2) (1,1,2) (2,1,2)

(0,2,2) (1,2,2) (2,2,2)};
\end{axis}
\end{tikzpicture}
```



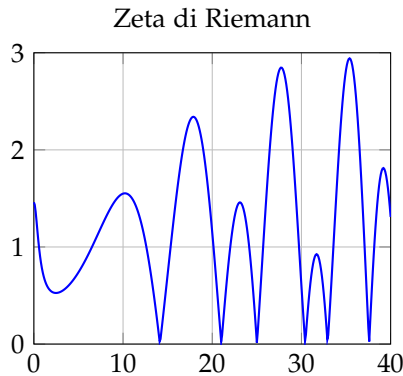
Si noti che in questo caso la sequenza delle coordinate assume la struttura di una matrice, nella quale ogni riga di valori va separata da quella successiva con una riga vuota.

10.5 CURVE E SUPERFICI CAMPIONATE DA FILE

A titolo d'esempio, si mostrano qui tre grafici (i primi due nel piano, il terzo nello spazio) costruiti con l'aiuto di file esterni ottenuti nei modi spiegati nel paragrafo 10.2.2 a pagina 196.

Il primo esempio mostra il grafico del valore assoluto della funzione Zeta di Riemann sulla retta $\text{Re } z = 1/2$.

```
\begin{tikzpicture}
\begin{axis} [xmin=0,xmax=40,
ymin=0,ymax=3,grid=major,
title={Zeta di Riemann}]
\addplot [thick,blue]
file {zeta.txt};
\end{axis}
\end{tikzpicture}
```

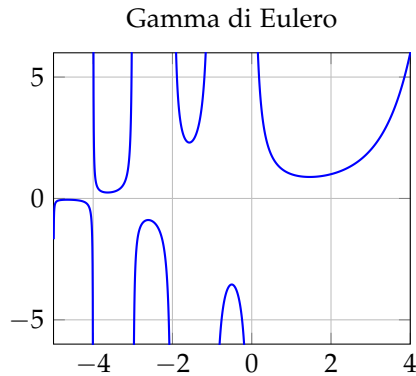


Di seguito si mostrano le prime righe del file `zeta.txt`:

```
0.0  1.460354508809587
0.1  1.433807867750897
0.2  1.362770945580488
0.3  1.266515016158303
```

Il prossimo esempio mostra il grafico della funzione Gamma di Eulero:

```
\begin{tikzpicture}
\begin{axis} [xmin=-5,xmax=4,
ymin=-6,ymax=6,grid=major,
title={Gamma di Eulero}]
\addplot
[unbounded coords=jump,
thick,blue]
file {gamma.txt};
\end{axis}
\end{tikzpicture}
```



dove l'opzione `unbounded coords=jump` gestisce i punti di discontinuità. Ed ecco le prime righe del file `gamma.txt`:

```
-5.000  NaN
-4.995  -1.6810104460206580
-4.990  -0.8478047198471037
-4.985  -0.5701560523263895
```

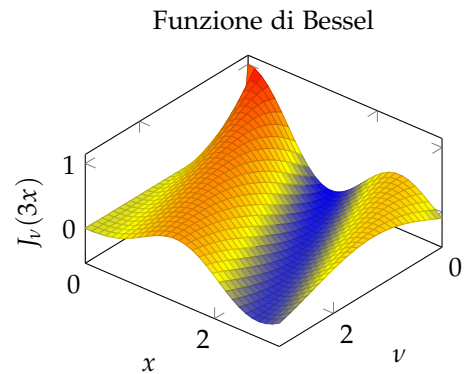
Si noti che le righe contenenti i valori NaN (*Not a Number*) e inf (*infinity*), indicanti rispettivamente un valore non numerico e infinito, sono sempre ignorate da `pgfplots`.

Il prossimo grafico, infine, mostra la funzione di Bessel (di prima specie):

```

\begin{tikzpicture}
\begin{axis} [view={130}{50},
  xlabel=$\nu$,ylabel=$x$,
  zlabel={$J_{\nu}(3x)$},
  title={Funzione di Bessel}]
\addplot3
[surf,z buffer=sort]
file {bessel.txt};
\end{axis}
\end{tikzpicture}

```



Di seguito si mostrano le prime righe del file `bessel.txt`:

```

0.000000 0.000000 1.000000
0.000000 0.103448 0.976066
0.000000 0.206897 0.905981
0.000000 0.310345 0.794755

```

Si noti che in questo caso le terne di coordinate che rappresentano i punti devono rispettare un ordine ben preciso ed essere in un formato opportuno (la documentazione del pacchetto spiega come farlo).

10.6 ALTRI SISTEMI DI RIFERIMENTO

In questa sezione si mostrano esempi degli altri sistemi di riferimento definiti da `pgfplots`. Alcuni di essi richiedono di caricare *nel preambolo* la libreria indicata con:

```

\usepgfplotslibrary{<libreria>}

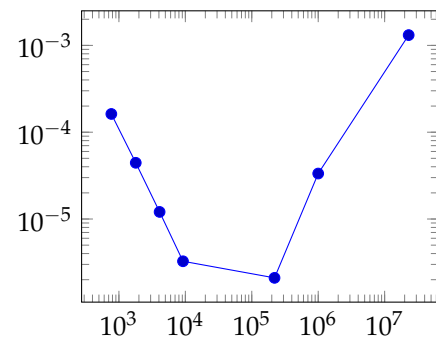
```

Per cominciare, un esempio di piano cartesiano logaritmico:

```

\begin{tikzpicture}
\begin{loglogaxis}
\addplot coordinates
{(769, 1.6227e-04)
(1793, 4.4425e-05)
(4097, 1.2071e-05)
(9217, 3.2610e-06)
(2.2e5, 2.1E-6)
(1e6, 0.00003341)
(2.3e7, 0.00131415)};
\end{loglogaxis}
\end{tikzpicture}

```

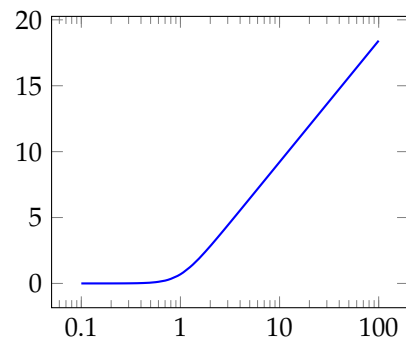


Ora un grafico con ascissa logaritmica:

```

\begin{tikzpicture}
\begin{semilogxaxis}
[log ticks with fixed point]
\addplot
[domain=0.1:100,
  thick,blue,smooth]
{ln(1+x^4)};
\end{semilogxaxis}
\end{tikzpicture}

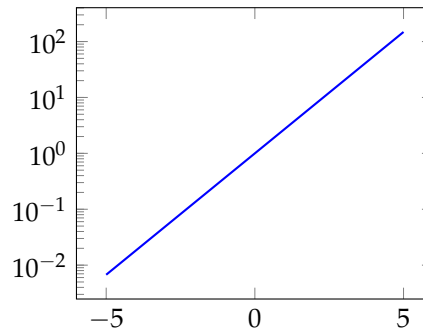
```



L'opzione `log ticks with fixed point` evita marcatori di tacca con esponenti.

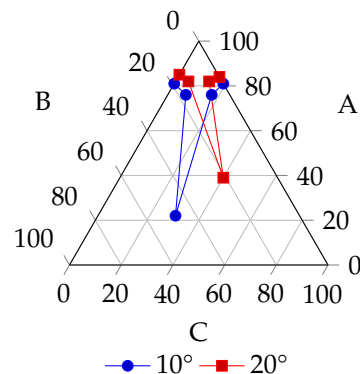
Il prossimo è un grafico con ordinata logaritmica:

```
\begin{tikzpicture}
\begin{semilogyaxis}
\addplot [thick,blue]
{exp(x)};
\end{semilogyaxis}
\end{tikzpicture}
```



Segue un esempio di diagramma ternario (richiede la libreria `ternary`):

```
\begin{tikzpicture}
\begin{ternaryaxis}
[xlabel=A,ylabel=B,zlabel=C,
legend style={anchor=north,
at={(0.5,-0.35)},draw=none},
legend columns=-1]
\addplot3 coordinates
{(0.81, 0.19, 0.00)
(0.76, 0.17, 0.07)
(0.22, 0.40, 0.30)
(0.76, 0.07, 0.17)
(0.81, 0.00, 0.19)};
\addplot3 coordinates
{(0.85, 0.15, 0.00)
(0.82, 0.13, 0.05)
(0.39, 0.30, 0.40)
(0.82, 0.06, 0.13)
(0.84, 0.00, 0.16)};
\legend{$10^\circ$ \textdegree,
$20^\circ$ \textdegree}
\end{ternaryaxis}
\end{tikzpicture}
```

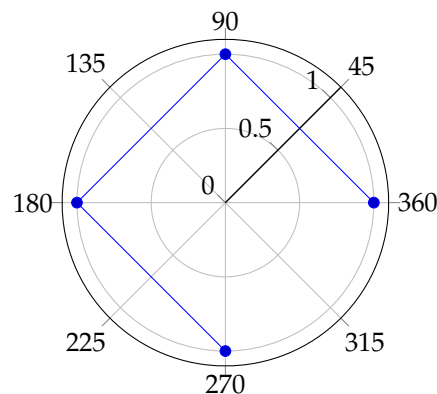


Si noti che;

- `draw=none` elimina il riquadro della legenda;
- `legend columns=-1` ne dispone gli elementi orizzontalmente.

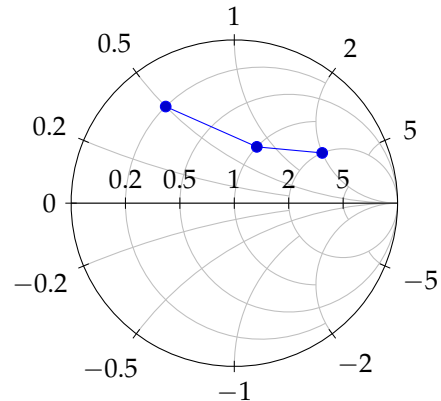
Il prossimo è un sistema di coordinate polari (richiede la libreria `polar`):

```
\begin{tikzpicture}
\begin{polaraxis}
[xmin=45,xmax=405]
\addplot coordinates
{(0, 1) (90, 1)
(180, 1) (270, 1)};
\end{polaraxis}
\end{tikzpicture}
```



Infine una carta di Smith (richiede la libreria `smithchart`):

```
\begin{tikzpicture}
\begin{smithchart}
\addplot coordinates
{(0.2, 0.5) (1, 0.8) (2, 2)};
\end{smithchart}
\end{tikzpicture}
```



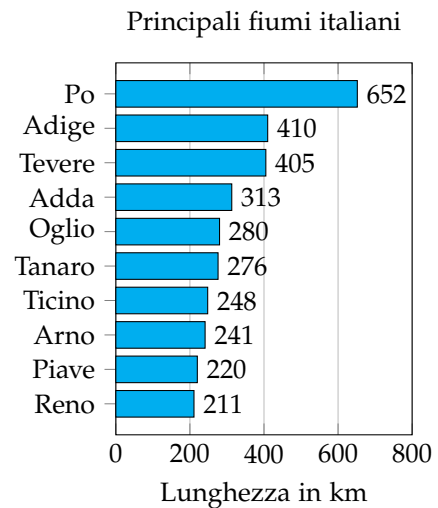
10.7 DIAGRAMMI A BARRE

Con `pgfplots` si possono realizzare anche diagrammi a barre. L'*ortogramma* si distingue dall'*istogramma* non solo per il diverso orientamento delle barre, ma anche perché di solito queste sono staccate tra di loro.

10.7.1 Ortogrammi

Un ortogramma:

```
\begin{tikzpicture}
\begin{axis}
[xbar,xmin=0,xmax=800,
height=6.5cm,
xmajorgrids=true,
ytick pos=left,
title={Principali fiumi
italiani},
xlabel={Lunghezza in km},
symbolic y coords={Reno,Piave,
Arno,Ticino,Tanaro,Oglio,Adda,
Tevere,Adige,Po},
ytick=data,nodes near coords,
nodes near coords align=
{horizontal}]
\addplot
[fill=cyan,draw=black]
coordinates
{(211,Reno) (220,Piave)
(241,Arno) (248,Ticino)
(276,Tanaro) (280,Oglio)
(313,Adda) (405,Tevere)
(410,Adige) (652,Po)};
\end{axis}
\end{tikzpicture}
```



Si noti che:

- `xbar` produce le barre orizzontali;

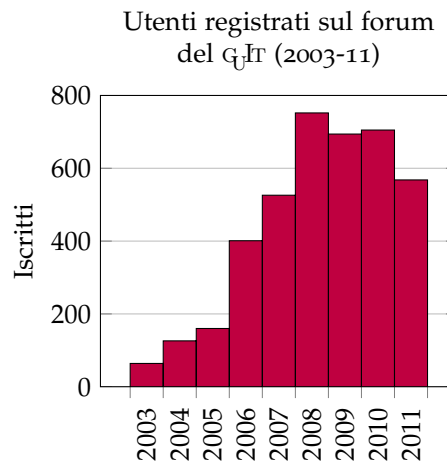
- `xmajorgrids` produce una “griglia” di sole linee verticali;
- `ytick pos` imposta la posizione delle tacche sull’asse *y*;
- `symbolic y coords` produce il proprio argomento come etichetta della barra;
- `data` attribuisce alle etichette i valori dichiarati come ordinate;
- `node near coords` mette il valore indicato nelle coordinate alla fine di ogni barra;
- `node near coords align` ne aggiusta l’allineamento in modo che non si sovrapponga alla barra;
- `fill` imposta il colore che riempirà le barre;
- `draw` ne imposta il colore di contorno.

Si possono creare ortogrammi anche con il pacchetto `bchart` (se ne veda la documentazione), che richiede una sintassi più semplice.

10.7.2 Istogrammi

Un istogramma:

```
\begin{tikzpicture}
\begin{axis} [ylabel={Iscritti},
ybar,ymin=0,ymax=800,xtick=data,
ymajorgrids=true,xtick pos=left,
x tick label style={
{rotate=90,anchor=east},
xticklabel interval boundaries,
symbolic x coords={$2003$,
$2004$, $2005$, $2006$, $2007$,
$2008$, $2009$, $2010$, $2011$,
$2012$},
title style={align=center},
title={Utenti registrati sul
forum\\ del \GuIT{} (2003-11)}]
\addplot
[ybar interval,fill=purple,
draw=black] coordinates
{($2003$, 64) ($2004$, 126)
($2005$, 160) ($2006$, 401)
($2007$, 526) ($2008$, 752)
($2009$, 694) ($2010$, 705)
($2011$, 568) ($2012$, 0)};
\end{axis}
\end{tikzpicture}
```



Alle osservazioni sul grafico precedente, qui riferibili all’asse delle *y*, si aggiungano le seguenti:

- `xtick pos` imposta la posizione delle tacche sull’asse *x*;
- `x tick label style` imposta l’aspetto delle etichette delle colonne (qui ruotate di 90° in senso antiorario);

- `xticklabel interval boundaries` centra l'etichetta di ogni colonna;
- `title style` imposta lo stile del titolo, necessario per averlo su due righe;
- `ybar interval` rende adiacenti le colonne, altrimenti staccate;
- un istogramma richiede di definire una colonna *in più* rispetto a quelle necessarie (qui è quella relativa all'anno 2012).

Parte V

BIBLIOGRAFIA E INDICE ANALITICO

La bibliografia è da sempre uno degli aspetti più delicati di un documento, e \LaTeX aiuta anche in questo caso, definendo tutti gli strumenti per realizzarla e gestirla con efficienza e flessibilità. L'argomento è tecnico, perciò è bene chiarire da subito alcuni termini usati (anche in forma abbreviata) in questo capitolo:

BIBLIOGRAFIA Elenco di opere scritte o di altro tipo che di solito occupa una sezione autonoma del documento con un titolo (in genere) omonimo.

RIFERIMENTO BIBLIOGRAFICO Serie di dati che permette di identificare ed eventualmente reperire un'opera. L'insieme dei riferimenti bibliografici costituisce la bibliografia di un documento.

STILE BIBLIOGRAFICO Modalità generale adottata per presentare al lettore i riferimenti bibliografici.

CITAZIONE BIBLIOGRAFICA Indicazione sintetica, data nel corpo del documento, che rinvia il lettore a un riferimento bibliografico.

SCHEMA DI CITAZIONE Modalità generale adottata per presentare al lettore una citazione bibliografica.

Con \LaTeX si può creare la bibliografia in due modi, essenzialmente:

- a mano, con l'ambiente `thebibliography`;
- automaticamente, con il pacchetto `biblatex`.

11.1 BIBLIOGRAFIA MANUALE

L'ambiente `thebibliography` gestisce la bibliografia di un documento molto facilmente, ma non è altrettanto flessibile. La sintassi generale è:

```
\begin{thebibliography}{\langle etichetta più lunga \rangle}
\bibitem[\langle etichetta personalizzata \rangle]{\langle chiave di citazione \rangle}
...
\end{thebibliography}
```

dove:

- `\langle etichetta più lunga \rangle` può essere un numero (di solito 9 se la bibliografia comprende meno di dieci opere, 99 se almeno dieci ma meno di cento, eccetera) oppure del testo (nel caso di etichette personalizzate: si scrive allora l'etichetta più lunga);
- `\bibitem` va premesso a ogni riferimento bibliografico;
- `\langle etichetta personalizzata \rangle` è un'etichetta personalizzata che *eventualmente* comparirà nella bibliografia e nelle citazioni al posto del numero predefinito;

- $\langle \text{chiave di citazione} \rangle$ è l'etichetta *univoca* per citare la fonte nel documento (si consiglia di usare la sintassi $\langle \text{autore} \rangle : \langle \text{titolo} \rangle$ analoga a quella di $\backslash \text{label}$).

Lo si vede all'opera nell'esempio seguente:

```
\begin{thebibliography}{9}
\bbibitem{eco:tesi}
Eco, Umberto (1977),
\emph{Come si fa una tesi di
laurea}, Bompiani, Milano.

\bbibitem{mori:tesi}
Mori, Lapo Filippo (2007),
"Scrivere la tesi di laurea
con \LaTeXe", \Ars~(3).
\end{thebibliography}
```

BIBLIOGRAFIA

- [1] Eco, Umberto (1977), *Come si fa una tesi di laurea*, Bompiani, Milano.
- [2] Mori, Lapo Filippo (2007), "Scrivere la tesi di laurea con \LaTeXe ", *\Ars* (3).

Si noti che:

- thebibliography si comporta in modo molto simile a un ambiente per elenchi;
- ciascun riferimento bibliografico va scritto per intero, regolandone *a mano* tutti gli aspetti (corsivo, virgolette, eccetera), compresa la posizione nell'ordine alfabetico;
- nel documento finito i riferimenti saranno contrassegnati con un numero tra parentesi quadre sia nella bibliografia sia nelle citazioni;
- thebibliography produce la sezione contenente la bibliografia con relativi titolo e testatina.

Per citare un riferimento bibliografico nel testo si usa il comando $\backslash \text{cite}$:

```
\cite[ $\langle \text{informazioni} \rangle$ ]{ $\langle \text{chiave di citazione} \rangle$ }
```

dove:

- $\langle \text{informazioni} \rangle$ sono ulteriori indicazioni (numero di pagina o di capitolo) che eventualmente si possono dare per completare la citazione;
- $\langle \text{chiave di citazione} \rangle$ si spiega da sé.

I prossimi esempi lo mostrano all'opera:

Si veda $\backslash \text{cite}\{\text{eco:tesi}\}$ per maggiori dettagli.

Si veda [1] per maggiori dettagli.

Si veda $\backslash \text{cite}[\text{p.~7}]\{\text{eco:tesi}\}$ per maggiori dettagli.

Si veda [1, p. 7] per maggiori dettagli.

Per mandare nell'indice generale il titolo della bibliografia del documento, *immediatamente prima* di aprire thebibliography vanno date sequenze di comandi diverse in base alla classe in uso:

```
\cleardoublepage
%\phantomsection
\addcontentsline{toc}{chapter}{\bibname}
```


se la classe è book o report, e

```
%\clearpage
%\phantomsection
\addcontentsline{toc}{section}{\refname}
```

se la classe è article, dove:

- `\cleardoublepage` fa cominciare la bibliografia in una pagina nuova dispari e assegna alla corrispondente voce nell'indice il numero di pagina corretto;
- `\clearpage` va dato per assicurare la corretta assegnazione del numero di pagina alla voce nell'indice *solo se* a fine composizione il corpo del documento terminasse esattamente a fine pagina e la bibliografia cominciasse all'inizio di una pagina nuova (in tal caso si decommenti la riga corrispondente);
- `\phantomsection` va dato *solo se* è caricato anche hyperref (in tal caso si decommenti la riga corrispondente);
- `chapter` e `section` indicano il livello della sezione bibliografica (un capitolo e un paragrafo, rispettivamente);
- `\bibname` e `\refname` producono nell'indice generale del documento le voci *Bibliografia* e *Riferimenti bibliografici* rispettivamente.

11.2 BIBLIOGRAFIA AUTOMATICA

L'ambiente thebibliography produce risultati dignitosi, ma a prezzo di alcuni inconvenienti non da poco:

- l'ordinamento alfabetico dei riferimenti nella bibliografia non è automatico;
- bisogna scrivere una bibliografia per ogni documento, anche se c'è solo qualche opera di differenza dall'uno all'altro;
- se si aggiorna un documento, bisogna modificare le bibliografie di *tutti* i documenti in cui esso è citato;
- per cambiare lo stile bibliografico bisogna modificare a mano *tutte* le voci, una per una.

Come si noterà, anziché compilare bibliografie indipendenti è decisamente meglio generarle *automaticamente* creando *una volta per tutte* un solo database: sistemandolo in un'opportuna cartella dell'*albero personale* (per esempio, \$HOME/texmf/bibtex/bib/mieibib) basterà aggiornarlo all'occorrenza.

In questo paragrafo si presentano gli elementi fondamentali del pacchetto biblatex, un potentissimo strumento per gestire automaticamente la bibliografia e personalizzare ogni aspetto degli stili bibliografici e di citazione con poche operazioni.

Per funzionare correttamente, il pacchetto richiede di caricare *anche* babel (o polyglossia, se si compone con X_YL^AT_EX, si veda il paragrafo 18.2.2 a pagina 297) e csquotes con le opzioni indicate di seguito, rispettando l'ordine dei comandi:

```
\usepackage[autostyle,italian=guillemets,<altre opzioni>]{csquotes}
\usepackage[<opzioni>,backend=biber]{biblatex}
```

Si noti che:

- `autostyle` adatta lo stile delle citazioni alla lingua corrente del documento;
- `italian=guillemets` racchiude automaticamente tra virgolette caporali i campi che prevedono le virgolette;
- `backend=biber` dice a `biblatex` che s'intende usare Biber come motore bibliografico (si veda il paragrafo successivo).

11.2.1 Biber

Da qualche anno il nuovo motore bibliografico predefinito da `biblatex` è Biber. `BIBTEX` è ancora utilizzabile, ma il fatto che lavori con una versione obsoleta di `biblatex` suggerisce di preferire senz'altro il primo, il cui sviluppo procede parallelamente a quello del pacchetto presentato in questo capitolo. Ed è questa la scelta qui raccomandata. Gli editor per `LATEX` presentati nel paragrafo 2.1.1 a pagina 13, tuttavia, sono ancora preimpostati per lavorare con `BIBTEX`. In attesa che la situazione si stabilizzi, si danno di seguito le semplici istruzioni per passare a Biber:

TEXSTUDIO Si segua il percorso opzioni → Configure TeXstudio... e nella riga BibTeX si sostituisca `biber` a `bibtex`.

TEXSHOP Si segua il percorso TeXShop → Preferenze... → Motore e nella riga BibTeX Engine si sostituisca `biber` a `bibtex`.

11.2.2 Database bibliografici

Un *database bibliografico* è un file da registrare con estensione `.bib` (si scrive con l'editor in uso e *non* va composto) che contiene un certo numero di *record* come i seguenti:

```
@book{eco:tesi,
  author    = {Eco, Umberto},
  title      = {Come si fa una tesi di laurea},
  publisher  = {Bompiani},
  date       = {1977},
  location   = {Milano},
}

@article{mori:tesi,
  author      = {Mori, Lapo Filippo},
  title       = {Scrivere la tesi di laurea con \LaTeXe},
  journaltitle = {\Ars},
  number      = {3},
  date        = {2007},
}

>manual{mori:poesie,
  author      = {Mori, Lapo Filippo},
```

```

    title      = {Scrivere poesie con \LaTeX},
    date       = {2007},
    url        = {http://www.guitex.org/},
}

@online{wiki:latex,
  title      = {\LaTeX{} su Wikipedia},
  date       = {2017},
  url        = {http://it.wikipedia.org/wiki/LaTeX},
  sortkey    = {wiki},
  label      = {wiki},
}

```

Si osservi che:

- Ogni record corrisponde a un riferimento bibliografico, il cui tipo va indicato come prima cosa (indifferentemente in maiuscolo o minuscolo) subito dopo @. Si tratta in questo caso di un libro (record di tipo @book), di un articolo (@article), di un manuale (@manual) e di un sito Web (@online).
- S'indica poi una *chiave*, di solito nella forma $\langle \text{cognome} \rangle : \langle \text{parola chiave} \rangle$ (dove $\langle \text{parola chiave} \rangle$ è un segnaposto univoco della fonte da citare), da usare nell'argomento dei comandi per le citazioni.
- Infine, si riempiono i *campi* che definiscono l'opera (autore, titolo, eccetera). Di qualunque tipo sia, un campo ha la forma

$\langle \text{nome del campo} \rangle = \{ \langle \text{contenuto del campo} \rangle \},$

e, come si vede, va terminato con la virgola, *anche se è l'ultimo*, pena un errore.

Per indicare a L^AT_EX quale o quali database usare per comporre la bibliografia si dà nel preambolo il comando

`\addbibresource{ $\langle \text{nome del database} \rangle$.bib}`

ripetendolo per ogni base di dati e *specificando l'estensione* .bib.

11.2.3 Record e campi

Ogni record contiene uno o più *campi*. Un campo può essere:

- *obbligatorio*, cioè indispensabile a biblatex per comporre il riferimento bibliografico;
- *opzionale*, cioè non indispensabile ma utile, se specificato, per aggiungerci informazioni.

I tipi di record e i relativi campi disponibili con biblatex sono numerosissimi e per tutte le esigenze; il loro elenco completo si trova nella documentazione del pacchetto.

Record

Di seguito si elencano i principali *record* standard riconosciuti da biblatex, indicando di ognuno campi obbligatori e principali campi opzionali (il loro significato verrà spiegato nel paragrafo successivo). In particolari situazioni, si possono sostituire i campi in corsivo con degli altri (si veda il paragrafo 11.2.7 a pagina 229).

@article Articolo apparso in una rivista o in un giornale.

Campi obbligatori: *author*, *title*, *journaltitle*, *date*.

Campi opzionali: *editor*, *series*, *volume*, *number*, *month*, *pages*, *note*, *url*.

@book Libro regolarmente pubblicato da una casa editrice.

Campi obbligatori: *author*, *title*, *date*.

Campi opzionali: *editor*, *volume*, *series*, *note*, *publisher*, *location*, *number*, *url*.

@collection Raccolta di contributi indipendenti di autori diversi.

Campi obbligatori: *editor*, *title*, *date*.

Campi opzionali: *volume*, *series*, *number*, *publisher*, *location*, *note*, *chapter*, *pages*, *url*.

@proceedings Atti di un convegno o conferenza.

Campi obbligatori: *editor*, *title*, *date*.

Campi opzionali: *volume*, *series*, *number*, *note*, *location*, *publisher*, *organization*, *chapter*, *pages*, *url*.

@inbook Parte di un libro con un titolo proprio.

Campi obbligatori: *author*, *title*, *booktitle*, *date*.

Campi opzionali: *editor*, *volume*, *series*, *note*, *publisher*, *location*, *chapter*, *number*, *pages*, *url*.

@bookinbook Opera pubblicata originariamente come libro autonomo e ristampata, per esempio, nell'*opera omnia* di un autore. Per i campi, si veda la voce precedente.

@incollection Parte di una raccolta con un titolo proprio.

Campi obbligatori: *author*, *title*, *booktitle*, *date*.

Campi opzionali: *volume*, *series*, *number*, *publisher*, *location*, *note*, *chapter*, *pages*, *url*.

@inproceedings Articolo negli atti di un convegno o intervento in una conferenza.

Campi obbligatori: *author*, *editor*, *title*, *booktitle*, *date*.

Campi opzionali: *volume*, *series*, *number*, *note*, *location*, *publisher*, *organization*, *chapter*, *pages*, *url*.

@booklet Libro distribuito in proprio.

Campi obbligatori: *author* o *editor*, *title*, *date*.

Campi opzionali: *howpublished*, *type*, *note*, *location*, *chapter*, *pages*, *url*.

@manual Documentazione tecnica.

Campi obbligatori: *author* o *editor*, *title*, *date*.

Campi opzionali: *type*, *version*, *series*, *number*, *organization*, *note*, *publisher*, *location*, *chapter*, *pages*, *url*.

@report Relazione pubblicata da università, scuola o altra istituzione.

Campi obbligatori: *author*, *title*, *type*, *institution*, *date*.

Campi opzionali: *number*, *note*, *location*, *chapter*, *pages*, *url*.

@thesis Tesi di laurea o di dottorato. Il campo *type* specifica il tipo di tesi.

Campi obbligatori: *author*, *title*, *type*, *institution*, *date*.

Campi opzionali: *note*, *location*, *chapter*, *pages*, *url*.

@online Risorsa disponibile su Internet.

Campi obbligatori: *author* o *editor*, *title*, *date*, *url*.

Campi opzionali: *note*, *organization*, *date*.

@unpublished Documento con autore e titolo, ma non pubblicato.

Campi obbligatori: *author*, *title*, *date*.

Campi opzionali: *howpublished*, *note*, *date*, *url*.

@misc Record da usare quando nessuno degli altri è appropriato.

Campi obbligatori: *author* o *editor*, *title*, *date*.

Campi opzionali: *howpublished*, *type*, *organization*, *location*, *note*, *date*, *url*.

Campi

Di seguito si riporta l'elenco dei principali *campi* riconosciuti da biblatex.

author Cognome e nome dell'autore (o degli autori, nel formato descritto più avanti) separati con la virgola.

booktitle Titolo dell'opera che contiene la fonte (se essa è solo una parte della pubblicazione).

chapter Numero del capitolo (o di una qualunque parte del documento).

date Anno di pubblicazione della fonte.

edition Numero di edizione della fonte.

editor Cognome e nome del curatore (o dei curatori) separati con la virgola.

howpublished Genere della pubblicazione.

institution Nome dell'università o dell'istituzione.

journaltitle Nome della rivista o del giornale.

label Etichetta per le citazioni, nel caso in cui manchino i dati necessari per formare l'etichetta "regolare" (si veda il paragrafo 11.2.7 a pagina 229).

location Indirizzo di editore (*publisher*) o istituzione (*institution*).

month Mese di pubblicazione dell'opera. Deve essere un numero intero; per esempio, non si scrive *month*={January}, ma *month*={1}.

note Informazioni supplementari che possono aiutare il lettore a identificare meglio l'opera.

number Numero della rivista, del giornale o della raccolta.

organization Organizzazione che pubblica il documento o patrocina la conferenza.

pages Uno o più numeri di pagina.

publisher Nome dell'editore.

series Nome della collezione di cui la fonte fa parte.

sortkey Ordina alfabeticamente le voci bibliografiche prive dell'indicazione di autore o curatore (si veda il paragrafo [11.2.7](#) a pagina [229](#)).

title Titolo della fonte.

type Tipo di manuale, relazione o tesi.

url Indirizzo Internet di riferimento per la fonte.

volume Numero di volume della fonte.

Alcune precisazioni

Gli stili *predefiniti* di biblatex producono il titolo dell'opera rispettando le maiuscole e le minuscole. Un titolo come

```
title = {TCP-IP e lo Zen di Confucio}
```

verrà prodotto esattamente com'è scritto.

Un comando che produce testo presente in un campo non va messo tra graffe se si lavora con uno stile predefinito:

```
title = {L'arte di scrivere con \LaTeX}
```

ma potrebbe essere necessario farlo se lo stile è personalizzato:

```
title = {L'arte di scrivere con {\LaTeX}}
```

Più nomi in un campo `author` o `editor` vanno separati con `and`:

```
author = {Mori, Lapo Filippo and Himmelmann, Maurizio}
```

Se l'elenco dei nomi degli autori o dei curatori è troppo lungo, può essere concluso da `and others`, che di regola viene reso da biblatex con *et al.*:

```
author = {Gregorio, Enrico and Mori, Lapo Filippo and  
Pantieri, Lorenzo and others}
```

I cognomi multipli si scrivono separandoli dal nome in questo modo:

```
author = {Levi Montalcini, Rita}
```

I cognomi preceduti da particelle con iniziale *minuscola* (*von* o *van*, per esempio) richiedono qualche attenzione. S'immagini che l'autore sia *Ludwig van Beethoven*: per impostazione predefinita gli stili standard ignorano la particella precedente il cognome, ordinandolo alfabeticamente di conseguenza. L'opzione `useprefix` evita questo comportamento e dice a biblatex di considerare il cognome per intero secondo l'uso italiano. (Si noti che l'opzione

useprefix=false, predefinita negli stili standard, ignora eventuali particelle *qualunque stile si usi*.)

Se la particella ha l'iniziale maiuscola, invece, l'inconveniente appena descritto non si presenta: *De Gasperi* cadrà sotto la lettera *d*.

Si noti che alcuni stili *personalizzati* di biblatex producono automaticamente il titolo in tutte lettere minuscole, con risultati a volte indesiderati. Si può risolvere il problema racchiudendo ulteriormente tra graffe solo le parole problematiche, così:

```
title = {{TCP-IP} e lo {Zen} di {Confucio}}
```

oppure l'intero titolo, così:

```
title = {{TCP-IP e lo Zen di Confucio}}
```

11.2.4 Stili bibliografici e schemi di citazione

L'aspetto dei riferimenti bibliografici e delle citazioni, che biblatex adatta automaticamente alla lingua principale del documento, si specificano in modi diversi. Il codice

```
\usepackage[bibstyle=authortitle,citestyle=verbose-trad1]{biblatex}
```

imposta per la bibliografia lo *stile bibliografico* (bibstyle) autore-titolo (ovvero authortitle), e per le citazioni uno *stile di citazione* (citestyle) che riporta per intero il riferimento solo nella prima citazione del documento e usa abbreviazioni in quelle successive (verbose-trad1). Il codice

```
\usepackage[style=alphanumeric]{biblatex}
```

invece, imposta con lo stile (style) alfabetico (alphanumeric) entrambi i parametri. Lo schema (e quindi lo stile) più adatto da usare dipende *anche* dal tipo di lavoro che si sta scrivendo.

Stili bibliografici

Il pacchetto biblatex definisce quattro *stili bibliografici* predefiniti, i quali agiscono *nella sezione bibliografica del documento*:

- ordinano alfabeticamente le opere in base al cognome di autore o curatore;
- possono contrassegnare o meno l'opera con un'etichetta;
- sistemano diversamente i dati nei riferimenti bibliografici.

Di seguito li si descrive brevemente.

numeric Anno di pubblicazione: in fondo al riferimento.

Etichetta: numero progressivo ([1], [2], eccetera).

Uso: documenti scientifici, dove importa conoscere subito non *chi* viene citato, ma che *qualcuno* viene citato e solo poi *che cosa* questo qualcuno ha scritto.

alphanumeric Anno di pubblicazione: in fondo al riferimento.

Etichetta: prima parte del cognome dell'autore e ultime due cifre dell'anno di pubblicazione ([Moro7]).

authoryear Anno di pubblicazione: presente solo nell’etichetta.

Etichetta: cognome di autore o curatore e anno di pubblicazione ([Becari, 2012]).

Uso: documenti in cui è importante indicare questi due dati direttamente nella citazione.

authortitle Anno di pubblicazione: in fondo al riferimento.

Etichetta: no.

Uso: la citazione è una nota al piede contenente cognome dell’autore e titolo dell’opera. Si usa quasi esclusivamente nei documenti umanistici.

Schemi di citazione

A ciascuno stile si possono associare (non contemporaneamente) uno o più *schemi di citazione*, che ne riproducono l’effetto nelle citazioni fatte *nel corpo del documento*, ma ogni schema può essere associato a *un solo* stile bibliografico alla volta. Di seguito si descrivono quattro schemi predefiniti da biblatex e alcune loro varianti.

numeric, numeric-comp Riferimento: numerico ([1], [2], eccetera).

Da associare a: stile numeric.

Varianti: numeric-comp, che ordina e comprime le citazioni multiple.

Per esempio, si ottiene [2–4, 8] al posto di [4, 2, 8, 3].

alphabetic Riferimento: misto ([Moro7]).

Da associare a: stile alphabetic.

authoryear, authoryear-comp Riferimento: autore, anno ([Mori, 2007]).

Da associare a: stile authoryear.

Varianti: authoryear-comp, che ordina e comprime le citazioni multiple con lo stesso autore (ed eventualmente lo stesso anno di pubblicazione).

Per esempio, si ottiene [Eco 1977; Mori 2007a,b] anziché [Mori 2007b; Eco 1977; Mori 2007a].

authortitle, verbose, verbose-trad1 Riferimento: autore, titolo (Mori, *Titolo*).

Da associare a: stile authortitle.

Varianti: verbose, che cita per intero il riferimento solo la prima volta e usa una forma abbreviata quelle successive; verbose-trad1, variante del precedente, che a seconda del contesto, usa le formule latine *idem*, *ibidem*, *op. cit.* e *loc. cit.*

Per l’elenco completo degli schemi di citazione di biblatex si rimanda alla documentazione del pacchetto.

Per ogni schema di citazione c’è uno stile bibliografico adatto, che biblatex imposta automaticamente in base allo schema scelto. Il codice

```
\usepackage[style=authoryear-comp]{biblatex}
```

infatti, equivale a

```
\usepackage[bibstyle=authoryear,citestyle=authoryear-comp]{biblatex}
```

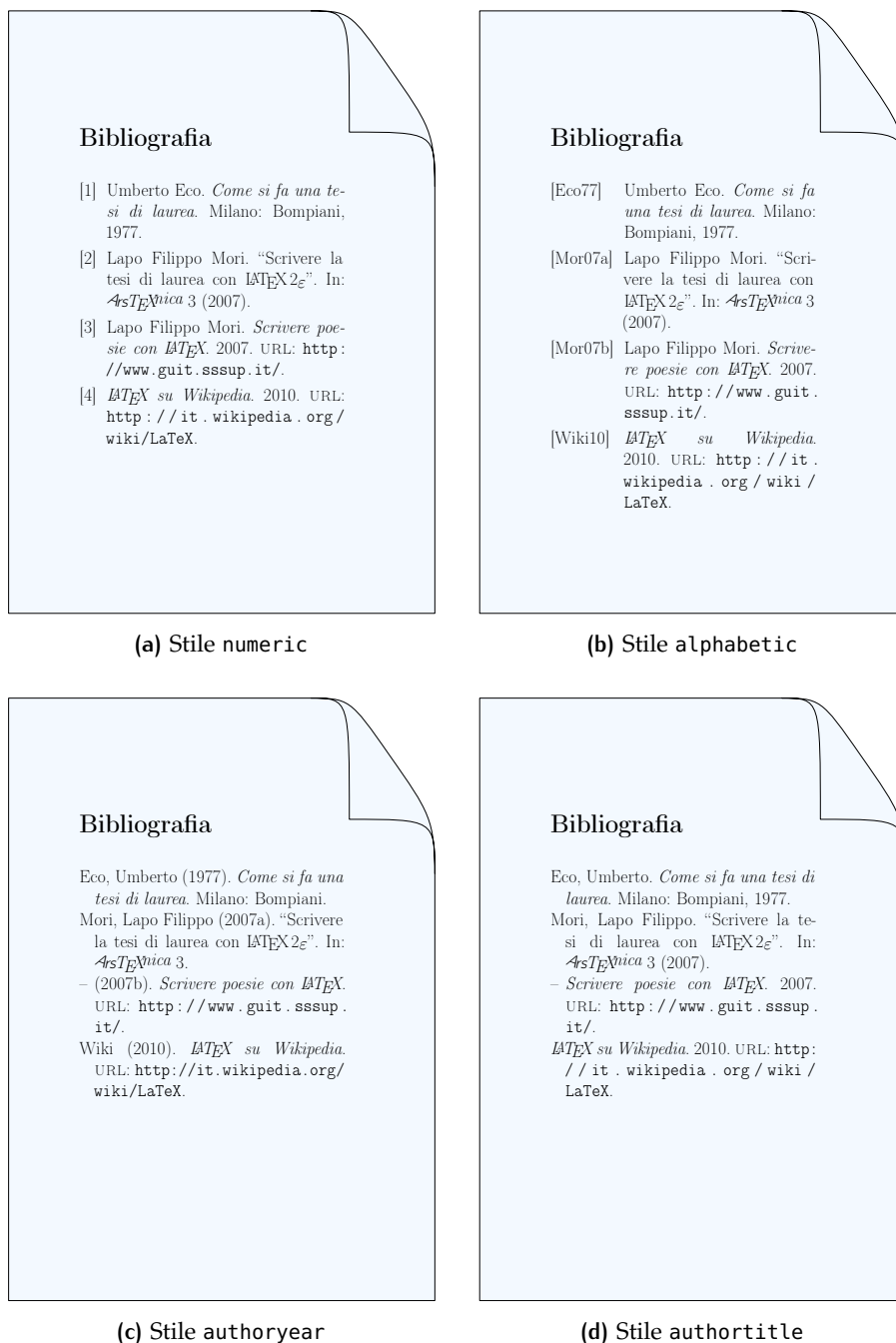



Figura 20: Esempi di stili bibliografici

Oltre a quelli predefiniti si possono usare molti altri stili: la bibliografia di questa guida, per esempio, è composta con lo stile `philosophy-modern`, presente in `TEX Live`. In Rete, infine, ce ne sono molti altri per comporre bibliografie conformi a precisi standard: si cerchi su  **CTAN**.

11.2.5 Comandi per le citazioni

Per citare un'opera nel documento si usano comandi particolari. Il più comune è `\cite`, che si usa come spiegato nel paragrafo 11.1 a pagina 217.

Il comando `\nocite`, invece, manda in bibliografia anche opere *non* citate nel documento (a patto, però, che figurino nel database):

- solo alcune, se dato nel corpo del documento al posto di `\cite` (come mostra il codice che produce la figura 21a a pagina 231);
- tutte, se dato prima o dopo `\printbibliography` (si veda più oltre) nella forma `\nocite{*}`.

Di seguito si mostrano gli stessi esempi d'uso di `\cite` con lo schema autore-anno compatto:

<code>\cite{eco:tesi} \\\</code>	Eco, 1977
<code>\cite[5]{eco:tesi} \\\</code>	Eco, 1977, p. 5
<code>\cite[5-9]{eco:tesi} \\\</code>	Eco, 1977, p. 5-9
<code>\cite[vedi][]{eco:tesi} \\\</code>	vedi Eco, 1977
<code>\cite[vedi][5]{eco:tesi}</code>	vedi Eco, 1977, p. 5

e con lo schema numerico compatto:

<code>\cite{eco:tesi} \\\</code>	[1]
<code>\cite[5]{eco:tesi} \\\</code>	[1, p. 5]
<code>\cite[5-9]{eco:tesi} \\\</code>	[1, pp. 5-9]
<code>\cite[vedi][]{eco:tesi} \\\</code>	[vedi 1]
<code>\cite[vedi][5]{eco:tesi}</code>	[vedi 1, p. 5]

(in modo del tutto analogo funzionano gli altri comandi: si facciano delle prove). Come si può osservare, `biblatex` produce automaticamente le abbreviazioni *p.* e *pp.* a seconda delle circostanze. (Si può ottenere anche la scrittura estesa *pagina* o *pagine*: si veda la documentazione del pacchetto.)

Oltre ai due comandi appena visti, `biblatex` ne definisce altri per citazioni ancora diverse:

- `\textcite` se la citazione è parte integrante del discorso: Eco [1977];
- `\parencite` racchiude la citazione fra parentesi: [Eco, 1977];
- `\footcite` mette la citazione in una nota, come qui¹;
- `\supercite` (solo per schemi numerici) mette la citazione in apice;
- `\fullcite` riporta nella citazione l'intero riferimento bibliografico nello stile impostato: Umberto Eco 1977 *Come si fa una tesi di laurea. Le materie umanistiche*, Bompiani, Milano.

¹ Eco, 1977.

Si noti che il tipo di parentesi dipende dall'opzione (square, in questa guida) assegnata a biblatex.

L'opzione natbib permette di mantenere una compatibilità quasi totale con i documenti la cui bibliografia è stata creata con il pacchetto natbib: la maggior parte dei nomi dei comandi per le citazioni da esso definiti, infatti, come `\citep` e `\citet`, sono conservati come alias.

Per citare singole parti di un riferimento ci sono `\citeauthor` e `\citeyear`:

<code>\citeauthor{eco:tesi}</code>	Eco
<code>\citeyear{eco:tesi}</code>	1977

11.2.6 Generare e collocare la bibliografia nel documento

Il comando `\printbibliography` produce la sezione bibliografica con relativi titolo e testatina. Per mandarne il relativo titolo nell'indice generale si usa l'istruzione:

<code>\printbibliography[heading=bibintoc]</code>

che funziona con le classi article, report e book.

Per generare effettivamente la bibliografia nel documento, infine, questa è la sequenza di composizione:

1. si compone il documento con \LaTeX una prima volta;
2. si lancia il programma Biber premendo l'apposito pulsante dell'editor;
3. si compone il documento (almeno) altre *due* volte con \LaTeX .

11.2.7 Specialità

In questo paragrafo si presentano alcuni argomenti avanzati relativi a biblatex.

Campi speciali

I campi elencati di seguito non contengono dati stampabili, ma servono per scopi diversi e si possono applicare a *tutti* i record.

hyphenation Imposta la lingua del riferimento. Il valore deve essere il nome di una lingua nota a babel.

sortkey Imposta l'ordine alfabetico dei riferimenti: serve come chiave di ordinamento nei record privi dell'indicazione di autore o curatore.

keywords Un elenco di parole chiave separate da virgole, da usare come *filtro* sulle voci bibliografiche da far andare in bibliografia.

Campi omissibili

Alcuni dei campi indicati come obbligatori nel paragrafo 11.2.3 a pagina 221 in realtà non lo sono sempre. Nella bibliografia del proprio lavoro ci possono essere un libro anonimo oppure la raccolta dei lavori di una conferenza senza un curatore. Di regola questo non è un problema nella composizione della bibliografia, ma può esserlo nelle citazioni: uno schema di

citazione autore-anno richiede *sempre* un campo `author/editor` e un campo `date`.

Se mancano i dati necessari per formare l’etichetta “regolare”, si può sostituire ogni dato mancante con il campo `label`, da usare nei modi propri di ogni schema di citazione. Negli schemi autore-anno sostituisce il campo `author/editor` oppure il campo `date` nelle opere che ne sono prive. Negli schemi numerici, invece, non viene usato, perché in questo caso il formato delle citazioni è indipendente dai dati della bibliografia. Lo stesso accade negli schemi autore-titolo, perché il solo titolo basta di solito per formare una citazione univoca (di regola, ogni fonte ha un titolo).

Riferimenti finali cliccabili e bibliografia multilingue

Le seguenti opzioni di `biblatex` permettono di personalizzare la bibliografia.

backref Indica accanto a ciascun riferimento le pagine del documento in cui è citato.

hyperref Rende cliccabili le citazioni e i riferimenti (richiede il pacchetto omonimo).

babel=hyphen Ordina a `LATEX` di verificare per quale o per quali riferimenti si è specificata una lingua diversa da quella principale e di sillabarla con le regole della nuova lingua.

babel=other Dice a `LATEX` di mettere il riferimento in un ambiente `otherlanguage` e di tradurre nella nuova lingua anche tutti i termini associati all’opera, come *curatore*, *volume*, eccetera.

L’esempio seguente mostra un record in cui si è specificata la lingua:

```
@book{lamport:latex,
  author      = {Lamport, Leslie},
  title       = {\LaTeX. A Document Preparation System},
  publisher    = {Addison-Wesley},
  date        = {1994},
  location     = {Reading (Massachusetts)},
  hyphenation = {english}
}
```

Suddividere la bibliografia

Talvolta è necessario suddividere la bibliografia in base a certi criteri. Per esempio, la si può ripartire in letteratura di base e di approfondimento, oppure elencare separatamente le risorse cartacee e quelle online, oppure ancora separare le opere citate esplicitamente nel documento da quelle che s’intendono semplicemente suggerire. Con `biblatex` tutto questo si può fare facilmente.

Si supponga di aver creato un database bibliografico `miabibliografia.bib` e si consideri il seguente codice:

```
\documentclass{article}
\usepackage[italian]{babel}
\usepackage[autostyle,italian=guillemets]{csquotes}
\usepackage[style=alphanumeric]{biblatex}
```

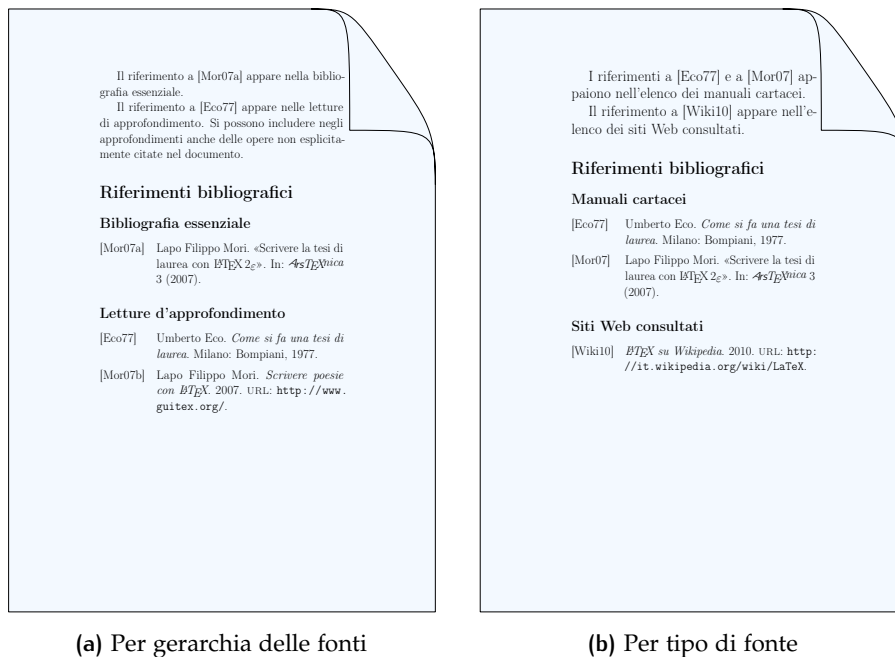


Figura 21: Bibliografie suddivise

```

\usepackage{guit}
\addbibresource{miabibliografia.bib}

\DeclareBibliographyCategory{basi}
\DeclareBibliographyCategory{approf}

\addtocategory{basi}{mori:tesi}
\addtocategory{approf}{eco:tesi,mori:poesie}

\defbibheading{basi}{\subsection*{Bibliografia essenziale}}
\defbibheading{approf}{\subsection*{Letture d'approfondimento}}

\begin{document}
Il riferimento a~\cite{mori:tesi} appare nella bibliografia essenziale.

Il riferimento a~\cite{eco:tesi} appare nelle letture di
approfondimento.
Si possono includere negli approfondimenti anche delle opere non
esplicitamente citate nel documento.
\nocite{mori:poesie}

\section*{\refname}
\printbibliography[heading=basi,category=basi]
\printbibliography[heading=approf,category=approf]
\end{document}

```

Si tratta di un articolo con due bibliografie separate. Si osservi ora quanto segue.

- Il comando `\DeclareBibliographyCategory` nel preambolo definisce due *categorie bibliografiche*: una per la letteratura di base, l'altra per gli approfondimenti. Il comando `\addtocategory` assegna le opere a ciascuna categoria.

- Il comando `\defbibheading` definisce il titolo delle due bibliografie (*Bibliografia essenziale* e *Letture d'approfondimento*), che vengono composte esattamente come due sottoparagrafi (è la prassi, in un articolo). (Scrivendo un book o un report con bibliografia suddivisa, nel codice precedente `\subsection*` e `\section*{\refname}` vanno sostituiti con `\section*` e `\chapter*{\bibname}` rispettivamente.)
- L'argomento facoltativo di `\printbibliography` specifica le istruzioni di controllo: `heading=<nome>` imposta il titolo della bibliografia come specificato con `\defbibheading`, `category=<categoria>` filtra le voci da mettere in bibliografia, selezionando soltanto quelle che appartengono alla `<categoria>` specificata.

Al posto dei due `\printbibliography` con i relativi argomenti facoltativi si può scrivere un semplice

```
\bibbycategory
```

che equivale a dare un `\printbibliography` per ogni categoria nell'ordine in cui le si è dichiarate. Il risultato è mostrato nella figura 21a nella pagina precedente.

Un altro esempio. Si supponga di partire ancora dallo stesso database bibliografico appena considerato e si osservi il codice seguente:

```
\documentclass{article}
\usepackage[italian]{babel}
\usepackage[autostyle,italian=guillemets]{csquotes}
\usepackage[style=alphabetic]{biblatex}
\usepackage{guit}
\addbibresource{Bibliografia.bib}

\defbibheading{cartaceo}{\subsection*{Manuali cartacei}}
\defbibheading{web}{\subsection*{Siti Web consultati}}

\begin{document}
I riferimenti a-\cite{eco:tesi} e a-\cite{mori:tesi} appaiono
nell'elenco dei manuali cartacei.

Il riferimento a-\cite{wiki:latex} appare nell'elenco dei siti Web
consultati.

\section*{\refname}
\printbibliography[heading=cartaceo,notttype=online]
\printbibliography[heading=web,type=online]
\end{document}
```

Si tratta di un articolo con due bibliografie separate, una per i manuali cartacei e l'altra per i siti Web consultati (una "sitografia"). Si osservi che:

- il comando `\defbibheading` definisce il titolo delle due bibliografie (*Manuali cartacei* e *Siti Web consultati*);
- l'argomento facoltativo di `\printbibliography` filtra le voci come prima, per cui `type=<tipo>` manda in bibliografia soltanto quelle del `<tipo>` specificato, mentre `notttype=<tipo>` omette quelle il cui tipo è `<tipo>`.

Il risultato è mostrato nella figura 21b nella pagina precedente.

Bibliografia di sezione

Talvolta può essere necessario realizzare una bibliografia separata per ogni capitolo di un libro o di una relazione, oppure per ogni paragrafo di un articolo. Per esempio, in una raccolta di articoli di diversi autori, come un volume dei lavori di una conferenza, è molto comune avere una bibliografia per ogni articolo invece di una bibliografia globale per l'intera raccolta.

Il pacchetto `biblatex` risolve questo problema, raccogliendo i riferimenti di ogni bibliografia *locale*, relativa cioè a ogni unità di sezionamento (capitolo o paragrafo), in file ausiliari che vengono elaborati da Biber.

Supponiamo, per esempio, di avere la solita base di dati di riferimenti bibliografici `Bibliografia.bib` definita come a pagina 220. Si consideri ora il seguente codice:

```
\documentclass{article}
\usepackage[italian]{babel}
\usepackage[babel]{csquotes}
\usepackage[style=alphanumeric,refsection=section]{biblatex}
\usepackage{guit}
\bibliography{Bibliografia}

\defbibheading{bibliography}%
  {\subsection*{Riferimenti bibliografici del paragrafo~\thesection}}

\begin{document}
\section{Un paragrafo}
Un paragrafo, con un riferimento a~\cite{bringhurst:elementi}, che
appare nella relativa bibliografia locale.

Si possono includere nella bibliografia locale anche delle opere
non esplicitamente citate nel documento.
\nocite{gregorio:breveguida}

\printbibliography

\section{Un altro paragrafo}
Un nuovo paragrafo, con un riferimento a~\cite{mori:tesi}, che
compare nella relativa bibliografia locale.

\printbibliography
\end{document}
```

Si tratta di un articolo diviso in due paragrafi, ciascuno dei quali ha la propria bibliografia.

- Il pacchetto `biblatex` è caricato con l'opzione `refsection=section`. Questa opzione comincia automaticamente una nuova sezione di riferimenti bibliografici (`refsection`) a ogni nuovo paragrafo (`section`). (L'opzione `refsection=chapter` inizia una nuova sezione di riferimenti a ogni capitolo.)
- Il comando `\defbibheading` definisce i titoli delle bibliografie locali. Il parametro `\thesection` è il contatore dei paragrafi, gestito automaticamente da L^AT_EX. (Analogamente, `\thechapter` è il contatore dei capitoli.)
- La posizione della bibliografia locale all'interno di ogni sezione si specifica con il comando `\printbibliography`.

1 Un paragrafo

Un paragrafo, con un riferimento a [Bri01], che appare nella relativa bibliografia locale.

Si possono includere nella bibliografia locale anche delle opere non esplicitamente citate nel documento.

Riferimenti bibliografici del paragrafo 1

- [Bri01] Robert BRINGHURST. *Gli Elementi dello Stile Tipografico*. Milano: Sylvestre Bonnard, 2001.
- [Gre08] Enrico GREGORIO. *LaTeX: breve guida ai pacchetti di uso più comune*. 2008. URL: <http://profs.sci.univr.it/~gregorio/>.

2 Un altro paragrafo

Un nuovo paragrafo, con un riferimento a [Mor07] che compare nella relativa bibliografia locale.

Riferimenti bibliografici del paragrafo 2

- [Mor07] Lapo Filippo MORI. “Scrivere la tesi di laurea con LaTeX 2_ε”. in: *ArsTeXnica* 3 (2007).

Figura 22: Bibliografie separate per sezione

Quando si compila con \LaTeX un documento come il precedente, per ogni unità di sezionamento viene generato un file ausiliario a sé stante (il cui nome è formato dal nome del documento principale seguito da un numero progressivo e dal suffisso `-blx.aux`); ciascuno di questi file deve essere elaborato da Biber. Nell'esempio in questione, se `prova.tex` è il nome del documento, si ottengono dunque due file ausiliari, chiamati `prova1-blx.aux` e `prova2-blx.aux`, da elaborare con Biber (si dà `biber prova1-blx.aux` e `biber prova2-blx.aux` dalla linea di comando dopo essersi portati nella stessa cartella dei file). Per generare le bibliografie è necessario compilare successivamente il documento principale con Biber; servono infine altre due compilazioni con \LaTeX .

Schematizzando, per generare le bibliografie locali con `biblatex` e includerle nel documento bisogna:

1. compilare il documento con \LaTeX ;
2. eseguire Biber su ciascuno dei file ausiliari;
3. eseguire Biber sul documento principale;
4. ricompilare due volte con \LaTeX per includere le bibliografie e aggiornare tutti i riferimenti.

Il risultato di queste operazioni applicate all'esempio considerato è visibile nella figura 22 nella pagina precedente.

11.2.8 L'indicizzazione automatica delle voci bibliografiche

Questo paragrafo presuppone una conoscenza di base degli strumenti per generare un indice analitico con \LaTeX , che verranno presentati nel capitolo successivo.

Un'interessante funzione di `biblatex` è l'indicizzazione automatica delle voci bibliografiche. L'indice analitico è utile al lettore di qualsiasi libro e di norma è presente in tutte le pubblicazioni scientifiche. L'opzione `indexing` di `biblatex` permette di inserire nell'indice analitico tutti gli autori citati nella bibliografia e nel testo. Il pacchetto `biblatex` si appoggia, per creare l'indice analitico, al pacchetto `makeidx` (per le funzioni di base) o al pacchetto `index` (per le funzioni avanzate, come gli indici multipli). Di seguito è riportato un semplice esempio di indicizzazione automatica dei nomi.

Supponendo di avere una base di dati `Bibliografia.bib` definita come a pagina 220, il seguente codice per il documento principale permette di provare le funzioni di indicizzazione automatica di `biblatex`.

```
\documentclass{article}
\usepackage[italian]{babel}
\usepackage[babel]{csquotes}
\usepackage[style=alphanumeric,indexing]{biblatex}
\usepackage{guil}
\usepackage{makeidx}
\makeindex
\bibliography{Bibliografia}

\begin{document}
Il riferimento a~\cite{gregorio:breveguida} appare nella
bibliografia e viene indicizzato automaticamente.
```

Il riferimento a [Gre08] appare nella bibliografia e viene indicizzato automaticamente.

Si possono includere nell'indice analitico anche gli autori delle opere non esplicitamente citate nel documento.

Riferimenti bibliografici

[Gre08] Enrico GREGORIO. *L^AT_EX: breve guida ai pacchetti di uso più comune*. 2008. URL: <http://profs.sci.univr.it/~gregorio/>.

[Mor07] Lapo Filippo MORI. "Scrivere la tesi di laurea con L^AT_EX 2_ε". in: *ArsT_EXnica* 3 (2007).

Indice analitico

Gregorio, Enrico, 1

Mori, Lapo Filippo, 1

Figura 23: Indicizzazione automatica delle voci bibliografiche

Si possono includere nell'indice analitico anche gli autori delle opere non esplicitamente citate nel documento. `\nocite{mori:tesi}`

```
\printbibliography
\printindex
\end{document}
```

A questo punto, si procede nel modo seguente:

1. si compila il documento con \LaTeX ;
2. si esegue Biber sul documento;
3. si ricompila un'altra volta con \LaTeX ;
4. si esegue MakeIndex sul documento;
5. si compila un'ultima volta con \LaTeX .

In questo modo si ottiene un documento con la normale bibliografia, seguita dall'indice analitico contenente tutti i nomi citati esplicitamente nel testo (con uno dei comandi presentati nel paragrafo 11.2.5 a pagina 228) e in bibliografia (con il comando `\nocite`). (Specificando `indexing=cite` è possibile inserire nell'indice analitico *solo* i nomi citati esplicitamente nel testo.)

Il risultato di tutte queste operazioni applicate all'esempio considerato è visibile nella figura 23.

12 | INDICE ANALITICO

L'indice analitico è un elenco alfabetico di parole o espressioni (anche con alcuni livelli di subordinazione) dette *voci*, posto di regola alla fine di un documento; accanto a ogni voce ci sono i numeri delle pagine in cui la voce in questione compare. In molti lavori questo indice è utilissimo, e \LaTeX è capace di gestirlo automaticamente e con grande efficienza, come si mostra in questo capitolo.

12.1 INDICI SEMPLICI

Per creare l'indice analitico con \LaTeX bisogna innanzitutto eseguire due operazioni preliminari:

1. caricare il pacchetto `imakeidx` per abilitare il programma alla composizione dell'indice;
2. dare *nel preambolo* il comando `\makeindex` (con eventuali opzioni, spiegate di seguito), per attivare i comandi dedicati che verranno inseriti nel corpo del testo.

A questo punto, *immediatamente dopo* ogni parola o espressione da indicizzare basterà dare il comando

```
\index{<voce>}
```

la cui sintassi è mostrata nella tabella 56 nella pagina successiva, nel cui argomento in teoria si può scrivere ciò che si vuole.

Per produrre l'indice analitico basta dare `\printindex` *immediatamente prima* di `\end{document}` e compilare (una volta) con \LaTeX .

Per esempio, il codice

```
\documentclass{article}
...
\usepackage{imakeidx}
\makeindex

\begin{document}
Si possono inserire nell'indice analitico singole parole
come \emph{arte}\index{arte}, oppure intere espressioni
come questa \index{interi espressioni come questa}.

\addcontentsline{toc}{section}{\indexname}
\printindex
\end{document}
```

crea un indice analitico con le voci *arte* e *interi espressioni come questa*.

Per produrre la sezione dell'indice analitico e mandarne il titolo nell'indice generale basta dare l'opzione `intoc` a `\makeindex`:

```
\makeindex[intoc]
```

Tabella 56: Sintassi del comando `\index`

Tipo di voce e codice	Risultato
Primaria <code>\index{Artisti}</code>	Artisti, 2
Sottovoce <code>\index{Artisti!Escher}</code>	Artisti, 2 Escher, 3
Voce in altro stile <code>\index{Gaudì@\textit{Gaudì}}</code>	<i>Gaudì</i> , 5
Pagina in altro stile <code>\index{Klimt textbf}</code>	Klimt, 7
Rimando <code>\index{Picasso see{Cubismo}}</code>	Picasso, <i>si veda</i> Cubismo
Intervallo di pagine <code>\index{Mirò ({}...\index{Mirò })}</code>	Mirò, 11–13

12.2 INDICI COMPLESSI

Con le semplici istruzioni presentate nel paragrafo precedente si può comporre un indice analitico più che dignitoso. Il pacchetto `imakeidx`, tuttavia, permette di gestire e personalizzare molto finemente l'indice analitico, e di produrre anche indici multipli. Di seguito si descrivono le principali funzioni di `imakeidx` e si rinvia alla sua documentazione per gli approfondimenti.

12.2.1 Comporre l'indice

Il comando

```
\makeindex[⟨opzioni⟩]
```

ha un argomento facoltativo costituito da un elenco di *⟨opzioni⟩*, separate da virgole. Un'opzione può essere un'unica parola o una voce *⟨chiave⟩=⟨valore⟩*. Le principali chiavi disponibili sono le seguenti.

name È il nome dell'indice; serve solo negli indici multipli.

Esempio: `name=persone`.

title È il titolo dell'indice. Il suo valore predefinito è `\indexname` (*Indice analitico*, in italiano).

Esempio: `title=Indice dei nomi`.

options È un elenco di opzioni di ordinamento e formato. Permette di usare uno stile personale, contenuto in un file *⟨stile⟩.ist* (si veda il paragrafo 12.2.6 a pagina 242).

Esempio: `options=stile.ist`.

intoc Manda il titolo dell'indice analitico nell'indice generale.

columns Indica il numero di colonne dell'indice.

Esempio: `columns=3`.

columnsep Indica la distanza tra le colonne dell'indice.

Esempio: `columnsep=15pt`.

columnseprule Stampa una riga tra le colonne dell'indice.

12.2.2 Indicizzare le voci

Il comando

```
\index[⟨nome⟩]{⟨voce⟩}
```

inserisce la *⟨voce⟩* nell'indice chiamato *⟨nome⟩*, se si usano indici multipli. Se non si specifica alcun *⟨nome⟩*, la *⟨voce⟩* va nell'indice predefinito.

12.2.3 Inserire una premessa nell'indice

Il comando

```
\indexprologue[⟨spazio⟩]{⟨testo⟩}
```

da dare subito prima di `\printindex`, si usa per scrivere del *⟨testo⟩* fra il titolo dell'indice e le voci. Lo *⟨spazio⟩* imposta la distanza verticale fra il *⟨testo⟩* e le voci.

12.2.4 Stampare l'indice

Il comando

```
\printindex[⟨nome⟩]
```

stampa l'indice chiamato *⟨nome⟩*. Se non si specifica alcuna opzione, stampa l'indice predefinito.

12.2.5 Un esempio

Un esempio chiarirà le idee. Il codice:

```
\documentclass{article}
...
\makeindex[title=Indice dei concetti]
\makeindex[name=persone,title=Indice dei nomi,columns=3]

\begin{document}
La teoria della relatività\index{relatività} è stata formulata da
Albert Einstein\index[persone]{Einstein, Albert}.

\printindex

\indexprologue{\small Questo indice contiene nomi di persone famose.}
\printindex[persone]
\end{document}
```

crea due indici analitici, il primo dei quali si intitola *Indice dei concetti* e contiene la voce *relatività*, mentre il secondo si intitola *Indice dei nomi*, contiene la voce *Einstein, Albert*, è disposto su tre colonne e ha una premessa.

12.2.6 Personalizzare l'indice analitico

L'indice analitico che si ottiene con le impostazioni predefinite non è molto elegante. Si può ricorrere a un apposito file, `stile.ist`, così definito:

```
headings_flag 1
heading_prefix "\\goodbreak\\textsc{"
heading_suffix "\\par\\nobreak\\vskip\\smallskipamount\\nobreak"
symhead_positive "Simboli"
symhead_negative "simboli"
numhead_positive "Numeri"
numhead_negative "numeri"
```

Una volta registrato e posizionato `stile.ist` nella cartella di lavoro, per produrre l'indice analitico personalizzato basta specificare

```
\makeindex[options=stile.ist]
```

e compilare con \LaTeX .

Parte VI

PERSONALIZZAZIONI E REVISIONE

13

PERSONALIZZAZIONI

Prima o poi, capita a tutti di aver bisogno di “cose” non contemplate da \LaTeX standard: di solito si tratta di comandi e ambienti ad hoc o personalizzazioni di quelli esistenti. In questo capitolo s’impara come farlo.

13.1 COMANDI E AMBIENTI PERSONALI

13.1.1 Nuovi comandi

S’immagini di scrivere un libro di botanica e di volere tutti i nomi scientifici di pianta in corsivo, come si fa di solito. Il modo più immediato per farlo è scrivere ogni nome nell’argomento di `\textit`. Successivamente, per scelte tipografiche imperscrutabili, l’editore chiede i nomi di pianta in nero. Che fare? Si potrebbero sostituire automaticamente tutti i `\textit` con altrettanti `\textbf`, ma ci sarebbe qualche problema, perché si potrebbe aver usato il corsivo anche in altre parti del libro, che invece debbono rimanere tali. Non rimane che sostituire un comando dopo l’altro perdendo un sacco di tempo.

\LaTeX risolve questi problemi molto più “filosoficamente” di quanto non facciano altri programmi. Anziché agire a mano, si può definire *una volta per tutte* un nuovo comando, che si chiamerà `\piana`, che produce il proprio argomento (un nome di pianta, in questo caso) nello stile deciso dall’utente (in corsivo, in questo caso). Se l’imposizione del nero avviene a documento completato, basterà modificare *una volta per tutte* la definizione del comando e non più tutti i `\piana` nel documento.

Il nuovo comando, dunque, non descrive come l’argomento debba essere *reso*, ma come lo si è *pensato*. Perciò in un documento scritto con \LaTeX i comandi per cambiare lo stile vanno usati molto di rado, in favore di comandi che rispecchino la logica di ciò che si sta scrivendo.

I comandi personali si definiscono *nel preambolo* con `\newcommand`:

```
\newcommand{<nome>}[<numero di argomenti>]{<definizione>}
```

Dove:

- `<nome>` è il nome che si dà al nuovo comando.
- `<numero di argomenti>` è il numero di argomenti obbligatori che gli si assegna, fino a un massimo di nove. Non specificando questo valore, il nuovo comando non avrà argomenti.
- `<definizione>` sono le istruzioni che specificano ciò che si vuole che il nuovo comando “faccia”. Gli argomenti eventualmente assegnati al comando s’indicano nella `<definizione>` con `#` seguito da un numero progressivo: `#1`, `#2`, eccetera.

Il comando appena esaminato serve a definire nuovi comandi senza e con argomenti. L’esempio seguente mostra la sintassi di un comando personale *senza argomenti*:

```
\newcommand{\arte}{\emph{L'arte di scrivere con \LaTeX}}
```

utile, per esempio, se in un documento si dovesse scrivere ripetutamente il titolo di questa guida (si noti che un comando di questo tipo “produce testo”, e va perciò terminato nei modi già visti nel paragrafo 3.4.1 a pagina 24):

```
Questo lavoro s'intitola \arte.
```

Questo lavoro s'intitola *L'arte di scrivere con L^AT_EX*.

Si faccia attenzione, però, a non esagerare con i comandi personali semplicemente per abbreviare il testo, come `\gb` per *Gran Bretagna*, per esempio: alla lunga rendono il codice illeggibile.

Di seguito si mostra la sintassi di un comando personale *con argomenti*:

```
\newcommand{\pianta}[1]{\textit{#1}}
```

Si è assegnato al nuovo comando un solo argomento obbligatorio ([1]). Quando si dà il comando, accade questo: il testo nell'argomento di `\pianta` viene “passato” a #1 e trattato secondo la *definizione*. In questo caso verrà reso in corsivo, come si vede nell'esempio seguente:

```
\pianta{Rosa canina}
```

Rosa canina

In modo analogo, anche se non è molto frequente, si possono definire nuovi comandi con più argomenti (specificando [2], [3], eccetera).

L^AT_EX impedisce di definire un comando già esistente, ma non di ridefinirlo: per farlo, si usa `\renewcommand`, che ha la stessa sintassi di `\newcommand`.

A chi volesse approfondire gli aspetti della programmazione avanzata di L^AT_EX si consiglia la lettura di [Gregorio, 2009].

13.1.2 Nuovi ambienti

Un ambiente personale si definisce con il comando `\newenvironment` *nel preambolo*:

```
\newenvironment{<nome>}[<numero di argomenti>]%  
  {<comandi di apertura>}{<comandi di chiusura>}
```

Dove:

- *<nome>* è il nome che si dà al nuovo ambiente.
- *<numero di argomenti>* è il numero di argomenti che gli si assegna. Non specificando questo valore, il nuovo ambiente non avrà argomenti.
- *<comandi di apertura>* sono le istruzioni, che usano gli argomenti eventualmente presenti, da eseguire all'inizio dell'ambiente.
- *<comandi di chiusura>* sono le istruzioni da eseguire alla chiusura dell'ambiente.

Si noti che si possono assegnare eventuali argomenti solo ai *<comandi di apertura>*. In pratica, definire un nuovo ambiente equivale a definire due nuovi comandi, uno di apertura e uno di chiusura, da usare come al solito.

L'esempio seguente mostra la sintassi di un ambiente personale *senza argomenti*. Scrivendo nel preambolo

```
\newenvironment{itaitemize}{\begin{itemize}\itshape}{\end{itemize}}
```

si potrà usare il nuovo ambiente `itaitemize` così:

<pre>\begin{itaitemize} \item Un elenco con voci\dots \item \dots automaticamente in corsivo. \end{itaitemize}</pre>	<ul style="list-style-type: none"> • <i>Un elenco con voci...</i> • <i>...automaticamente in corsivo.</i>
--	---

Per ridefinire ambienti già esistenti, si usa il comando `\renewenvironment` (analogo a `\renewcommand`), che ha la stessa sintassi di `\newenvironment`.

13.2 PERSONALIZZARE IL TESTO

13.2.1 Font

In tipografia la parola *font* indica un insieme eterogeneo di caratteri (detti anche *glifi*) accomunati da un certo stile grafico. Più font con caratteristiche comuni costituiscono una *famiglia di font*, che comprende senz'altro la variante regolare (detta anche *tondo*) e almeno il *corsivo*, il **MAIUSCOLETTO** e il **nero**. Tra le caratteristiche più importanti di un font ci sono la *larghezza* e la presenza o meno delle *grazie*.

I caratteri dei font a larghezza fissa (o *monospaced* o *typewriter*, usati nei codici d'esempio di questa guida) hanno tutti la stessa larghezza e sono adatti per scrivere in un editor di testi o per incolonnare dati numerici, caratteristica che non hanno i font a larghezza *variabile*, più versatili e indicati per riempire la riga in modo ottimale.

I caratteri di un font *con grazie* (o *serif*, usati di solito nel testo principale) presentano piccole rifiniture alle estremità che alla lunga li rendono più leggibili su carta, al contrario di quelli di un font senza grazie (o *sans serif*, in questa guida usati per i nomi dei pacchetti), più indicati per la lettura a schermo o per corpi molto piccoli. Le figure 24 e 25 nella pagina seguente mostrano le quattro categorie di font appena esaminate.

Di regola, \LaTeX sceglie la variante appropriata in base alla struttura logica del documento (capitoli, paragrafi, testatine, eccetera), ma può accadere di doverne modificare a mano stile e corpo.

Scegliere i font

A volte, per i motivi più vari, potrebbe essere necessario usare font diversi da quelli predefiniti: per una precisa richiesta dell'editore, per aumentare la leggibilità del documento, per "alleggerire" il proprio lavoro di qualche pagina caricando un font più compatto, eccetera.

\LaTeX permette di usare praticamente *ogni tipo* di font in circolazione ma, si noti bene, un font *non* va usato per differenziare parti di testo (si usino per questo i comandi descritti nel paragrafo 4.4 a pagina 60) né tanto meno per abbellirle. Analogamente, non si considerino i vari stili come una "tavolozza" da cui attingere a piacere: tondo, corsivo e maiuscoletto, da usare secondo i criteri spiegati nell'appendice A a pagina 337, dovrebbero bastare per la gran parte delle esigenze.

In linea generale, in un documento servono diverse famiglie di font:




Figura 24: Famiglie di font a larghezza variabile



Figura 25: Famiglie di font a larghezza fissa

- una famiglia con grazie per il testo principale;
- una famiglia senza grazie per scopi particolari;
- una famiglia a larghezza fissa per indirizzi Internet, codici e alcune parole di ambito informatico;
- *una serie di font per scrivere la matematica.*

Scegliere famiglie che stiano bene insieme richiede abilità ed esperienza: perciò, *specie se si è alle prime armi*, si consiglia di ricorrere senz'altro a classi e pacchetti — anche non standard, come `ClassicThesis` (si veda il capitolo 22), `ArsClassica` (si veda il capitolo 23 a pagina 335) o `suftesi` — che abbiano già risolto questo problema.

Si può vedere una panoramica (quasi) completa dei font latini liberamente disponibili per l'uso con \LaTeX già presenti in \TeX Live su  **FONTS**.

13.2.2 Lingue straniere

Se in un documento le parti in lingua straniera sono molte, anziché usare i comandi standard per le lingue descritti nel paragrafo 3.2 a pagina 22 è più conveniente ridefinirli come si spiega di seguito.

Si può ottenere una versione personale di `\foreignlanguage` con

```
\newcommand{\inglese}[1]{\foreignlanguage{english}{\em #1}}
```

da usare come mostra l'esempio seguente:

Come scrisse una volta Donald Knuth, «`\inglese{we have seen that computer programming is an art}`».

Come scrisse una volta Donald Knuth, «*we have seen that computer programming is an art*».

Con le opportune sostituzioni si possono definire comandi per tutte le lingue di cui si ha bisogno. Nell'esempio si è usata la dichiarazione `\em` per mettere in evidenza il testo straniero, secondo le consuetudini italiane. In modo analogo si può ridefinire l'ambiente `otherlanguage*` (o `otherlanguage`, se serve) da usare come al solito.

13.3 SPECIALITÀ

Il tutto è maggiore della somma
delle parti.

Aristotele
Metafisica

In questo paragrafo si descrivono soltanto alcune delle numerose possibilità offerte da \LaTeX per personalizzare il proprio documento. Per ulteriori modifiche, si rimanda al paragrafo 13.4 a pagina 253.

13.3.1 Epigrafi

Esigenze *molto* particolari potrebbero richiedere di scrivere un'epigrafe in testa a un capitolo o un paragrafo: per realizzarla facilmente c'è il pacchetto `epigraph`. Un'epigrafe come quella che si vede dopo il titolo di questa sezione si ottiene con il seguente codice:

```
\section{Specialità}
\epigraph{Il tutto è maggiore della somma delle parti}%
{Aristotele\ \emph{Metafisica}}
```

13.3.2 Scritture curiose

Con \LaTeX , il pacchetto `shapepar` permette di “sagomare” un testo in molte forme diverse: cuori (come qui), quadrati, diamanti, cerchi, stelle a cinque punte, rettangoli, esagoni, candele che bruciano, triangoli diversamente orientati (si può anche ottenere un testo nella forma del logo $T_E X$). Basta scrivere il testo da sagomare nell'argomento del comando corrispondente (`\heartpar` in questo caso, se ne veda la documentazione). Forse questa possibilità non verrà molto usata, ma è un chiaro esempio di come con pochissima fatica si possano ottenere risultati impensabili con altri programmi.

Avete notato il piccolo cuore proprio qui sotto?

♡

13.3.3 Capilettera

LE CONVENZIONI tipografiche permettono, a scopo decorativo, di sostituire la prima lettera di un capitolo con un *capolettera*, cioè una lettera di corpo maggiore delle altre (come qui). I capilettera hanno le forme più svariate: vanno dai caratteri nel font in uso a quelli elaboratissimi disegnati appositamente per pubblicazioni particolarmente ricercate.

Ne offre un vasto assortimento il pacchetto `lettrine` (si veda il capitolo 21 a pagina 311 per gli approfondimenti), che per funzionare al meglio richiede a propria volta di caricare *prima* di `fontenc` con l'opzione `T1` il pacchetto `type1ec` (per scalare “a piacimento” i font standard di \LaTeX disegnati in vari corpi; usando font disegnati in un solo corpo come per esempio quelli dei pacchetti `txfonts`, `pxfonts`, `mathpazo` e `fourier`, invece, `type1ec` non serve):

```
\usepackage{type1ec}
\usepackage[T1]{fontenc}
\usepackage{lettrine}
```

Il comando `\lettrine`, la cui sintassi dovrebbe essere chiara, produce il capolettera:

```
\lettrine{<capolettera>}{<eventuale testo in maiuscoletto>}
```

Di seguito si riporta un esempio di testo con capolettera.

```
\lettrine{L}{e convenzioni}
tipografiche permettono, a scopo
decorativo, di sostituire la
prima lettera di un capitolo con
un \emph{capolettera}, cioè una
lettera di corpo maggiore delle
altre (come qui).
```

LE CONVENZIONI tipografiche permettono, a scopo decorativo, di sostituire la prima lettera di un capitolo con un *capolettera*, cioè una lettera di corpo maggiore delle altre (come qui).

Una consuetudine molto seguita prevede di sfumare il passaggio dal capolettera al testo normale con qualche parola in maiuscoletto, come si è fatto in questo paragrafo: si può fare, ma *non* è una regola.

13.3.4 Documenti multicolonna

Scrivere su più colonne

L'opzione di classe `twocolumn` permette di comporre l'intero documento su due colonne, ma ha il difetto di non bilanciare il testo nell'ultima pagina.

Il pacchetto `multicol` (se ne veda la documentazione) risolve il problema. Scrivendo semplicemente

```
\begin{multicols}{<numero>}
<testo multicolonna>
\end{multicols}
```

dove:

- `<numero>` è il numero delle colonne (da 2 a 10) in cui s'intende dividere il testo;
- `<testo multicolonna>` si spiega da sé;

si ottiene un testo su più colonne, bilanciate automaticamente e sezionabili con i comandi consueti. Si noti che l'ambiente `multicols` può essere usato per un intero documento o solo per alcune sezioni di esso, e che non accetta note a margine né oggetti mobili che non siano a tutta pagina, ciò che lo rende adatto piuttosto a documenti di solo testo o a sezioni particolari come l'indice analitico.

Inserire oggetti mobili

L'inserimento di oggetti mobili in contesti multicolonna richiede un'attenzione particolare, pena risultati indesiderati. Le seguenti indicazioni dovrebbero risolvere i casi problematici.

Se il documento è composto su due colonne con `twocolumn`:

- si possono inserire oggetti nelle singole colonne mettendoli negli ambienti `table` e `figure` con una larghezza riferita a `\columnwidth` (che è la giustezza della colonna di composizione corrente);
- lì si possono inserire a piena pagina mettendoli negli ambienti `table*` e `figure*` con una larghezza riferita a `\textwidth`, ma si noti che nel documento finito compariranno *sempre* in testa alla pagina (le eventuali preferenze indicate, infatti, vengono ignorate) o in una pagina *p*;
- si noti che la contemporanea presenza di entrambe le forme degli ambienti altera l'ordine di apparizione degli oggetti nel documento finito: ri-

solve il problema il pacchetto `dblfloatfix`, che permette di indicare le preferenze `[tbp]`.

L'ambiente `multicols`, invece, non è indicato per contenere oggetti mobili. Infatti:

- `table` e `figure` non funzionano e al loro posto vanno usa-

te le forme asteriscate, che si comportano come sopra;

- eventuali oggetti nelle singole colonne si possono solo collocare a mano e “proprio lì” con la preferenza `H` del pacchetto `float` (si veda il paragrafo 8.2.2 a pagina 122).

13.3.5 Colori

Il pacchetto `xcolor` permette di usare i colori in un documento scritto con \LaTeX , anche se in linea generale si sconsiglia di farlo, a meno che non sia davvero indispensabile. Con `xcolor`:

- si possono usare i colori il loro nome, scelto in tavolozze predefinite corrispondenti ad altrettante opzioni;
- si può definire un *qualunque* colore, al quale assegnare un nome arbitrario, secondo svariati modelli;
- si possono definire i colori direttamente nell'argomento dei comandi dedicati via via che servono.

Il funzionamento del pacchetto è piuttosto complicato: se ne veda la ricca documentazione per gli approfondimenti. Di seguito si mostreranno soltanto i comandi più usati e assumendo di usare i colori con il loro nome (quello predefinito oppure quello scelto per un colore personale).

Per **colorare del testo** si usa il comando

```
\textcolor{<colore>}{<testo>}
```

la cui sintassi si spiega da sé (all'interno di un gruppo si può usare la corrispondente dichiarazione `\color`).

Per racchiudere del testo **in un riquadro colorato** nel quale si può determinare la distanza dell'area colorata dal testo, si usa il comando

```
\colorbox{<colore>}{<testo>}
```

tenendo presente che il riquadro modificherà l'interlinea del capoverso e non verrà spezzato su più righe.

Il comando

```
\fcolorbox{<colore_1>}{<colore_2>}{<testo>}
```

racchiude del `<testo>` in un riquadro colorato di `<colore_1>`, del quale si può determinare l'estensione, a propria volta **incorniciato** in una cornice di `<colore_2>`, della quale si può determinare lo spessore. Valgono anche in questo caso le osservazioni fatte per il comando precedente.

Infine, in rari casi potrebbe essere necessario modificare il colore di sfondo di un'intera pagina (come si è fatto nell'ultima pagina di questo capitolo). Per farlo si usa la *dichiarazione*

```
\pagecolor{<colore>}
```

Tabella 57: Parole fisse italiane di babel

Comando	Voce	Comando	Voce
<code>\abstractname</code>	Sommario	<code>\indexname</code>	Indice analitico
<code>\alsoname</code>	vedi anche	<code>\listfigurename</code>	Elenco delle figure
<code>\appendixname</code>	Appendice	<code>\listtablename</code>	Elenco delle tabelle
<code>\bibname</code>	Bibliografia	<code>\pagename</code>	Pag.
<code>\ccname</code>	e p. c.	<code>\partname</code>	Parte
<code>\chaptername</code>	Capitolo	<code>\prefacename</code>	Prefazione
<code>\contentsname</code>	Indice	<code>\proofname</code>	Dimostrazione
<code>\enclname</code>	Allegati	<code>\refname</code>	Riferimenti bibliografici
<code>\figurename</code>	Figura	<code>\seename</code>	vedi
<code>\glossaryname</code>	Glossario	<code>\tablename</code>	Tabella
<code>\headtoname</code>	Per		

Si noti che si tratta di una dichiarazione *globale*, e dunque metterla in un gruppo non ne limita l'effetto. Risolvono il problema il pacchetto `afterpage` e i seguenti comandi, dati in un punto (meglio tra capoversi "normali") che cadrà nella pagina di cui si vuole cambiare lo sfondo:

```
\pagecolor{<nuovo colore di sfondo>}\afterpage{\pagecolor{white}}
```

In questo modo, dalla pagina successiva lo sfondo ritornerà bianco.

13.3.6 Filigrane

Un semplice modo per proteggere il proprio lavoro ed evitarne usi impropri è quello di contrassegnarne le pagine con una filigrana (come si è fatto in questa pagina). Il pacchetto `draftwatermark`, che agisce anche solo caricandolo, (e il più potente `xwatermark`) permette di farlo facilmente.

13.4 ALTRE PERSONALIZZAZIONI

13.4.1 Modificare le parole fisse

Per modificare le parole fisse generate da babel (sostituire *Capitolo* con *Unità* oppure *Bibliografia* con *Opere consultate*, per esempio), si scrive nel *preambolo* il codice

```
\addto\captions<lingua>\{<testo>\}
```

che comanda a \LaTeX di aggiungere alle definizioni specifiche della *<lingua>* il *<testo>*.

Volendo sostituire la voce *Capitolo* con *Unità* basta scrivere

```
\addto\captionsitalian{\renewcommand{\chaptername}{Unità}}
```

mentre per sostituire *Bibliografia* con *Opere consultate* si può scrivere

```
\addto\captionsitalian{\renewcommand{\bibname}{Opere consultate}}
```

La tabella 57 elenca i comandi di queste voci con la relativa traduzione italiana.

13.4.2 Aggiungere spazio tra le voci dell'indice generale

Se la classe in uso non lo fa già automaticamente, si può rendere la consultazione dell'indice generale più agevole aggiungendo dello spazio tra le voci: di solito lo si mette prima dei titoli di capitolo (come si è fatto in questa guida). Basta caricare il pacchetto `etoolbox` e dare *nel preambolo*:

```
\preto\chapter{\addtocontents{toc}{\protect\addvspace{\langlelunghezza\rangle}}}
```

dove $\langle lunghezza \rangle$ va espressa in una qualsiasi delle unità di misura tipografiche accettate da \LaTeX .

13.4.3 Personalizzazioni avanzate

Di seguito si descrivono brevemente alcuni tra i pacchetti più usati per personalizzare in modo avanzato certi aspetti del documento. Si raccomanda di usarli, se proprio necessario, quando il proprio lavoro è in forma *definitiva* e dopo averne letto attentamente la documentazione o la descrizione contenuta in [Gregorio, 2010].

`titlesec`, `titletoc` Permettono di personalizzare ogni aspetto dei titoli di sezione, dell'indice generale, delle testatine e dei piedi.

`sectsty` Permette di cambiare il font dei titoli di sezione e la loro posizione sulla pagina.

`tocloft` Permette di cambiare la resa tipografica degli indici (generale, delle tabelle e delle figure) e di definirne di nuovi.

`toctibind` Permette di inserire o eliminare i titoli delle sezioni dall'indice generale e di modificarlo in altri aspetti.

`fancyhdr` Permette di personalizzare ogni aspetto di testatine e piedi.

`enumitem` Permette di personalizzare ogni aspetto dei tre ambienti standard per gli elenchi `itemize`, `enumerate` e `description`.

Come si è detto più volte, anche se \LaTeX induce a privilegiare la struttura logica del documento e a trascurarne l'aspetto finale, nemmeno lui, tuttavia, riesce a risolvere in una volta sola *tutti* i problemi tipografici evitando d'intervenire in prima persona: è il piccolo prezzo da pagare per un prodotto di altissima qualità. La revisione finale del documento è una fase delicata: si tratta, a volte, di risolvere grandi problemi d'impaginazione con piccoli (o piccolissimi) aggiustamenti. Un'arte complicata ma ricca di soddisfazioni, che dà i propri frutti migliori solo quando il documento è nella sua forma *definitiva*.

Prima di mettere in pratica i suggerimenti proposti nei prossimi paragrafi si tenga presente che *quasi* sempre si può correggere un difetto d'impaginazione semplicemente riformulando il testo nel punto o nei dintorni del punto critico. Si risolva un problema alla volta, un capoverso alla volta: *probabilmente* anche qualche altro problema più in là nel testo andrà a posto da solo.

14.1 PROBLEMI ORIZZONTALI

I difetti d'impaginazione orizzontali riguardano la formazione dei capoversi: di solito consistono in righe sporgenti nel margine destro.

Titoli problematici

Se troppo lunghi, i titoli creano problemi almeno in due luoghi:

- nel corpo del documento, perché \LaTeX potrebbe spezzarli in modo non soddisfacente o addirittura non spezzarli del tutto se non trova un punto di sillabazione adatto;
- nell'indice generale e nelle eventuali testatine, riproducendo il problema appena visto.

Un titolo di sezione non dovrebbe *mai* andare a capo. Si risolvono tutti i problemi in una volta sola riformulandolo: si può fare praticamente sempre. Se invece il titolo lungo fosse davvero necessario, lo si mandi a capo con un `\` esplicito nell'argomento del comando di sezionamento e se ne usi l'argomento facoltativo per indice generale e testatine:

```
\chapter[Titolo breve per indice e testatine]%  
  {Titolo lungo\ da mandare a capo}
```

Capoversi problematici

Un capoverso che \LaTeX non riesce a comporre bene può influenzare la divisione in pagine del documento, per cui sistemarlo *potrebbe* risolvere au-

tomaticamente anche alcuni dei problemi descritti nel prossimo paragrafo. Di seguito si danno alcuni consigli per le situazioni più frequenti.

Un documento di una certa lunghezza conterrà quasi certamente qualche riga che L^AT_EX non è riuscito a comporre “bene” e che si vedrà sporgere (ma non sempre, si noti) nel margine destro. Per essere sicuri di “curarle” tutte, si possono evidenziare dando subito prima dell’inizio del documento il comando

```
\overfullrule=<lunghezza>
```

che accanto a ciascuna riga che sporge dal margine stampa un ■ (più o meno grande a seconda del valore assegnato a *<lunghezza>*, esprimibile in una qualunque delle unità di misura tipografiche accettate da L^AT_EX).

Possono causare lo stesso problema anche le parole che L^AT_EX per vari motivi non riesce a spezzare e che dopo la composizione si trovano a fine riga: si provi a suggerirne la sillabazione con degli \- espliciti là dove serve.

Si possono risolvere altre situazioni problematiche eliminando qualche spazio indivisibile non necessario (*mai*, però, tra l’iniziale puntata di un nome e il relativo cognome) o mettendo in display alcune formule matematiche poco leggibili in linea. Infine, si usino il meno possibile lunghi URL nel corpo del testo: li si metta in una nota al piede (se sono pochi ed è proprio necessario) oppure, se molti, in un elenco alla fine del documento.

Gli ultimi tocchi al documento sono puramente estetici. Si eviti che due righe consecutive comincino o finiscano con la stessa o le stesse parole riformulando il testo; si eviti che un capoverso termini con una sola parola molto breve (o, peggio, con una sola sillaba) racchiudendola in \mbox e unendola alla penultima con uno spazio indivisibile. Lo spazio sottile \, è preziosissimo per risolvere altri inestetismi: può servire per separare un apostrofo e le successive virgolette inglesi; per evitare fastidiose sovrapposizioni in certe sequenze: tra una lettera corsiva con tratti discendenti come la *f* e una parentesi tonda immediatamente prima, per esempio.

14.2 PROBLEMI VERTICALI

I difetti d’impaginazione verticali riguardano la divisione in pagine del documento.

Oggetti in testo e fuori testo

Gli oggetti in testo causano problemi d’impaginazione a volte irrisolvibili perché, di regola, non possono essere spezzati *in nessun modo* tra due pagine. Basta evitarli il più possibile e non si avranno di questi problemi.

Gli oggetti fuori testo, invece, sono molto più flessibili. Tenendo presente che si può considerare “ottimale” il risultato fintanto che oggetto e relativo riferimento si trovano sulla stessa pagina o al massimo in due pagine opposte (a libro aperto non dovrebbe essere necessario voltare pagina, per intenderci), anche se purtroppo non sempre è possibile, qualche oggetto potrebbe comunque non piacere dove L^AT_EX ha pensato di metterlo:

- potrebbe essere finito un po’ troppo lontano dal punto “ottimale”;

- ci si potrebbe trovare con due oggetti in una pagina e nessuno in quella successiva, quando li si preferirebbe distribuiti più omogeneamente;
- si potrebbe volere in una pagina di soli oggetti un oggetto che invece L^AT_EX ha messo in una pagina con del testo.

Come fare? I primi due casi si risolvono semplicemente arretrando il codice dell'oggetto di qualche capoverso, mentre il quarto dandogli la preferenza *p*, che L^AT_EX riesce a soddisfare *sempre* anche se da sola. Si possono sbrogliare tutte le altre situazioni intricate usando oculatamente le preferenze di collocazione e facendo qualche prova, considerando che molto testo e pochi oggetti garantiscono risultati migliori della situazione contraria. A volte può risolvere la situazione `\clearpage`, spiegato nel paragrafo 14.2 a pagina 261.

Note a piè di pagina

Le note al piede sono problematiche specie se lunghe più di un capoverso, perché sono ottime candidate a finire tra due pagine moltiplicando i difetti. Le si eviti il più possibile, dunque.

Orfani e vedove nel corpo del documento

In tipografia si usa chiamare *orfano* la prima e unica riga di un capoverso in fondo alla pagina e *vedova* l'ultima riga di un capoverso in cima a una pagina nuova. Che siano gli orrori tipografici da evitare con la cura più certolina lo conferma la terminologia tedesca: una riga orfana si chiama *Schusterjunge* ("apprendista ciabattino"), mentre una riga vedova *Hurenkind* ("figlio di p***"). L^AT_EX è programmato per evitare automaticamente queste due situazioni: quando non ce la fa, si ricorra ai consigli descritti di seguito.

Il metodo più semplice per risolvere la faccenda è riformulare il capoverso in questione per diminuirlo o aumentarlo di una riga (funziona in entrambi i casi) oppure valutare la possibilità di spezzare un capoverso in due o di riunirne due in uno nei pressi del problema.

Se il capoverso che contiene l'orfano o la vedova è abbastanza lungo, si può provare a dare *immediatamente prima o dopo* il testo del capoverso (o anche in mezzo) uno dei due comandi

```
\looseness=1
\looseness=-1
```

con cui si chiede a L^AT_EX di provare ad allungarlo o accorciarlo del numero di righe indicato (che è 0 per impostazione predefinita). Un tentativo con -1 potrebbe funzionare se l'ultima riga è *molto* breve, uno con 1 se è quasi piena. Si ricordi che il comando agisce solo sul capoverso cui è applicato.

Si possono eliminare orfani e vedove anche spostando una riga di testo dalla pagina *precedente* a quella in cui si trovano con `\pagebreak`. Il comando va dato prima della riga orfana nella versione composta e funziona tanto meglio quanto più la pagina contiene spazi flessibili. Lo si usi senz'altro quando un display sta in cima alla pagina e il capoverso precedente è lungo una o due righe soltanto.

Se nessuno degli strumenti visti fin qui funziona, si può provare con uno tra

```
\enlargethispage{1\baselineskip}
\enlargethispage{-1\baselineskip}
```

che dicono a L^AT_EX di allungare o accorciare *la pagina* del numero di righe indicato nell'argomento: è bene che l'allungamento (o l'accorciamento) ammonti a una riga soltanto o, *in casi particolarissimi*, a due, ma non di più. Il comando *modifica la gabbia del testo* e va dato *tra due capoversi*: perciò *va usato in coppia su due pagine opposte*, il cui piede non sia occupato da note o numeri di pagina e con margini inferiori sufficientemente alti. Se per la stampa si userà una carta di qualità, infine, l'aggiustamento non si noterà nemmeno. La variante asterisco del comando si comporta allo stesso modo, provando prima a comprimere o ad allargare gli eventuali spazi bianchi sulla pagina a seconda dei casi.

Si ricordi, infine, che salvo situazioni fortunate i capoversi brevi non sono adatti per simili acrobazie.

Orfani e vedove nell'indice generale

Il problema appena descritto può affliggere anche le voci dell'indice generale del documento. Per riportare all'ovile una voce vedova si può dare

```
\addtocontents{toc}{\protect\enlargethispage*{\baselineskip}}
```

nel corpo del documento *immediatamente prima* di un titolo di sezione che nella versione composta del documento cade nella pagina dell'indice generale precedente la vedova. Si facciano delle prove aumentando il valore di `\baselineskip` finché la voce in questione non rientra.

Per dare un po' di compagnia a una voce orfana, invece, si darà

```
\addtocontents{toc}{\protect\pagebreak}
```

nel corpo del documento *immediatamente prima* del titolo di sezione incriminato, tenendo presente che la pagina dell'indice verrà stiracchiata con tutte le conseguenze del caso.

Pagine di solo testo

I capoversi più problematici per il buon riempimento della pagina sono quelli di solo testo, specie se lunghi, numerosi e contigui. Infatti, gli eventuali spazi aggiunti dal programma visti nel paragrafo 3.6.2 a pagina 35 cadrebbero inevitabilmente tra un capoverso e l'altro, risultando immediatamente (e fastidiosamente) visibili. Si risolve il problema in uno dei modi seguenti:

- riformulando il testo in uno o più capoversi;
- inserendo nella pagina un oggetto mobile o uno o più display;
- facendo in modo che nella pagina ci sia almeno un titolo di sezione.

Gli ultimi due rimedi, in particolare, permettono al programma di ripartire lo spazio in più in modo ottimale.

Interrompere la pagina corrente

Di seguito si descrivono brevemente i principali comandi per cambiare pagina. Tutti interrompono la pagina corrente *nel punto in cui vengono dati*, ma:

- `\newpage` comincia semplicemente una pagina nuova;
- `\pagebreak` prima di cominciare la nuova pagina stiracchia in verticale il contenuto di quella in cui viene dato per riempirla al meglio;
- `\clearpage` prima di cominciare la nuova pagina dice a L^AT_EX di stampare tutti gli oggetti già definiti e che non hanno ancora trovato posto sulle pagine, se ce ne sono (si veda il paragrafo 8.2.2 a pagina 119);
- `\cleardoublepage` si comporta come il comando precedente ma inserendo, se necessario, una pagina bianca prima di cominciare quella nuova (utile nei documenti impostati per la stampa in fronte/retro).

Si tenga presente che L^AT_EX interrompe automaticamente la pagina nel punto ritenuto migliore: i comandi descritti ne modificano il comportamento predefinito e possono risolvere situazioni problematiche, ma vanno usati con accortezza.

14.3 ALTRE INDICAZIONI

Controllare l'indice analitico

Per controllare l'indice analitico può essere utile il pacchetto `showidx`, che ne visualizza le voci nel margine sinistro delle pagine del documento.

E per finire...

Infine, non rimane che rileggere il tutto *più volte* per stanare i refusi: è praticamente impossibile non scovarne in ogni documento che superi la decina di pagine.

Parte VII

DOCUMENTI PARTICOLARI

15 | VIDEOPRESENTAZIONI

15.1 INTRODUZIONE

Questo capitolo descrive nei suoi elementi essenziali beamer, una classe di documento molto potente e altamente versatile dedicata alle videopresentazioni. Fra i suoi punti di forza ci sono la gestione semplice degli effetti dinamici tra le diapositive, la possibilità di sezionare il testo con i comandi standard, la piena compatibilità con gli altri pacchetti e un ventaglio di personalizzazioni virtualmente illimitato. Si rimanda chi voglia approfondire l'argomento alla corposa e dettagliata documentazione della classe.

Si noti che beamer carica automaticamente i pacchetti `hyperref`, `graphicx`, `amsmath`, `amssymb`, `amsthm`.

Qualche consiglio

Una presentazione efficace è il risultato dell'esposizione chiara di contenuti essenziali, possibilmente riprodotti con diapositive poco elaborate: beamer aiuta, ma da solo non basta. Si tenga presente di:

- scrivere e parlare il più semplicemente possibile;
- ridurre il contenuto della diapositiva al minimo indispensabile (il pubblico deve ascoltare chi parla e non leggere);
- proiettare una diapositiva al minuto, non di più;
- suddividere la presentazione in frazioni di circa 5-7 minuti l'una;
- usare un font senza grazie e di corpo molto grande (leggibile senza fatica anche nelle ultime file);
- evitare capoversi giustificati (difficili da ottenere e inutili);
- usare lo stesso formato per tutte le diapositive.

15.2 FONDAMENTALI

In linea generale, una presentazione realizzata con beamer consiste in una serie di *quadri* (*frame*, in inglese), ciascuno dei quali è composto da una o più *diapositive* (*slide*).

15.2.1 Per cominciare

L'esempio

```
\documentclass{beamer}

% dati generali
```

```

\title{La nostra prima presentazione}
\author{Lorenzo Pantieri \and Tommaso Gordini}
\date{1 maggio 2011}

\begin{document}

% quadro 1
\begin{frame}
\maketitle
\end{frame}

% quadro 2
\begin{frame}
\frametitle{Un esempio}
\begin{itemize}
\item<1-> Mane
\item<2-> Tekel
\item<3-> Fares
\end{itemize}
\end{frame}

\end{document}

```

riproduce il codice della presentazione di quattro diapositive in due quadri mostrata nella figura 26 a fronte: il primo quadro ne contiene il titolo generale; il secondo un elenco di tre elementi. Si osservi che le cose ancora sconosciute sono poche:

- l'ambiente `frame` racchiude un quadro;
- il comando `\frametitle` produce il titolo del quadro: *ogni* quadro dovrebbe averne uno;
- se le diapositive di un quadro sono più d'una, si mettono in un ambiente `itemize`, dove ogni `\item` produce una diapositiva;
- le espressioni racchiuse tra i segni di `<` e `>` verranno spiegate nel paragrafo 15.2.3 a pagina 271.

Anche in questo caso, infine, un uso oculato dei commenti facilita la lettura del sorgente.

15.2.2 Una presentazione articolata

Il professor Euclide dell'Università di Alessandria d'Egitto è pronto per rivelare al mondo la sua ultima scoperta: i numeri primi sono infiniti. Euclide parlerà durante il VII Simposio Internazionale sui Numeri Primi dove avrà a disposizione venti minuti, cinque dei quali riservati alle domande del pubblico. La presentazione che ha preparato è la seguente:

```

\documentclass{beamer}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}

% dati generali
\title{I numeri primi sono infiniti}

```

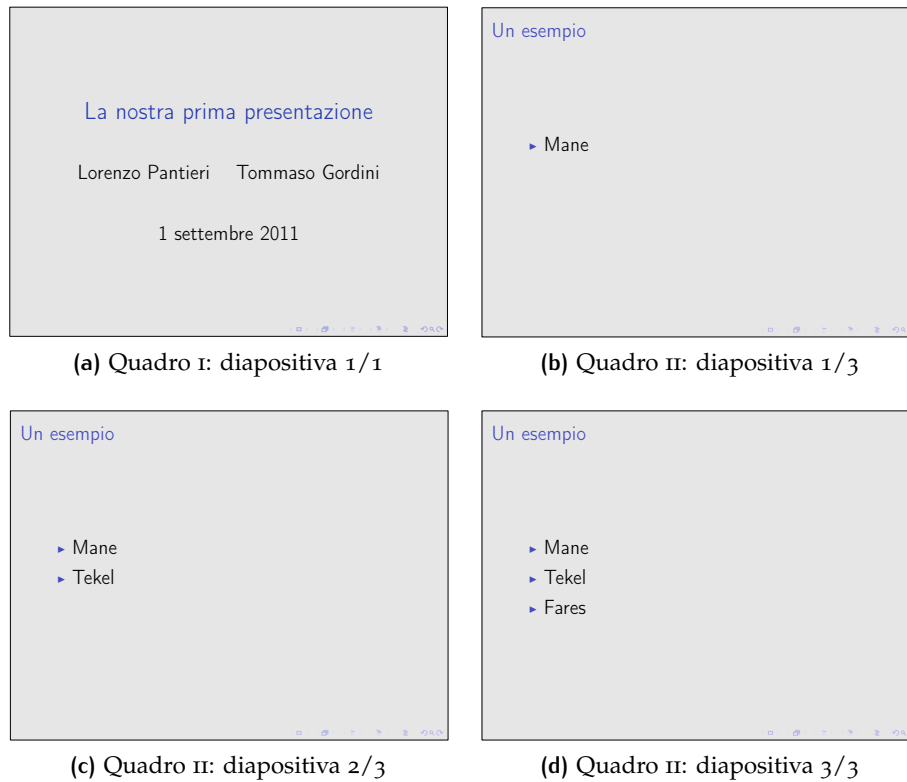



Figura 26: Presentazione semplice

```

\author[Euclide]{Euclide di Alessandria \\\
\texttt{euclide@alessandria.edu}}
\date[VII SINP]{VII Simposio Internazionale sui Numeri Primi}
\institute[UniAlessandria]{Università di Alessandria}
\logo{\includegraphics[width=0.2\textwidth]{sigillo}}

% temi e aspetto del testo
\usetheme{AnnArbor}
\useoutertheme[right]{sidebar}
\setbeamercovered{dynamic}

% definizione degli enunciati matematici
\theoremstyle{definition}
\newtheorem{definizione}{Definizione}
\theoremstyle{plain}
\newtheorem{teorema}{Teorema}

\begin{document}

% quadro 1
\begin{frame}
\maketitle
\end{frame}

% quadro 2
\begin{frame}
\frametitle{Piano della presentazione}
\tableofcontents
\end{frame}

```

```

% quadro 3
\section{Introduzione}
\begin{frame}
\frametitle{Che cosa sono i numeri primi?}
\begin{definizione}
Un \alert{numero primo} è un intero  $>1$  che ha esattamente
due divisori positivi.
\end{definizione}
\end{frame}

% quadro 4
\section{L'infinità dei primi}
\begin{frame}
\frametitle{I numeri primi sono infiniti}
\framesubtitle{Ne diamo una dimostrazione diretta}
\begin{teorema}
Non esiste un primo maggiore di tutti gli altri.
\end{teorema}
\pause
\begin{proof}
\begin{enumerate}[<+>]
\item Sia dato un elenco di primi.
\item Sia  $q$  il loro prodotto.
\item Allora  $q+1$  è divisibile per un primo  $p$ 
che non compare nell'elenco. \qedhere
\end{enumerate}
\end{proof}
\end{frame}

% quadro 5
\section{Problemi aperti}
\begin{frame}
\frametitle{Che cosa c'è ancora da fare?}
\begin{block}{Problemi risolti}
Quanti sono i numeri primi?
\end{block}
\begin{block}{Problemi aperti}
Un numero pari  $>2$  è sempre la somma di due primi?
\end{block}
\end{frame}

\end{document}

```

è una presentazione di otto diapositive in cinque quadri (mostrata nella figura 27 a fronte) che di seguito si analizza negli elementi nuovi.

Preambolo

Contiene come al solito le istruzioni generali del documento. Si noti che:

- gli argomenti facoltativi di `\author` e `\date` mettono il loro contenuto in punti particolari della diapositiva (di solito in basso).
- `\institute` (il cui argomento facoltativo si comporta come descritto sopra) inserisce nella diapositiva il nome dell'istituzione di appartenenza e `\logo` ne mette il logo. (Se lo si desidera, si può anche inserire una

I numeri primi sono infiniti

Euclide di Alessandria
euclide@alessandria.edu
Università di Alessandria

VII Simposio Internazionale sui Numeri Primi

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 1 / 5

(a) Quadro I: diapositiva 1/1

I numeri primi sono infiniti

Ne diamo una dimostrazione diretta

Teorema
Non esiste un primo maggiore di tutti gli altri.

Dimostrazione.

1. Sia dato un elenco di primi.
2. Sia q il loro prodotto.
3. Allora $q + 1$ è divisibile per un primo p che non compare nell'elenco.

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 4 / 5

(e) Quadro IV: diapositiva 2/4

Piano della presentazione

1. Introduzione
2. L'infinità dei primi
3. Problemi aperti

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 2 / 5

(b) Quadro II: diapositiva 1/1

I numeri primi sono infiniti

Ne diamo una dimostrazione diretta

Teorema
Non esiste un primo maggiore di tutti gli altri.

Dimostrazione.

1. Sia dato un elenco di primi.
2. Sia q il loro prodotto.
3. Allora $q + 1$ è divisibile per un primo p che non compare nell'elenco.

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 4 / 5

(f) Quadro IV: diapositiva 3/4

Che cosa sono i numeri primi?

Definizione
Un numero primo è un intero > 1 che ha esattamente due divisori positivi.

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 3 / 5

(c) Quadro III: diapositiva 1/1

I numeri primi sono infiniti

Ne diamo una dimostrazione diretta

Teorema
Non esiste un primo maggiore di tutti gli altri.

Dimostrazione.

1. Sia dato un elenco di primi.
2. Sia q il loro prodotto.
3. Allora $q + 1$ è divisibile per un primo p che non compare nell'elenco.

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 4 / 5

(g) Quadro IV: diapositiva 4/4

I numeri primi sono infiniti

Ne diamo una dimostrazione diretta

Teorema
Non esiste un primo maggiore di tutti gli altri.

Dimostrazione.

1. Sia dato un elenco di primi.
2. Sia q il loro prodotto.

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 4 / 5

(d) Quadro IV: diapositiva 1/4

Che cosa c'è ancora da fare?

Problemi risolti
Quanti sono i numeri primi?

Problemi aperti
Un numero pari > 2 è sempre la somma di due primi?

Euclide (Un'Alessandria) I numeri primi sono infiniti VII SINP 5 / 5

(h) Quadro V: diapositiva 1/1

Figura 27: Presentazione complessa

figura nella diapositiva iniziale scrivendo nel preambolo l'istruzione `\titlegraphic{\includegraphics{\langle figura \rangle}}`.)

- Con `\usetheme` si sceglie il *tema* della presentazione, cioè l'aspetto generale delle diapositive per colori e disposizione degli elementi, e con `\useoutertheme` si scelgono ulteriori elementi di contorno (si veda il paragrafo 15.3 a pagina 275).
- `\setbeamercovered` regola l'aspetto del testo nella proiezione. Accetta tre valori: con `transparent` il testo non ancora proiettato è (semi)trasparente (quel tanto che basta a suggerire a chi parla come proseguire l'esposizione); con `invisible` (predefinito) è completamente nascosto; con `dynamic` è tanto più trasparente quanto più tempo deve rimanere nascosto, e viceversa.

Primo quadro

Composto di una sola diapositiva (figura 27a), contiene il titolo della presentazione. In basso a destra si vede il contatore dei quadri in forma di frazione (1/5): quello proiettato, dunque, è il primo di cinque quadri in totale.

Secondo quadro

Composto di una sola diapositiva (figura 27b), contiene l'indice generale dell'esposizione, utile all'inizio per illustrare il piano della proiezione. L'indice è riportato esattamente sulla *barra di navigazione* laterale, le cui voci (cliccabili per portarsi rapidamente da un punto all'altro della presentazione) s'illuminano progressivamente all'esaurirsi dei quadri.

Intermezzo

Tra il secondo e il terzo quadro c'è un comando di sezionamento standard che va dato, si noti bene, *prima* che il quadro cominci. In beamer questi comandi servono solo a comporre l'indice generale: il contenuto del loro argomento non verrà *mai* mostrato. Si ricordi che una presentazione non è un documento tradizionale, per cui si raccomanda di valutarne attentamente la struttura e di non sezionarla eccessivamente.

Terzo quadro

Composto di una sola diapositiva (figura 27c), contiene una definizione in cui due parole sono evidenziate con il comando `\alert`, che marca il proprio argomento diversamente dai comandi standard (in blu, in questo caso).

Quarto quadro

È il quadro più articolato, composto di quattro diapositive (figure 27d-g). Si noti che qualunque sia il numero di diapositive che lo compongono, il suo contenuto va scritto *una sola volta*: le tecniche di esposizione incrementale spiegate nel prossimo paragrafo si occuperanno di dilazionarne la proiezione.

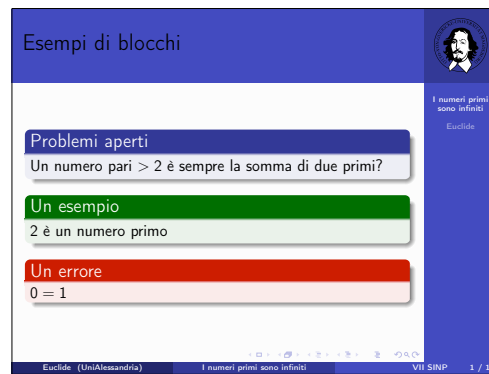


Figura 28: Esempi di blocchi

Quinto quadro

Composto di una sola diapositiva, evidenzia il proprio contenuto in due *blocchi* (i due ambienti `block`), per i quali si rimanda al paragrafo 15.2.4 nella pagina seguente.

15.2.3 Esposizione incrementale

Rivelare progressivamente il contenuto di un quadro (specie se si tratta di un ragionamento svolto per punti o di un elenco) proiettandone gli elementi uno alla volta può aiutare il pubblico a seguire meglio il discorso e a limitare il carico d'attenzione. Con beamer questo effetto si ottiene in diversi modi, anche se si raccomanda di non abusarne e di riservarlo ai casi di vera necessità.

Il metodo più semplice è frazionare il contenuto del quadro dando nei punti opportuni altrettanti comandi `\pause`. Ogni diapositiva mostrerà il testo corrente più tutto quello precedente, e così via.

Il secondo metodo è dare dopo ogni `\item` il comando `\onslide`, la cui sintassi generale è

```
\onslide<intervallo>>{<testo>}
```

dove il `<testo>` verrà mostrato solo nelle diapositive indicate nell'`<intervallo>`. Nelle altre, anche se non presente sullo schermo, avrà comunque lo spazio corrispondente. Per esempio, un elenco i cui punti scompaiano uno dopo l'altro dopo essere stati proiettati si ottiene con:

```
\begin{frame}
\begin{itemize}
\item \onslide<1>{Mane}
\item \onslide<2>{Tekel}
\item \onslide<3>{Fares}
\end{itemize}
\end{frame}
```

Il terzo metodo consiste nello scrivere subito dopo ciascun `\item` un'espressione che nella sua forma più completa è:

```
<numero>-<numero>
```

dove:

- `<\numero>` è il numero progressivo della diapositiva nella presentazione;
- un semplice numero (`<4>`, per esempio) fa proiettare l'elemento una sola volta nella quarta diapositiva del quadro;
- un intervallo numerico separato con un trattino (`<4-6>`, per esempio) mantiene l'elemento visibile nelle diapositive dalla quarta alla sesta comprese, ma non prima né dopo;
- non scrivere nulla prima del trattino (`<-4>`) equivale a scrivere `<1-4>`;
- non scrivere nulla dopo il trattino (`<4->`) rende visibile l'elemento dalla diapositiva in cui compare la prima volta (la quarta, in questo caso) fino all'ultima diapositiva del quadro (come nella prima presentazione analizzata).

Si poteva scrivere più sinteticamente il secondo quadro della prima presentazione con:

```
\begin{itemize}[<+>]
\item Mane
\item Tekel
\item Fares
\end{itemize}
```

15.2.4 Blocchi

In beamer un *blocco* è una cornice più o meno marcata che evidenzia una porzione di testo. La classe definisce tre ambienti standard per i blocchi, resi diversamente in base al tema caricato: `block` per i blocchi generici, `exampleblock` per gli esempi e `alertblock` per gli avvisi. Per esempio, in una presentazione con il tema Madrid il codice

```
\begin{frame}
\begin{block}{Problemi aperti}
Un numero pari  $>2$  è sempre la somma di due primi?
\end{block}

\begin{exampleblock}{Un esempio}
 $2$  è un numero primo
\end{exampleblock}

\begin{alertblock}{Un errore}
 $0=1$ 
\end{alertblock}
\end{frame}
```

produce la diapositiva mostrata nella figura 28 nella pagina precedente, in cui si vede che la differenza fra i blocchi è soltanto cromatica. Per le innumerevoli personalizzazioni possibili si veda la documentazione della classe.

15.2.5 Bibliografia

Non dovrebbe *mai* essere necessario proiettare i riferimenti bibliografici del proprio lavoro. Nei rari casi in cui servisse, si può usare un codice come

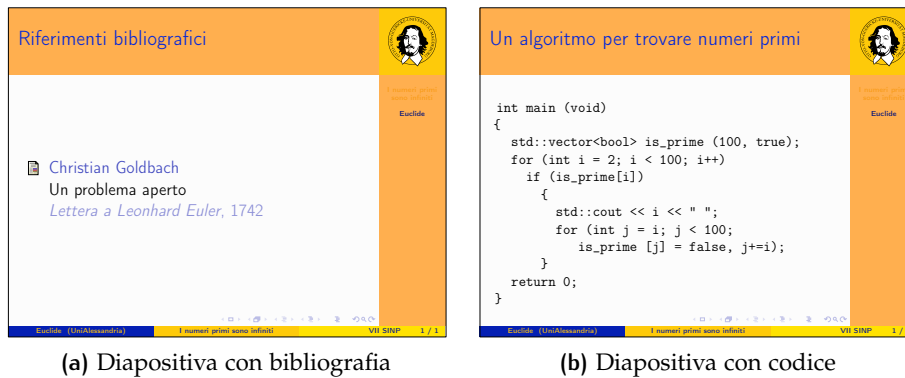


Figura 29: Diapositive particolari

```

\begin{frame}
\frametitle{\refname}
\begin{thebibliography}{9}
\bibitem{goldbach:congettura} Christian Goldbach
\newblock Un problema aperto
\newblock \emph{Lettera a Leonhard Euler}, 1742
\end{thebibliography}
\end{frame}

```

che produce la diapositiva mostrata nella figura 29a, dove il comando standard `\newblock` separa i diversi elementi della bibliografia. Va da sé che in una presentazione non vanno *mai* usati nemmeno i comandi come `\cite` o analoghi, perché durante la proiezione non avrebbe modo di verificare a quale opera si riferiscono.

15.2.6 Codici

Per scrivere del codice in una diapositiva, come mostra la figura 29b, si può usare l'ambiente standard `verbatim`, ricordandosi di dare a `frame` l'opzione `fragile`:

```

\begin{frame}[fragile]
\frametitle{Un algoritmo per trovare numeri primi}
\begin{verbatim}
int main (void)
{
    std::vector<bool> is_prime (100, true);
    for (int i = 2; i < 100; i++)
        if (is_prime[i])
        {
            std::cout << i << " ";
            for (int j = i; j < 100; is_prime[j] = false, j+=i);
        }
    return 0;
}
\end{verbatim}
\end{frame}

```

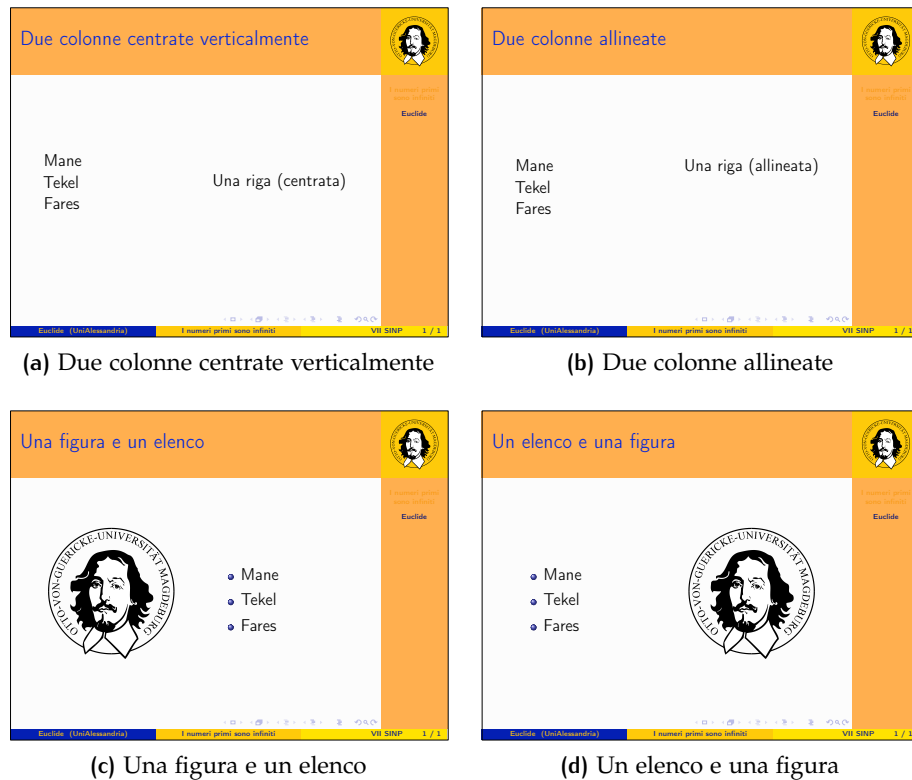


Figura 30: Diapositive strutturate su più colonne

15.2.7 Disporre il contenuto su più colonne

Disporre il contenuto di una diapositiva su più colonne è utile soprattutto quando si devono inserire tabelle o figure con la relativa descrizione, che in una presentazione va messa sempre *accanto*. Gli ambienti mobili `table` e `figure`, infatti, non hanno motivo di essere usati: chi parla non dirà mai «come abbiamo visto nella figura n », perché *nessuno* durante la spiegazione si ricorderà qual è la figura n .

Le diverse colonne sono prodotte da altrettanti ambienti `column` (ai quali va *sempre* assegnata una larghezza) annidati in un unico ambiente `columns`. Il codice seguente

```
\begin{frame}
\begin{columns}
\begin{column}{0.4\textwidth}
Mane \\ Tekel \\ Fares
\end{column}

\begin{column}{0.4\textwidth}
Una riga (centrata)
\end{column}
\end{columns}
\end{frame}
```

produce la diapositiva mostrata nella figura 30a, nella quale le righe delle due colonne sono centrate verticalmente l'una rispetto all'altra (è l'impostazione predefinita). Se si vogliono le prime righe di ciascuna colonna allineate come nella figura 30b, invece, basta dare a `columns` l'opzione `t`.

Una figura (a sinistra) con accanto un elenco (a destra), come nella figura 30c a fronte, si ottiene con

```
\begin{frame}
\begin{columns}
\begin{column}{0.4\textwidth}
\includegraphics[width=\columnwidth]{figura}
\end{column}

\begin{column}{0.4\textwidth}
\begin{itemize}
\item Mane
\item Tekel
\item Fares
\end{itemize}
\end{column}
\end{columns}
\end{frame}
```

Si ottiene l'effetto contrario (mostrato nella figura 30d nella pagina precedente) invertendo il contenuto dei due ambienti `column`.

15.2.8 Stampare la presentazione

Assegnando a beamer l'opzione `handout` si ottiene facilmente anche una versione della presentazione adatta alla stampa (un *handout*, in gergo), ovviamente priva di (eventuali) effetti dinamici. A questo scopo il pacchetto `pgfpages` permette di raccogliere in una stessa pagina 2, 4, 8 o 16 diapositive caricandolo così:

```
\usepackage{pgfpages}
\pgfpagesuselayout{4 on 1}[a4paper,border shrink=5mm,landscape]
```

dove si chiede a \LaTeX di mettere quattro diapositive in ogni pagina (4 on 1) di formato A4 disposta in orizzontale (*landscape*), con un piccolo spazio (`border shrink`) di 5 mm attorno a ciascuna diapositiva. (L'opzione *landscape* è richiesta soltanto se in una pagina si vogliono 4 o 16 diapositive, non negli altri casi.)

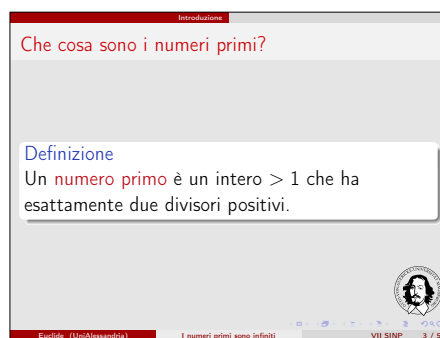
15.3 PRESENTAZIONI PERSONALIZZATE

In questa sezione si mostreranno solo alcuni esempi delle numerose possibilità offerte da beamer per personalizzare le presentazioni.

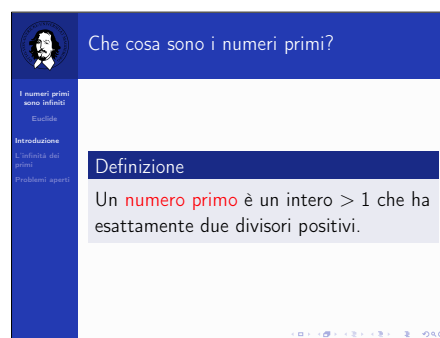
Il *tema* gestisce l'aspetto grafico della presentazione definendo i colori che appariranno nelle diapositive, il formato di elenchi e blocchi e infine l'aspetto e la disposizione degli elementi di contorno. Si noti che tutti questi elementi possono essere regolati uno per uno, moltiplicando le possibili personalizzazioni. Se non dovessero bastare i temi precaricati in beamer, se ne possono scaricare moltissimi altri da Internet o ancora crearsene di personali da zero.

Il tema si seleziona scrivendo nel preambolo

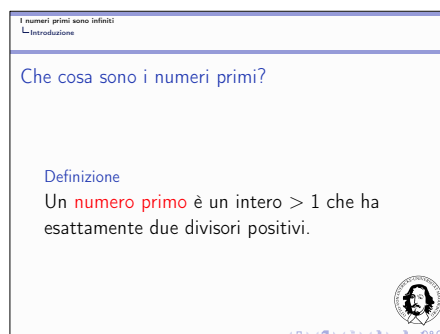
```
\usetheme{<tema della presentazione>}
```



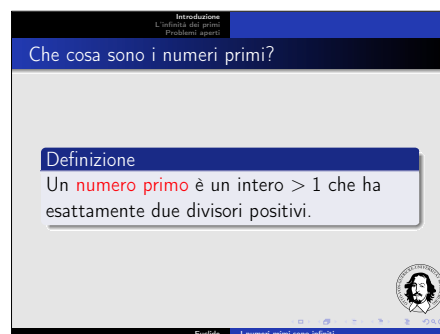
(a) CambridgeUS



(b) Berkeley



(c) Montpellier



(d) Warsaw

Figura 31: Alcuni temi predefiniti di beamer

dove *tema della presentazione* è il nome del tema prescelto che, per cominciare, si può scegliere tra quelli predefiniti dalla classe, identificati di regola dal nome di una città.

Non potendoli qui riportare tutti, si tenga presente che a grandi linee i temi possono essere:

- senza barra di navigazione (AnnArbor, CambridgeUS, Madrid, per esempio);
- con barra di navigazione laterale (Berkeley, Goettingen, Marburg);
- con barra di navigazione ad albero (Antibes, JuanLesPins, Montpellier);
- con quadro di navigazione (Berlin, Dresden, Warsaw, Singapore).

Alcuni di essi vengono mostrati nelle diapositive della figura 31.

16 | CURRICULUM

Questo articolo spiega come scrivere un curriculum, sia in generale (fornendo indicazioni che possono essere sfruttate con qualunque programma di videoscrittura), che, nello specifico, con \LaTeX .

16.1 INTRODUZIONE

Il *curriculum vitae* (o semplicemente *curriculum*) è un documento che contiene l'elenco delle esperienze scientifiche e professionali di un individuo. Di norma lo si scrive per ottenere un colloquio con potenziali datori di lavoro. I candidati possono aggiungere informazioni personali, slegate dalla propria vita professionale, per fornire un'immagine più completa di sé.

16.2 STILE E CONTENUTI

16.2.1 Stile

Per quanto riguarda lo stile del curriculum, la scelta è fra due alternative essenzialmente.

CRONOLOGICO Nello stile cronologico le esperienze del candidato vengano elencate in ordine cronologico inverso. È lo stile più diffuso e mostra l'evoluzione professionale del candidato.

FUNZIONALE In questo stile le esperienze sono raggruppate per aree tematiche. Si usa per evidenziare l'esperienza maturata dal candidato in alcuni ambiti specifici.

16.2.2 Contenuti

I contenuti di un curriculum variano in funzione del contesto. Ci sono informazioni essenziali e facoltative.

Informazioni essenziali

DATI ANAGRAFICI Sono nome, sesso, data e luogo di nascita, cittadinanza, stato civile, indirizzo, numero di telefono, indirizzo di posta elettronica. Questi dati non sono tutti necessari: talvolta si richiede di ometterne alcuni, per esempio il sesso e la data di nascita, per evitare possibili discriminazioni.

ISTRUZIONE Si elencano le qualifiche possedute, sia scientifiche che professionali.

ESPERIENZE LAVORATIVE Per ogni tappa della propria carriera si indicano le mansioni svolte e i risultati ottenuti.

Informazioni facoltative

PROFILO PERSONALE È una descrizione sintetica del candidato.

OBIETTIVI PERSONALI Si tratta di una breve descrizione degli obiettivi professionali del candidato.

FOTOGRAFIA La presenza di una fotografia dipende dal contesto: negli Stati Uniti di norma non c'è, perché potrebbe suggerire che il datore di lavoro sia interessato all'aspetto fisico del candidato, mentre in Italia non ci sono consuetudini al riguardo, per cui la scelta è lasciata al candidato.

LINGUE STRANIERE È bene riportare le lingue conosciute, specificando la capacità di comunicazione scritta e orale.

CONOSCENZE INFORMATICHE Si possono indicare se riguardano software tecnici, linguaggi di programmazione o strumenti che non fanno parte delle conoscenze informatiche di base.

16.3 INDICAZIONI

Il curriculum è spesso il primo contatto con il futuro datore di lavoro e perciò è importante curarlo sia nella forma che nei contenuti. È opportuno concentrare l'attenzione sulle esperienze che danno valore aggiunto alla candidatura.

Si consiglia di ridurre al minimo la lunghezza del curriculum, in modo da mettere in risalto gli aspetti più significativi, correlati all'impiego per cui ci si sta candidando. Di regola è bene non superare le due pagine (talvolta solo una), ma in certi altri casi il curriculum deve essere dettagliato. Una possibilità consiste nel preparare due versioni del curriculum, una sintetica e una estesa, in modo che il datore di lavoro possa esaminarle entrambe.

\LaTeX , grazie all'alta qualità dei documenti prodotti, si presta bene per la redazione di un curriculum efficace.

16.4 CURRICULUM A REGOLA D'ARTE

La classe `moderncv`, consigliata in questa guida, è un'ottima soluzione per scrivere un curriculum con \LaTeX . Caratterizzata da uno stile chiaro e accattivante, ha numerosi comandi dedicati che permettono di soddisfare le esigenze più disparate. Il modello fornito come esempio è ben strutturato e commentato: l'utente deve solo sostituire con i propri i dati fittizi presenti all'interno dell'esempio, per disporre in pochi minuti di un curriculum gradevole.

La classe `moderncv` ha alcune opzioni *globali*, che hanno cioè effetto sull'intero documento.

- `10pt`, `11pt`, `12pt` sono analoghe a quelle standard (il valore predefinito è `11pt`);
- `a4paper` e `a5paper` sono analoghe a quelle standard (la dimensione predefinita è `a4paper`);

- sans scrive il curriculum usando un font sans serif; in alternativa si può usare roman, che usa un font con grazie.

Nel preambolo si danno le seguenti istruzioni:

```
\moderncvstyle{<stile del curriculum>}
\moderncvcolor{<colore degli elementi di separazione>}
\firstname{<nome>}
\familyname{<cognome>}
\title{<titolo del curriculum>}
\address{<indirizzo>}{<codice postale>}
\mobile{<numero di cellulare>}
\phone{<numero di telefono>}
\fax{<numero di fax>}
\email{<indirizzo di posta elettronica>}
\homepage{<sito Web>}
\photo[<altezza della fotografia>][<spessore della cornice>]{<nome del file>}
```

Il loro funzionamento si spiega da sé, ma si osservi quanto segue:

- lo `<stile del curriculum>` può essere casual (lo stile predefinito), classic, oldstyle o banking;
- il `<colore degli elementi di separazione>` può essere blue (il colore predefinito), orange, green, red, purple, grey e black;
- i vari `\title`, `\address`, `\mobile`, `\phone`, `\fax`, `\email`, `\homepage` e `\photo` sono facoltativi (se si vuole dare un titolo al curriculum occorre anche dare `\makecvtitle` subito dopo `\begin{document}`);
- per l'`<altezza della fotografia>` e lo `<spessore della cornice>` si può usare una qualsiasi unità tipografica riconosciuta da L^AT_EX (per non avere alcuna cornice basta dare al suo spessore il valore 0);
- i margini del documento si impostano con il pacchetto geometry (si veda il paragrafo 3.6.1 a pagina 34).

Dopo di che, le varie voci del curriculum si inseriscono dando nel corpo del testo le istruzioni seguenti:

```
\cvitem{<voce>}{<descrizione>}
\cvdoubleitem{<voce>}{<descrizione>}{<voce>}{<descrizione>}
\cventry{<anno>}{<voce in neretto>}{<descrizione in corsivo>}{<descrizione>}%
    {<altra descrizione>}{<descrizione a capo>}
\cvlistitem{<voce di un elenco>}
```

Qualche prova chiarirà le idee, ma si osservi che:

- `\cvitem` stampa voci a tutta pagina, `\cvdoubleitem` le stampa su due colonne;
- gli argomenti di `\cventry` non vanno tutti necessariamente riempiti (per lasciarne qualcuno vuoto basta specificare {});
- i comandi di sezionamento (`\section` e `\subsection`) producono sezioni non numerate.

Il curriculum mostrato nella figura 32 è stato prodotto con il codice 3 a pagina 281.

La classe moderncv mette anche a disposizione un modello per scrivere una lettera di presentazione, il cui stile richiama nelle linee essenziali quello del curriculum.

Lorenzo Pantieri

Curriculum

viale dei Giardini 1, Cesena
CAP 47521

+39 333 1234567
+39 0547 123456

lorenzo.pantieri@gmail.com
www.lorenzopantieri.net



Dati anagrafici

Nascita Cesena, 30 gennaio 1973

Titoli di studio

2000 **Laurea in Matematica**, *Università degli Studi*, Bologna, votazione 110/110 e lode.
 1992 **Maturità classica**, *Liceo classico "Vincenzo Monti"*, Cesena, votazione 60/60.

Abilitazioni

- Abilitazione all'insegnamento della Matematica
- Abilitazione all'insegnamento della Fisica

Esperienze lavorative

Nella scuola

2001–2017 **Docente di Matematica**, *Istituto "Versari-Macrelli"*, Cesena.
 1999–2001 **Giornalista**, rivista *"PC Magazine"*.
 1997–1999 **Sviluppatore di siti Web**.

Lingue

Italiano Madrelingua
 Inglese Buono, affinato con soggiorni in Inghilterra

Conoscenze informatiche

Programmi Mathematica, Photoshop Linguaggi Pascal, C

Collaborazioni

- Membro del Gruppo Utilizzatori Italiani di \TeX e \LaTeX

Pubblicazioni

- L'arte di scrivere con \LaTeX , 2017.
- Matematica per Istituti professionali, 2016

Figura 32: Curriculum

Codice 3: Codice per un curriculum

```

\documentclass[11pt,a4paper,sans]{moderncv}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\moderncvstyle{classic}
\moderncvcolor{blue}
\usepackage[scale=0.8]{geometry}
\firstname{Lorenzo}
\familyname{Pantieri}
\title{Curriculum}
\address{viale dei Giardini 1, Cesena}{CAP 47521}
\mobile{+39 333 1234567}
\phone{+39 0547 123456}
\email{lorenzo.pantieri@gmail.com}
\homepage{www.lorenzopantieri.net}
\photo[64pt][0.4pt]{LP}

\begin{document}
\makecvtitle

\section{Dati anagrafici}
\cvitem{Nascita}{Cesena, 30 gennaio 1973}

\section{Titoli di studio}
\cventry{2000}{Laurea in Matematica}%
        {Università degli Studi}{Bologna}%
        {votazione 110/110 e lode}{}
\cventry{1992}{Maturità classica}%
        {Liceo classico "Vincenzo Monti"}{Cesena}%
        {votazione 60/60}{}

\section{Abilitazioni}
\cvlistitem{Abilitazione all'insegnamento della Matematica}
\cvlistitem{Abilitazione all'insegnamento della Fisica}

\section{Esperienze lavorative}
\cventry{2001--2017}{Docente di Matematica}%
        {Istituto "Versari-Macrelli"}{Cesena}{}{}
\cventry{1999--2001}{Giornalista}%
        {rivista "PC Magazine"}{}{}{}
\cventry{1997--1999}{Sviluppatore di siti Web}{}{}{}{}

\section{Lingue}
\cvitem{Italiano}{Madrelingua}
\cvitem{Inglese}{Buono, affinato con soggiorni in Inghilterra}

\section{Conoscenze informatiche}
\cdoubleitem{Programmi}{Mahematica, Photoshop}{Linguaggi}{Pascal, C}

\section{Collaborazioni}
\cvlistitem{Membro del Gruppo Utilizzatori Italiani di \TeX{} e \LaTeX{}}

\section{Pubblicazioni}
\cvlistitem{L'arte di scrivere con \LaTeX, 2017.}
\cvlistitem{Matematica per Istituti professionali, 2016.}
\end{document}

```

17 | LETTERE

Questo articolo presenta alcuni strumenti messi a disposizione da \LaTeX per scrivere una lettera.

17.1 LETTERE STANDARD

Per scrivere lettere, \LaTeX offre la classe standard `letter`, che ha alcune particolarità:

- non definisce alcun comando di sezionamento;
- non accetta l'opzione `twoside`.
- l'indirizzo del mittente (`\address`), la firma (`\signature`) e la data (`\date`, opzionale) si scrivono prima di `\begin{document}` (nell'indirizzo e nella firma, righe diverse vanno separate con `\\`);
- dopo `\begin{document}` segue l'istruzione `\begin{letter}` e, fra parentesi graffe (come argomento dell'ambiente `letter`), l'indirizzo del destinatario; quindi viene posta l'istruzione di apertura `\opening` e, a seguire, il testo della lettera, che termina con `\closing`, che è la parte riservata per i saluti;
- dopo i saluti, si può scrivere un poscritto (con il comando `\ps`), specificare degli allegati (`\encl`, *enclosures*) o notificare che si è inviata ad altri destinatari una copia della lettera (`\cc`; p. c., "per conoscenza"; in inglese *cc*, *carbon copy*).

Di seguito si riporta un esempio di lettera redatta con la classe `letter` (la figura 33 nella pagina successiva mostra il risultato).

```
\documentclass[a4paper]{letter}

\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{layaureo}
\usepackage{lipsum}

\address{Dott.~Lorenzo Pantieri \\
         Dipartimento di Matematica \\
         Università degli Studi di Bologna \\
         Piazza di Porta S.~Donato, 5 \\
         40125 Bologna}
\signature{Lorenzo Pantieri}

\begin{document}
\begin{letter}{Prof.~Enrico Gregorio \\
              Dipartimento di Informatica \\
              Università di Verona \\
```

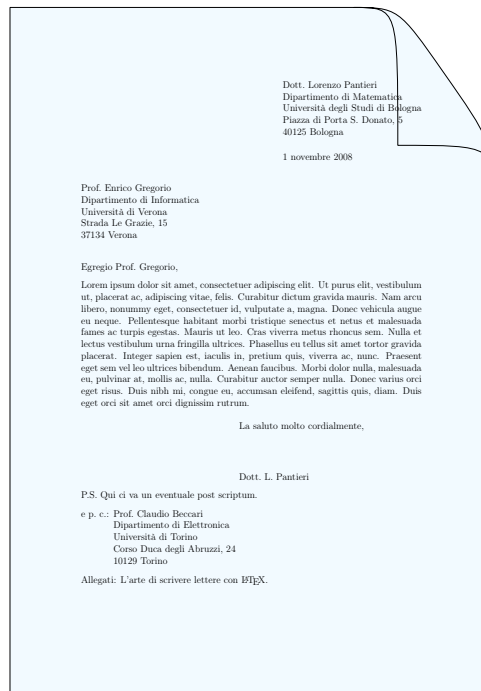


Figura 33: Un esempio di lettera composta con la classe letter.

```

Strada Le Grazie, 15 \\
37134 Verona}
\opening{Gentile prof.-Gregorio,}

\lipsum[1]

\closing{La saluto molto cordialmente,}

\ps{Qui ci va un eventuale poscritto.}

\cc{Prof.-Claudio Beccari \\
Dipartimento di Elettronica \\
Università di Torino \\
Corso Duca degli Abruzzi, 24 \\
10129 Torino}

\encl{L'arte di scrivere lettere con \LaTeX.}

\end{letter}
\end{document}

```

Si noti che \LaTeX stampa come predefinita la data del giorno in cui si compone la lettera. Come al solito, per specificare una $\langle data \rangle$ diversa si dà l'istruzione $\text{\date}\{\langle data \rangle\}$, mentre per ometterla si scrive $\text{\date}\{\}$.

Ci sono anche i comandi \location e \telephone (che indicano rispettivamente l'indirizzo e il numero di telefono del mittente), che però hanno effetto solo in assenza del comando \address . C'è anche \name (indica il nome del mittente), usato in assenza di \signature .

La classe letter permette di scrivere a destinatari diversi lettere con lo stesso testo, inserendole nello stesso sorgente:

```

\documentclass[...]{letter}
\usepackage{...}

```

```

\address{<...>}
\signature{<...>}

\begin{document}

\newcommand{\body}{<Testo della lettera>}

\begin{letter}{<indirizzo del primo destinatario>}
\opening{Gentile professor Gregorio,}
\body
\closing{<...>}
\end{letter}

\begin{letter}{<indirizzo del secondo destinatario>}
\opening{Gentile professor Beccari,}
\body
\closing{<...>}
\end{letter}
\end{document}

```

17.2 LETTERE CONFORMI ALLO STILE ITALIANO

Lo stile della classe `letter` è tipicamente americano. La classe `letteracdp`, invece, permette di scrivere lettere conformi allo stile italiano. Questa classe fa parte di un *bundle* usato dal Coordinamento dei Dottorandi e dei Dottori di Ricerca dell'Università di Padova, ed è preinstallata nelle più diffuse distribuzioni di L^AT_EX. La classe `letteracdp` si carica nel solito modo:

```
\documentclass[<opzioni>]{letteracdp}
```

Di seguito si riportano le opzioni principali della classe `letteracdp`. Si descrivono solo le opzioni che si comportano in modo diverso da quelle standard; le altre sono semplicemente solo elencate.

- 10pt, 11pt, 12pt
- letterpaper, legalpaper, a4paper, executivepaper, a5paper, b5paper
L'opzione predefinita è `a4paper`.
- final, draft
- oneside, twoside
- onecolumn, twocolumn L'opzione `twocolumn` è incompatibile con la classe `letteracdp`; l'opzione predefinita è `onecolumn` (è l'unica accettabile).
- leqn
- fleqn
- mediumsubject, boldsubject L'opzione `mediumsubject` (è predefinita) scrive l'oggetto (*subject*) della lettera con un font di spessore medio, mentre `boldsubject` lo compone in neretto.
- uprightsignature, italicsignature L'opzione `uprightsignature` (è predefinita) scrive la firma in tondo, mentre `italicsignature` la scrive in corsivo.

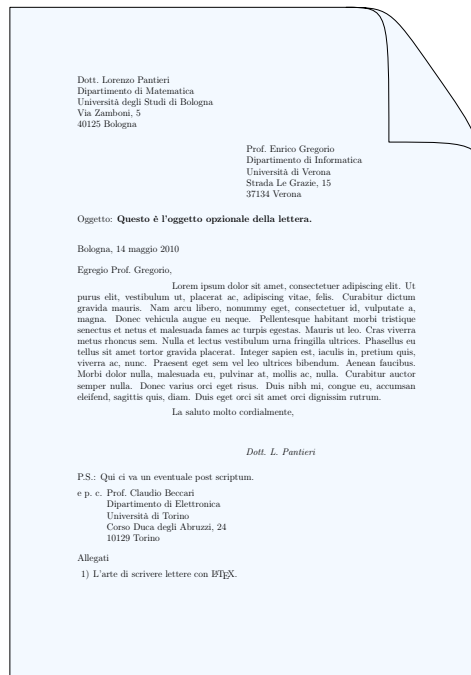


Figura 34: Una lettera composta con la classe letteracdp

- `indent`, `noindent`, `smartindent`, `shortindent` Impostano il rientro dei capoversi. L'opzione `indent` (predefinita) imposta il rientro dei capoversi a un decimo della larghezza del testo; `noindent` elimina del tutto il rientro; `smartindent` imposta il rientro pari alla larghezza dell'ultima riga della dichiarazione di apertura; `shortindent` imposta lo stesso rientro delle classi standard.

Di seguito si riporta un esempio di lettera redatta con la classe `letteracdp` (la figura 34 mostra il risultato).

```
\documentclass[boldsubject,italicsignature,smartindent]{letteracdp}

\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{layaureo}
\usepackage{lipsum}

\address{Dott.~Lorenzo Pantieri \\\
Dipartimento di Matematica \\\
Università degli Studi di Bologna \\\
Via Zamboni, 5 \\\
40125 Bologna}
\signature{Lorenzo Pantieri}
\place{Bologna}

\begin{document}
\begin{letter}{Prof.~Enrico Gregorio \\\
Dipartimento di Informatica \\\
Università di Verona \\\
Strada Le Grazie, 15 \\\
37134 Verona}%
[Questo è l'oggetto opzionale della lettera.]
```

```

\opening{Gentile professor Gregorio,}

\lipsum[1]

\closing{La saluto molto cordialmente,}

\PS
Qui ci va un eventuale post scriptum.

\cc{Prof.~Claudio Beccari \\
    Dipartimento di Elettronica \\
    Università di Torino \\
    Corso Duca degli Abruzzi, 24 \\
    10129 Torino}

\begin{enclosures}
    \item L'arte di scrivere lettere con \LaTeX.
\end{enclosures}

\end{letter}
\end{document}

```

Per scrivere il poscritto e per specificare degli allegati si usano rispettivamente la dichiarazione `\PS` e l'ambiente `enclosures`. Il comando opzionale `\place` indica il luogo in cui la lettera è stata scritta.

Parte VIII

LINGUE ESOTICHE

18 | LATINO E GRECO

Questo capitolo presenta gli strumenti essenziali per scrivere in latino e in greco, le due lingue fondamentali per chi si occupa di scienze umane. Se non indicato diversamente, si assume che la lingua principale del documento sia l'italiano.

18.1 SCRIVERE IN LATINO

La lingua latina ha diverse ortografie, che differiscono per lessico e sillabazione. Ci sono:

1. il latino moderno, usato negli scritti eruditi, scientifici e religiosi dal XVI secolo a oggi;
2. il latino classico, usato dal I secolo a.C. fino alla caduta dell'Impero romano d'Occidente;
3. il latino medievale, usato nel Medioevo come lingua erudita, letteraria e religiosa;
4. il latino con le *marche prosodiche* (che permettono di segnare le vocali brevi e lunghe), usato nelle moderne grammatiche;
5. il latino ecclesiastico, usato nei messali e nei breviari della Chiesa cattolica;
6. il latino liturgico, impiegato per scrivere i canti gregoriani.

Sei ortografie diverse richiedono sei diversi nomi di lingua da indicare a babel. Bisogna specificare `latin` tra le opzioni del pacchetto e scegliere una variante ortografica indicando uno dei sei *attributi* seguenti:

modern per il latino moderno (essendo predefinito, si può omettere);

classic per il latino classico;

medieval per il latino medievale;

withprosodicmarks per il latino con le marche prosodiche;

ecclesiastic per il latino ecclesiastico;

liturgic per il latino liturgico.

Per esempio, un documento in italiano come lingua principale che contiene qualche brano in latino classico si imposta con:

```
\usepackage[latin.classic,italian]{babel}
```

oppure con

```
\usepackage[latin,italian]{babel}
\languageattribute{latin.classic}{ancient}
```

Si passa da una lingua all'altra con gli strumenti standard di L^AT_EX come, per esempio:

```
\dots testo in italiano.
```

```
\begin{otherlanguage*}{latin}
\em Gallia est omnis divisa in
partes tres, quarum unam incolunt
Belgae, aliam Aquitani, tertiam
qui ipsorum lingua Celtae, nostra
Galli appellantur.
\end{otherlanguage*}
```

```
Altro testo in italiano\dots
```

...testo in italiano.

Gallia est omnis divisa in partes tres, quarum unam incolunt Belgae, aliam Aquitani, tertiam qui ipsorum lingua Celtae, nostra Galli appellantur.

Altro testo in italiano...

e si può inserire un testo breve o brevissimo in un capoverso scrivendolo nell'argomento di un comando personale `\latino`:

```
\newcommand{\latino}[1]{\foreignlanguage{latin}{\em #1}}
```

da usare come segue:

```
L'attacco della prima
\emph{Catilinaria}, \latino{Quo
usque tandem abutere, Catilina,
patientia nostra?}, è celebre.
```

L'attacco della prima *Catilinaria*,
Quo usque tandem abutere, Catilina,
patientia nostra?, è celebre.

Di seguito si descrivono alcune *proprietates* di babel.

18.1.1 Latino medievale

Per dare al testo latino una patina medievaleggiante, l'utente deve servirsi delle seguenti varianti (tutte o solo alcune):

- usare un unico segno, *u*, per *u* e *v* minuscole: *Nouembris* anziché *Novembris*, per esempio;
- usare un unico segno, *V*, per *U* e *V* maiuscole: *IVLIVS* anziché *Iulius*;
- usare le legature per i dittonghi *æ* e *œ* (minuscoli e maiuscoli), che nel latino classico e in quello moderno (ma non in quello ecclesiastico e in quello liturgico) sono *sempre* sciolti: *æ* (`\ae`), *œ* (`\oe`), *Æ* (`\AE`), *Œ* (`\OE`).

L'attributo `medieval` di babel definisce un paio di automatismi utili per scrivere in questa varietà di latino:

- gestisce correttamente la conversione da minuscolo a maiuscolo nelle stringhe automatiche come le testatine: per esempio, un titolo come `\chapter{C\ae sar et Heluetii}` nella corrispondente testatina maiuscola diventa *CÆSAR ET HELVETII*;
- ridefinisce `\prefacename` per ottenere *Præfatio* anziché *Praefatio* (le altre parole fisse non subiscono variazioni).

18.1.2 Marche prosodiche

L'attributo `withprosodicmarks` permette a \LaTeX di riconoscere la marca della quantità sulle vocali *minuscole* (spesso segnata nei testi didattici o di linguistica), che si ottiene premettendo loro il carattere `^` per la quantità breve `˘` o il carattere `=` per quella lunga `¯`:

```
\begin{otherlanguage*}{latin}
\ProsodicMarksOn\em
Galli^a est omn^is divis^a in
part=es tr=es, qu=arum un^am
inc^olunt Belgae, ali^am Aquitan=i,
terti^am qu=i ips=orum lingu=a
Celtae, nostr=a Gall=i
appell=antur.
\end{otherlanguage*}
```

*Galliā est omnīs divisā in partēs trēs,
quārum unām incōlunt Belgae, aliām
Aquitānī, tertiām quī ipsōrum lin-
guā Celtae, nostrā Gallī appellāntur.*

Si ricordi, però, di attivarne *effettivamente* la composizione anteponendo al testo interessato la *dichiarazione* `\ProsodicMarksOn`.

Si noti che \LaTeX potrebbe sillabare in modo insoddisfacente o non sillabare per nulla le parole con la quantità segnata: si risolve il problema indicando esplicitamente i punti di cesura con `"|` là dove serve.

18.1.3 Documenti interamente in latino

Chi dovesse scrivere un documento completamente in latino (seguendo indifferentemente qualunque delle sue sei varianti ortografiche) può contare su qualche agevolazione in più. L'opzione `latin` di `babel`, infatti:

- converte automaticamente la data prodotta da `\today` nella corrispondente data "latina", nella forma *vi Ianuarii MMXVIII*;
- traduce in latino tutte le parole fisse delle classi di documento standard: *Capitolo* diventa *Caput*, per esempio.

18.1.4 Preambolo tipico

Si fornisce infine il preambolo tipo per comporre un documento in italiano come lingua principale che contiene qualche brano in latino classico.

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[latin.classic,italian]{babel}

\begin{document}
...
\end{document}
```

18.2 SCRIVERE IN GRECO

Il greco, che com'è noto si scrive con un alfabeto diverso da quello latino, ha diverse ortografie, che differiscono per lessico e sillabazione.

1. C'è il greco moderno monotónico, che ha un solo tipo di accento (in genere acuto) e la dièresi, ma non usa gli spiriti.
2. C'è il greco moderno politónico, che usa tutta la collezione di accenti, spiriti, dièresi e iota sottoscritto e ascritto, ma esplicita gli iati con la dièresi anche se non sempre serve. Per esempio: ἄλλος non richiederebbe la dièresi sulla υ, ma questa varietà preferisce scrivere ἄλλος.
3. C'è infine il greco antico, che si scrive come il greco moderno politónico, ma usa la dièresi solo se è strettamente necessario.

Tre ortografie diverse richiedono tre diversi nomi di lingua da indicare a babel. Bisogna specificare greek tra le opzioni del pacchetto e scegliere una variante ortografica indicando uno dei tre attributi seguenti:

monotoniko per il greco moderno monotónico (essendo predefinito, si può omettere);

polutoniko per il greco moderno politónico;

ancient per il greco antico.

Per esempio, un documento in italiano come lingua principale che contiene qualche brano in greco antico si imposta con:

```
\usepackage[greek.ancient,italian]{babel}
```

oppure con

```
\usepackage[greek,italian]{babel}
\languageattribute{greek}{ancient}
```

Una volta scelto il sistema ortografico che s'intende seguire, si possono scrivere le parti in greco in due modi:

- *traslitterando* il testo greco secondo una particolare corrispondenza tra i caratteri;
- scrivendo *direttamente* i caratteri greci.

Se non diversamente indicato, gli esempi di questo paragrafo sono in greco antico.

Si passa da una lingua all'altra con gli strumenti standard di L^AT_EX come, per esempio:

```
\dots testo in italiano.
```

```
\begin{otherlanguage*}{greek}
Th| p'anta dido'ush| ka'i
>apolambano'ush| f'usei <o
pepaideum'enos ka'i a>id'hmn
l'egai; <<d'os, <o j'eleis,
>ap'olabe, <o j'eleic>>.
L'egai d'e to~uto o>u
katajrasun'omenos, >all'a
peijarq~wn m'onon ka'i
e>uno~wn a>ut~h|.
\end{otherlanguage*}
```

```
Altro testo in italiano\dots
```

... testo in italiano.

Τη πάντα διδούση καὶ ἀπολαμβάνου-
 σι φύσει ὁ πεπαιδευμένος καὶ
 αἰδήμων λέγει· “δὸς, ὃ θέλεις, ἀπό-
 λαβε, ὃ θέλεις”. Λέγει δὲ τοῦτο οὐ
 καταθρασυνόμενος, ἀλλὰ πειθαρχῶν
 μόνον καὶ εὐνοῶν αὐτῇ.

Altro testo in italiano...

\	!	" ..	£	\$	%	&	/	()	=	? ;	^
1	2	3	4	5	6	7	8	9	0	' ,	ì	
Q X	W Ω	E Ε	R Ρ	T Τ	Y Ψ	U Υ	I Ι	O Ο	P Π	é	*	
q χ	w ω	e ε	r ρ	t τ	y ψ	u υ	i ι	o ο	p π	è [+]
A Α	S Σ	D Δ	F Φ	G Γ	H Η	J Θ	K Κ	L Λ	ç	°	§	
a α	s σ	d δ	f φ	g γ	h η	j θ	k κ	l λ	ò @	à #	ù	
> ' <	Z Ζ	X Ξ	C	V	B Β	N Ν	M Μ	; ·	:	—		
	z ζ	x ξ	c ç	v	b β	n ν	m μ	,	.	-		

Figura 35: Corrispondenza tra la tastiera italiana e i caratteri greci

e si può inserire un testo breve o brevissimo in un capoverso scrivendolo nell'argomento di un comando personale `\greco` (leggermente diverso dall'analogo `\latino`):

```
\newcommand{\greco}{\foreignlanguage{greek}}
```

da usare come segue:

In lettere minuscole, `\TeX{}` si scriverebbe `\greco{teq}`. È la radice della parola greca `\greco{t'eqnh}` che vuol dire "arte" e "mestiere".

In lettere minuscole, \TeX si scriverebbe $\tau\epsilon\chi$. È la radice della parola greca $\tau\acute{\epsilon}\chi\nu\eta$ che vuol dire "arte" e "mestiere".

18.2.1 Traslitterare i caratteri

Greco antico

Come si osserva negli esempi precedenti, il sorgente "in greco" è un insieme di caratteri latini e altri segni (tutti conosciuti, ma qui con funzioni diverse) che poco assomiglia a ciò che si vede a destra. Lo si è scritto, infatti, con il metodo più immediato per chi possiede una tastiera italiana, cioè la *traslitterazione dei caratteri*. La figura 35 mostra come l'opzione `greek` di `babel` interpreta i caratteri latini (alfabetici e non) che hanno una corrispondenza in greco (per scrivere la virgoletta alta aperta e la tilde si veda il paragrafo 3.4.2 a pagina 28 sui caratteri speciali di \LaTeX). Tutti gli altri, compresi i caratteri speciali, sono uguali nelle due lingue (per la tilde, però, si veda sotto). Si noti che la lettera italiana *v* non ha un corrispettivo greco perché è una specie di carattere greco "speciale": la scrittura *sv*, infatti, è l'unico modo per ottenere un sigma iniziale o mediale isolato σ , altrimenti interpretato dal programma sempre come finale ς .

La tabella 58 nella pagina successiva raccoglie i segni di punteggiatura e i diacritici necessari per scrivere in greco. Si noti che:

- I font greci predefiniti permettono di combinare i diacritici in *tutti* i modi previsti dalla grammatica semplicemente premettendoli a una lettera in una sequenza qualsiasi.
- L'unico diacritico da posporre alla lettera è lo ι sottoscritto.

Tabella 58: Diacritici e punteggiatura del greco antico

Segno		Codice	Esempio	Risultato
Accento	acuto	'	'w	ώ
	grave	`	`w	ὠ
	circonflesso	~	~w	ῶ
Spirito	dolce	>	>w	ὦ
	aspro	<	<w	ὡ
Dieresi		"	"i	ï
ι sottoscritto			w	ϝ
Apostrofo		"_	d" >eg'w	δ' ἐγώ
Punto	in alto	;		·
	e virgola	?		;

- Il carattere " nel sorgente restituisce un apostrofo soltanto se seguito da uno spazio (mantenuto anche nel testo composto, a differenza dell'italiano): "_.
- Il carattere ~, che in greco produce l'accento circonflesso, produce il consueto spazio indivisibile (che scrivendo in greco si ottiene con `\nobreakspace`) nelle seguenti sequenze: ~, ~", "~", ~<, <~, ~> e >~. Si risolve il problema premettendo a ciascuna una barra rovescia.
- Il greco antico non prevedeva nella scrittura le virgolette, che si possono comunque scrivere: quelle caporali si ottengono con (()).

L'opzione `greek` di `babel` definisce ulteriori caratteri utili per scopi particolari:

- i caratteri minuscoli per indicare alcuni numeri (il *qoppa* Ϟ ottenuto con `\qoppa` per il numero 6, lo *stigma* Ϛ con `\stigma` per il 90 e il *sampi* ϛ con `\sampi` per il 900); quelli per il *digamma* minuscolo ϝ, ottenuto con `\ddigamma` e maiuscolo Ϟ, ottenuto con `\Digamma`; il carattere ϙ, ottenuto con `\vardigamma`;
- l'apice ' e il pedice , per scrivere i numerali, ottenuti rispettivamente con `\anwtonos` e `\katwtonos`;
- i due comandi `\greeknumeral` e `\Greeknuneral`, che trasformano automaticamente il numero arabo scritto nel loro argomento (da 1 a 999 999, in greco lo zero non esisteva) nel corrispondente numero greco secondo il sistema di numerazione alfabetico, minuscolo e maiuscolo rispettivamente:

<pre>\begin{otherlanguage*}{greek}% \greeknumeral{996} \\\ \Greeknuneral{123} \end{otherlanguage*}</pre>	ϛϙϚ' PKT'
--	--------------

Greco moderno monotónico

L'ortografia monotónica è molto più semplice delle altre, avendo bisogno di soli due segni oltre a lettere e punteggiatura:

- ' per l'accento acuto;
- "␣ (lo spazio, evidenziato, è necessario) per l'apostrofo, oppure " seguito da ι/I o υ/Υ per la dieresi.

Un testo in questa varietà di greco appare così:

```
\begin{otherlanguage*}{greek}%
Pr'epei na skefto'ume mia
sun'arthsh ths opo'ias
gnwr'izoume 'oti up'arqei
to olokl'hwrma.
\end{otherlanguage*}
```

Πρέπει να σκεφτούμε μια συνάρτηση
της οποίας γνωρίζουμε ότι υπάρχει
το ολοκλήρωμα.

18.2.2 Scrivere in greco nel sorgente

Il modo appena descritto per scrivere in greco non è particolarmente difficile: la corrispondenza di lettere e diacritici sulla tastiera si memorizza in poco tempo, ma la traslitterazione rende comunque il sorgente difficile da leggere, specie per chi il greco lo conosce.

Quando in un documento le parti in greco sono lunghe o lunghissime (o quando il documento è in greco per intero) poterle scrivere direttamente in lingua diventa una comodità quasi irrinunciabile. I metodi per farlo sono più d'uno: si può usare una tastiera bilingue o una tastiera virtuale che non richieda di ricordare tutti gli abbinamenti, oppure si possono ridefinire allo scopo i tasti di una tastiera italiana (con i modi propri di ogni sistema operativo). Qualunque metodo si scelga tra quelli spiegati in questo paragrafo, l'editor in uso deve supportare *pienamente* la codifica Unicode.

Scritto in greco, l'esempio iniziale diventa:

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[greek.ancient,italian]{babel}

\begin{document}

\begin{otherlanguage*}{greek}
Τη πάντα διδούση καὶ ἀπολαμβάνουσα φύσει ὁ πεπαιδευμένος καὶ αἰδήμων
λέγει· “δὸς, ὁ θέλεις, ἀπόλαβε, ὁ θέλεις”. Λέγει δὲ τοῦτο οὐ
καταθρασυνόμενος, ἀλλὰ πειθαρχῶν μόνον καὶ εὐνοῶν αὐτῇ.
\end{otherlanguage*}

\end{document}
```

L'alternativa è il programma \LaTeX (da scegliere tra i motori gestiti dall'editor in uso), con il quale è facile usare i font già presenti nel proprio sistema operativo, che richiede un preambolo come il seguente:

```
\documentclass[a4paper]{article}
\usepackage{fontspec}
\setmainfont{Linux Libertine 0}
\usepackage{polyglossia}
\setmainlanguage{italian}
\setotherlanguage[variant=ancient]{greek}
```

```
\begin{document}

\begin{greek}
Τη πάντα διδούση καὶ ἀπολαμβάνουσα φύσει ὁ πεπαιδευμένος καὶ αἰδήμων
λέγει· “δὸς, ὁ θέλεις, ἀπόλαβε, ὁ θέλεις”. Λέγει δὲ τοῦτο οὐ
καταθρασυνόμενος, ἀλλὰ πειθαρχῶν μόνον καὶ εὐνοῶν αὐτῇ.
\end{greek}

\end{document}
```

Si noti che:

- il pacchetto fontenc è sostituito da fontspec, che gestisce i font del sistema in modo molto potente;
- il pacchetto inputenc *non* va caricato, perché X_YL^AT_EX lavora solo con la codifica UTF-8;
- `\setmainfont` imposta il font principale del documento con le eventuali opzioni (qui si è scelto il font Linux Libertine, già presente nel sistema e dotato di una collezione completa di caratteri greci);
- il pacchetto babel è sostituito da polyglossia;
- `\setmainlanguage` imposta la lingua principale del documento con le eventuali opzioni;
- `\setotherlanguage` carica l'altra lingua del documento con le eventuali opzioni, in questo caso il greco antico, impostato con la variante ancient (le varianti mono e poly impostano il greco moderno monotonico e politonico, rispettivamente); si possono caricare in una volta sola più lingue *senza opzioni* con il comando `\setotherlanguages`;
- l'ambiente greek equivale a otherlanguage* con l'argomento {greek}.

Si noti che usando X_YL^AT_EX nelle parti in italiano microtype permette solo la protrusione dei caratteri, ma non l'espansione e altre finzze.

18.2.3 Documenti interamente in greco

Chi dovesse scrivere un documento completamente in greco (seguendo indifferentemente qualunque delle sue tre varianti ortografiche) può contare su qualche agevolazione in più. L'opzione greek di babel, infatti:

- definisce due comandi per convertire automaticamente la data corrente nella corrispondente data greca: `\dategreek` la mette in numerali arabi (6 Ἰανουαρίου 2018), `\Grtoday` in numerali alfabetici (Γ' Ἰανουαρίου ,ΒΗ');
- traduce in greco le parole fisse delle classi di documento standard in accordo con l'ortografia dichiarata: per esempio, *Indice* diventa Εὑρετήριο in greco moderno monotonico, Εὔρετήριον in greco moderno politonico e Εὔρετήριος in greco antico;
- definisce altri due comandi utili: `\euro`, che produce il simbolo dell'euro €, e `\permill`, che produce il simbolo di per mille ‰.

Si fornisce infine il preambolo tipo per comporre *con* \LaTeX un documento in italiano come lingua principale che contiene qualche brano in greco antico.

```
\documentclass[a4paper]{article}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[greek.ancient,italian]{babel}

\begin{document}
...
\end{document}
```

Naturalmente, anche per \XeLaTeX valgono le osservazioni di questo paragrafo: basterà modificare opportunamente il preambolo appena esemplificato.

18.2.4 Specialità

Chi avesse la necessità di comporre documenti in greco di altro tipo (in particolare linguistici, filologici e letterari), può usare il pacchetto *teubner*, che comprende una collezione di caratteri inclinati particolarmente raffinati, disegnati originariamente nella tipografia Teubner di Lipsia (in onore della quale è stato scelto il nome del pacchetto), e tra le altre cose permette di trattare il testo greco classico con i numerosissimi segni filologici usati dagli specialisti della materia.

Si raccomanda infine di caricare *teubner* *dopo* *amsmath*, se dovesse servire anche quest'ultimo pacchetto. Per maggiori dettagli si rimanda alla sua documentazione.

19 | CIRILLICO

Questo capitolo, rivolto agli utenti che hanno la necessità di riportare brevi tratti di testo in russo, bulgaro, ucraino o in una delle altre lingue scritte in cirillico, presenta alcuni strumenti offerti da L^AT_EX per comporre un testo con questo alfabeto.

19.1 INTRODUZIONE

L'alfabeto cirillico, chiamato anche *Азбука* (si pronuncia “azbùka”) dall'antico nome delle prime due sue lettere, si usa per scrivere varie lingue slave (il russo, il bulgaro, l'ucraino, il bielorusso e il serbo, per esempio) e altre lingue non slave parlate in territori appartenenti all'ex Unione Sovietica e nell'odierna Federazione russa.

19.2 CODIFICA

Per comporre un testo in cirillico è conveniente abilitare la codifica OT2 dei font scrivendo nel preambolo

```
\usepackage[OT2,T1]{fontenc}
```

e definire la dichiarazione `\cyr` e il corrispondente comando con argomento `\textcyr` nel modo seguente:

```
\newlanguage\fakeLanguage
\newcommand\cyr{\fontencoding{OT2}\fontfamily{wncyr}\selectfont
\language\fakeLanguage}
\DeclareTextFontCommand{\textcyr}{\cyr}
```

In questo modo, i caratteri cirillici sono composti usando i font *wncyr*, disegnati dall'*AMS* e dalla Washington University, particolarmente adatti per essere accostati ai Computer Modern. La cesura delle parole scritte in cirillico è disabilitata, poiché (ovviamente) non si può usare quella della lingua corrente. Si può scrivere il testo in cirillico con i caratteri latini, usando una traslitterazione messa a punto dalla American Mathematical Society. Per esempio:

Tabella 59: Corrispondenza delle lettere cirilliche con le lettere della tastiera italiana

q	w	e	r	t	y	u	i	o	p
		e	p	т	ы	y	и	о	п
a	s	d	f	g	h	j	k	l	
а	с	д	ф	г	х	ј	к	л	
z	x	c	v	b	n	m			
з			в	б	н	м			

La “X” di `\TeX{}` è un suono che si trova in svariate lingue, per esempio nel russo `\textcyr{horosho}` (“bene”).

La “X” di `\TeX` è un suono che si trova in svariate lingue, per esempio nel russo хорошо (“bene”).

Allo stesso modo, la parola победа (“vittoria”) si ottiene con l’istruzione `\textcyr{pobeda}`, союз (“unione”) si ottiene con `\textcyr{soyuz}`, mentre `\textcyr{Solzhenitsyn}` produce il nome del celebre scrittore Solzhenitsyn (nella traslitterazione “americana”; Solženicyn in quella “scientifica”) in cirillico, Солженицын.

Non a tutte le lettere dell’alfabeto latino si può far corrispondere un carattere cirillico in base al suono; è il caso della Q, della W e della X. Neppure alla lettera C corrisponde un carattere cirillico, perché non sarebbe immediato ricordarsi quale lettera produca, la Ч o la Ц, poiché la traslitterazione basata sulla scrittura delle lingue slave che usano l’alfabeto latino riserva la “C” latina per il secondo suono. Questo vale sia per il maiuscolo sia per il minuscolo.

L’alfabeto completo della lingua russa si ha con:

`{\cyr A B V G D E E0 Zh Z I I0
K L M N O P R S T U F Kh Ts Ch
Sh Shch P2 Y P1 E1 Yu}`

А Б В Г Д Е Ё Ж З И Й К Л
М Н О П Р С Т У Ф Х Ц Ч Ш
Щ Ъ Ы Ь Э Ю

Le minuscole si fanno allo stesso modo:

`{\cyr a b v g d e e0 zh z i i0
k l m n o p r s t u f kh ts ch
sh shch p2 y p1 e1 yu}`

а б в г д е ё ж з и й к л м н о п
р с т у ф х ц ч ш щ ъ ы ь э ю

19.3 CARATTERI SPECIALI

L’alfabeto cirillico è ricco di simboli speciali, che si possono ottenere semplicemente accostando nel sorgente differenti caratteri uno dopo l’altro: per esempio `\textcyr{i0}` produce la “i” breve й, mentre il nome (spesso storpiato) del matematico Chebyshev (nella traslitterazione americana; Čebyšev in quella scientifica) si ottiene con

`\textcyr{Pafnutii0 Lp1vovich
Chebyshe0v}`

Пафнутий Львович Чебышёв

Basta un po’ di pratica per imparare a scrivere rapidamente.

Talvolta bisogna evitare queste “legature”: la parola советский (“sovietico”), per esempio, si ottiene con `\textcyr{sovet\skii0}`, dove `\` si usa per spezzare la legatura `ts` (si veda la tabella 60 a fronte).

Tabella 60: Principali caratteri cirillici. Nelle legature, il simbolo 0 è la cifra zero.
Per spezzare la legatura ts si usa \/.

Input	Output	Input	Output	Input	Output	Input	Output
A	А	B	Б	C		D	Д
E	Е	F	Ф	G	Г	H	Х
I	И	J	Ј	K	К	L	Л
M	М	N	Н	O	О	P	П
Q		R	Р	S	С	T	Т
U	У	V	В	W		X	
Y	Ы	Z	З				
a	а	b	б	c		d	д
e	е	f	ф	g	г	h	х
i	и	j	ј	k	к	l	л
m	м	n	н	o	о	p	п
q		r	р	s	с	t	т
u	у	v	в	w		x	
y	ы	z	з				
C1	Ѓ	Ch	Ч	D1	Ђ	D2	Џ
D3	Ѕ	Dj	Ђ	E0	Ё	E1	Э
E2	Є	I0	Й	I1	І	J1	Ј
J2	ЈО	Kh	Х	L1	Љ	Lj	Љ
N0	Њ	N1	Њ	Nj	Њ	P1	Ь
P2	Ђ	Sh	Ш	Shch	Ш	Ts	Ц
Ya	Ј	Yu	ЈО	Z1	Ж	Zh	Ж
c1	ђ	ch	ч	d1	ђ	d2	џ
e3	ѕ	dj	ђ	e0	ё	e1	э
e2	є	i0	й	i1	і	j1	ј
j2	јо	kh	х	l1	љ	lj	љ
		n1	њ	nj	њ	p1	ь
p2	ђ	sh	ш	shch	ш	ts	ц
ya	ј	yu	јо	z1	ж	zh	ж
\'C	Ѐ	\'G	Ґ	\'K	Ѓ	\u{U}	Ў
\'c	ё	\'g	ґ	\'k	ќ	\u{u}	ў
[[«]]	»	<	«	>	»
\#	Ђ	+	Ѓ				

20 | EBRAICO E ARABO

Questo capitolo, rivolto agli utenti che devono riportare brevi tratti di testo in ebraico o in arabo, presenta i pacchetti `cjhebrew` e `arabtex`, che permettono di comporre un testo con questi alfabeti.

20.1 SCRIVERE IN EBRAICO

20.1.1 Introduzione

L'alfabeto ebraico si usa per scrivere in lingua ebraica, in yiddish e in altre lingue parlate dagli Ebrei nel mondo. Si scrive da destra a sinistra ed è formato da 22 consonanti e due semiconsonanti, **ו** (*vav*) e **י** (*yod*). Non ci sono lettere per le vocali, che si indicano con segni speciali scritti sotto le consonanti. Certe consonanti hanno una forma diversa se sono scritte in fine di parola. Le consonanti **א** (*alef*) e **ע** (*ayin*) non hanno un proprio suono, ma servono da "appoggio" per la vocale successiva. La pronuncia di alcune consonanti si modifica a seconda della presenza di determinate vocali.

Il pacchetto `cjhebrew`, scritto da Christian Justen e preinstallato nelle moderne distribuzioni di \LaTeX , si carica semplicemente con

```
\usepackage{cjhebrew}
```

Il pacchetto definisce il comando `\cjRL`, che permette di scrivere, all'interno di un testo in caratteri latini, delle parole ebraiche nel verso corretto (da destra a sinistra, *Right to Left*): l'istruzione `\cjRL{'bgd}` produce **אבגד**. C'è anche la forma abbreviata `\<>`: le istruzioni `\<'bgd>` e `\cjRL{'bgd}` danno lo stesso risultato. Per scrivere un intero brano in ebraico si usa l'ambiente `cjhebrew`.

20.1.2 Consonanti

La tabella 61 mostra come scrivere le consonanti nel sorgente. Le lettere che si trovano alla fine di una parola sono scritte automaticamente nella forma corretta: `\<mlk>` produce **מלך**. A volte, però, si può voler inserire una lettera nella sua forma "finale" in un punto dove essa non sarebbe scritta automaticamente in quel modo, per esempio nel mezzo di una parola. A tal fine si usano i comandi riportati nella tabella 61. In alternativa, si mette

Tabella 61: Codifica delle consonanti ebraiche

א	ב	ג	ד	ה	ו	ז	ח	ט	י	כ	ך	ל	מ	ם
'	b	g	d	h	w	z	.h	.t	y	k	K	l	m	M
נ	ן	ס	ע	פ	ף	צ	ץ	ק	ר	ש	שׁ	שׂ	ת	
n	N	s	'	p	P	.s	.S	q	r	/s	,s	+s	t	

Tabella 62: Vocali ebraiche e altri simboli

◌ִ	◌ֵ	◌ֶ	◌ִּ	◌ֵּ	◌ֶּ	◌ִּֿ	◌ֵּֿ	◌ֶּֿ	◌ִּֿֿ	◌ֵּֿֿ	◌ֶּֿֿ	◌ִּֿֿֿ	◌ֵּֿֿֿ	◌ֶּֿֿֿ	◌ִּֿֿֿֿ	◌ֵּֿֿֿֿ	◌ֶּֿֿֿֿ
i	e	E	E:	a	/a	a:	A	A:	o	u	*	:	0	U			
:	-	◌ִּֿֿֿֿ															
;	-	\dottedcircle															

un punto esclamativo dopo la consonante. Per esempio, una ם (*mem*) finale si ottiene con \<M> o con \<m!>. Se invece si vuole evitare la sostituzione automatica di una lettera nella sua forma finale, basta scrivere | dopo la lettera: \<m|> produce una *mem* “normale”, ם.

20.1.3 Vocali e altri simboli

La tabella 62 mostra come scrivere le vocali nel sorgente. Le vocali si scrivono *dopo* la relativa consonante (אֱלֹהִים si ottiene con \<'E:lohiym>). L'unica eccezione è il cosiddetto *patah furtivum*, come in רוּחַ (\<rU/a.h>). Il *dageš* si ottiene con *, che si scrive *dopo* la relativa consonante (\<b*:> produce בָּ).

20.1.4 Un esempio

Questo è l'inizio della Bibbia:

בְּרֵאשִׁית בָּרָא אֱלֹהִים אֶת הַשָּׁמַיִם וְאֶת הָאָרֶץ: וְהָאָרֶץ הָיְתָה תֹהוּ וָבֹהוּ
וְהָשָׁךְ אֶל־פְּנֵי תְהוֹם וְרוּחַ אֱלֹהִים מְרַחֶפֶת אֶל־פְּנֵי הַמַּיִם:

L'esempio precedente è stato ottenuto con il codice:

```
\begin{cjhebrew}
b*:re'+siyt b*ArA' 'E:lohiym 'et ha+s*Amaysim w:'et hA'ArE.s; w:hA'ArE.s
hAy:tAh tohU wAbohU w:.ho+sEk: 'al--p*:ney t:hOm w:rU/a.h 'E:lohiym
m:ra.hEpEt 'al--p*:ney ham*Ayim;
\end{cjhebrew}
```

20.2 SCRIVERE IN ARABO

20.2.1 Introduzione

Le versioni moderne dell'arabo oggi in uso in vari paesi del mondo derivano da un primo alfabeto comune, ma hanno subito nei secoli diverse modifiche, sia nella grafia che nella pronuncia.

L'alfabeto arabo si scrive da destra a sinistra ed è composto da 28 lettere di base più la *hamza*, cui corrisponde un solo segno grafico (e che a volte è posizionata su una *waw*, *ya* o *alif*). Adattamenti della lingua scritta per altri idiomi, come il persiano e l'urdu, hanno ulteriori lettere.

La scrittura araba è corsiva, e tutte le lettere fondamentali hanno forme variabili, a seconda che siano scritte all'inizio, nel mezzo o alla fine di una parola.

Tabella 63: Codifica delle consonanti arabe

ي	ن	گ	ق	ع	ض	س	ذ	ح	ت	أ
y	n	g	q	'	.d	s	_d	.h	t	a
ى	ه	ل	ف	غ	ط	ش	ر	خ	ث	ب
_A	h	l	v	.g	.t	^s	r	_h	_t	b
ة	و	م	ك	ف	ظ	ص	ز	د	ج	پ
T	w	m	k	f	.z	.s	z	d	^g	p

Il pacchetto `arabtex` definisce il comando `\RL`, che permette di scrivere, all'interno di un testo in caratteri latini, delle parole in arabo nel verso corretto (da destra a sinistra, *Right to Left*). Per scrivere un intero brano in arabo si usa l'ambiente `RLtext`. Il pacchetto (se ne veda la documentazione) permette anche di traslitterare facilmente le parole arabe in caratteri latini.

20.2.2 Consonanti

La tabella 63 mostra come scrivere le consonanti nel sorgente. L'alfabeto arabo è un alfabeto consonantico e quindi il lettore deve conoscere la lingua per ricostruire le vocali (le vocali brevi non sono scritte, mentre quelle lunghe lo sono).

20.2.3 Un esempio

Ecco un esempio di brano in arabo.

صَدِيقُ مُحَمَّدٍ يَطْلُبُ مِنْهُ حِمَارَهُ لِيَرْكَبَهُ فِي سَفَرَةٍ قَصِيرَةٍ وَقَالَ لَهُ: سَوْفَ فِي
الْمَسَاءِ لَكَ فَقَالَ مُحَمَّدًا: جِدًّا لَا لَكَ رَغَبَتِكَ، فَالْحِمَارُ لَيْسَ هُنَا الْيَوْمَ. وَقَبْلَ
يَتِمُّ مُحَمَّدًا كَلَامَهُ بَدَّ الْحِمَارُ يَتَهَقُّ فِي اصْطَبِيلِهِ. فَقَالَ لَهُ صَدِيقُهُ: حِمَارَكَ يَا مُحَمَّدًا
يَتَهَقُّ. فَقَالَ لَهُ مُحَمَّدًا: غَرِيبٌ يَا صَدِيقِي! الْحِمَارُ وَتُكَذِّبُنِي؟

L'esempio precedente è stato ottenuto con il codice:

```
\begin{RLtext}
'at_A .sadIquN 'il_A ^gu.hA ya.tlubu minhu .himArahu li-yarkabahu fI
safraTiN qa.sIraTiN wa-qAla lahu: sawfa 'u'Iduhu 'ilayka fI al-masA'i,
wa-'adfa'u laka 'u^graTaN. fa-qAla ^gu.hA: 'anA 'AsifuN ^giddaN 'annI
lA 'asta.tI'u 'an 'u.haqqiqa laka ra gbataka, fa-al.himAru laysa hunA
al-yawma. wa-qabla 'an yutimmu ^gu.hA kaLAmahu bada'a al-.himAru yanhaqu
fI i.s.tablihi. fa-qAla lahu .sadIquhu: 'innI 'asma'u .himAraka yA
^gu.hA yanhaqu. fa-qAla lahu ^gu.hA: .garIbuN 'amruka yA .sadIqi!
'a-tu.saddiqu al-.himAra wa-tuka_d_dibunI?
\end{RLtext}
```


Parte IX

SPECIALITÀ

Questo capitolo presenta il pacchetto `lettrine`, che permette di inserire capilettera con \LaTeX .

21.1 INTRODUZIONE

LE CONVENZIONI tipografiche permettono, a scopo decorativo, di sostituire la prima lettera di un capitolo con un capolettera, cioè una lettera di corpo maggiore delle altre (come qui). I capilettera hanno le forme più svariate: vanno dai caratteri nel font in uso a quelli elaboratissimi disegnati appositamente per pubblicazioni particolarmente ricercate.

Ne offre un vasto assortimento il pacchetto `lettrine`, che per funzionare al meglio richiede a propria volta di caricare *prima* di `fontenc` con l'opzione `T1` il pacchetto `type1ec` (per scalare “a piacimento” i font standard di \LaTeX disegnati in vari corpi; usando font disegnati in un solo corpo come per esempio quelli dei pacchetti `txfonts`, `pxfonts`, `mathpazo` e `fourier`, invece, `type1ec` non serve):

```
\usepackage{type1ec}
\usepackage[T1]{fontenc}
\usepackage{lettrine}
```

Il pacchetto `lettrine` definisce il comando `\lettrine`, che ha due argomenti obbligatori e uno opzionale:

```
\lettrine[<opzioni>]{<capolettera>}{<testo in maiuscoletto>}
```

Se non si specifica diversamente nelle *<opzioni>*, il comando produce un *<capolettera>* dell'altezza di due righe, seguito da un *<testo in maiuscoletto>*, mentre il resto del capoverso è avvolto attorno al capolettera. Le *<opzioni>* permettono di controllare il formato e la posizione del capolettera; ogni opzione è costituita da una voce del tipo *<chiave>=<valore>*.

```
\lettrine{E}{cco un primo}
esempio che mostra come si può
ottenere un capolettera con
un uso minimo di codice.
```

ECCO UN PRIMO esempio che mostra come si può ottenere un capolettera con un uso minimo di codice.

I caratteri in maiuscoletto che compaiono dopo il capolettera sono una consuetudine spesso usata in tipografia, anche se non costituiscono una regola ferrea.

21.2 PERSONALIZZARE IL CAPOLETTERA

Il pacchetto `lettrine` fornisce diversi strumenti con cui personalizzare l'aspetto del capolettera. Di seguito se ne presentano alcuni.

21.2.1 Dimensione del capolettera

La chiave `lines` definisce l'altezza del capolettera in termini di numero di righe: `lines=<numero intero>` produce un capolettera con altezza pari al `<numero intero>` di righe specificato (il valore predefinito è 2).

```
\lettrine[lines=3]{U}{n} secondo
esempio dell'uso del pacchetto,
questa volta con un capolettera
dell'altezza di tre righe.
Spesso un capolettera di grandi
dimensioni è gradevole, ma
bisogna avere un capoverso
sufficientemente lungo, in modo
che il testo riesca ad avvolgere
il capolettera.
```

UN secondo esempio dell'uso del pacchetto, questa volta con un capolettera dell'altezza di tre righe. Spesso un capolettera di grandi dimensioni è gradevole, ma bisogna avere un capoverso sufficientemente lungo, in modo che il testo riesca ad avvolgere il capolettera.

Con `lines=1` si ottiene l'effetto seguente:

```
\lettrine[lines=1]{C}{ome si
osserva}, si ottiene un
capolettera alto due righe,
ma con la base allineata con
quella della prima riga del
capoverso.
```

COME SI OSSERVA, si ottiene un capolettera alto due righe, ma con la base allineata con quella della prima riga del capoverso.

21.2.2 Capolettera che sporge dal margine

La chiave `lhang` stabilisce quanto un capolettera debba sporgere dal margine sinistro. La sintassi è `lhang=<numero decimale>`. Il `<numero decimale>` della chiave `lhang` stabilisce la frazione di capolettera che deve fuoriuscire nel margine. Per esempio, ponendo `lhang=0.5` il capolettera sporge per metà dal margine, mentre con `lhang=1` il capolettera fuoriesce interamente dal margine (il valore predefinito 0 lascia il capolettera completamente nel testo).

```
\lettrine[lhang=.3,lines=3]{O}{ppure} si può far
fuoriuscire il capolettera
dal margine sinistro. In questo
caso il capolettera sporge dal
margine del~30\%$.
```

OPPURE si può far fuoriuscire il capolettera dal margine sinistro. In questo caso il capolettera sporge dal margine del 30%.

```
\lettrine[lhang=1,lines=3]{I}{n questo} esempio il
capolettera fuoriesce
completamente dal margine.
È un effetto gradevole
soprattutto con lettere
strette e lunghe come la~I
o la~J. In più, così non si
pone il problema della
lunghezza del capoverso.
```

IN QUESTO esempio il capolettera fuoriesce completamente dal margine. È un effetto gradevole soprattutto con lettere strette e lunghe come la I o la J. In più, così non si pone il problema della lunghezza del capoverso.

21.2.3 Dilatare il capolettera

L'opzione `loversize=<numero decimale>` dilata il capolettera della frazione definita da `<numero decimale>` (il valore predefinito è 0). Per esempio:

```
\lettrine[loversize=.6]%
{A}{desso} un esempio che
mostra il capolettera-A con
un'altezza aumentata del~$60\%$
rispetto al numero di righe
dato, in modo che sporga dalla
riga superiore del capoverso.
Il pacchetto gestisce gli spazi
in modo da non interferire con
il capoverso precedente.
```

ADESSO un esempio che mostra il capolettera A con un'altezza aumentata del 60% rispetto al numero di righe dato, in modo che sporga dalla riga superiore del capoverso. Il pacchetto gestisce gli spazi in modo da non interferire con il capoverso precedente.

21.2.4 Spostare verticalmente il capolettera

L'opzione `lraise=<numero decimale>` sposta il capolettera verso l'alto se il `<numero decimale>` è positivo, o verso il basso se è negativo (il valore predefinito è 0). L'opzione, che non influisce sull'altezza del capolettera, è indispensabile con lettere come la Q, infatti non usandola si ottiene:

```
\lettrine[lines=4]{Q}{uando}
si ha a che fare con lettere
come la-Q, bisogna traslare il
capolettera verso l'alto per
evitare una brutta
sovrapposizione del capolettera
con il testo del capoverso.
```

QUANDO si ha a che fare con lettere come la Q, bisogna traslare il capolettera verso l'alto per evitare una brutta sovrapposizione del capolettera con il testo del capoverso.

Poiché agendo sulla chiave `lraise` l'altezza del capolettera resta invariata, se si vuole che il capolettera non sporga dalla riga superiore del capoverso bisogna anche riscalarlo usando valori negativi di `loversize`. Di seguito si ripropone l'esempio precedente, con l'aggiustamento:

```
\lettrine[lines=5,%
loversize=-0.2,lraise=0.2]%
{Q}{uando} si ha a che fare con
lettere come la-Q, bisogna
traslare il capolettera verso
l'alto per evitare una brutta
sovrapposizione del capolettera
con il testo del capoverso.
```

QUANDO si ha a che fare con lettere come la Q, bisogna traslare il capolettera verso l'alto per evitare una brutta sovrapposizione del capolettera con il testo del capoverso.

21.2.5 Rientrare le righe del capoverso

Ci sono tre opzioni che permettono di controllare i rientri delle righe del capoverso del capolettera.

- `findent=<dimensione>` imposta lo spazio orizzontale presente fra il capolettera e il testo del capoverso che lo avvolge; la `<dimensione>` può essere positiva o negativa (il valore predefinito è 0);

- `nindent=<dimensione>` rientra della quantità *<dimensione>* (positiva o negativa) tutte le righe tranne la prima (il valore predefinito è 0.5em).
- `slope=<dimensione>` si può usare con lettere come la A o la V per aggiungere un rientro pari a *<dimensione>* (positiva o negativa) a ogni riga a partire dalla terza (il valore predefinito è 0; l'opzione non ha effetto se `lines=2`).

```
\lettrine[lines=4,%
slope=0.6em,findent=-1em]%
{A}{anche se} il codice è
semplice, gli esempi aiutano.
Qui il capolettera è una-A
maiuscola alta quattro righe, con
l'aggiunta di ~$0.6$-em di rientro
a ogni riga, a partire dalla
terza. Bisogna avvicinare il
testo al capolettera (il lettore
provi senza la correzione).
```

ANCHE SE il codice è semplice, gli esempi aiutano. Qui il capolettera è una A maiuscola alta quattro righe, con l'aggiunta di 0.6 em di rientro a ogni riga, a partire dalla terza. Bisogna avvicinare il testo al capolettera (il lettore provi senza la correzione).

```
\lettrine[lines=4,%
slope=-0.5em,lhang=0.5,%
nindent=0pt,loversize=.1]%
{V}{eramente} non è facile
trovare un capoverso che inizi
con la-V. Gli effetti più
elaborati si ottengono come in
questo esempio combinando le
opzioni. Qui il capolettera
sporge per metà dal margine
sinistro e il testo si "incastra"
con il contorno della lettera.
```

VERAMENTE non è facile trovare un capoverso che inizi con la V. Gli effetti più elaborati si ottengono come in questo esempio combinando le opzioni. Qui il capolettera sporge per metà dal margine sinistro e il testo si "incastra" con il contorno della lettera.

21.2.6 Testo prima di un capolettera

L'opzione `ante=<testo>` stampa del testo prima del capolettera. È utile soprattutto per inserire le virgolette di un dialogo o di una citazione.

```
\lettrine[ante=«»,lhang=.3]%
{E}{cco} un esempio. Anche se
prevedibilmente l'opzione verrà
usata di rado, tuttavia può
servire per ottenere effetti
particolari.»
```

«**E**CCO un esempio. Anche se prevedibilmente l'opzione verrà usata di rado, tuttavia può servire per ottenere effetti particolari.»

21.2.7 Formato del capolettera

Il comando `\LettrineTextFont` stabilisce il font usato nel secondo argomento obbligatorio di `\lettrine`. La sua definizione iniziale è:

```
\newcommand{\LettrineTextFont}{\scshape}
```

Esso può essere ridefinito con il comando standard `\renewcommand`.

Il comando `\LettrineFont` stabilisce invece il font impiegato per stampare il capolettera. In mancanza di istruzioni specifiche, si usa il font corrente

con il corpo opportunamente ingrandito. Il comando `\LettrineFontHook` permette di cambiare il font usato per il capolettera.

```
\renewcommand{\LettrineFontHook}
{\fontfamily{calligra}}
\renewcommand
{\LettrineTextFont}{\itshape}

\lettrine[findent=1.5em]{L}{a
possibilità} di usare font
inusuali ed elaborati per
il capolettera può essere
utile: in particolare, i font
calligrafici danno risultati
piuttosto eleganti.
```

*L*a possibilità di usare font inusuali ed elaborati per il capolettera può essere utile: in particolare, i font calligrafici danno risultati piuttosto eleganti.

Se si desidera che la ridefinizione dei font sia solo locale, basta scrivere il codice all'interno di due parentesi graffe.

Il comando `\LettrineFontHook` permette di avere un capolettera in grigio (serve il pacchetto `xcolor`).

```
\renewcommand%
{\LettrineFontHook}%
{\color{gray}{0.5}}

\lettrine[lines=4]{S}{e il
capolettera} è di grandi
dimensioni, può essere
bene "alleggerirlo" scrivendolo
in grigio: l'effetto estetico
che ne risulta è
particolarmente gradevole.
```

*S*E IL CAPOLETTERA è di grandi dimensioni, può essere bene "alleggerirlo" scrivendolo in grigio: l'effetto estetico che ne risulta è particolarmente gradevole.

21.2.8 Usare immagini come capilettera

Il comando `\LettrineFontEPS` permette, a dispetto del nome, di usare come capilettera immagini PDF, JPG e PNG, oltre che EPS. Si usa per ridefinire il comando `\LettrineFont`.

```
\renewcommand{\LettrineFont}%
{\LettrineFontEPS}

\lettrine[lines=7,nindent=0pt,
findent=2pt,loversize=0.2,%
lraise=-0.05,lhang=0.1]%
{D}{avvero} originale, questo
capolettera. L'iniziale gotica
usata in questo esempio,
contenuta in un file a sé
chiamato \texttt{D.pdf}, è stata
disegnata da Yannis Haralambous.
```



AVVERO originale, questo capolettera. L'iniziale gotica usata in questo esempio, contenuta in un file a sé chiamato `D.pdf`, è stata disegnata da Yannis Haralambous.

Bisogna che sia caricato il pacchetto `graphicx` e che l'immagine si trovi in una cartella cercata da \LaTeX (la cartella di lavoro, per esempio).


21.3 SCRIVERE IN GOTICO

Si dicono *gotiche* alcune grafie dell'alfabeto latino, diffuse anticamente nei paesi di lingua tedesca. Le lettere, caratterizzate da una minore spaziatura rispetto al "tondo", sono rimarcate da spessi tratti. L'effetto che ne risulta è quello di una scrittura alta e spigolosa, elegante, ma scura e di non agevole lettura. Una delle grafie gotiche più diffusa è la *Fraktur*, parola tedesca che deriva dal latino *fractus* ("rotto").

Per scrivere in gotico si può usare il pacchetto `yfonts`, che definisce il comando `\yinipar`; quest'ultimo, analogo a `\lettrine`, permette di inserire un capolettera.

```
\large\fraklines\frakfamily
```

```
\yinipar{G}eometrie kann  
lesbare Buchstaben  
hervorbringen, aber einzig die  
Kunst verleiht ihnen Sch"onheit.  
Die Kunst beginnt, wo die  
Geometrie aufh"ort, und verleiht  
den Buchstaben einen Charakter,  
der nicht messbar ist.
```



eometrie kann lesbare
Buchstaben hervorbringen,
aber einzig die Kunst
verleiht ihnen Schönheit.
Die Kunst beginnt, wo die Geometrie
aufhört, und verleiht den Buchstaben
einen Charakter, der nicht messbar ist.

Per maggiori dettagli si rimanda alla documentazione di `yfonts`.

Questo capitolo presenta alcuni strumenti per scrivere un documento utilizzando lo stile ClassicThesis, di André Miede, ispirato all'opera di Robert Bringhurst *Gli Elementi dello Stile Tipografico* [1992].

22.1 BASI

Questo paragrafo fornisce una visione d'insieme di ClassicThesis e ne descrive le caratteristiche fondamentali e le peculiarità.

22.1.1 Storia e filosofia

La *suite* ClassicThesis, sviluppata per \LaTeX da André Miede, è composta da un pacchetto, ClassicThesis, e da un modello di tesi pronto per l'uso. ClassicThesis ha due obiettivi:

- fornire un modello, pronto per l'uso, di tesi di laurea o di dottorato (usabile anche per libri e relazioni);
- fornire uno stile “classico” e di alta qualità, ispirato al capolavoro di Robert Bringhurst *Gli Elementi dello Stile Tipografico* [1992], cui si rimanda per gli approfondimenti.

La prima versione della *suite* è stata pubblicata nel 2006. La versione attuale è stabile e collaudata: sono stati composti con ClassicThesis numerosi documenti, e lo stile è diffuso in tutto il mondo.

La *suite* fa parte delle più diffuse distribuzioni di \LaTeX (\TeX Live e \MiKTeX *complete*) e usa font liberamente disponibili.

IMPORTANTE Alcuni aspetti di questo stile possono sembrare insoliti, a una prima occhiata. Tuttavia, sono caratteristiche appositamente studiate.

- Lo stile non usa il neretto: il corsivo o il maiuscoletto lo sostituiscono egregiamente.
- I margini ampi sono voluti. Così vengono prodotti documenti eleganti e leggibili. E, no: le righe *non* sono troppo corte.
- Le tabelle non usano linee verticali. A questo proposito si consiglia di leggere la documentazione del pacchetto booktabs.
- Negli indici, i numeri di pagina si trovano accanto alle relative voci. Sì, i numeri *non* sono allineati a destra e *non* sono collegati alle rispettive voci con punti che guidano l'occhio lungo una distanza non necessaria. Per chi non fosse ancora convinto: il lettore vuole sapere il numero di pagina o deve sommare quei numeri?

Se si modificano queste impostazioni si rischia di distruggere la bellezza e l'armonia dello stile, per cui si consiglia di lasciarle invariate, a meno che non si sappia *davvero* che cosa si sta facendo.

22.1.2 Uso

ClassicThesis si carica semplicemente con

```
\documentclass[...]{scrreprt} % classe report di KOMA-Script
%\documentclass[...]{scrbook} % classe book di KOMA-Script
\usepackage{...}
\usepackage[...]{classicthesis}

\input{classicthesis-config}

\begin{document}
...
\end{document}
```

oppure con

```
\documentclass[...]{scrartcl} % classe article di KOMA-Script
\usepackage{...}
\usepackage[...]{classicthesis}

\input{classicthesis-config}

\begin{document}
...
\end{document}
```

Se si sta scrivendo una tesi, si consiglia di modificare uno alla volta i file del modello: solo dopo essersi assicurati che tutto funzioni, si può eventualmente personalizzare il proprio documento.

22.1.3 Organizzazione del lavoro

Un fattore importante per il successo nella scrittura di una tesi (ma anche di un libro o di una relazione) è l'organizzazione del proprio materiale. Si suggerisce di adottare una struttura come la seguente:

- La cartella Capitoli contiene il materiale principale, suddiviso in capitoli, come Introduzione.tex, Basi.tex, e così via.
- La cartella MaterialeInizialeFinale contiene il materiale iniziale e finale, come i ringraziamenti, la dedica, la bibliografia.
- La cartella Immagini contiene tutte le figure usate nella tesi. Se si hanno molte figure, può essere conveniente organizzare la cartella in sottocartelle.
- Bibliografia.bib: la base di dati di biblatex che contiene i riferimenti bibliografici del documento.
- ClassicThesis.tex (o un file .tex dal nome appropriato): il file principale della tesi, che richiama tutti gli altri.

- `classicthesis-config.tex`: file che contiene le impostazioni dei pacchetti, i comandi e gli ambienti personali; deve essere caricato *dopo* `classicthesis`.

Questo piccolo accorgimento permette di semplificare notevolmente il proprio lavoro.

22.1.4 Opzioni

Il pacchetto `classicthesis` ha alcune opzioni che permettono di modificare (moderatamente) l'aspetto dei propri documenti:

- Impostazioni generali:
 - `drafting` Stampa la data e l'ora in fondo a ogni pagina del documento, così si ha sempre sott'occhio la versione a cui si sta lavorando.
- Parti e capitoli:
 - `parts` Va specificata se il documento è suddiviso in parti. L'opzione non può essere usata assieme a `nochapters`.
 - `nochapters` Va data se si usa `ClassicThesis` con una classe che non ha capitoli (come `scrartcl`). Disattiva automaticamente le opzioni `eulerchapternumbers`, `linedheaders`, `listsseparated` e `parts`.
 - `linedheaders` Cambia l'aspetto della pagina del titolo dei capitoli, aggiungendo una linea orizzontale sopra e sotto il titolo. Il numero del capitolo viene spostato in alto, sopra il titolo.
- Tipografia:
 - `eulerchapternumbers` Imposta i font \mathcal{AMS} Euler per i numeri dei capitoli. Il font predefinito è il Palatino.
 - `beramono` Imposta i font Bera Mono come font a larghezza fissa. Non c'è alcun font a larghezza fissa predefinito.
 - `eulermath` Imposta i font \mathcal{AMS} Euler per le formule matematiche. I font predefiniti per la matematica sono i Palatino.
 - `pdfspacing` Spazia i caratteri del maiuscoletto con il pacchetto `microtype` (scelta consigliata: si veda il paragrafo [22.2.2](#) a pagina [328](#)).
 - `minionprospacing` Spazia i caratteri del maiuscoletto con il pacchetto `MinionPro`. Abilita automaticamente l'opzione `minionpro` e annulla `pdfspacing`.
- Indici:
 - `tocaligned` Allinea a sinistra tutte le voci dell'indice.
 - `dottedtoc` Allinea a destra i numeri di pagina nell'indice generale.
 - `many chapters` Nell'indice generale, aumenta lo spazio tra il numero e il titolo dei capitoli (utile se i capitoli sono dieci o più).
- Oggetti mobili:

- listings Carica il pacchetto listings e configura di conseguenza l'elenco dei codici.
- floatperchapters ClassicThesis numera progressivamente le figure e le tabelle ("Figura 1", "Figura 2", e così via); in un libro o una relazione, floatperchapters ripristina la numerazione standard collegata al capitolo corrente ("Figura 1.1", "Figura 1.2", eccetera).
- subfig Attiva la compatibilità con il pacchetto omonimo.

Il modo migliore per capire come funzionano queste opzioni è provarle e scegliere quelle più gradite.

22.1.5 Impostazioni

A differenza del pacchetto classicthesis, che non va *mai* modificato, il file classicthesis-config si può personalizzare: per esempio, per impostare il titolo del documento e il nome dell'autore basta scrivere:

```
\newcommand{\myTitle}{L'arte di scrivere con ClassicThesis}
\newcommand{\mySubTitle}{Un omaggio agli Elementi di Bringham}
```

22.1.6 Non solo tesi

Si può usare ClassicThesis per scrivere articoli, relazioni e libri. Questo paragrafo presenta un paio di semplici esempi.

Un articolo

```
\documentclass[a4paper]{scrartcl} % Classe article di KOMA-Script

\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{lipsum}
\usepackage[nochapters,pdfspacing]{classicthesis}

\input{classicthesis-config}

\begin{document}
\title{Titolo}
\author{Tyler Durden}

\maketitle

\begin{abstract}
\lipsum[1]
\end{abstract}

\tableofcontents

\section{Un paragrafo}
\lipsum[2]

\subsection{Un sottoparagrafo}
\lipsum[3]
```

```
\section{Un altro paragrafo}
\lipsum[4]
\end{document}
```

Un libro

```
\documentclass[a4paper,cleardoubleempty]{scrbook}
% Classe book di KOMA-Script

\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{lipsum}
\usepackage[parts,pdfspacing]{classicthesis}

\input{classicthesis-config}

\begin{document}
\tableofcontents
\cleardoublepage
\part{Una parte}

\chapter{Un capitolo}
\lipsum[1]

\chapter{Un paragrafo}
\lipsum[2]

\cleardoublepage
\part{Un'altra parte}

\chapter{Un altro capitolo}
\lipsum[3]

\appendix
\chapter{Appendice}
\lipsum[4]
\end{document}
```

22.1.7 Comandi e ambienti

Con \LaTeX si fa un uso intensivo di comandi e ambienti. Questo paragrafo presenta alcuni comandi e ambienti specifici di ClassicThesis.

Tabella 64: Varianti di stile del font Palatino usate da ClassicThesis

Comando	Stile
<code>\textit</code>	<i>Corsivo</i>
<code>\textsc</code>	MAIUSCOLETTO
<code>\spacedlowsmallcaps</code>	MAIUSCOLETTO BASSO SPAZIATO
<code>\spacedallcaps</code>	MAIUSCOLETTO SPAZIATO

Tabella 65: Corrispondenza tra le classi KOMA-Script e le classi standard

Classe standard	Classe KOMA-Script
article	scrartcl
report	scrreprt
book	scrbook
letter	scrlettr

Elenchi numerati e descrizioni

Gli elenchi sono molto utili. Permettono infatti:

- A. di “dare respiro” al testo;
- B. di migliorarne la leggibilità;
- C. di strutturare le proprie idee.

L’elenco precedente, con le etichette in maiuscoletto, è stato ottenuto con l’ambiente `aenumerate` di `ClassicThesis`, analogo allo standard `enumerate`.

22.1.8 Note a margine

Le note a margine danno un tocco di vitalità alla pagina

Una nota a margine si ottiene semplicemente con il comando

```
\graffito{⟨...⟩}
```

Nei documenti fronte-retro, le note sono poste nel margine destro nelle pagine dispari, e nel margine sinistro nelle pagine pari. Nei documenti solo-fronte, le note sono sempre poste nel margine destro.

Comandi di cambiamento di stile

Di regola, \LaTeX sceglie il carattere appropriato in base alla struttura logica del documento (capitoli, paragrafi, testatine, eccetera). In alcuni casi, si potrebbe però voler cambiare lo stile dei caratteri, manualmente; per farlo con `ClassicThesis` si usano i comandi elencati nella tabella 64 nella pagina precedente.

22.1.9 Classi

\LaTeX offre quattro classi standard per la composizione di libri, relazioni, articoli e lettere: esse sono rispettivamente `book`, `report`, `article` e `letter`. `ClassicThesis`, però, funziona non con le usuali classi standard, ma con le classi KOMA-Script, scritte da Markhus Kohm (da cui il prefisso “Koma”).

Le classi KOMA-Script sono simili a quelle standard. Alle classi `book`, `report`, `article` e `letter` corrispondono rispettivamente le classi `scrbook`, `scrreprt`, `scrartcl` e `scrlettr` (tabella 65). I nomi di queste classi sono composti dal prefisso `scr` e dal nome abbreviato della corrispondente classe standard (per accorciare la lunghezza dei nomi a otto lettere).

Tabella 66: Opzioni più comuni delle classi KOMA-Script; i simboli ●, ◐ e ○ indicano rispettivamente che l'opzione è predefinita, applicabile (anche se non predefinita), non applicabile

Opzione	scrbook	scrreprt	scrartcl
11pt	●	●	●
a4paper	●	●	●
oneside	◐	●	●
twoside	●	◐	◐
openany	◐	●	○
openright	●	◐	○
headexclude	●	●	●
footexclude	●	●	●
cleardoublepage=plain	●	●	●
titlepage	●	●	◐
notitlepage	◐	◐	●
abstractoff	○	●	●
numbers=noenddot	●	●	●
final	●	●	●

Opzioni

Le classi KOMA-Script hanno diverse opzioni *globali*, che hanno cioè effetto sull'intero documento. La tabella 66 riporta le opzioni più comuni delle classi scrartcl, scrreprt e scrbook (la classe scrlettr fa storia a sé: se ne veda la documentazione), mostrando per ciascuna se l'opzione è predefinita, applicabile (anche se non predefinita) o non applicabile.

- 10pt, 11pt, 12pt Sono analoghe a quelle standard. Il valore predefinito è 11 punti.
- a4paper e a5paper. Sono analoghe a quelle standard. La dimensione predefinita è a4paper.
- oneside, twoside Sono analoghe a quelle standard.
- openany, openright Sono analoghe a quelle standard.
- BCOR<correzione> Lascia uno spazio uguale a <correzione> per la rilegatura del documento. Per la <correzione> si può usare una qualsiasi unità tipografica riconosciuta da L^AT_EX.
- headinclude, headexclude, footinclude, footexclude Per determinare i margini di pagina bisogna specificare se le testatine e i piè di pagina vanno considerati parte del corpo del testo (lo fanno headinclude e footinclude) o dei bordi (lo fanno headexclude e footexclude).
- <valore>headlines Per definire i margini di pagina bisogna anche specificare l'altezza delle testatine; lo si fa con l'opzione <valore>headlines, dove <valore> indica il numero di righe di ciascuna testatina. Il valore predefinito è 1.25, e va modificato solo se serve.
- cleardoublepage=empty, cleardoublepage=plain Se si vuole che le pagine vuote non abbiano né testatina né piè di pagina, con le classi standard bisogna provvedere a mano (caricando con il pacchetto

Tabella 67: Alcuni stili dei font Computer Modern

Dritto	Corsivo	Inclinato	Neretto	MAIUSCOLETTA	Sans serif	Typewriter
--------	---------	-----------	---------	--------------	------------	------------

emptypage, per esempio). Con le classi KOMA-Script, invece, l'opzione `cleardoublepage=empty` applica alle pagine vuote lo stile `empty`, mentre `cleardoublepage=plain` dà loro lo stile `plain`.

- `titlepage`, `notitlepage` Sono analoghe a quelle standard.
- `abstracton`, `abstractoff` Se si scrive un sommario (in un articolo o una relazione), con `abstracton` si antepone a esso l'intestazione centrata "Sommario", con `abstractoff` la si omette.
- `numbers=enddot`, `numbers=noenddot` L'opzione `numbers=enddot` fa seguire un punto alla numerazione di sezioni e didascalie, mentre con `numbers=noenddot` (scelta consigliata) non c'è alcun punto.
- `fleqn`, `leqno` Sono analoghe a quelle standard.
- `draft`, `final` Sono analoghe a quelle standard.

Stili di pagina

Le classi KOMA-Script, analogamente a quelle standard, accettano diverse combinazioni di testatina/piè di pagina (i cosiddetti *stili di pagina*). L'argomento `<stile>` del comando

```
\pagestyle{<stile>}
```

stabilisce quale stile sarà usato. Gli stili di pagina più diffusi delle classi di KOMA-Script sono i seguenti.

PLAIN Stampa i numeri di pagina al piè di pagina, lasciando vuota la testatina. È lo stile predefinito.

EMPTY Non stampa nulla nelle testatine e nei piè di pagina.

HEADINGS Stampa il titolo del capitolo corrente sulla testatina di ciascuna pagina e il numero di pagina al piè di pagina.

SCRHEADINGS Permette di personalizzare lo stile di pagina.

22.1.10 Font

La scelta dei font

In mancanza di istruzioni specifiche, L^AT_EX impiega i font Computer Modern (si veda la tabella 67), disegnati da Donald Knuth, e precisamente (si veda il paragrafo 13.2.1 a pagina 247):

- i Computer Modern Roman, con grazie;
- i Computer Modern Sans Serif, senza grazie;
- i Computer Modern Typewriter, a larghezza fissa;

- i Computer Modern Mathematics, per la matematica.

ClassicThesis, invece, usa i seguenti font:

- i Palatino, con grazie;
- i Bera Mono, a spaziatura fissa;
- i Palatino (o, in alternativa, gli \mathcal{AMS} Euler) per la matematica.

ClassicThesis non usa alcun font sans serif.



Figura 36: Alcuni font disegnati da Hermann Zapf

LA FAMIGLIA PALATINO I Palatino sono font con grazie disegnati da Hermann Zapf nel 1948. Equilibrati ed eleganti, sono tra i caratteri di ispirazione neumanista più apprezzati, diffusi e imitati.

Chiamati così in onore di Giambattista Palatino, un maestro della calligrafia italiano del sedicesimo secolo, i Palatino sono basati sui caratteri del Rinascimento italiano, che imitano la scrittura calligrafica. L'origine del nome è il *Mons Palatinus*, il colle Palatino a Roma, sito del tempio di Apollo e di molti palazzi imperiali.

I font Palatino sono stati adattati praticamente a tutte le tecnologie tipografiche; ne esistono versioni liberamente disponibili e commerciali (le fonderie Linotype e Adobe

commercializzano entrambe versioni autentiche del Palatino).

I Palatino sono i font usati per comporre le parti testuali di questo lavoro.

LA FAMIGLIA BERA MONO Bera Mono è una versione della famiglia di font Bitstream Vera Mono ottimizzata per \LaTeX , con una licenza d'uso libera. Disegnati da Jim Lyles della Bitstream, sono font attraenti e leggibili.

I Bera Mono sono i font usati per comporre i codici riportati in questo lavoro.

LA FAMIGLIA EULER Gli \mathcal{AMS} Euler sono font matematici commissionati dalla American Mathematical Society e disegnati da Hermann Zapf e Donald Knuth. Emulano la calligrafia di un matematico che scrive sulla lavagna, che è dritta, piuttosto che inclinata. Si mescolano bene con altri tipi di carattere disegnati da Zapf, come i Palatino, ma male con i Computer Modern (i font predefiniti di \LaTeX).

Il nome \mathcal{AMS} Euler è stato scelto in onore di Leonhard Euler. I font \mathcal{AMS} Euler sono stati usati per la prima volta nel libro *Concrete Mathematics* (1988), di cui Knuth fu co-autore, che era dedicato ad Euler.

I font \mathcal{AMS} Euler sono gradevoli ed eleganti. Tuttavia, si segnala che non sono corsivi (in contrasto con quanto prescrivono le norme ISO-UNI) e che

manca qualche carattere speciale. Se ciò costituisce un ostacolo, si possono usare i font Palatino per la matematica, che non hanno questi problemi.

LA FAMIGLIA MINION PRO I Minion Pro sono caratteri neumanisti disegnati da Robert Slimbach e commercializzati dalla Adobe dal 1989. Sono molto economici per quel che riguarda la composizione: permettono di inserire qualche carattere in più per riga, in ogni corpo, rispetto alla maggior parte degli altri caratteri da testo, senza apparire schiacciati o compressi. La famiglia comprende anche un font di ornamenti tipografici, caratteri con svolazzi e una serie cirillica. Il Minion Pro greco, tondo e corsivo, esiste in forma sperimentale.

I Minion Pro sono i font usati da Robert Bringhurst per comporre i suoi *Elementi dello Stile Tipografico*.



Figura 37: La famiglia Minion Pro

22.2 PERSONALIZZAZIONI

Presto o tardi tutti cercano di personalizzare particolari oggetti o l'intero documento. Questo capitolo dà alcuni suggerimenti su come fare in modo che ClassicThesis produca risultati diversi da quelli predefiniti.

22.2.1 Margini di pagina

Una delle lamentele più frequenti di chi si accosta a ClassicThesis è che i margini sarebbero troppo ampi e che il foglio non sarebbe sufficientemente ben riempito. Prima di buttarsi nella frenesia dell'«allarghiamo un po' questa strettissima pagina» è però bene riflettere. Come per la maggior parte delle cose in \LaTeX , ci sono ottime ragioni per cui i margini sono quelli che sono.

Un punto di forza di \LaTeX è che permette di disinteressarsi (quasi) completamente delle questioni tipografiche, per dar modo all'autore di concentrarsi sui contenuti del proprio documento: usando uno stile scritto da altri, l'utente accetta le impostazioni tipografiche scelte per lui dall'autore dello stile e non è più tenuto a studiare la tipografia per mettere a punto l'aspetto delle proprie pubblicazioni. Questo vale anche per ClassicThesis: modificarne i margini significa contraddire questa filosofia e, di conseguenza, comporta l'obbligo di studiare un (bel) po' di tipografia per ottenere risultati accettabili.

L'esperienza mostra che leggere diventa difficile se ci sono troppi caratteri per riga: l'occhio si affatica spostandosi dalla fine di una riga all'inizio della successiva (è il motivo per cui i giornali vengono stampati su più colonne). Bringhurst ha codificato questa esperienza nella celebre regola che considera

come ottimale il valore di circa 66 caratteri per riga, contando anche gli spazi, indipendentemente dal font usato.

La lunghezza di riga adottata da ClassicThesis, di suo, è leggermente superiore a quanto stabilito dalla regola di Bringham, per consentire un maggiore riempimento delle pagine in formato A4, su cui si assume avvenire la stampa (come di norma, in Europa, in ambiente universitario o casalingo).

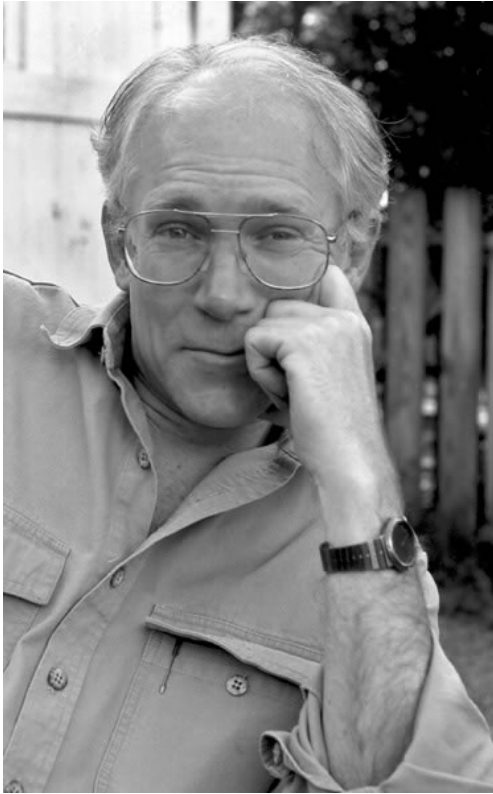


Figura 38: Robert Bringham

Gli eleganti margini ampi, inoltre, permettono l'impiego delle note a margine, utili per riassumere il testo e per dare un tocco di vitalità alla pagina, e lasciano spazio per eventuali annotazioni del proprio relatore (se si sta scrivendo una tesi).

Per queste ragioni *si sconsiglia* di modificare i margini predefiniti di ClassicThesis. Se l'utente è soddisfatto delle proporzioni di pagina di ClassicThesis e trova allettante l'idea di disinteressarsi del problema della loro definizione, allora potrà usare con soddisfazione ClassicThesis. Altrimenti, se si hanno esigenze diverse o non si è soddisfatti dalla resa grafica dello stile, allora ci si può rivolgere ad altre classi o pacchetti, o ci si può costruire da soli le dimensioni di pagina desiderate.

Se un utente necessita di margini differenti da quelle predefiniti di ClassicThesis, per esempio perché è obbligato a seguire delle indicazioni

imposte dalla propria università, va tenuto presente che le classi KOMAScript, su cui ClassicThesis si basa, hanno il proprio sistema per impostare i parametri tipografici: per determinare le proporzioni di pagina, esse usano il pacchetto typearea (se ne veda la documentazione), che ha una serie di impostazioni predefinite. Se serve, si possono impostare a mano le dimensioni del corpo del testo, mediante il comando:

```
\areaset[⟨rilegatura⟩]{⟨larghezza⟩}{⟨altezza⟩}
```

dove i parametri *⟨rilegatura⟩*, *⟨larghezza⟩* e *⟨altezza⟩* rappresentano rispettivamente la correzione per la rilegatura, la larghezza e l'altezza del corpo del testo. I valori predefiniti per una pagina A4 con il font Palatino in corpo 10 (e le opzioni di classe `headinclude` e `footinclude`) sono

```
\areaset{336pt}{750pt}
```

In alternativa al comando `\areaset` di typearea si possono usare i pacchetti standard `layaureo` e `geometry` (se ne veda la documentazione).

22.2.2 Microtipografia

Un titolo di sezione non dovrebbe *mai* andare a capo. Si risolvono tutti i problemi in una volta sola riformulandolo: si può fare praticamente sempre. Nelle prime versioni di ClassicThesis, la fuoriuscita di una riga dal margine destro nei titoli dei paragrafi e dei capitoli era abbastanza comune. Questi problemi erano dovuti al fatto che ClassicThesis usava il pacchetto soul per spaziare i caratteri del maiuscoletto: la spaziatura (formata da uno spazio fisso più un'eventuale correzione, la cosiddetta "crenatura") era realizzata "carattere per carattere" e non teneva conto di tutti i fattori necessari.

Questi problemi sono stati risolti dal pacchetto microtype, che realizza la spaziatura agendo direttamente al livello del font. Perché ciò avvenga, basta caricare classicthesis con l'opzione pdfspacing:

```
\usepackage[⟨...⟩,pdfspacing]{classicthesis}
```

22.2.3 Collegamenti ipertestuali

Supponiamo di avere un documento composto da un'introduzione e da diversi capitoli:

```
\documentclass{scrreprt}
\usepackage{lipsum}
\usepackage{classicthesis}

\begin{document}
\chapter{Introduzione}
\lipsum

\chapter{Pizza}
\lipsum

\chapter{Birra}
\lipsum
\end{document}
```

Se si desidera inserire nell'introduzione un'indicazione del tipo:

```
L'esposizione del lavoro è articolata come segue:
\begin{description}
\item[Il primo capitolo] parla di pizza.
\item[Il secondo capitolo] parla di birra.
\end{description}
```

in modo che cliccando su "primo capitolo" e "secondo capitolo" il lettore venga portato nelle pagine corrispondenti, si strutturi il documento così:

```
\documentclass{scrreprt}
\usepackage{lipsum}
\usepackage{classicthesis}

\begin{document}
\chapter{Introduzione}
\begin{description}
\item[Il primo capitolo] parla di pizza.
\item[Il secondo capitolo] parla di birra.
\end{description}
\end{document}
```

```

\chapter{Pizza}
\label{cap:pizza}
\lipsum

\chapter{Birra}
\label{cap:birra}
\lipsum
\end{document}

```

22.2.4 Matematica

Un punto di forza di \LaTeX è la sua capacità nella composizione delle formule matematiche. Se si sta scrivendo un lavoro scientifico, è bene caricare i pacchetti `amsmath` e `amssymb`, che ne semplificano la gestione.

Personalizzare lo stile degli enunciati

Il comando standard `\newtheorem` permette di introdurre e numerare definizioni, teoremi ed enunciati simili. Per personalizzarne lo stile il pacchetto `amsthm` definisce il comando `\newtheoremstyle`:

```

\newtheoremstyle{<nome dell'enunciato>}%
  {<spazio che precede l'enunciato>}%
  {<spazio che segue l'enunciato>}%
  {<stile del font dell'enunciato>}%
  {<rientro (se vuoto, non c'è rientro)>}%
  {<stile del font dell'intestazione>}%
  {<punteggiatura che segue l'intestazione>}%
  {<spazio che segue l'intestazione>}%
  {<intestazione dell'enunciato>}

```

la cui sintassi si spiega da sé.

Con `ClassicThesis`, per evidenziare adeguatamente definizioni e teoremi si può impiegare, al posto del neretto (che `ClassicThesis` non usa), il maiuscolletto per l'intestazione, accompagnandolo con una spaziatura prima e dopo l'enunciato, per “staccarlo” visivamente dal resto (si veda la figura 39 nella pagina successiva). Ecco il codice per farlo (serve `amsthm`):

```

\newtheoremstyle{classicdef}{12pt}{12pt}{\scshape}{:}{1em}{}
\newtheoremstyle{classicthm}{12pt}{12pt}{\itshape}{\scshape}{:}{1em}{}

\theoremstyle{classicdef}
\newtheorem{definizione}{Definizione}

\theoremstyle{classicthm}
\newtheorem{teorema}{Teorema}

```

Lettere greche minuscole

Alcune lettere greche minuscole hanno delle forme varianti i cui nomi cominciano con l'abbreviazione `var`; esse negli Stati Uniti sono delle varianti, ma in Europa (tranne `\varpi` e `\varsigma`, che non si usano praticamente mai) sono considerate le forme principali (si veda la tabella 68 a pagina 331).

Veniamo ora alle definizioni di continuità, derivata ed integrale per una funzione standard $f: {}^*(a, b) \rightarrow {}^*\mathbb{R}$.

DEFINIZIONE A.7 (continuità): Sia $x \in {}^*(a, b) \cap \mathbb{R}$; f si dice *continua* in x se

$$f(x + \varepsilon) \approx f(x) \quad \forall \varepsilon \approx 0, \quad (\text{A.10})$$

ovvero se $\text{st}[f(x + \varepsilon)] = f(x)$ per ogni $\varepsilon \approx 0$.

DEFINIZIONE A.8 (continuità uniforme): f è *continua uniformemente* in ${}^*(a, b)$ se $x', x'' \in {}^*(a, b)$ e $x' \approx x''$ implica $f(x') \approx f(x'')$.

DEFINIZIONE A.9 (derivata): Sia $x \in {}^*(a, b) \cap \mathbb{R}$; il numero reale $c \in \mathbb{R}$ si dice *derivata* di f in x e si scrive $c = f'(x)$ se

$$\frac{f(x + \varepsilon) - f(x)}{\varepsilon} \approx c \quad \forall \varepsilon \approx 0, \quad (\text{A.11})$$

ovvero se $\text{st}\left[\frac{f(x + \varepsilon) - f(x)}{\varepsilon}\right] = c$ per ogni $\varepsilon \approx 0$.

DEFINIZIONE A.10 (integrale): Ci limitiamo per semplicità a funzioni continue in ${}^*[a, b]$. Sia $\omega \in {}^*\mathbb{N} \setminus \mathbb{N}$ un intero infinito e poniamo $\alpha_k = a + \frac{k}{\omega}(b - a)$, $k = 1, 2, \dots, \omega$. I punti α_k costituiscono una suddivisione infinita di $[a, b]$, tale che $\alpha_k - \alpha_{k-1} = \frac{b-a}{\omega} \approx 0$. Poniamo

$$S_\omega(f) = \sum_{k=1}^{\omega} f(\alpha_k)(\alpha_k - \alpha_{k-1}) = \frac{b-a}{\omega} \sum_{k=1}^{\omega} f(\alpha_k). \quad (\text{A.12})$$

Si osservi che $S_\omega(f)$ è somma di infiniti termini infinitesimi. Il numero reale I si dice *integrale* di f in $[a, b]$ se

$$S_\omega(f) \approx I \quad \forall \omega \in {}^*\mathbb{N} \setminus \mathbb{N}, \quad (\text{A.13})$$

ovvero se $\text{st}[S_\omega(f)] = I$ per ogni $\omega \in {}^*\mathbb{N} \setminus \mathbb{N}$.

Come si vede l'operazione di parte standard sostituisce l'operazione di limite che qui non è necessaria.

Terminiamo questo flash sull'analisi non-standard con una dimostrazione allo scopo di dare un'idea, seppure minima, di ciò che Gödel intendeva con le parole riportate all'inizio. È la dimostrazione del seguente teorema.

TEOREMA A.2 (di Cantor-Heine): *Ogni funzione continua su un compatto è uniformemente continua.*

Dimostrazione (non-standard). Sia $K \subset {}^*\mathbb{R}$ compatto e $f: K \rightarrow {}^*\mathbb{R}$ continua. Se x' e $x'' \in K$ e $x' \approx x''$, sia $x = \text{st}(x') = \text{st}(x'')$. Allora $x \in K$ essendo K compatto ed essendo f continua si ha

$$f(x') \approx f(x) \approx f(x''), \quad (\text{A.14})$$

e il teorema è dimostrato. \square

Figura 39: Definizioni e teoremi dallo stile personalizzato

Tabella 68: Lettere greche minuscole: forme principali e varianti

(a) Font Palatino				(b) Font \mathcal{AMS} Euler			
ϵ	<code>\epsilon</code>	ε	<code>\varepsilon</code>	ϵ	<code>\epsilon</code>	ε	<code>\varepsilon</code>
θ	<code>\theta</code>	ϑ	<code>\vartheta</code>	θ	<code>\theta</code>	ϑ	<code>\vartheta</code>
π	<code>\pi</code>	ϖ	<code>\varpi</code>	π	<code>\pi</code>	ϖ	<code>\varpi</code>
ρ	<code>\rho</code>	ϱ	<code>\varrho</code>	ρ	<code>\rho</code>	ϱ	<code>\varrho</code>
σ	<code>\sigma</code>	ς	<code>\varsigma</code>	σ	<code>\sigma</code>	ς	<code>\varsigma</code>
ϕ	<code>\phi</code>	φ	<code>\varphi</code>	ϕ	<code>\phi</code>	φ	<code>\varphi</code>

(Si noti che con i font \mathcal{AMS} Euler, `\rho` e `\varrho` producono lo stesso risultato; lo stesso accade per `\sigma` e `\varsigma`. Poiché in un documento per ciascuna lettera si sceglie in alternativa la forma principale o la sua variante, non c'è pericolo di confondersi.)

Per scrivere documenti in accordo con il gusto europeo, è utile ridefinire le lettere greche varianti come normali. A tal fine, basta scrivere

```
\renewcommand{\epsilon}{\varepsilon}
\renewcommand{\theta}{\vartheta}
\renewcommand{\rho}{\varrho}
\renewcommand{\phi}{\varphi}
```

dopo aver caricato il pacchetto `classicthesis`. Il codice deve seguire il caricamento del pacchetto in quanto `ClassicThesis` si occupa di scegliere i font matematici, e questa azione definisce anche il significato delle lettere greche.

22.2.5 Appendici

Poiché `ClassicThesis` non scrive l'intestazione "Capitolo" né nei titoli dei capitoli né nelle testatine né nell'indice, si consiglia, se il proprio documento contiene delle appendici, di adottare lo stesso criterio. A tal fine, è sufficiente strutturare il proprio documento così:

```
\documentclass{scrreprt}
\usepackage{lipsum}
\usepackage{classicthesis}

\begin{document}
\tableofcontents

\chapter{Pizza}
\lipsum

\chapter{Birra}
\lipsum

\appendix
\chapter{Caffè}
\lipsum
\end{document}
```

22.2.6 Indice analitico

Generare l'indice analitico

Generare l'indice analitico con ClassicThesis è semplice: si scrive il codice

```
\manualmark
\markboth{\spacedlowsmallcaps{\indexname}}%
          {\spacedlowsmallcaps{\indexname}}
\phantomsection
\pagestyle{scrheadings}
\addcontentsline{toc}{chapter}{\tocEntry{\indexname}}
\printindex
```

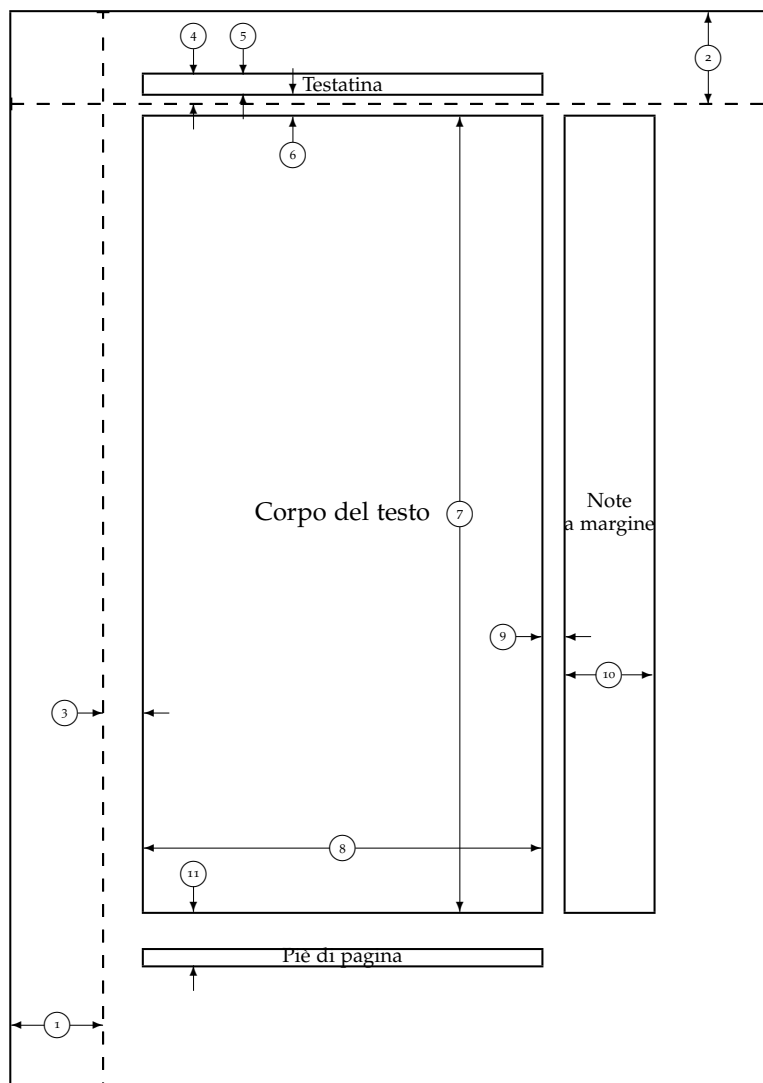
dove si desidera che compaia l'indice analitico (di solito, alla fine del documento, subito prima di `\end{document}`). Così, nel documento finito vengono creati il relativo segnalibro e le testatine. (Servono il pacchetto `makeidx` e il programma `MakeIndex`, e nel preambolo si deve dare `\makeindex.`)

Personalizzare l'indice analitico

L'indice analitico che si ottiene con le impostazioni predefinite non è molto elegante. Per migliorarne la resa tipografica si consiglia di seguire le indicazioni presentate nel paragrafo 12.2.6 a pagina 242.

22.3 PROPORZIONI DI PAGINA

ClassicThesis determina i margini di pagina secondo le direttive di Bringhurst [1992, p. 26]. Per esempio, la lunghezza dell'alfabeto latino minuscolo "abcdefghijklmnopqrstuvwxyz" del font Palatino in corpo 10 è di 133 punti. Ne segue che una buona lunghezza di riga è di 288-312 punti. Usando un corpo del testo "a doppio quadrato" con un rapporto tra larghezza e altezza di 1:2, il risultato è un blocco di 312:624 punti. Una valida alternativa può essere un corpo del testo con larghezza e altezza in sezione aurea, con un rapporto di $1:\varphi \simeq 1:1,62$, che conduce a dimensioni di 312:505 punti (si veda la figura 40 a fronte).



- | | |
|-------------------------|-------------------------------------|
| 1 un pollice + \hoffset | 2 un pollice + \voffset |
| 3 \oddsidemargin = 32pt | 4 \topmargin = -23pt |
| 5 \headheight = 15pt | 6 \headsep = 18pt |
| 7 \textheight = 624pt | 8 \textwidth = 312pt |
| 9 \marginparsep = 19pt | 10 \marginparwidth = 69pt |
| 11 \footskip = 42pt | \marginparpush = 5pt (non mostrato) |
| \hoffset = 0pt | \voffset = 0pt |
| \paperwidth = 597pt | \paperheight = 845pt |

Figura 40: Proporzioni di pagina di ClassicThesis

23 | ARSCLASSICA

Questo capitolo presenta le semplici nozioni che sono alla base del funzionamento dello stile ArsClassica e ne presenta le peculiarità.

23.1 INTRODUZIONE

Il pacchetto ArsClassica modifica alcuni aspetti tipografici di ClassicThesis (si veda il capitolo precedente). Riproduce la veste grafica di questa guida.

Il pacchetto, preinstallato nelle moderne distribuzioni di L^AT_EX, funziona con le classi KOMA-Script e richiede ClassicThesis (va caricato *dopo* quest'ultimo). Per esempio, questo documento è stato prodotto con il codice seguente:

```
\documentclass{scrbook}

\usepackage{...}
\usepackage[eulerchapternumbers,beramono,pdfspacing]{classicthesis}
\usepackage{arsclassica}

\begin{document}
...
\end{document}
```

Si raccomanda di usare le opzioni `eulerchapternumbers` e `beramono` di ClassicThesis, assieme ad ArsClassica.

23.2 PRINCIPI

Lo stile tipografico di ArsClassica si differenzia da quello di ClassicThesis per i seguenti aspetti:

- usa i font sans serif Iwona per i titoli delle sezioni, per le testatine e le intestazioni delle didascalie (ClassicThesis non usa alcun font sans serif);
- i numeri dei capitoli sono diversi;
- le testatine sono semitrasparenti;
- le didascalie hanno intestazioni in neretto (che ClassicThesis non usa);
- gli elenchi puntati hanno etichette semitrasparenti;
- i margini sono più stretti di quelli di ClassicThesis.

ArsClassica offre all'utente uno stile tipografico pronto per l'uso: modificandolo si rischia di distruggerne l'equilibrio, per cui si consiglia caldamente

di *non* farlo. ArsClassica non è né configurabile né personalizzabile: se l'utente è soddisfatto del pacchetto e trova allettante l'idea di disinteressarsi del problema della definizione dello stile, userà ArsClassica con soddisfazione. Chi invece ha esigenze diverse è bene che provi a rivolgersi ad altre classi o pacchetti, eventualmente costruendosi da solo il proprio stile.

23.3 REGOLE

Per scrivere un documento secondo lo stile di ArsClassica si devono seguire alcune semplici regole.

- Non si devono modificare *per nessun motivo* le impostazioni del pacchetto (font, margini, colori, eccetera).
- I titoli delle sezioni devono essere *lunghi una sola riga* (non devono andare a capo) e devono essere *semplice testo* (non devono contenere simboli, formule o frammenti di codice). Se un titolo non soddisfa questi requisiti, si provi a riformularlo: si può fare praticamente sempre.
- Nell'indice generale e nell'elenco delle tabelle e delle figure, le voci devono essere *lunghe una sola riga* e devono essere *semplice testo*. Se serve, si può usare l'argomento facoltativo dei comandi di sezionamento e di `\caption`.
- Non vanno scelte le opzioni `tocaligned` e `dottedtoc` di ClassicThesis: l'indice predefinito è eccellente (si veda al riguardo la documentazione di ClassicThesis).
- Non si usino filetti verticali o doppi nelle tabelle (si veda in proposito la documentazione di booktabs).
- Si usino il meno possibile le note al piede e a margine.
- Se il proprio documento contiene disegni e grafici, essi vanno realizzati direttamente con \LaTeX (con `TikZ` e `pgfplots`). Solo così si ottiene un risultato tipografico ottimale.

Quest'appendice, basata su [Cevolani, 2006], cui si rimanda per gli approfondimenti, descrive sinteticamente e senza pretese di completezza le tradizioni tipografiche più seguite nella redazione di un documento in italiano. Di ogni regola discussa si mostra, per quanto possibile, l'applicazione in L^AT_EX. La parola *norma* va qui intesa in senso piuttosto lato: anche nella nostra lingua, come in tutte le altre, non esistono che pochissime regole tipografiche realmente universali e vincolanti, mentre molti aspetti del testo finito dipendono da tradizioni e abitudini o dai gusti di utente o editore.

Naturalmente è superfluo ricordare che per risolvere dubbi di altro tipo il mezzo più rapido e sicuro rimane la consultazione di grammatica e dizionario.

A.1 ACCENTO E APOSTROFO

Accento

In italiano ci sono due accenti:

- *grave*, ` , che indica suono aperto;
- *acuto*, ´ , che indica suono chiuso.

Un terzo accento, il *circonflesso*, ^ , usato in passato soprattutto per distinguere gli omografi uscenti in *-ii* al plurale – *odî* (plurale di *odio*) da *odi* (voce del verbo *udire*) – oggi non viene quasi più usato.

La loro applicazione non è univoca, e oscilla tra le prescrizioni UNI (per cui l'accento grave può colpire *tutte* le vocali: à/À, è/È, ì/Ì, ò/Ò, ù/Ù; e l'accento acuto preferibilmente solo le vocali *e* e *o*: é/É, ó/Ó) e le tradizioni di alcune case editrici (per cui *i* e *u* portano sempre l'accento acuto, per esempio). Qualunque sistema si scelga, l'importante è seguirlo coerentemente in *tutto* il documento.

Gli accenti vanno apposti *sopra* la vocale, minuscola o maiuscola che sia: il paragrafo 3.2 a pagina 22 mostra come impostare i pacchetti che permettono di interpretare correttamente le lettere accentate italiane. Al contrario di quanto accade con le vecchie macchine per scrivere o con altri elaboratori di testo, L^AT_EX permette di accentare senza problemi *anche le lettere maiuscole*, che spesso vengono erroneamente apostrofate (come in *E'*, che sta per *Ei* cioè "egli").

Apostrofo

L'*apostrofo*, ´ , segnala normalmente la caduta della parte iniziale, come in *'sta* (per *questa*), o finale di una parola, come in *un'altra*, *un po'*, eccetera.

Quando cadono lettere o sillabe iniziali l’apostrofo è preceduto da uno spazio; quando cadono sillabe finali l’apostrofo è seguito da uno spazio o da un segno d’interpunzione.

Quando la vocale finale di una parola cade di fronte a quella iniziale della parola successiva (come in *un’altra* o in *quell’uomo*), l’apostrofo sta al posto della vocale caduta e la parola seguente comincia senza nessuno spazio intermedio. Se babel è caricato con l’opzione *italian*, L^AT_EX evita automaticamente l’apostrofo in fine di riga: *quell’uomo* viene sillabato “quel-l’uo-mo”, come se fosse un’unica parola.

A.2 PUNTEGGIATURA E SPAZIATURA

La punteggiatura italiana comprende segni d’interpunzione, parentesi, virgolette, puntini di sospensione, trattini e altri simboli come asterisco e barra. Ci sono alcune regole fisse sull’uso degli spazi prima e dopo i segni di punteggiatura.

A.2.1 Segni d’interpunzione

Tutti i segni d’interpunzione seguono immediatamente la parola precedente e vanno separati con uno spazio da quella successiva.

A.2.2 Virgolette

Nel testo di regola non si lasciano spazi bianchi tra virgolette e contenuto (anche se è ammesso e talvolta richiesto lo spazio sottile), mentre li si lascia tra virgolette e testo esterno.

Normalmente si evidenziano tra virgolette parole o frasi in relazione al loro significato:

- citazioni dirette e citazioni “annidate”, cioè citazioni dentro a citazioni («Gli ho sentito dire: “Verrò di sicuro”»);
- termini ed espressioni che specificano il significato di altre parole: «La parola *box* significa “scatola”»;
- espressioni figurate o gergali: «Si prevedono scioperi “a singhiozzo”»;
- termini correnti a cui si attribuisce un significato diverso da quello abituale: «Questo ragazzo non è certo una “cima”»;
- parole usate in senso ironico.

I casi sopra elencati non sono regole vincolanti: alcuni di essi, infatti, trovano un ottimo concorrente nel *corsivo*, con l’avvertenza di evitare *sempre* di usarli contemporaneamente. Non esistono nemmeno regole fisse per usare i vari tipi di virgolette (si veda il paragrafo 4.3.1 a pagina 58): nella pubblicazione dei testi a stampa, infatti, si adottano convenzioni tipografiche variabili dall’uno all’altro editore e perfino tra l’una e l’altra collana dello stesso marchio editoriale. L’unico consiglio che qui si può dare è di scegliere un sistema di virgolettatura *prima* di cominciare a scrivere il documento e di usarlo con coerenza.

A.2.3 Parentesi

In italiano si usano di solito le parentesi *tonde* () e le parentesi *quadre* []. Le parentesi *graffe* { } e *angolate* < > (i comandi per queste ultime sono `\langle` e `\rangle`, da usare in modo matematico) si usano solo in matematica e in informatica rispettivamente.

In genere le parentesi racchiudono un inciso nel discorso (cioè una parte *che si può omettere*, come questa). Seguono alcune indicazioni generali sul loro impiego.

- Le parentesi vanno attaccate al loro contenuto e separate con uno spazio dal testo precedente e seguente (a meno che non chiudano un enunciato, come qui).
- Quando un segno d'interpunzione chiude il testo tra parentesi, si richiede, come qui, un ulteriore punto *fuori* dalla parentesi di chiusura (ovviamente!). Si noti che di norma solo i punti interrogativo ed esclamativo possono stare dentro le parentesi.
- L'uso di racchiudere tra parentesi un intero enunciato è lecito ma poco diffuso in italiano, a differenza di altre lingue. (In tal caso, anche la punteggiatura rientra nelle parentesi, come qui.)

Le parentesi quadre si usano quasi esclusivamente in due casi.

- Come parentesi interne a parentesi (come [anche se non è molto frequente] in questo caso).
- Per introdurre «il commento di una persona diversa dall'autore del testo cui il commento si riferisce [in questo modo]». In quest'ultimo caso rientra anche quello dell'*omissione* volontaria (che è comunque un commento), segnalata con [...] (si veda il paragrafo 4.3.1 a pagina 58).

A.2.4 Puntini di sospensione

I puntini di sospensione sono *sempre e solo tre* e, come gli altri segni di interpunzione, seguono immediatamente la parola che li precede e sono separati con uno spazio da quella che li segue... in questo modo. Con \LaTeX , il comando da usare è `\dots` (non vanno *mai* inseriti a mano tre punti separati), eventualmente chiudendolo con un gruppo vuoto {} se i puntini non si trovano a fine enunciato.

Se usati per indicare un'omissione in una citazione «è bene [...] inserirli tra parentesi quadre o tonde» con il comando `\omissis` come in questo caso (si veda lo stesso paragrafo).

A.2.5 Trattino, tratto e lineetta

Si è già spiegato (nel paragrafo 4.3.1 a pagina 58) come realizzare questi tre segni con \LaTeX . Di seguito se ne descrive l'uso in termini generali.

Il *trattino* di solito divide le parti di un'espressione composta e nella scrittura s'interpone tra di esse senza ulteriori spazi bianchi. Si usa:

- Nei termini formati da due parole autonome, come «guerra-lampo».

- Negli intervalli numerici o di tempo i cui estremi siano espressi in cifre, come «1915-1918» e «pagine 2-11».

Il *tratto*, preceduto e separato da uno spazio bianco, di solito suddivide elementi come titoli e didascalie, come in «Varianti di carattere – Il maiuscoletto».

La *lineetta* isola nettamente un inciso all'interno del testo — ma in quest'uso la tipografia italiana preferisce il tratto — o segnala le battute in un dialogo. Nella scrittura è separata dal testo precedente e seguente da uno spazio bianco normale, come qui.

A.2.6 Barra e asterisco

La *barra* (o *sbarretta*) può essere usata nelle seguenti circostanze ordinarie.

- Senza spazi né prima né dopo, per un'alternativa tra due termini: «i passeggeri diretti a Torino/Milano».
- Preceduta e seguita da uno spazio bianco, separa i versi di una poesia in linea: «M'illumino / d'immenso».
- Senza spazi prima e dopo, si usa nelle frazioni numeriche in modo testuale: «i 3/4 della popolazione».

Si ricordi che la barra (/), inseribile direttamente dalla tastiera, *non* va confusa con il carattere di barra rovescia (\), riservato ai comandi di L^AT_EX.

L'*asterisco* ha un uso limitato alle seguenti situazioni:

- separato da quanto precede con uno spazio sottile e ripetuto tre volte, indica omissione volontaria («Il padre Cristoforo da *** era un uomo più vicino ai sessanta che ai cinquanta», codice: *\, *\, *);
- per indicare in linguistica forme non attestate, scorrette o inaccettabili: «* che io vadi».

A.3 STILE DEL FONT

Oltre a quello normale usato per il testo, di solito un font prevede lo stile *corsivo*, **nero** e MAIUSCOLETTA. Si usano poco lo stile *inclinato* (da non confondere con il corsivo *inclinato*) e altri stili, utili solo per esigenze particolari: per esempio, in questo documento si usa lo stile dattilografico per evidenziare il codice L^AT_EX. Lo stile “evidenziato” (o “enfaticizzato”) è reso normalmente con il corsivo, ma riveste un ruolo logico differente.

A.3.1 Corsivo

Normalmente si evidenziano in corsivo parole o frasi in relazione alla loro presenza nel testo:

- termini tecnici: «Una *distribuzione* è una raccolta di programmi...»;
- parole o frasi straniere di uso non comune con le quali si ritiene che il lettore non abbia affinità: «Questa tecnica di *engraving* è considerata...»;

- parole e lettere a cui ci si riferisce come tali nel testo: «La lettera *e* non compare nella parola *parola*»;
- parole o frasi da evidenziare: «Questo *non* si fa».

Oltre che nel testo principale, si può usare il corsivo per comporre i titoli.

A.3.2 Nero

Le classi standard di L^AT_EX usano il nero per i titoli di sezione. Di norma non va impiegato per evidenziare parole o interi capoversi del testo principale (il corsivo va benissimo), e comunque va utilizzato con moderazione per non appesantire la pagina.

A.3.3 Maiuscoletto

Si usa il maiuscoletto quasi esclusivamente per i nomi degli autori citati in bibliografia, come in BRINGHURST (1992). Questa convenzione, tuttavia, dipende dallo stile bibliografico scelto. Oppure concorre con il maiuscolo per segnalare gli acronimi nel testo. Talvolta si usa per i titoli di certe sezioni.

A.4 TRATTAMENTO DEL TESTO

Ci limiteremo a prendere in considerazione il caso delle parole straniere, uno dei più frequenti.

A.4.1 Parole straniere

Le parole straniere vanno in corsivo, a meno che non vengano esplicitamente “quoted” (“virgolettate”, come in questo caso) o non siano d’uso comune. Quindi si scriverà «ho visto un bel film» (comune) ma «ho mangiato un *pudding*» (non comune). In realtà, essendo molto difficile stabilire che cosa sia “comune” o meno, la cosa più corretta è scrivere in corsivo le parole straniere che si presumono di uso non corrente *per il lettore a cui ci si rivolge*. In un libro d’informatica, perciò, *software* e *computer* saranno in tondo.

I nomi propri e le denominazioni ufficiali (come Stanford University e Magna Charta) non sono considerati parole straniere e vanno in tondo.

La traduzione straniera di un’espressione italiana usata nel testo può essere semplicemente messa in corsivo e fra parentesi (*bracket*), come in questo caso. Se invece l’espressione tradotta ricorre in una citazione e se ne vuole indicare la forma originale, va tra parentesi quadre come ogni altro commento. Per esempio: «La visione del mondo [*Weltanschauung*]. . . ».

Ovviamente ogni lingua straniera va sillabata a sé, ed è altrettanto ovvio che scrivendo in lingue diverse dall’italiano alcune cesure nel documento finito potrebbero risultare errate. Si noti che è ammesso sillabare secondo le nostre regole parole straniere isolate nel discorso, ma non un testo in lingua più esteso (si veda il paragrafo 3.2 a pagina 22).

A.4.2 Numeri

Scrivere i numeri

I caratteri numerici possono essere *maiuscoli* e *minuscoli*. Si osservi la resa tipografica del numero 1821 nei due esempi seguenti:

Il numero \$1821\$ non è primo. \\ Napoleone morì nel 1821.	Il numero 1821 non è primo. Napoleone morì nel 1821.
--	---

Nel primo, 1821 è maiuscolo; nel secondo, 1821 è minuscolo.

I numeri maiuscoli hanno tutti la stessa altezza e sono indicati:

- nelle formule matematiche;
- nelle tabelle composte di dati numerici;
- se intesi in senso aritmetico (come nell'indicazione di quantità esatte).

I numeri minuscoli presentano tratti ascendenti e discendenti per integrarsi al meglio con i caratteri minuscoli del font corrente, e non sono indicati in tutti gli altri casi. Di solito per ottenerli non serve fare nulla, perché sono predefiniti nel font scelto: si consiglia in ogni caso di fare una prova per verificarlo.

Spaziare le cifre

La corretta scrittura dei numeri interi *di cinque o più cifre* prevede uno spazio sottile ogni tre cifre a partire da destra (come in 1 500 000). Per ottenerlo ci sono diverse possibilità: se i numeri da scrivere nel documento non sono molti, si può inserire a mano lo spazio sottile \, ; in caso contrario, risolve il problema il comando \num del pacchetto siunitx. Si osservi il risultato tipografico dei numeri negli esempi seguenti:

\$1500000\$ \\	1500000
\$1\,500\,000\$ \\	1 500 000
\num{1500000}	1 500 000

Come si può notare, se non altrimenti specificato \num produce numeri maiuscoli.

Si evitino il punto e la virgola per separare le cifre, perché questi due segni servono già da separatore decimale nel mondo anglosassone ed europeo rispettivamente. Lo spazio sottile è l'unico metodo universalmente corretto.

Numeri decimali

Nell'uso italiano, il separatore fra parte intera e fratta di un numero decimale è una virgola: «21,12». Si noti che in modo matematico L^AT_EX considera la virgola un normale segno di punteggiatura e mette dopo uno spazio extra, con un risultato insoddisfacente. Il pacchetto siunitx caricato con l'opzione output-decimal-marker={,} e \num risolvono anche questo problema:

\$21,12\$ \\	21,12
\num{21,12}	21,12

Si osservi che nel primo caso dopo la virgola si ha una spaziatura (leggermente) sbagliata: la scrittura corretta è la seconda.

A.4.3 Frazioni, percentuali, unità di misura

Le frazioni si esprimono in lettere, «tre quarti», a meno che non indichino una quantità numerica precisa. In questo caso si possono scrivere usando la barra, «3/4», o la forma frazionaria vera e propria, « $\frac{3}{4}$ », ottenibile con il codice visto nel paragrafo 6.2.3 a pagina 80.

Nella scrittura del testo il simbolo di percento segue immediatamente il numero cui si riferisce senza spazi intermedi: «30%» (codice: `\%`; attenzione ai caratteri speciali).

Le *quantità misurate* sono costituite da numeri seguiti da un'unità di misura espressa di solito con il simbolo relativo, come in «20 cm» o «15 kg». Si noti che tra numero e simbolo c'è uno spazio sottile e che il simbolo non vuole il punto dopo. Per scrivere correttamente le unità di misura del Sistema Internazionale si consiglia di usare il pacchetto `siunitx` o lo spazio sottile (si veda il paragrafo 6.10 a pagina 100).

A.4.4 Acronimi e abbreviazioni

Gli *acronimi* sono espressioni formate dalle lettere o sillabe iniziali delle parole di un'espressione che si vuole abbreviare, come `GjTr`, HTML o PDF. Dovrebbero essere composti interamente da maiuscole (si ammette anche il maiuscoletto), *non* spaziate e *senza* punti d'abbreviazione. Se l'acronimo è entrato nell'uso e si può pronunciare come parola, è ammissibile (e spesso preferibile) scriverlo come tale: *Fiat* e *radar* vanno benissimo. Si consiglia di citare per esteso gli acronimi meno noti la prima volta che compaiono nel testo, mettendone la forma estesa fra parentesi: `GjTr` (Gruppo Utilizzatori Italiani di $\text{T}_{\text{E}}\text{X}$ e $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$).

Le *abbreviazioni* si ottengono invece dal troncamento di una parola mantenendone una o più lettere iniziali seguite dal punto (come *sig.* o *prof.*). Se l'abbreviazione dovesse cadere a fine enunciato (caso raro, forse possibile con il solo *ecc.*) il punto di abbreviazione funziona anche da punto fermo (cioè, ovviamente, non si scrivono due punti successivi).

B | GALLERIA DEGLI ORRORI

Quest'appendice, ispirata a [Gregorio, 2003], raccoglie alcuni esempi tratti da documenti \LaTeX trovati in Rete. Presento questa galleria non per svergognare gli autori, naturalmente; piuttosto, l'obiettivo è di mostrare come *non* si scrive in \LaTeX e che con qualche minuto di riflessione e il pacchetto "giusto" si può spesso ottenere un risultato migliore e più semplice, evitando di prodursi in acrobazie " \TeX niche". Di ciascun esempio si mostra il testo originale; lo si discute e poi si dà una versione corretta, con l'indicazione di quali pacchetti richiamare e quali comandi definire nel preambolo.

B.1 BASI

B.1.1 Accenti

Consideriamo il codice seguente:

```
Basta! Non se ne pu\`o pi\`u!  
Perch\`e non c'\`e piet\`a  
per chi deve scrivere cos\`i?
```

Basta! Non se ne può più! Perché
non c'è pietà per chi deve scrivere
così?

Con i pacchetti `inputenc` e `fontenc` opportunamente configurati si possono inserire i caratteri accentati direttamente da tastiera (si veda il paragrafo 3.2 a pagina 22):

```
Però: caricando il pacchetto  
giusto la vita diventerà così  
facile che c'è molto più gusto.
```

Però: caricando il pacchetto giusto
la vita diventerà così facile che c'è
molto più gusto.

B.1.2 Virgolette

Si consideri il seguente esempio:

```
\thanks{  
Grazie a Dirac e alle sue "funzioni improprie".}
```

Non si deve mai usare il carattere " per indicare le virgolette (si veda il paragrafo 3.4.2 a pagina 28). Va inoltre eliminato quello spazio spurio. Il codice proposto va sostituito con:

```
\thanks{Grazie a Dirac e alle sue ``funzioni improprie''.}
```

B.1.3 Classi di documento

Si considerino i seguenti esempi:

```
\documentstyle[11pt,twoside]{article}  
\input{amssymb.sty}
```

```
\documentclass[11pt,bezier]{article}
```

Il comando `\documentclass` è stato introdotto nel 1994 (paragrafo 3.5 a pagina 32): gli utenti dovrebbero ormai essersene impadroniti. Quanto al secondo esempio, come faccia un autore a ignorare bellamente, ogni volta che compila il documento, il messaggio

```
LaTeX Warning: Unused global option(s) [bezier].
```

e a non prendere il semplice provvedimento di cancellare quella parola, è cosa che non riusciamo a immaginare. I codici proposti vanno sostituiti rispettivamente con:

```
\documentclass[11pt,twoside]{article}
\usepackage{amssymb}
```

```
\documentclass[11pt]{article}
```

B.1.4 Margini di pagina

Si consideri il seguente esempio:

```
\addtolength{\textwidth}{1cm}
\addtolength{\oddsidemargin}{-0.5cm}
```

Per impostare i margini di pagina non si devono toccare comandi interni di L^AT_EX come `\textwidth` o `\oddsidemargin`, perché la loro azione non tiene conto delle proporzioni di pagina, ma bisogna rivolgersi a pacchetti come LayAureo o geometry (paragrafo 3.6 a pagina 34).

B.1.5 Sezionamenti

Si considerino i seguenti esempi:

```
\section{\bigskip Introduzione}
```

```
\vspace{3ex}
{\large\bfseries Titolo di un paragrafo}
\vspace{3ex}
```

Gli autori qui volevano nel primo caso aumentare lo spazio fra il titolo e l'introduzione, e nel secondo avere un titolo di sezione senza la numerazione automatica.

Lo spazio prima di un titolo di sezionamento è una faccenda su cui va lasciata la decisione alla classe del documento, mentre se non vuole la numerazione automatica basta dare `\section*` (paragrafo 3.7 a pagina 36).

I codici proposti vanno sostituiti con:

```
\section{Introduzione}
```

```
\section*{Titolo di un paragrafo}
```

B.1.6 Indirizzi Internet

Consideriamo il seguente esempio:

<code>http://profs.sci.univr.it/% \$\sim\$gregorio/</code>	<code>http://profs.sci.univr.it/~gregorio/</code>
--	---

Per scrivere un indirizzo Internet è opportuno servirsi del pacchetto `url` (caricato automaticamente da `hyperref`), che definisce il comando `\url` (paragrafo 3.11 a pagina 44):

<code>\url{http://profs.sci.univr.it/% ~gregorio/}</code>	<code>http://profs.sci.univr.it/ ~gregorio/</code>
---	--

In questo modo l'indirizzo Internet è composto con il font a spaziatura fissa usato, viene mandato a capo se necessario e non occorre fare peripezie per inserire la tilde.

B.1.7 Sillabazione

Si consideri il seguente esempio:

<code>\hyphenation{sil-la-ba-zio-ne sim-pa-ti-ca-men-te}</code>

Questo uso di `\hyphenation` è infelice, perché le parole indicate vengono sillabate correttamente da `babel`. Il comando `\hyphenation` serve con nomi propri o tecnicismi come *nitroidrossilamminico* o *macroistruzione*, per i quali a volte si richiede la sillabazione etimologica anziché quella che `LATEX` esegue di default (paragrafo 4.2 a pagina 53). L'esempio proposto va sostituito con un altro del tipo seguente:

<code>\hyphenation{nitro-idrossil-amminico ma-cro-istru-zio-ne}</code>
--

B.1.8 Caratteri particolari

Si consideri il seguente esempio:

<code>Sm\/{o}rrebr\/{o}d</code>	Smorrebrod
---------------------------------	------------

Il nome citato è "Smørrebrød": non si capisce che ci stia a fare il comando `\/{o}`. L'autore evidentemente si è detto: «La sbarra attraverso è una specie di accento; si farà con la sbarra, no?». No. Il codice corretto è il seguente (paragrafo 4.3.2 a pagina 59):

<code>Sm{\o}rrebr{\o}d</code>	Smørrebrød
-------------------------------	------------

B.1.9 Titoli

Si consideri il seguente esempio:

<code>\title{{\bfseries\Large Titolo del documento}}</code>

Anche queste sono faccende sulle quali va lasciata la decisione alla classe del documento (paragrafo 4.5 a pagina 62). Le doppie graffe, poi, sono inutili. Il codice corretto è il seguente:

<code>\title{Titolo del documento}</code>

B.1.10 Elenchi

Si considerino i seguenti esempi:

<code>\begin{enumerate}</code>	
<code>\item[\emph{a}]] Mane</code>	<i>a) Mane</i>
<code>\item[\emph{b}]] Tekel</code>	<i>b) Tekel</i>
<code>\item[\emph{c}]] Fares</code>	<i>c) Fares</i>
<code>\end{enumerate}</code>	

<code>\emph{a}) Mane \[4pt]</code>	<i>a) Mane</i>
<code>\emph{b}) Tekel \[4pt]</code>	<i>b) Tekel</i>
<code>\emph{c}) Fares</code>	<i>c) Fares</i>

Gli autori qui volevano produrre un elenco personalizzato, ma l'hanno fatto in modo orrendo. Per modificare il comportamento dell'ambiente `enumerate` c'è il pacchetto `enumerate` (paragrafo 4.8.1 a pagina 65). È conveniente definire nel preambolo un ambiente elenco:

```
\newenvironment{elenco}{\begin{enumerate}[label=\emph{\alph*}]]}%
{\end{enumerate}}
```

e usarlo in modo coerente nel corpo del testo:

<code>\begin{elenco}</code>	
<code>\item Mane</code>	<i>a) Mane</i>
<code>\item Tekel</code>	<i>b) Tekel</i>
<code>\item Fares</code>	<i>c) Fares</i>
<code>\end{elenco}</code>	

Si noti anche che ogni volta che si ha voglia di usare il comando `\[` bisogna riflettere: questo comando infatti va usato solo negli ambienti che lo richiedono (`center`, `flushleft`, `flushright`, `tabular`, `array` e, in genere, quelli degli allineamenti di formule matematiche).

B.2 TABELLE E FIGURE

B.2.1 Regole generali

Consideriamo il seguente esempio:

```
\begin{tabular}{|p{2cm}|p{2cm}|}|
\hline\hline
Alcaloide & Origine \\
\hline\hline
atropina & belladonna \\
morfina & papavero \\
nicotina & tabacco \\
\hline\hline
\end{tabular}
```

Alcaloide	Origine
atropina	belladonna
morfina	papavero
nicotina	tabacco

Si noti che:

- la tabella precedente è stata composta senza seguire le regole generali, che impongono di non usare filetti verticali e di evitare filetti doppi (paragrafo 8.3.1 a pagina 123);

- le tabelle migliori si ottengono lasciando loro la propria larghezza naturale, indicando larghezza delle colonne solo se necessario (paragrafo 8.3.3 a pagina 126);
- per ottenere i filetti orizzontali è opportuno usare i comandi del pacchetto booktabs invece dello standard `\hline`, dalla resa tipografica insoddisfacente per via dello spazio troppo risicato che risulta tra filetti e testo nelle celle (paragrafo 8.3.1 a pagina 124);
- anche se \LaTeX non richiede di incolonnare le celle nel sorgente, si consiglia di farlo ugualmente: un codice ordinato facilita eventuali modifiche, diminuisce la probabilità di commettere errori e aumenta quella di scovarli (paragrafo 8.3.1 a pagina 125).

Per queste ragioni è opportuno riscrivere le tabelle nel modo seguente:

```
\begin{tabular}{ll}
\toprule
Alcaloide & Origine \\
\midrule
atropina & belladonna \\
morfina & papavero \\
nicotina & tabacco \\
\bottomrule
\end{tabular}
```

Alcaloide	Origine
atropina	belladonna
morfina	papavero
nicotina	tabacco

B.2.2 Riferimenti incrociati

Capita spesso di imbattersi nei seguenti codici:

```
\begin{table}
\label{tab:esempio}
\caption{Esempio}
...
\end{table}
```

```
\begin{figure}
...
\label{fig:esempio}
\caption{Esempio}
\end{figure}
```

Il comando `\label` va sempre dato *dopo* il corrispondente `\caption` (paragrafo 3.10 a pagina 43). I codici proposti vanno sostituiti con:

```
\begin{table}
\caption{Esempio}
\label{tab:esempio}
...
\end{table}
```

```
\begin{figure}
...
\caption{Esempio}
\label{fig:esempio}
\end{figure}
```

Consideriamo ora il seguente esempio:

Si veda la Fig.~1.

Si noti che:

- i riferimenti incrociati si fanno con i comandi standard `\label` e `\ref` (paragrafo 3.10 a pagina 43);
- in italiano non si usa la maiuscola nell'abbreviazione di *figura* e *tabella*, come avviene invece in inglese.

Il codice proposto va sostituito con un altro del tipo seguente:

Si veda la figura~\ref{fig:esempio}.

B.2.3 Collocazione

Consideriamo il seguente esempio:

```
\begin{figure}[!h]
\begin{center}
\includegraphics[width=10cm]{esempio.jpg}
\end{center}
\caption{Esempio}
\label{fig:esempio}
\end{figure}
```

Il codice precedente contiene diversi errori:

- gli oggetti mobili vanno inseriti senza specificare alcuna preferenza di collocazione, oppure, se proprio necessario, con `tp` o con `htp` (paragrafo 8.2.2 a pagina 122);
- per centrare un oggetto mobile sulla pagina si usa `\centering`, perché l'ambiente `center` lascia tra testo e oggetto uno spazio verticale eccessivo (paragrafo 8.2.2 a pagina 120);
- è opportuno assegnare all'oggetto una larghezza relativa (espressa con una frazione della giustezza stabilita dalla classe in uso): una larghezza assoluta causerebbe ovvi inconvenienti cambiando classe di documento o aumentando le colonne di composizione (paragrafo 8.1 a pagina 117);
- nell'argomento obbligatorio di `\includegraphics` ci va il nome dell'immagine senza specificarne l'estensione (paragrafo 8.4.5 a pagina 142).

In conclusione, per introdurre una figura mobile è opportuno servirsi di un codice del tipo seguente:

```
\begin{figure}
\centering
\includegraphics[width=0.5\columnwidth]{esempio}
\caption{Esempio}
\label{fig:esempio}
\end{figure}
```

B.3 MATEMATICA

B.3.1 Formule in display

Si consideri il seguente esempio:

```
Poiché
$$
a=b,
$$
allora
$$
b=a.
$$
```

Poiché $a = b,$
allora $b = a.$

Per scrivere una formula non si devono usare i dollari doppi $$$\dots$$$, che potrebbero compromettere la corretta spaziatura verticale delle formule o il funzionamento dell'opzione di classe `fleqn` (paragrafo 6.1.2 a pagina 78). Il codice corretto è il seguente:

```
Poiché
\[
a=b
\]
allora
\[
b=a
\]
```

Poiché $a = b$
allora $b = a$

Nelle formule precedenti abbiamo eliminato la punteggiatura (paragrafo 6.1.2 a pagina 78).

Si consideri ora il seguente esempio:

```
\begin{eqnarray}
e^{x+y} &=& e^x e^y \\\
\ln xy &=& \ln x + \ln y
\end{eqnarray}
```

$$e^{x+y} = e^x e^y \quad (\text{B.1})$$

$$\ln xy = \ln x + \ln y \quad (\text{B.2})$$

Gli ambienti standard `eqnarray` e `eqnarray*` non si devono usare, perché prima e dopo `=` inseriscono più spazio del dovuto (paragrafo 6.1.2 a pagina 78). Per incolonnare due o più formule si usa l'ambiente `align` del pacchetto `amsmath` (paragrafo 6.7.2 a pagina 94). L'esempio precedente si scrive correttamente:

```
\begin{align}
e^{x+y} &= e^x \,, e^y \\\
\ln xy &= \ln x + \ln y
\end{align}
```

$$e^{x+y} = e^x e^y \quad (\text{B.3})$$

$$\ln xy = \ln x + \ln y \quad (\text{B.4})$$

Si noti che nella prima formula è stata modificata la spaziatura automatica, qui leggermente insoddisfacente (paragrafo 6.2.10 a pagina 86).

B.3.2 Riferimenti

Si consideri il seguente esempio:

B.3.5 Nuovi operatori

Si consideri il seguente esempio:

```


$$\mathrm{cov}(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \overline{x})(y_i - \overline{y})$$


```

$$\mathrm{cov}(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

Per definire una nuova funzione c'è il comando `\DeclareMathOperator` di `amsmath` (si veda il paragrafo 6.3.2 a pagina 88). Per esempio, per definire la funzione matematica “cov” che denoti la covarianza di due variabili aleatorie (funzione non prevista né da $\mathrm{\LaTeX}$ né da `amsmath`), si scrive nel preambolo:

```
\DeclareMathOperator{cov}{cov}
```

Nel documento, poi, si darà `\cov` per ottenere “cov” nel font corretto e adeguatamente spaziato su entrambi i lati. L'esempio precedente si scrive allora:

```

\l
\cov(X, Y) =
\frac{1}{n} \sum_{i=1}^n
(x_i - \bar{x})(y_i - \bar{y})
\r

```

$$\mathrm{cov}(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

È stato usato il comando `\bar` al posto di `\overline` per indicare il valor medio delle variabili aleatorie X e Y : il simbolo \bar{x} indica un operatore (coniugio di numeri complessi, per esempio) applicato alla variabile x , mentre il simbolo \bar{x} indica un nome di variabile distinto da x .

B.3.6 Parentesi

Si consideri la seguente formula:

```

\l
\left(\sum_n x_n^2\right)^{1/2}
\r

```

$$\left(\sum_n x_n^2\right)^{1/2}$$

I comandi standard `\left` e `\right` inseriscono quasi sempre spaziature indesiderate e parentesi più grandi del necessario: perciò va preferito loro il metodo manuale (paragrafo 6.4 a pagina 89). La formula precedente si scrive allora:

```

\l
\biggl(\sum_n x_n^2\biggr)^{1/2}
\r

```

$$\left(\sum_n x_n^2\right)^{1/2}$$

B.3.7 Matrici

Si consideri il seguente esempio:

```


$$\mathcal{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$


```

$$\mathcal{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Per scrivere matrici ci sono gli appositi ambienti di `amsmath` (paragrafo 6.5 a pagina 90), che sono facili da usare e producono un risultato tipografico ottimale. La matrice precedente si scrive allora:

```

\begin{pmatrix}
\mathcal{A}=
\begin{pmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{pmatrix}
\end{pmatrix}

```

$$\mathcal{A} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

Si noti anche che `\mathcal` è un comando e non una dichiarazione.

B.3.8 Dividere una formula in più righe, con allineamenti

Si consideri la seguente formula:

```

\begin{eqnarray}
&\int_1^2 x^2 dx \\
&= \left[ \frac{x^3}{3} \right]_1^2 \\
&= \frac{2^3}{3} - \frac{1^3}{3} \\
&= \frac{8}{3} - \frac{1}{3} \\
&= \frac{7}{3}
\end{eqnarray}

```

$$\begin{aligned} \int_1^2 x^2 dx &= \left[\frac{x^3}{3} \right]_1^2 \\ &= \frac{2^3}{3} - \frac{1^3}{3} \\ &= \frac{8}{3} - \frac{1}{3} \\ &= \frac{7}{3} \end{aligned} \quad (\text{B.7})$$

Si noti che:

- per dividere una formula in più righe, con le righe da allineare, si usa l'ambiente `split` di `amsmath` (paragrafo 6.6 a pagina 92);
- come già detto, poiché i comandi standard `\left` e `\right` inseriscono quasi sempre spaziature indesiderate e parentesi più grandi del necessario va preferito loro il metodo manuale (paragrafo 6.4 a pagina 89).

La formula precedente si scrive allora:


```

\begin{equation}
\begin{split}
\int_1^2 x^2 dx
&= \biggl[\frac{x^3}{3}\biggr]_1^2 \\
&= \frac{2^3}{3} - \frac{1^3}{3} \\
&= \frac{8}{3} - \frac{1}{3} \\
&= \frac{7}{3}
\end{split}
\end{equation}

```

$$\begin{aligned}
 \int_1^2 x^2 dx &= \left[\frac{x^3}{3} \right]_1^2 \\
 &= \frac{2^3}{3} - \frac{1^3}{3} \\
 &= \frac{8}{3} - \frac{1}{3} \\
 &= \frac{7}{3}
 \end{aligned}
 \tag{B.8}$$

Nell'esempio precedente sono state eliminate svariate parentesi graffe superflue, in particolare per gli esponenti.

B.3.9 Dividere una formula in più righe, senza allineamenti

Si consideri la seguente formula:

```

\begin{eqnarray}
\lefteqn{f=a+b+c}
\nonumber \\
&+ i+j+k+l \\
\nonumber \\
&+ x+y+z
\end{eqnarray}

```

$$\begin{aligned}
 f &= a + b + c \\
 &+ i + j + k + l \\
 &+ x + y + z
 \end{aligned}
 \tag{B.9}$$

Per dividere una singola formula in più righe, senza particolari allineamenti fra le varie righe, si usa l'ambiente `multline` di `amsmath` (paragrafo 6.6.1 a pagina 92):

```

\begin{multline}
f=a+b+c \\
+i+j+k+l \\
+x+y+z
\end{multline}

```

$$\begin{aligned}
 f &= a + b + c \\
 &+ i + j + k + l \\
 &+ x + y + z
 \end{aligned}
 \tag{B.10}$$

B.3.10 Casi

Si consideri il seguente esempio:

```

$$
|x| =
\left\{
\begin{array}{rl}
x & \text{se } x \geq 0 \\
-x & \text{se } x < 0
\end{array}
\right.
.
$$

```

$$|x| = \begin{cases} x & \text{se } x \geq 0 \\ -x & \text{se } x < 0 \end{cases}$$

Si noti che:

- per scrivere il valore assoluto si consiglia di definire nel preambolo un comando ad hoc `\abs` servendosi del pacchetto `mathtools` come spiegato nel paragrafo 6.2.8 a pagina 83;

- per le definizioni fatte per casi si usa l'ambiente `cases` di `amsmath` (paragrafo 6.7.4 a pagina 94), che è facile da usare e produce un risultato tipografico ottimale.

La formula precedente si scrive allora:

```
\[
\abs{x} =
\begin{cases}
x & \& \text{se } x \ge 0 \\
-x & \& \text{se } x < 0
\end{cases}
\]
```

$$|x| = \begin{cases} x & \text{se } x \geq 0 \\ -x & \text{se } x < 0 \end{cases}$$

B.3.11 Dimostrazioni

Si consideri il seguente esempio:

```
\newenvironment{dimostrazione}%
{\vspace{1mm}{\it Dimostrazione}.\ }%
{\nolinebreak $\Box$ \par \medskip}

\begin{dimostrazione}
Per esercizio.
\end{dimostrazione}
```

Dimostrazione. Per esercizio. \square

L'idea dell'autore era di lasciare un piccolo spazio prima della dimostrazione e di terminare la stessa con un quadratino. Tutto ciò è già previsto nell'ambiente `proof` di `amsthm`, che permette di evitare di doversi definire un ambiente per conto proprio e che dà un risultato superiore (paragrafo 6.9.2 a pagina 100). Non si reinventi la ruota e si usi `amsthm`.

```
\begin{proof}
Per esercizio.
\end{proof}
```

Dimostrazione. Per esercizio. \square

B.4 BIBLIOGRAFIA

Si consideri il seguente esempio:

```
\section*{Riferimenti bibliografici}
\begin{enumerate}
\item U.Eco (1977), \emph{Come si fa una tesi}, Bompiani, Milano.
\item D.E.Knuth (1984), \emph{The \TeX book}, Addison-Wesley, Reading.
\end{enumerate}
```

Per motivi misteriosi l'autore non usa l'ambiente `thebibliography`, che gestisce la bibliografia molto facilmente (paragrafo 11.1 a pagina 217). Si notino anche le spaziature errate nei nomi. Il codice corretto è il seguente:

```
\begin{thebibliography}{9}
\bibitem{eco:tesi} Eco, Umberto (1977),
\emph{Come si fa una tesi di laurea}, Bompiani, Milano.
\bibitem{knuth:book} Knuth, Donald Ervin (1984)
\emph{The \TeX book}, Addison-Wesley, Reading (Massachusetts).
\end{thebibliography}
```

Quest'appendice, ispirata a [Fleck, 2008], traccia una breve storia della scrittura, di cui \LaTeX rappresenta una delle tappe più felicemente riuscite.

L'*excursus* è molto sintetico: ci sono fior di manuali dedicati all'argomento, e non si può discuterne a fondo in una veloce carrellata. Mi interessa solo mostrare che la tipografia è un'arte tramandata da secoli: i buoni documenti si sviluppano sempre secondo canoni obbedienti a quest'arte.

C.1 ORIGINI DELLA SCRITTURA

L'uomo, inteso come *homo sapiens*, ha vissuto duecentomila anni prima di cominciare a scrivere (si veda la tabella 69 nella pagina seguente): fino ad allora, si era accontentato di parlare e il suo sapere era limitato alla capacità di ricordare i racconti ascoltati. Questo frutto del pensiero andava però in gran parte perduto: se le idee non vengono *fissate* per essere apprese e tramandate, la generazione successiva deve cominciare ogni volta da capo.

c.1.1 Dalla preistoria alla storia

Nel *Fedro* di Platone, Socrate racconta la nascita leggendaria della scrittura. L'invenzione è attribuita a Thot, dio egizio della sapienza, già scopritore della matematica, dell'astronomia e della musica.



Figura 41: Figura bovina presente nelle grotte di Lascaux, in Francia (30 000 a.C.)

L'invenzione della scrittura (e con essa il passaggio dalla preistoria alla storia) è il frutto non di un'evento isolato, ma di un'evoluzione millenaria, preceduta dalla comparsa di disegni e simboli. Le origini della scrittura si possono cercare nell'arte rupestre degli uomini del Paleolitico, che ritraevano oggetti della vita quotidiana come alberi, monti, animali, scene di caccia.

Le prime evidenze di una "proto-scrittura" risalgono, secondo gli esami al radiocarbonio, al 6600 a.C. Si tratta di segni incisi su gusci di tartaruga scoperti nel sito archeologico

di Jiahu, nei pressi del Fiume Giallo, in Cina.

Una linea di sviluppo tradizionalmente articolata dagli storici è quella che ordina in serie cronologica i sistemi *logografici* (a un segno corrisponde una parola), quelli *sillabici* (a un segno corrisponde una sillaba) e quelli *alfabetici* (a un segno corrisponde un suono). Questa linea evolutiva va considerata con qualche distinzione: alcuni segni del sistema logografico egizio dei ge-

Tabella 69: Cronologia della scrittura

200 000 a. C.	Primi esemplari di <i>homo sapiens</i>
fino al 20 000	Arte rupestre del Paleolitico
6600	Segni cinesi su gusci di tartaruga
3400	Scrittura cuneiforme mesopotamica
3000	Geroglifici egizi
1400	Iscrizioni cretesi in lineare B
1400	Alfabeto di Ugarit
1300	Scrittura cinese
1000	Alfabeto fenicio
IX secolo	I greci adottano l'alfabeto dai fenici
VIII secolo	Gli etruschi adottano l'alfabeto dai greci
VII secolo	I latini adottano l'alfabeto dagli etruschi
VI secolo	Scrittura maya
VI secolo	Dall'aramaico nasce l'ebraico
II secolo	Pergamena
I secolo d. C.	Passaggio dal <i>volumen</i> al <i>codex</i>
III secolo	Scrittura onciale romana
V secolo	Gli ideogrammi cinesi si usano in Giappone
V secolo	Dall'aramaico nasce l'arabo
VII secolo	Nasce in India la scrittura Devanagari
VII secolo	I Cinesi inventano la carta
VIII secolo	Gli Arabi introducono la carta in Europa
VIII secolo	Scrittura carolingia
IX secolo	Alfabeto cirillico
X secolo	Primi documenti in italiano
XI secolo	Scrittura gotica
XI secolo	Sviluppo della xilografia
XII secolo	A Fabriano sorge la prima cartiera
XIV secolo	Scrittura umanistica
1455	Gutenberg stampa la Bibbia
1501	Aldo Manuzio introduce nella stampa il corsivo
1530	Claude Garamond disegna l'omonimo carattere
1540	Giambattista Palatino pubblica il suo trattato
1757	John Baskerville disegna l'omonimo carattere
1796	Litografia
1783	Firmin Didot disegna l'omonimo carattere
1798	Giambattista Bodoni disegna l'omonimo carattere
1814	Pressa a vapore
1843	Rotativa
1846	Macchina per scrivere
1851	Scrittura rongorongo nell'Isola di Pasqua
1883	Penna stilografica
1886	Linotype
1889	Monotype
1904	Stampa offset su carta
1938	Penna biro
1980	Prima fonderia digitale
1982	Internet
1982	Prima versione di T _E X
1985	Prima versione di L ^A T _E X

roglifici avevano valore fonetico; ancor oggi sono usate forme di scrittura logografica (come il sistema numerico), mentre cinesi e giapponesi moderni resistono con forza ad adottare sistemi alfabetici.

La scrittura è una tecnologia: è stata inventata, raffinata, si è imposta e diffusa. Secondo le ricerche più recenti, l'invenzione della scrittura è sorta nel mondo (cioè inventata da zero, in modo completamente indipendente) almeno due volte: in Mesopotamia, a opera degli antichi Sumeri, intorno al 3400 a.C., e nel sud del Messico, ai tempi dell'impero maya, intorno al 600 a.C.

Alcuni studiosi ritengono che siano originali anche l'invenzione dei geroglifici fatta in Egitto attorno al 3000 a.C., della scrittura fatta in Cina intorno al 1300 a.C. e della scrittura rongorongo (tuttora non decifrata) usata dai nativi polinesiani nell'Isola di Pasqua in tempi moderni (il reperto più antico è del 1851). È tuttavia probabile che la comparsa della scrittura in questi luoghi sia dovuta al contatto con altre civiltà alfabetizzate.



Figura 42: Tavoletta con iscrizioni in rongorongo, scavata con un dente di squalo, nell'Isola di Pasqua

c.1.2 Scrittura cuneiforme

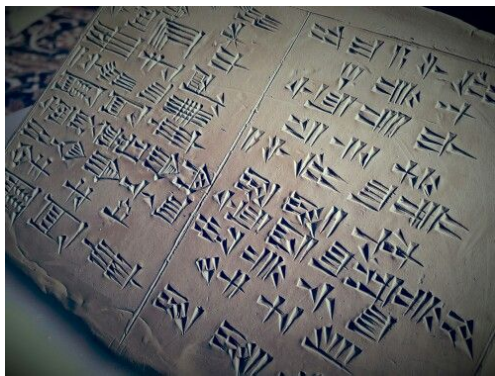


Figura 43: Il Codice di Hammurabi (XVIII secolo a.C.), scolpito in caratteri cuneiformi, è una fra le più antiche raccolte di leggi scritte a noi pervenute

si servivano di un oggetto appuntito (di solito la punta di una canna) per incidere la creta.

Le tavolette più antiche sono semplici elenchi. Ci troviamo, per esempio, un disegno che rappresenta una pecora e poi un numero; poi il simbolo di un volatile e un altro numero, eccetera. Col passare del tempo i disegni diven-

Cominciamo dalla prima delle scritture, quella cuneiforme. Attorno al 3400 a.C. i Sumeri in Mesopotamia cominciarono a tener traccia per iscritto dei loro commerci. Avevano un modo standard per scrivere i numeri e lo usavano per registrare su tavolette d'argilla l'andamento della produzione del grano e dell'allevamento delle pecore. Migliaia di tavolette sono state ritrovate in eccellenti condizioni nelle rovine della antica città di Uruk sul fiume Eufrate, a circa trecento chilometri da Bagdad.

La scrittura cuneiforme deriva il proprio nome dal latino *cuneus* (chiodo), in quanto i primi incisori

tano simboli, sempre più stilizzati. Poi piano piano sorgono simboli astratti. Per esempio il segno che significa *testa* e il segno che significa *pane* vengono combinati e nasce un segno che significa *mangiare*. I Sumeri inventano così i *logogrammi*: simboli grafici che rappresentano una parola (un nome o un verbo). Il sistema dei logogrammi è il complesso modo di scrivere che usano ancor oggi i cinesi e i giapponesi: i logogrammi di quelle due lingue in italiano si chiamano *ideogrammi*, *hanzi* in cinese e *kanji* in giapponese (anche i coreani fino al XV secolo d. C. usavano degli ideogrammi, detti *hanja*).

I Sumeri fanno un altro passo avanti: cominciano a usare una rappresentazione fonetica. In lingua sumera la parola che si pronuncia *tai* ha due significati: *freccia* e *vita*. Nelle tavolette sumere qualcuno comincia a disegnare una freccia tutte le volte che deve scrivere *vita*. Poco dopo, i Sumeri cominciano a disegnare una freccia per rappresentare anche singole parti di parole, quelle che contengono la sillaba *tai*. Immaginate che per scrivere in italiano la parola *restia* si disegni una testa coronata e poi una gabbia che contiene due volatili. È il principio del rebus.



Figura 44: Scrittura maya

L'alfabeto cuneiforme che si incontra verso il 3000 a. C. è un complicato miscuglio di logogrammi, simboli sillabici e lettere. Non si semplificherà mai; forse volontariamente, perché sapere è potere, e la casta degli scribi godeva di notevoli vantaggi nel restare poco numerosa.

c.1.3 Scrittura maya

L'altra scrittura originale, quella maya, ha un alfabeto composto da sillabe. Ogni segno rappresenta una sillaba della parola (per esempio, ci sono cinque segni per rappresentare *ta*, *te*, *ti*, *to* e *tu*). Anche in questo caso si arriva ai segni con un processo di astrazione: in lingua maya la parola *coda* si dice *neh*, quindi la sillaba *ne* viene rappresentata dal disegno stilizzato di una coda. Ancor oggi ci sono alfabeti sillabici sulla Terra: in Giappone, per esempio, accanto agli ideogrammi *kanji* si usano due alfabeti di questo tipo, il *katakana* e lo *hiragana*, che costituiscono il livello minimo di conoscenza per tutti i giapponesi e che sono impiegati fra l'altro per traslitterare i nomi stranieri e per scrivere al computer.

c.1.4 Sistemi logografici e alfabetici

I sistemi di scrittura logografici sono molto difficili da imparare. Un bambino giapponese quando entra nella prima classe elementare si vede assegnare un compito a casa: imparare a memoria alcuni ideogrammi entro qualche giorno. L'incarico si ripeterà poi regolarmente, sinché il giovane non completa le scuole superiori. A colpi di gruppi di ideogrammi da studiare a memoria, impara a scrivere.

Gli alfabeti sillabici sono decisamente più semplici, ma l'innovazione che rende l'alfabeto davvero facile a dominarsi e quindi alla portata di chiunque è l'alfabeto fonetico, in cui ogni segno rappresenta un suono. Sono alfabeti pressapoco fonetici tutti quelli più diffusi oggi al mondo: il nostro alfabeto latino, il cirillico usato nell'Europa dell'Est, il greco, l'arabo, l'ebraico.

愛	和	美
amore	armonia	bellezza
仁	清	慈
benevolenza	chiarezza	compassione
英	氣	永
coraggio	energia	eternità
福	華	恩
felicità	gloria	grazia
明	實	貴
intuizione	onestà	onore
恕	孚	德
perdono	verità	virtù

Figura 45: Ideogrammi giapponesi

Diciamo *pressappoco* perché in realtà noi italofoni usiamo gruppi di lettere per rappresentare un singolo suono. Per esempio, in italiano noi usiamo la combinazione di lettere *g* e *n* per rappresentare il suono centrale della parola *signore*, ma quel suono non è lo stesso suono che si ottiene avvicinando la *g* di *gatto* alla *n* di *noce*. Lo stesso principio vale per *g* e *l* (il suono di *foglia*) e per *s* e *c* (il suono di *scettro*). Nello spagnolo, che usa il nostro stesso alfabeto, il suono *gn* si scrive con una tilde sopra la lettera *n*, così: *señor*. Nel cirillico (che ha più lettere del latino, ben trentatré) il suono *sc* si scrive *ш*. Il cirillico, inventato nella seconda metà del IX secolo d. C., è sostanzialmente un perfetto alfabeto fonetico per il russo, nel senso che ha un segno per ogni suono esistente nella lingua russa.

Perché l'alfabeto latino è meno adatto all'italiano di quanto il cirillico sia adatto al russo? Perché è un alfabeto più antico, nato per un'altra lingua (il latino, appunto) e poi adattato alle lingue posteriori come la nostra.

L'alfabeto latino era un perfetto alfabeto fonetico per i contemporanei di Cicerone. Chi ha visto un'epigrafe latina sa che i Romani scrivevano con la stessa lettera sia la consonante *V* che la vocale *U*. Il motivo è semplice: in latino non c'era la *V*, e tutte quelle che a noi sembrano *V* sono delle *U*. Per esempio, la famosa frase di Giulio Cesare *veni, vidi, vici* — che significa *venni* (sul campo di battaglia), *vidi* (l'andamento della battaglia) e *vinsi* — veniva pronunciata dai Romani dell'epoca dei Cesari *ueni, uidi, uiki*. Ai Romani mancava anche la nostra *c* morbida: avevano solo la *c* dura, ecco perché nel latino classico non c'è la lettera *kappa*. Anche nello spagnolo moderno manca il suono *v*, come nella nostra comune antenata, la lingua latina. Gli spagnoli bevono come noi il vino, ma lo pronunciano pressappoco *bino*.

Il più semplice di tutti gli alfabeti possibili è un alfabeto perfettamente fonetico. Un esempio da manuale ci viene dalla Turchia. Per motivi religiosi, la lingua turca deve molto all'arabo. Tradizionalmente, il turco si è sempre scritto usando quell'alfabeto. L'attuale alfabeto turco è stato però ideato a tavolino nel 1928 da un gruppo internazionale di linguisti, per volere del "padre della patria" Mustafa Kemal Atatürk, che voleva occidentalizzare il proprio Paese. Il turco usa le lettere latine (meno *q*, *w* e *x*) e aggiunge i

Tabella 70: Scrittura lineare B

Segno	Valore	Segno	Valore	Segno	Valore	Segno	Valore
	da				ai		
	ro		mu		ke		ju
	pa		ne		de		ta ₂
	te		a		je		ki
	to		ru				ro ₂
	na		re		nwa		tu
	di		i				ko
	a		pu ₂		pu		pe
	se		ni		du		mi
	u		sa		no		ze
	po		qo		ri		we
	so		ra ₃		wa		ra
	me				nu		ka
	do				pa ₃		qe
	mo		jo		ja		zu
	pa ₂		ti		su		ma
	za		e		ta		
			pi		ra		
			pi		o		
	zo		si		pte		
	qi		wo				

Tabella 71: Geroglifici egizi

Segno	Valore	Nome	Segno	Valore	Nome
	<i>a</i>	avvoltoio <i>aleph</i>		<i>h</i>	treccia
	<i>i</i>	canna <i>yod</i>		<i>h</i>	setaccio
	<i>y</i>	due canne doppio <i>yod</i>		<i>h</i>	coda
	<i>'</i>	braccio <i>ajin</i>		<i>s</i>	stoffa
	<i>w</i>	pulcino <i>waw</i>		<i>š</i>	stagno
	<i>b</i>	piede		<i>k</i>	pendio
	<i>p</i>	stuoia		<i>k</i>	cesto
	<i>f</i>	vipera		<i>g</i>	vaso
	<i>m</i>	civetta		<i>t</i>	focaccia
	<i>n</i>	acqua		<i>t</i>	briglia
	<i>r</i>	bocca		<i>d</i>	mano
	<i>h</i>	tettoia		<i>d</i>	cobra

simboli *ü, ç, ö, ş, ğ, ı*. Le regole per la pronuncia sono semplicissime: a ogni lettera corrisponde uno e un solo suono (per esempio, la lettera *c* si pronuncia come la nostra *g* morbida, mentre la *ç* si pronuncia come la nostra *c* morbida). Atatürk vinse la scommessa, perché la semplice corrispondenza tra il modo in cui una parola si pronuncia e il modo in cui si scrive nel turco moderno permise ad amplissimi strati della popolazione di imparare a leggere e a scrivere.

c.1.5 Scritture dell'Egeo






























Con questo nome si indicano le scritture dei popoli che si affacciavano sul Mar Egeo nel II millennio a. C., che ci hanno lasciato documenti composti su tavolette.

GEROGLIFICI CRETESI Si tratta di una scrittura logografica ancora non decifrata, databile fra il 2000 e il 1000 a. C. circa.

LINEARE A Alla scrittura logografica si affiancò una scrittura sillabica, anche questa ancora non decifrata malgrado numerosi tentativi.

LINEARE B Sviluppata fra il 1400 e il 1150 a. C., questa scrittura è stata decifrata, è scritta da sinistra a destra e conta circa 200 segni, di cui una novantina sono segni sillabici con valore fonetico (si veda la tabella 70 nella pagina precedente) mentre i rimanenti sono logogrammi.

Tabella 72: Alfabeto di Ugarit

Segno	Valore	Nome	Segno	Valore	Nome
	<i>a</i>	aleph		<i>d</i>	<i>d</i>
	<i>b</i>	beth		<i>n</i>	nun
	<i>g</i>	gimel		<i>z</i>	
	<i>h</i>			<i>s</i>	samekh
	<i>d</i>	dalet		<i>'</i>	ayin
	<i>h</i>	he		<i>p</i>	pe
	<i>w</i>	vau		<i>ṣ</i>	sade
	<i>z</i>	zayin		<i>q</i>	qoph
	<i>h</i>	het		<i>r</i>	resh
	<i>t</i>	tet		<i>t</i>	
	<i>y</i>	yod		<i>g</i>	
	<i>k</i>	kaph		<i>t</i>	tav
	<i>l</i>	lamed		<i>i</i>	
	<i>m</i>	mem		<i>u</i>	
				<i>ṣ</i>	

c.1.6 Scritture dell'Egitto

La prima scrittura che si afferma in Egitto, com'è noto, è la *geroglifica* (dal greco *ἱερός*, pron. *ierós*, "sacro", e *γλύφω*, pron. *glifo*, "incidere"): di solito i geroglifici erano incisi sulla pietra e usati come scrittura monumentale.

All'inizio ciascun geroglifico rappresentava un'idea o un fatto; una successione di geroglifici sviluppava un discorso. Gli Egizi si dovettero però accorgere presto che non si poteva procedere su questa linea, perché l'elevato numero degli ideogrammi diveniva ingestibile. Accanto agli ideogrammi comparvero dei fonogrammi (a ogni fonogramma era associato un suono).

La scrittura geroglifica egizia comprende 24 caratteri principali (simboli per un singolo suono: si veda la tabella 71 nella pagina precedente), cui si aggiungono svariati segni biconsonantici (simboli per due suoni combinati) e triconsonantici (tre suoni). In totale, la scrittura geroglifica era composta da quasi *settemila* caratteri. Pur con l'elevato numero di simboli, la capacità espressiva era molto limitata, non permettendo di esporre concetti e svolgere ragionamenti.

c.1.7 Alfabeto di Ugarit

Ugarit è un'antica città del Vicino Oriente, sulla costa della Siria. Fu la capitale di un regno fiorito dal 1500 al 1200 a.C., che confinava a nord con la potenza ittita, a est con la Mesopotamia e a sud con l'Egitto. Ugarit è la città da cui nascono le tradizioni socio-culturali dei Fenici.

L'alfabeto di Ugarit, sviluppato intorno al 1400 a.C., è uno dei primi alfabeti del Vicino Oriente (si veda la tabella 72). Comprende una trentina di caratteri cuneiformi, ordinati in una sequenza che si ritrova nella maggior

Tabella 73: Alfabeto fenicio (*vau* si scriveva a volte come F)

Segno	Valore	Nome	Segno	Valore	Nome
𐤀	<i>a</i>	aleph	𐤁	<i>l</i>	lamed
𐤂	<i>b</i>	beth	𐤃	<i>m</i>	mem
𐤄	<i>g</i>	gimel	𐤅	<i>n</i>	nun
𐤆	<i>d</i>	dalet	𐤇	<i>s</i>	samekh
𐤈	<i>h</i>	he	𐤉	<i>ʿ</i>	ayin
𐤊	<i>w</i>	vau	𐤋	<i>p</i>	pe
𐤌	<i>x</i>	zayin	𐤍	<i>ṣ</i>	sade
𐤎	<i>ḥ</i>	het	𐤏	<i>q</i>	qoph
𐤐	<i>t</i>	tet	𐤑	<i>r</i>	resh
𐤒	<i>y</i>	yod	𐤓	<i>š</i>	shin
𐤔	<i>k</i>	kaph	𐤕	<i>t</i>	tav

parte degli alfabeti successivi (greco, etrusco, latino). Dopo la distruzione di Ugarit a opera dei “Popoli del Mare”, la scrittura ugaritica cessò di essere usata ma l’alfabeto, che ispirò quello fenicio, si diffuse in Europa.

c.1.8 Alfabeto fenicio



Figura 46: Stele fenicia ritrovata a Nora, in Sardegna

I Fenici abitavano, dal 1200 a.C., in una striscia di terra compresa tra la Siria e il Libano. Erano pro-vetti marinai, e perciò avevano contatti con molti popoli, così che poterono maturare delle conoscenze pressoché uniche per quei tempi. Il loro commercio era molto florido e c’erano colonie fenicie sparse in tutto il bacino del Mediterraneo.

La notevole idea dei Fenici fu di assegnare a ciascun segno grafico un suono, il cui valore resta costante e invariabile. Byblos, una prospera città fenicia, fu il centro di diffusione di questa nuova invenzione.

L’alfabeto fenicio, che risale al 1000 a.C., era composto da 22 lettere (si veda la tabella 73), che prendevano il nome da oggetti. Per esempio, le prime due lettere erano *aleph* (bue) e *beth* (casa). Non c’erano vocali, che venivano aggiunte durante la lettura. C’erano solo le lettere maiuscole. I Fenici scrivevano da destra a sinistra.

L’alfabeto fenicio ha dato origine all’alfabeto greco e a quello aramaico. Questi a loro volta hanno portato ai sistemi di scrittura usati in tutte le regioni che vanno dall’Asia occidentale verso l’Africa e l’Europa.

c.1.9 Altri alfabeti

Nel IX secolo a.C. i Greci adottarono l’alfabeto fenicio e vi aggiunsero le vocali. All’inizio il greco si scriveva da destra a sinistra, ma verso il VI secolo a.C. divenne *bustrofedico* (dal greco βούς, pron. *bus*, “bue”, e στρέφω, pron. *stréfo*, “volgere”): il verso delle righe era alternato, come quello dei

buoi che arano un campo. Nel VI secolo a.C. alla scrittura boustrophedica succede, dopo aver coesistito con essa, quella da sinistra a destra. I Greci mantennero il nome delle lettere fenicie, opportunamente grecizzato, così *aleph* e *bet* divennero *alfa* e *beta* (a quel punto il nome delle lettere non aveva più alcun significato). C'erano diverse varianti dei caratteri greci finché vennero definitivamente fissati ad Atene nel 403 a.C.; i Greci non svilupparono le lettere minuscole fino al 700-600 a.C.

Deriva dal fenicio anche l'aramaico, una lingua dalla lunga storia (fu, tra l'altro, la lingua ufficiale dell'impero persiano), da cui sono derivati l'ebraico (VI secolo a.C.), l'arabo (V secolo d.C.) e il Devanagari, usato in diverse lingue dell'India (VII secolo d.C.). L'ebraico e l'arabo sono lingue semitiche che si scrivono da destra a sinistra.

Concludiamo questa velocissima carrellata storica ricordando che il nostro attuale alfabeto deriva dal latino, che a sua volta discende dall'etrusco che originò dal greco. Gli etruschi scrivevano da destra a sinistra, così i caratteri presi in prestito erano immagini speculari di quelli greci; i Romani, che scrivevano da sinistra a destra (sebbene le prime iscrizioni romane fossero boustrophediche), li rovesciarono nuovamente.

All'inizio l'alfabeto latino era composto esclusivamente da lettere maiuscole e solo successivamente si ebbero altre scritture (si veda il paragrafo C.1.12 a pagina 368). La scrittura era detta *capitale quadrata*: il nome (dal latino *caput*, "capo" ma anche "capitolo") deriva dal fatto che quando, nel Medioevo, cadde in disuso, restò per i titoli delle opere e dei capitoli.



Figura 47: Iscrizione sul Pantheon di Roma (I secolo a.C.)

C.1.10 Materie

Una materia scrittoria diffusissima nell'antichità fu l'*ostraca* (dal greco ὄστρακα, pron. *óstraca*, plurale di ὄστρακον, pron. *óstrakon*, che significa "conchiglia" ma anche "frammento"). Si tratta di resti di vasi di terracotta, scritti nella parte interna (si usavano, fra l'altro, come schede elettorali nelle procedure di *ostracismo*).

I Romani scrivevano su tavolette cerate, costituite da un supporto che poteva essere di pietra, legno o avorio, con un bordo rialzato su cui veniva colata la cera incisa poi con lo *stilo*, un'asta (per lo più di osso) dalla punta aguzza.

Un supporto a lungo diffuso fu il *papiro*, ricavata dalla corteccia della pianta dall'omonimo nome. Questo veniva scritto su un solo lato, e per essere conservato era arrotolato su asticcioli di cedro e riposto in custodie di legno. Da qui origina il termine *volume*, dal latino *volvere*, perché per leggere bisognava srotolare il papiro.

Con l'avvento dell'era cristiana il supporto scrittorio muta forma. Si passa infatti dal *volumen* al *codex*, costituito da fascicoli cuciti assieme, avvicinandosi alla forma che ancor oggi conserva. I vantaggi sono evidenti: il documento

era più maneggevole (non andava più srotolato e si riponeva facilmente) e per la prima volta poteva essere scritto davanti e dietro. Prendevano così forma anche nel libro, com'era fino ad allora solo per le monete, il *recto* e il *verso*. È da questa nuova organizzazione che prendono forma consuetudini protrattesi fino a oggi, come la divisione dell'opera in sezioni.

Da Pergamo, città dell'Asia Minore dove il supporto venne realizzato la prima volta nel II secolo a. C., deriva invece il nome della *pergamena*. Questa era fabbricata con pelle di pecora, di vitello o di capra. Le pelli venivano lavate con acqua bollente, sgrassate con calce, lisceate con pietra pomice e infine distese su appositi telai e fatte seccare al sole. La pergamena, al contrario del papiro, era scritta sui due lati.



Figura 48: Manoscritto miniato

La pergamena domina in tutto il Medioevo, imponendo la forma al libro che viene quasi sempre realizzato *in quarto* (si otteneva piegando due volte un foglio intero, prima lungo il lato minore e poi su quello maggiore, ricavando così quattro carte, pari a otto pagine). Il libro è scritto su colonne (due o tre, di solito). Non ha frontespizio e neppure titolo, e solo raramente presenta

nelle sue ultime pagine l'indicazione del luogo e dell'anno di esecuzione (il cosiddetto *colophon*).

La pergamena, per la sua capacità di resistere nel tempo, fu un fattore determinante nello sviluppo del libro. Una variante della pergamena fu il *palinsesto*: con questo termine si indicava la pergamena su cui era stata raschiata la prima scrittura per poterci scrivere nuovamente.

La materia scrittoria per eccellenza è comunque la carta. Inventata dai Cinesi e usata dai Giapponesi già nel VII secolo, fu introdotta dagli Arabi in Africa e in Europa nell'VIII secolo nelle terre da loro occupate. Per una produzione industriale della carta bisogna attendere il XII secolo, quando a Fabriano sorse la prima cartiera che impose la sua produzione in tutta Europa.

Altri passi in avanti nel processo di produzione della carta si ebbero prima a opera dei cartai olandesi, che introdussero metodi di fabbricazione usati ancor oggi, salvo poche varianti, e poi grazie all'idea di affiancare al procedimento di produzione naturale fino ad allora impiegato un processo chimico, usando la cellulosa ricavata dall'abete e dal pioppo.

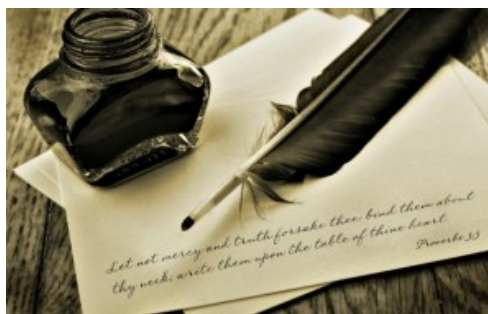


Figura 49: Carta, penna e calamaio

c.1.11 Strumenti

Dello stilo s'è già detto. Lo scolaro romano teneva la tavoletta cerata sulle ginocchia, seguendo gli insegnamenti del maestro che insegnava la grammatica e l'arte della scrittura.



Figura 50: Forme scritte

Per tracciare segni sul supporto si usavano pennelli, penne di volatili e calami (pezzi di canna o giunco con un'estremità appuntita). Dalla custodia del calamo prese poi il nome l'odierno *calamaio*, il recipiente che conteneva il liquido tintore, l'*inchiostro*, parola che in greco significa *bruciato*. Nel Medioevo il liquido era detto *encaustum* (da cui deriva l'inglese *ink*), a significare che era prodotto al fuoco.

Nell'inchiostro c'erano vari materiali: fuliggine, resine, mosto d'uva, nerofumo disciolto in gomma, polvere di seppia, anilina. Per gli inchiostri colorati si usavano, fin dall'antichità, il cinabro, il minio, il tannino, la biacca. La composizione sostanzialmente naturale di questi inchiostri permetteva di cancellarli quando erano ancora freschi, con un panno imbevuto d'acqua.

Nel Medioevo compaiono altri elementi quali birra, aceto, noce di galla e addirittura vetriolo, mentre cominciano a delinearsi le tecniche per la composizione di inchiostri per il colore: il rosso, il verde e il giallo-oro. Attualmente l'inchiostro è prodotto mediante processi chimici.

Nel Rinascimento avanzato (metà Cinquecento) si diffuse l'uso di bastoncini di grafite (le matite) che quasi subito soppiantarono le lamine di piombo e stagno usate nel Medioevo.

C.1.12 Forme

Tralasciamo l'evoluzione della scrittura dopo che l'alfabeto fenicio si diffuse nel Mediterraneo, per saltare a un'epoca relativamente vicina considerando la vasta misura temporale su cui si sta lavorando, per arrivare al periodo romano. Qui troviamo imperante la scrittura capitale quadrata cui si era accennato. Questa scrittura, nata per essere tracciata su pietra e basata su quadrati, decadde con l'impero e sopravvisse solo nei titoli delle opere.

In seguito si affacciano altre forme di scrittura conducendo gradatamente al modo moderno di scrivere. Qui di seguito se ne descrivono alcune.

SCRITTURA ONCIALE Dalla scrittura romana nacque in Africa nel III secolo d.C. un carattere detto *onciale* (dal latino *uncia*, "uncia", unità di misura di circa due centimetri e mezzo, da cui deriva l'inglese *inch*). Le lettere, ancora maiuscole, sono però arrotondate e facili da tracciare con una penna d'oca, su papiro o pergamena, vicine al nostro corsivo. Fu diffusa fino al IX secolo.

SCRITTURA CAROLINGIA Dopo la caduta dell'Impero Romano, il primo regno a riemergere dalle rovine fu quello di Carlo Magno che diede i natali, intorno alla metà del VIII secolo, a un nuovo carattere che da lui prese il nome. La scrittura *minuscola carolingia*, semplice ed elegante, si diffuse in tutta Europa fino alla comparsa dei caratteri da stampa introdotti nel Rinascimento.

SCRITTURA GOTICA Nei Paesi del nord Europa si diffuse, a partire dall'XI secolo, una scrittura che perse l'inscrivibilità delle lettere nel quadrato, tipica sia dello stile romano che di quello carolingio, per assumere forme verticali, compatte e spigolose, di non agevole lettura; questo stile, denominato *gotico*, risente delle forme architettoniche dell'epoca.

SCRITTURA UMANISTICA Gli umanisti italiani, che non amavano le spigolosità e i tratti cupi del gotico, si ispirarono ai caratteri della scrittura carolingia, dando vita nel XIV secolo a una scrittura raffinata e armonica, che noi definiamo *umanistica* e che loro nominarono *littera antiqua*, in contrapposizione al gotico che chiamavano *littera moderna*.

C.2 NASCITA DELLA TIPOGRAFIA

Dopo l'anno mille la richiesta di documenti e libri s'era fatta pressante, specie in relazione al sorgere delle prime università. Si disponeva di carta in quantità sufficiente, ma il lavoro dei copisti non era più in grado di soddisfare le richieste.



Figura 51: Pagina della Bibbia di Gutenberg

S'iniziarono così a *stampare*, il termine non è improprio, a pressa, i primi documenti. Questi lavori erano composti su tavole di legno, che venivano incise e poi stampate su carta (o su seta), con una tecnica chiamata *xilografia*.

La tecnica, di origine cinese (le prime stampe su stoffa risalgono al V secolo d. C.), si sviluppò con la diffusione della carta. All'inizio si usavano legni morbidi e facili da lavorare: ciò facilitava il lavoro degli incisori, ma le matrici si deteriorano velocemente (nel XVI secolo il legno verrà abbandonato, sostituito dalle matrici in metallo).

La nascita vera e propria della *tipografia* (dal greco τύπος, "segno", e γράφω, "scrivere") è dovuta a un orafo tedesco di Magonza, Johann Gutenberg, che ebbe l'intuizione di prendere strumenti e metodi già esistenti (fra cui una caratteristica della

propria arte, ovvero punzonare i monili) e applicarli alla stampa, servendosi di matrici in rilievo composte da *caratteri mobili*.

L'idea di Gutenberg era semplice: a ogni lettera dell'alfabeto corrispondeva un *carattere* (dal greco *χαρακτήρ*, pron. *caractér*, "impronta"), un blocchetto forgiato in una lega di piombo, in rilievo e invertito, di dimensioni pressoché costanti; per comporre ciascuna riga di testo si sceglievano uno a uno i caratteri corrispondenti; una volta che tutte le righe di una pagina erano state composte, i caratteri venivano legati strettamente per tenerli fermi, e la matrice era pronta. In questo modo si potevano correggere gli errori senza impostare di nuovo l'intera pagina. La matrice veniva poi ricoperta d'inchiostro (adeguato ai caratteri in metallo, non più ad acqua ma a olio) con pennelli di crine di cavallo e si posizionava su di essa una pagina, che veniva compressa con un torchio di legno, modellato sul torchio da vino dei coltivatori renani.

Il libro non esauriva la sua lavorazione con la stampa. Ci potevano essere interventi manuali successivi quali miniature e capilettera.

Le righe della Bibbia di Gutenberg erano giustificate, come nei migliori manoscritti. Per ottenere la giustificazione, Gutenberg non usò spazi di dimensione variabile tra le parole, ma distribuì segni di punteggiatura più o meno larghi, impiegò delle legature e sostituì alcune parole con le loro abbreviazioni.

Gutenberg ebbe successo con la stampa di 180 copie della famosa Bibbia a quarantadue righe, che si concluse nel 1455, dopo tre anni di lavoro, un periodo in cui un amanuense avrebbe portato a termine la riproduzione di una sola copia. L'invenzione impiegò tempo a farsi strada — da Magonza (1455) a Roma (1467), Venezia (1469), Parigi (1470), Londra (1476) e Vienna (1482) — tanto che all'inizio la tipografia si presentò come un'arte girovaga, perché non avendo clientela fissa i primi stampatori si spostavano dove c'era lavoro.

Comunque sia, agli inizi del Cinquecento l'Europa è ormai permeata dell'invenzione di Gutenberg. In quel tempo, oltre duecento città avevano già prodotto i cosiddetti *incunaboli*, libri che derivano il loro nome dal latino *in cuna*, "neonato", "in fasce", alludendo al metodo con cui i caratteri mobili venivano legati. I caratteri usati erano essenzialmente di due tipi: umanistici per i libri classici e gotici per quelli religiosi.

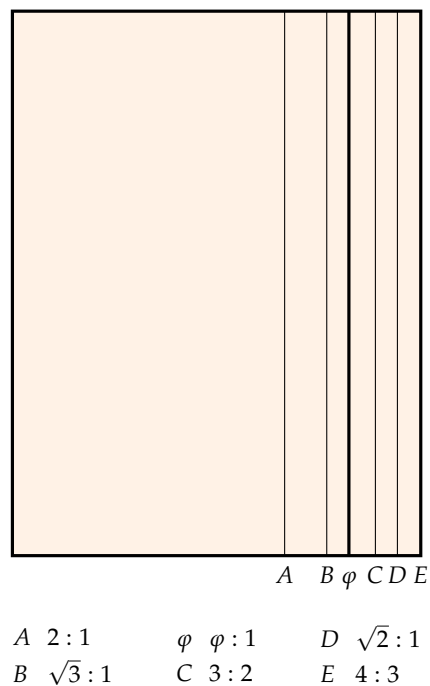


Figura 52: Proporzioni di pagina

c.2.1 Formato del libro

La tipografia non poteva prescindere dal contributo che durante il basso impero e per tutto il Medioevo gli amanuensi avevano portato in questo campo, individuando nelle opere copiate pazientemente le proporzioni di pagina adeguate al carattere usato, che era di un corpo straordinariamente grande rispetto a quello odierno.

Con l'introduzione di un carattere dal corpo contenuto i libri assunsero dimensioni ridotte, e ciò contribuì a farli divenire oggetti non più riservati alle biblioteche dei conventi e di poche case patrizie, ma accessibili anche alla borghesia. Questo processo si completò solo nel XIX secolo.

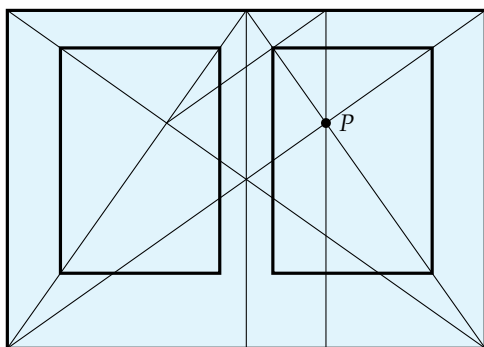


Figura 53: Pagina-tipo di Gutenberg

L'individuazione della forma rettangolare della pagina e del rapporto altezza/larghezza fu dunque mutuata da opere che a loro volta avevano risentito dell'influsso delle misure delle tavolette cerate usate dagli scolari dell'antica Roma, un migliaio di anni prima. I primi tipografi non si basarono però solo sulla tradizione dei formati librari degli amanuensi, ma cercarono proporzioni di pagina che fossero gradevoli e di facile lettura, permettendo nel contempo di

economizzare la carta, materiale non diffuso quanto oggi.

Nel fervore rinascimentale di studio del mondo classico, era stata fra l'altro recuperata la *sezione aurea* (la parte di un segmento che è media proporzionale tra il segmento stesso e la parte che ne resta), ribattezzata *divina proporzione*. Il rapporto tra un segmento e la sua sezione aurea si indica con φ , che vale circa 1,618. C'è una formula suggestiva che esprime φ usando solo il numero 1:

$$\varphi = 1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{1 + \frac{1}{\dots}}}}$$

Su questo rapporto altezza/larghezza si fondò la costruzione di molti libri, anche se furono ammessi anche altri formati (si veda la figura 52 a fronte). Secondo gli standard ISO attuali, il rapporto altezza/larghezza del foglio è pari a $\sqrt{2}$, che vale circa 1,414.

La figura 53 rappresenta un esempio storico usato da Gutenberg nei propri libri, assieme alla costruzione geometrica che mostra come si pervenne a quelle impostazioni.

c.2.2 Tipografia in Italia

È a Venezia che si assiste al massimo splendore della tipografia italiana nel Cinquecento. Fra tutti, emerge il nome di Aldo Manuzio, che fondò nella città il più grande centro tipografico d'Europa, introducendo nella stampa il carattere corsivo (1501), che da allora gli anglosassoni chiamarono (bontà loro) *italico*. Il corsivo presentava alcuni caratteri innovatori, tra cui il puntino sulle *i* e i trattini obliqui aggiunti ai tratti discendenti delle lettere.

Manuzio fu anche il primo a usare il formato detto *ottavo* (ottenuto piegando tre volte un foglio intero) e a porre in commercio libri rilegati, mentre precedentemente all'acquirente venivano fornite opere in fogli sciolti.

Oltre a Venezia, anche Roma ebbe un ruolo fondamentale nello sviluppo della tipografia italiana del Cinquecento, grazie anche a Giambattista Palatino. Maestro della calligrafia, Palatino è autore del più noto trattato di scrit-

tura del Rinascimento, il *Libro nuovo d'imparare a scrivere* (1540). I caratteri disegnati da Palatino si diffusero in tutta Europa.

I primi esemplari di libri a stampa, ricalcando nella struttura i manoscritti dei copisti, non avevano frontespizio e iniziavano con la dedica e l'indice: talvolta subito con il testo preceduto dal verbo *Incipit*, da *incipere*, cioè *inizia*. Il titolo compare stabilmente sulla fine del Cinquecento, mentre soltanto nel secolo successivo si prende l'abitudine di numerare le pagine.

c.2.3 Tipografia in Europa

Adattando modelli italiani di caratteri, il francese Claude Garamond disegna nel 1530 quello che diventerà il prototipo europeo di carattere, che verrà usato insieme al corsivo.

Se i primi incunaboli cercavano di presentarsi, per forma dei caratteri e disposizione generale, come i manoscritti, nel XVI secolo l'editoria comincia a essere un'industria matura e si sforza di affrancarsi dall'eredità del passato. Così le righe si spaziano, i caratteri si rimpiccioliscono e la presentazione dei testi mira alla leggibilità. Così, nel Settecento, l'inglese John Baskerville (1757), i francesi François e Firmin Didot (1781) e l'italiano Giambattista Bodoni (1798) introducono nuovi caratteri ispirati a rigorose proporzioni geometriche.

c.2.4 Sviluppi tecnologici

Nel 1796 nasce la *litografia* (dal greco λίθος, pron. *lithos*, "pietra", e γράφω, pron. *gráfo*, "scrivere"), un metodo di stampa basato sulla repulsione tra acqua e sostanze grasse. Il principio è semplice: un particolare tipo di pietra, levigata e poi disegnata con una matita grassa, ha la peculiarità di trattenere nelle parti non disegnate un sottile velo d'acqua, che il segno grasso invece respinge. L'inchiostro passato sulla pietra così trattata è respinto dalle parti inumidite e trattenuto dalle parti grasse. Al torchio, perciò, il foglio di carta riceve solo l'inchiostro che si deposita sulle parti disegnate e non sulle altre.

Con la rivoluzione industriale del XIX secolo anche la tipografia compì notevoli progressi. All'inizio del secolo la pressa in legno, rimasta sostanzialmente immutata dai tempi di Gutenberg, fu sostituita dalla pressa con struttura di metallo.

La prima pressa piano-cilindrica a vapore fu realizzata nel 1814 per il Times di Londra; questa tecnica permise di aumentare la capacità di stampa da poche centinaia a oltre mille copie all'ora. Sempre al Times venne introdotta pochi anni dopo, nel 1828, una macchina in grado di produrre diverse migliaia di copie all'ora.

Un decisivo passo in avanti si ebbe con l'introduzione della rotativa (1843), una macchina in cui il testo veniva stampato su un cilindro meccanizzato (all'inizio a vapore e in seguito elettricamente), con un movimento di rotazione continuo e un'alimentazione a bobina anziché a fogli singoli.



Figura 54: Caratteri mobili in una tipografia del XX secolo

Gli sviluppi tecnologici portano nel 1886 alla realizzazione della Linotype, basata sulla fusione di intere righe già composte, e poco dopo, nel 1889, alla Monotype, basata sulla fusione non delle righe ma dei singoli caratteri. Sempre alla fine dell'Ottocento diventa standard l'uso di carta da cellulosa.

La parte meccanica delle tecnologie di stampa subirà poi solo piccoli cambiamenti, fino alla diffusione della stampa offset negli anni Sessanta del secolo scorso. L'*offset*, metodo inventato nel 1875 (stampa su stagno) e adattato alla stampa su carta nel 1904, è una tecnica basata sullo stesso principio della litografia, con la differenza che l'inchiostro non viene trasferito dalla matrice (pietra o lastra) direttamente alla carta, ma è trasferito prima a un sistema di tre cilindri (un cilindro umidificatore, un cilindro inchiostatore di caucciù e un cilindro di pressione), e poi da questi alla carta.

c.2.5 Altri strumenti

Parallelamente allo sviluppo delle tecnologie di stampa si assiste a un'evoluzione degli strumenti per la scrittura.

Per molti secoli il principale strumento di scrittura è stata la penna d'oca intinta nel calamaio, di cui s'è già detto. Le prime notizie storiche di una penna a serbatoio d'inchiostro risalgono al X secolo. Nel 953 il califfo egiziano al-Mu'izz commissionò una penna che non macchiasse le mani e i vestiti, e fu provvisto di una penna con un serbatoio d'inchiostro collegato al pennino per gravità e capillarità. È tuttavia possibile che altri tentativi di usare una penna del genere vadano molto più indietro nel tempo.

Ci sono dei disegni di Leonardo da Vinci che descrivono una penna a serbatoio, ed è probabile che Leonardo l'abbia prodotta e se ne sia servito per i suoi lavori. Alcuni documenti attestano l'uso di penne di questo tipo nel XVII secolo, ma non ne è rimasto alcun esemplare. Nel 1780 vennero sviluppati alcuni prototipi in bronzo e corno.



Figura 55: Penna stilografica

Per ottenere una penna affidabile bisogna attendere il XIX secolo. La *penna stilografica* divenne uno strumento largamente diffuso in seguito a tre innovazioni: il pennino dorato con la punta in iridio (molto resistente), l'ebanite (facilmente lavorabile, fino ad allora usata per fabbricare parti di strumenti musicali) e l'inchiostro a flusso libero (che

permetteva una scrittura lineare e costante).

Nonostante l'esistenza di vari precursori, l'invenzione della moderna penna stilografica è generalmente attribuita allo statunitense Lewis Edson Waterman, che nel 1883 sviluppò il primo modello veramente funzionante. Il serbatoio, fino all'invenzione delle cartucce precaricate e sostituibili (1936), era riempito con un contagocce.

La *penna a sfera* è uno strumento per scrivere su carta che rilascia inchiostro da un serbatoio interno attraverso l'azione di rotolamento di una sfera metallica a contatto con la carta. La sfera, di diametro variabile, era in origine costruita in ottone (poi in acciaio o in carburo di tungsteno).

L'idea di usare una sfera all'interno di uno strumento di scrittura come metodo per applicare dell'inchiostro su carta risale al XIX secolo. Il primo

brevetto di una penna a sfera è stato depositato nel 1888 da John Jacob Loud, un conciatore di pelli statunitense che voleva costruire uno strumento in grado di scrivere sui suoi prodotti in pelle, cosa che le penne stilografiche di allora non potevano fare. La penna di Loud era composta essenzialmente da un tubo contenente l'inchiostro e da una piccola sfera rotante di acciaio inserita sulla punta. Anche se poteva essere usata per marcare superfici ruvide come il cuoio, si rivelò troppo grossa per scrivere sulla carta.

La fabbricazione di penne a sfera economiche e affidabili come le conosciamo ora risale al XX secolo, quando László Bíró, un giornalista ungherese frustrato dalla quantità di tempo speso a caricare le penne stilografiche e pulire le pagine macchiate, notò che gli inchiostri usati nella stampa dei giornali asciugavano rapidamente, lasciando la carta senza sbavature. Bíró combinò un inchiostro ad alta viscosità con un meccanismo di precisione, in modo da evitare che l'inchiostro si asciugasse all'interno del serbatoio, permettendo così un flusso controllato. Bíró brevettò la propria penna, da allora detta penna *biro*, nel 1938.



Figura 56: Penna biro



Figura 57: Caratteri di una macchina per scrivere

Sul finire del XIX secolo si diffuse la *macchina per scrivere*, inventata nel 1846, che fu uno dei primi strumenti di largo uso per comporre rapidamente documenti in formati standardizzati. Era dotata di una tastiera collegata a dispositivi (prima meccanici, poi elettrici ed elettronici) che permettevano di stampare su un

supporto, di solito un foglio di carta, caratteri simili a quelli tipografici (lettere, numeri, segni di punteggiatura). Gli accessori di uso più frequente erano la *gomma* (a forma di sottile dischetto, per rimuovere con precisione gli errori) e il *bianchetto* (per coprire gli errori e poter poi battere il carattere corretto).

C.3 TIPOGRAFIA DIGITALE

L'introduzione e lo sviluppo degli strumenti informatici comporta profondi cambiamenti anche nella tipografia e porta alla nascita, negli anni Settanta del secolo scorso, della cosiddetta *tipografia digitale*.

c.3.1 Software

Nasce un nuovo strumento *impalpabile*: il software, lo strumento scrittorio per eccellenza di questi ultimi decenni. Con il software, il tipografo dà istruzioni all'hardware del calcolatore che produce il documento finale pensato dall'autore.

La tastiera del computer sostituisce il gesto del vecchio tipografo di prendere i singoli caratteri dalla cassa e di posizionarli sul righello. Lo schermo sostituisce la matrice con i caratteri rovesciati che il tipografo inumidiva d'inchiostro e poneva sotto la pressa.

Grazie all'informatica, la tipografia si è affrancata da molte limitazioni di carattere tecnico. Mentre i costi di fabbricazione dei caratteri in piombo erano elevatissimi, con gli strumenti offerti dall'informatica l'unico costo è quello di progettazione.



Figura 58: Personal computer

In tipografia un carattere è detto *tipo*. In informatica una famiglia di caratteri si chiama *font*, termine che deriva dal francese medioevale *fonte*, che discende a propria volta dal latino *fundere*, che indica il processo di fusione attraverso cui si ottenevano i caratteri. Alcuni font moderni si ispirano a caratteri storici (come Garamond, Baskerville, Didot, Bodoni: si veda la figura 60 nella pagina seguente), altri sono nuovi. Negli anni Ottanta nascono le prime

“fonderie digitali” (la prima è la società Bitstream, fondata nel 1980), che distribuiscono direttamente i propri caratteri.

Sostanziali progressi per la tipografia digitale si devono all'introduzione di programmi specifici, come T_EX (1982) e L^AT_EX (1985).

Dopo il 2000 si afferma Unicode, uno standard informatico che permette di codificare, rappresentare e gestire praticamente tutti i sistemi di scrittura attualmente usati, comprese diverse lingue morte, l'alfabeto Braille e una vasta raccolta di simboli matematici, scientifici e cartografici (l'ultima versione contiene più di centomila caratteri appartenenti a più di cento alfabeti diversi).

c.3.2 Hardware

Il prodotto finale di questi processi è generalmente la stampante (sia essa una comune stampante per uso domestico o un plotter professionale da tipografia), che è una propaggine, una periferica del computer.

Una tappa fondamentale nel percorso evolutivo della stampa è rappresentato dalla *stampa digitale*, in cui la forma da stampare è generata attraverso processi elettronici e impressa direttamente sul supporto, senza l'uso di lastre o matrici. Rientrano in questo ambito le stampanti *ad aghi* (in cui degli aghi mossi da elettromagneti battono sulla carta attraverso un nastro inchiostroato,



Figura 59: Tablet

mentre si spostano lateralmente sul foglio), *a getto di inchiostro* (in cui una testina, muovendosi sulla superficie da stampare, rilascia l'inchiostro liquido attingendolo da un serbatoio) e *laser* (in cui l'inchiostro in polvere, detto *toner*, è attratto da un rullo caricato elettricamente e poi trasferito sulla carta).

Negli anni Ottanta e Novanta del secolo scorso, con la disponibilità di piccole stampanti ad aghi (in una prima fase) e poi a getto di inchiostro e a laser, si diffonde, prima negli uffici e poi nelle abitazioni, la pratica della



(a) Garamond



(b) Baskerville



(c) Didot



(d) Bodoni

Figura 60: Font moderni ispirati a caratteri storici

stampa personale: con competenze relativamente contenute i singoli utenti possono scrivere e stampare lettere, relazioni e testi a tiratura limitata. È la nascita del *desktop publishing* (dall'inglese "editoria da scrivania"), l'insieme delle tecniche di realizzazione di prodotti editoriali mediante l'impiego di un personal computer. Il *desktop publishing* ha sostituito in maniera pressoché totale le tecnologie precedenti, segnando una profonda rivoluzione della tipografia, a distanza di cinque secoli da Gutenberg.

Dopo il 2000 si diffondono nuovi hardware come i *tablet*, computer di dimensioni compatte, che usano come sistema di input uno schermo controllato da una penna o dalle dita, invece che una tastiera e un mouse. Il loro nome deriva dalla forma di tali dispositivi, che assomiglia a quella di una tavoletta usata per la scrittura. L'interfaccia grafica differisce da quella dei personal computer ed è molto simile a quella usata dagli *smartphone*.

c.3.3 Tipografia come modalità di manifestazione del pensiero

La tipografia, tanto quella classica quanto quella digitale, non ha solo un valore tecnico, ma anche umanistico, poiché è un modo di manifestazione del pensiero. La scrittura, così come il pensiero, se non viene comunicata non stimola la riflessione e non produce effetti sensibili. Se il testo scritto non è letto e assimilato, se rimane confinato nel cassetto della nostra scrivania, è lettera morta.

Poiché la scrittura è una forma di comunicazione, dev'essere presentata nel migliore dei modi, per risultare accattivante nei confronti del lettore. Anche se ciò che conta è la comunicazione e non il mezzo, tuttavia il buon uso del mezzo è fondamentale. Benché la stampa non sia più l'unico modo di scambiarsi conoscenze, il ruolo svolto dalla tipografia nella comunicazione e nella diffusione del sapere è fuori discussione.

c.3.4 Scrittura in Rete: un nuovo umanesimo

Nella seconda metà degli anni Novanta del secolo scorso, con la diffusione di Internet, i documenti prodotti elettronicamente si possono distribuire senza sostanziali limitazioni di distanza fra autore e lettori.

Oggi, grazie al Web, chiunque può diffondere i risultati delle proprie ricerche. In questi anni sta fiorendo in Rete un rinnovato umanesimo, che non ha bisogno di mecenati e che spesso si fonda sulla volontà di trasmettere conoscenze in puro spirito di liberalità. Molto spesso gli autori che sposano questa nuova forma di comunicazione si distinguono non solo per l'originalità dei propri contributi ma anche per la loro qualità formale.

c.3.5 Scomparsa dell'editore

L'autore di un libro può attrarre il potenziale lettore in due modi: il contenuto dell'opera e la sua presentazione. Fino a poco tempo fa, la mediazione dell'editore, cui l'autore affidava l'opera per ottenerne una buona riuscita grafica e la debita pubblicità, era indispensabile.

Oggi, con la possibilità di fare egregia tipografia digitale anche a livello domestico, il ruolo di questo terzo (incomodo o comodo che sia) non è più indispensabile. Specie nella "piccola" divulgazione (inteso l'aggettivo in

sensu numerico e non come sinonimo di scarsa qualità), l'autore è spesso l'editore di se stesso, usando la Rete come la bottega del libraio in cui offrire le proprie opere.

S'è affermata una nuova figura d'autore, che spesso non intende trarre alcun vantaggio commerciale dalle proprie creazioni, ma solo la gratificazione nell'essere letto ed eventualmente apprezzato. Nasce un nuovo concetto di *copyright*, che pur tutelando la paternità dell'opera non fa scaturire da questa automatici effetti economici, non richiesti e forse non desiderati.

E se fino a poco tempo fa chi usava il computer per scrivere ricorreva a un'editoria povera, badando soprattutto al contenuto, ora, grazie all'insegnamento magistrale di Knuth e Lamport, si è consapevoli che le due cose (contenuto significativo e presentazione grafica ottimale) possono andare d'accordo, e il secondo elemento (la forma) può accrescere il primo (la sostanza) divenendone parte integrante.

c.3.6 Libro elettronico

In questo *excursus* non può mancare un cenno a una categoria che da qualche anno si è affacciata all'orizzonte: il *libro elettronico* (o *e-book*), un documento composto per essere consultato sullo schermo di un computer o di uno strumento dedicato.

Dal suo apparire, il libro elettronico ha visto dividersi due schieramenti: da una parte c'è chi sostiene l'improprietà della definizione, dovendosi intendere *libro* solo quello ottenuto col procedimento a stampa, dall'altra c'è chi vede nell'e-book il futuro prossimo, la dimensione verso cui ci si dovrà incamminare.

A nostro avviso, per risolvere la questione basta una semplice considerazione, più logica che sintattica: *elettronico* è solo un aggettivo che nulla aggiunge o toglie alla sostanza del libro, specificando unicamente una caratteristica diversa da quella cartacea. Un libro contiene parole e idee: non ha importanza se si scrive con una penna o con la tastiera di un computer, o se il pensiero dell'autore è espresso in un libro stampato o in un file.



Figura 61: e-book

Le potenzialità dell'e-book sono molte: esso permette di mostrare testo, suoni, immagini, filmati, collegamenti ipertestuali. D'altra parte la leggibilità di un'opera a stampa non è eguagliabile da quella di un libro elettronico.

c.3.7 Conclusioni

Queste poche pagine possono solo delineare una piccola panoramica della storia della scrittura e della tipografia, ma speriamo che stimolino il lettore a osservare con più attenzione i documenti che ha sotto mano.

Il mondo, specialmente quello del software e di \LaTeX in particolare, è sempre in fermento, e promette nuovi traguardi e possibilità per chi ama comporre i propri testi secondo i canoni dell'arte tipografica tramandata da secoli.

ACRONIMI

\mathcal{AMS}	American Mathematical Society Fondata nel 1888, con svariate decine di migliaia di soci la Società Matematica Americana è una delle più importanti associazioni di matematici nel mondo. L' \mathcal{AMS} ha sostenuto attivamente lo sviluppo di \LaTeX ed è stata tra i primi organismi scientifici a sollecitare gli autori a scrivere con questo programma.
CTAN	Comprehensive \TeX Archive Network La “rete di archivi completi di \TeX ” è, nel Web, il luogo di riferimento da cui scaricare software e materiale su \TeX e \LaTeX .
\GfIt	Gruppo Utilizzatori Italiani di \TeX e \LaTeX È un’associazione senza fini di lucro con lo scopo di aumentare la diffusione di \TeX e \LaTeX in Italia attraverso la condivisione di informazioni legate al loro uso, conciliando il vantaggio dell’apprendimento con il piacere dell’insegnamento.
HTML	Hyper Text Mark-up Language Il “linguaggio di marcatura degli ipertesti” è un linguaggio di pubblico dominio inventato da Tim Berners-Lee al CERN di Ginevra nel 1989 per descrivere i documenti ipertestuali che popolano il Web.
ISO	International Standard Organization L’ “organizzazione internazionale per le standardizzazioni” è la più importante associazione a livello mondiale per la definizione di standard tecnico-scientifici. Suoi membri sono gli organismi nazionali di standardizzazione di 163 Paesi del mondo.
JPEG	Joint Photographic Experts Group È il più usato standard di compressione delle immagini fotografiche. Esegue una compressione con perdita di informazioni (cioè di tipo <i>lossy</i>). Insieme con il PNG, è il formato standard delle immagini bitmap da inserire in un documento da comporre con \LaTeX .
PDF	Portable Document Format È il formato di file più versatile per la stampa e la distribuzione elettronica, introdotto dalla Adobe Systems nel 1993 per rappresentare documenti indipendentemente dall’hardware e dal software usati per generarli o visualizzarli. Il PDF eredita molte delle funzioni del PostScript, un linguaggio di descrizione della pagina sviluppato dalla stessa azienda. È il formato standard delle immagini vettoriali da inserire in un documento da comporre con \LaTeX .
PNG	Portable Network Graphics Creato nel 1995, è un formato di file particolarmente adatto per rappresentare disegni e icone. Esegue una compressione senza perdita di informazioni (cioè di tipo <i>lossless</i>). Con il JPG, è il formato standard delle immagini bitmap da inserire in un documento da comporre con \LaTeX .

TUG **T_EX User Group**

L'espressione ("gruppo di utenti di T_EX") indica le associazioni di persone accomunate dalla passione per T_EX e L^AT_EX sparse nei principali Paesi del mondo. Il loro scopo è quello di diffondere l'uso di questi due programmi e di fornire supporto alle rispettive comunità di utenti.

UNI **Ente Nazionale Italiano di Unificazione**

È un'associazione privata senza scopo di lucro che svolge attività normativa in tutti i principali settori tecnico-scientifici. Rappresenta l'Italia in seno all'iso.

URL **Uniform Resource Locator**

È una stringa di caratteri che identifica in modo univoco l'indirizzo di una qualunque risorsa in Rete. Ogni URL (per esempio <http://www.tug.org/texlive/>) si compone normalmente di tre parti: il protocollo usato per indirizzare la risorsa (http, nell'esempio considerato), il nome dell'*host* o del server o del dominio (www.tug.org), e infine il nome del file che contiene la risorsa (/texlive/).

UTF **Unicode Transformation Format**

Unicode è un sistema di codifica che assegna una combinazione di bit a ogni carattere indipendentemente da programma, piattaforma e lingua usati. La codifica UTF-8 (*Unicode Transformation Format-8 bit*) è una particolare realizzazione di Unicode.

WYSIWYG **What You See Is What You Get**

L'espressione indicata dall'acronimo ("ciò che vedi è ciò che ottieni") ha sostanzialmente due significati.

Il primo si riferisce al problema di ottenere in stampa testo e immagini che abbiano una disposizione grafica uguale a quella visualizzata sullo schermo del calcolatore. I primi software e le prime stampanti per uso domestico non davano risultati pienamente soddisfacenti e si superò il problema introducendo nuovi dispositivi e software (pionieri furono il sistema di codifica dei caratteri TrueType sviluppato dalla Apple e il programma Adobe TypeManager).

Con il tempo, il significato dell'acronimo si è esteso per analogia anche ad alcune problematiche connesse alla creazione dei documenti. Nei comuni elaboratori di testo (come per esempio Microsoft Word), l'utente agisce direttamente sul testo già composto così come appare sullo schermo, e ogni sua azione si traduce in un'immediata variazione di quel testo. Programmi di questo tipo vengono detti WYSIWYG (in questo secondo significato), e il tipo di composizione che adottano viene denominato "composizione sincrona". L'acronimo che si riferisce al concetto opposto è WYSIWYM.

WYSIWYM **What You See Is What You Mean**

L'acronimo ("ciò che vedi è ciò che intendi") è stato coniato espressamente per L^AT_EX in contrapposizione a WYSIWYG (nel senso di "programma di videoscrittura caratterizzato da una composizione sincrona").

La caratteristica che più differenzia L^AT_EX dagli altri elaboratori di testo è il fatto che per realizzare un documento con questo programma bisogna agire *in tempi diversi* per introdurre il testo e per comporlo. È la "composizione asincrona": prima si scrive il testo badando soltanto a contenuto e scansione logica; poi lo si dà "in pasto" a L^AT_EX, che lo compone e lo impagina per produrre il documento finito.

ELENCO DEI SITI INTERNET

CTAN

Sito ufficiale di CTAN.

<http://www.ctan.org/>

FONTS

Catalogo dei font per L^AT_EX.

<http://www.tug.dk/FontCatalogue/>

GUI

Sito ufficiale del G_UI_T.

<http://www.guitex.org/>

LP

Sito di Lorenzo Pantieri.

<http://www.lorenzopantieri.net/>

MACTEX

Pagina Web di MacT_EX.

<http://www.tug.org/mactex/>

TEXCATALOGUE

Catalogo L^AT_EX online.

<http://texcatalogue.ctan.org/>

TEXAMPLE

Esempi di disegni fatti con TikZ.

<http://www.texample.net/>

TEXLIVE

Pagina Web di T_EX Live.

<http://www.tug.org/texlive/>

TEXSHOP

Pagina Web di T_EXShop.

<http://pages.uoregon.edu/koch/texshop/>

TEXSTUDIO

Pagina Web di T_EXstudio.

<http://www.texstudio.org/>

BIBLIOGRAFIA

Beccari, Claudio

- 2016 *Introduzione all'arte della composizione tipografica con L^AT_EX*, <http://www.guitex.org/home/images/doc/guidaguit-b5.pdf>.

Beccari, Claudio e Tommaso Gordini

- 2015 *L'arte di scrivere in diverse lingue con X_YL^AT_EX*, <http://www.guitex.org/home/images/doc/ArteLingue.pdf>.
2016 *Codifiche in T_EX e L^AT_EX. Dal sorgente al PDF. Guida pratica per lavorare con successo*, <http://www.guitex.org/home/images/doc/GuideGuIT/introcodifiche.pdf>.

Bringhurst, Robert

- 1992 *The Elements of Typographic Style*, Hartley & Marks, Vancouver; trad. it. *Gli Elementi dello Stile Tipografico*, Sylvestre Bonnard, Milano 2001.

Cevolani, Gustavo

- 2006 «Norme tipografiche per l'italiano in L^AT_EX», *ArsT_EXnica*, 1, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica01.pdf.

De Marco, Agostino e Roberto Giacomelli

- 2011 «Creare grafici con pgfplots», *ArsT_EXnica*, 12.

Eco, Umberto

- 1977 *Come si fa una tesi di laurea. Le materie umanistiche*, Bompiani, Milano.

Fairbairns, Robin

- 2014 *The UK T_EX FAQ*, <http://texdoc.net/texmf-dist/doc/generic/FAQ-en/newfaq.pdf>.

Feuersänger, Christian

- 2011 *Manual for Package pgfplots*, <http://www.ctan.org/tex-archive/graphics/pgf/contrib/pgfplots/doc/latex/pgfplots/pgfplots.pdf>.

Fleck, Heinrich

- 2008 *Appunti L^AT_EX*, <http://www.heinrichfleck.net/latex/latex.html>.

Goossens, Michel, Frank Mittelbach e Johannes Braams

- 2004 *The L^AT_EX Companion*, Addison-Wesley, Reading (Massachusetts).

Gregorio, Enrico

- 2003 *Galleria degli orrori. Come maltrattare L^AT_EX e rendere infelice un copy editor*, <http://profs.sci.univr.it/~gregorio/orreri.pdf>.
2009 *Appunti di programmazione in L^AT_EX e T_EX*, <http://profs.sci.univr.it/~gregorio/introtex.pdf>.
2010 *L^AT_EX. Breve guida ai pacchetti di uso più comune*, <http://profs.sci.univr.it/~gregorio/breveguida.pdf>.

Gregorio, Enrico

- 2013 *Installare T_EXLive 2012 su Ubuntu*, <http://profs.sci.univr.it/~gregorio/texlive-YEAR-ubuntu.pdf>.

Guiggiani, Massimo e Lapo Filippo Mori

- 2008 «Come *non* maltrattare le formule matematiche», *ArsT_EXnica*, 5, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica05.pdf.

Knuth, Donald Ervin

- 1973 *Computer Programming as an Art*, Addison-Wesley, Reading (Massachusetts), vol. 3.
1984 *The T_EXbook*, Addison-Wesley, Reading (Massachusetts).

Lamport, Leslie

- 1994 *L^AT_EX. A Document Preparation System*, Addison-Wesley, Reading (Massachusetts).

Miede, André

- 2015 *A Classic Thesis style*, Manuale d'uso dello stile ClassicThesis, <http://www.ctan.org/tex-archive/macros/latex/contrib/classicthesis/ClassicThesis.pdf>.

Mittelbach, Frank, Gianluca Pignatelli e Dave Walden

- 2007 «Intervista a Frank Mittelbach», *ArsT_EXnica*, 3, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica03.pdf.

Mori, Lapo Filippo

- 2006 «Tabelle su L^AT_EX 2_ε: pacchetti e metodi da utilizzare», *ArsT_EXnica*, 2, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica02.pdf.
2007 «Scrivere la tesi di laurea con L^AT_EX 2_ε», *ArsT_EXnica*, 3, http://www.guit.sssup.it/arstexnica/download_ars/arstexnica03.pdf.

Oetiker, Tobias, Hubert Partl, Irene Hyna e Elisabeth Schlegl

- 2011 *The Not So Short Introduction to L^AT_EX 2_ε*, <http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf>.

Pakin, Scott

- 2015 *The Comprehensive L^AT_EX Symbol List*, <http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf>.

Sabatini, Francesco e Vittorio Coletti

- 1997 *il Sabatini Coletti. Dizionario della Lingua Italiana*, Giunti, Firenze.

Serianni, Luca

- 1989 *Grammatica italiana. Italiano comune e lingua letteraria*, con la collaborazione di Alberto Castelvetti, UTET, Torino.

Tantau, Till

- 2013 *TikZ and PGF manual*, <http://ftp.uniroma2.it/TeX/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf>.

INDICE ANALITICO

COMANDI SPECIALI

$\backslash!$, 86, 87
 $\backslash\#$, 25, 27
 $\backslash\$$, 25, 27
 $\backslash\%$, 25, 27
 $\backslash\&$, 25, 27
 \backslash' , 60
 $\backslash($, 77
 $\backslash)$, 77
 $\backslash,$, 81, 86, 87, 258, 340, 342
 $\backslash-$, 56, 258
 $\backslash.$, 60
 $\backslash=$, 60
 $\backslash@$, 57
 $\backslash[$, 78, 118
 $\backslash\backslash$, 28, 54, 55, 62, 67, 68, 124, 133, 139, 140, 257
 $\backslash\{$, 25, 27
 $\backslash\}$, 25, 27
 $\backslash]$, 78, 118
 $\backslash^$, 60
 $\backslash_$, 25, 27
 \backslash' , 60
 $\backslash\sim$, 25, 60

A

$\backslash AA$, 60
 $\backslash aa$, 60
 $\backslash abs$, 355
 $\backslash abstractname$, 253
 $\backslash addcontentsline$, 41, 44
 $\backslash addplot$, 196–198, 200
 $\backslash addplot/***/$, 198
 $\backslash addplot/*3*/$, 197, 203
 $\backslash address$, 279, 283, 284
 $\backslash addtocategory$, 231
 $\backslash AE$, 60, 292
 $\backslash ae$, 60, 292
 $\backslash aleph$, 92
 $\backslash alert$, 270
 $\backslash alpha$, 83
 $\backslash alsoname$, 253
 Ambiente
 abstract, 37
 acronym, 71, 72
 aenumerate, 322
 alertblock, 272
 align, 43, 92–94, 351

align*, 94
 aligned, 94
 array, 117, 118, 120, 125, 126, 348
 axis, 192–194
 block, 271, 272
 Bmatrix, 90
 bmatrix, 90
 cases, 90, 94, 356
 center, 118, 121, 348, 350
 cjhebrew, 305
 column, 274, 275
 columns, 274
 comment, 31
 description, 65, 71, 254
 document, 48, 49
 doublespace, 35
 elenco, 348
 enclosures, 287
 enumerate, 43, 65, 254, 322, 348
 eqnarray, 78, 351
 eqnarray*, 78, 351
 equation, 43, 78, 94
 equation*, 78
 exampleblock, 272
 figure, 43, 120, 144, 148, 150, 154, 192, 251, 252, 274
 figure*, 251
 flushleft, 348
 flushright, 348
 frame, 266, 273
 gather, 43, 92, 93
 gather*, 94
 gathered, 94
 greek, 298
 itaitemize, 247
 itemize, 65, 254, 266
 landscape, 140
 letter, 283
 loglogaxis, 192
 lstlisting, 103, 104, 106, 108
 matrix, 90
 multicolors, 251, 252

multiline, 43, 92, 355
 multiline*, 93
 nota, 149
 onehalfspace, 35
 otherlanguage, 230
 otherlanguage, 24, 249
 otherlanguage*, 249, 298
 pascal, 112
 pmatrix, 90
 polaraxis, 192
 proof, 100, 356
 quotation, 67
 quote, 67
 quoting, 68
 RLtext, 307
 SCfigure, 144
 SCfigure*, 144
 scope, 154
 SCTable, 144
 SCTable*, 144
 semilogxaxis, 192
 semilogyaxis, 192
 sidewaysfigure, 138
 sidewaysstable, 138
 singlespace, 35
 smallmatrix, 91
 smithchart, 192
 split, 43, 92, 93, 354
 subequations, 95
 table, 43, 119, 137, 144, 148, 150, 251, 252, 274
 table*, 251
 tabular, 90, 94, 117, 118, 120, 125, 126, 137, 138, 348
 tabularx, 127
 ternaryaxis, 192
 thebibliography, 37, 217–219, 356
 tikzpicture, 153, 154, 168, 174, 175, 192
 titlepage, 63

verbatim, 69, 103, 273
 verse, 68
 Vmatrix, 90
 vmatrix, 90
 wrapfloat, 147, 148
 \AmS, 59
 \and, 30, 62
 \angle, 92
 \anwtonos, 296
 \ap, 60
 \appendix, 26, 38
 \appendixname, 253
 \approx, 93
 \arccos, 88
 \arcsin, 88
 \arctan, 88
 \areaset, 327
 \arg, 88
 \arraybackslash, 136
 \Ars, 25, 59
 \ast, 92
 \asympt, 93
 \author, 29, 268

B

\b, 60
 \backmatter, 38
 \bar, 83, 84, 353
 \beta, 83
 \bfseries, 61
 \bibitem, 217
 \bibname, 219, 253
 \BibTeX, 59
 \Big, 89
 \big, 89
 \bigcap, 82
 \bigcup, 82
 \Bigg, 89
 \bigg, 89
 \Biggl, 89
 \biggl, 89
 \Biggr, 89
 \biggr, 89
 \Bigl, 89
 \bigl, 89
 \bigodot, 92
 \bigoplus, 92
 \bigotimes, 92
 \Bigr, 89
 \bigr, 89
 \bigskip, 55
 \binom, 89
 \blacksquare, 92
 \bmod, 88
 \boldsymbol, 90, 96
 \bottomrule, 125
 \Bra, 90

\Braket, 90
 \bullet, 92
 C
 \c, 60
 \cap, 82
 \caption, 105, 120, 195, 336, 349
 \captionsetup, 121
 \cc, 283
 \ccname, 253
 \cdot, 92
 \centering, 121, 135, 350
 \chapter, 37, 42, 43, 64
 \chaptername, 253
 \check, 84
 \chi, 83
 \circ, 92
 \cite, 218, 228, 273
 \citeauthor, 229
 \citep, 229
 \citel, 229
 \citeyear, 229
 \cjrL, 305
 Classe

article, 33, 38, 39, 219, 229, 322
 beamer, 32, 187, 265, 270–272, 275, 276
 book, 33, 36, 39, 40, 219, 229, 232, 322
 KOMA-Script, 32, 322–324, 327, 335
 letter, 33, 283–285, 322
 letteracdp, 285, 286
 memoir, 32
 moderncv, 278, 279
 report, 33, 39, 219, 229, 232, 322
 scrartcl, 319, 322, 323
 scrbook, 322, 323
 scrlettr, 322, 323
 scrreprt, 322, 323
 standalone, 154
 suftesi, 32, 249
 toptesi, 32
 \cleardoublepage, 219, 261
 \clearpage, 219, 259, 261
 \closing, 283
 \cmidrule, 125

\colon, 85
 \color, 252
 \columnwidth, 251
 \complement, 82
 \cong, 93
 Contatore
 secnumdepth, 41, 42
 tocdepth, 41
 tocnumdepth, 42
 \contentsname, 253
 \coordinate, 170
 \cos, 87, 88
 \cosh, 88
 \cot, 88
 \coth, 88
 \cov, 353
 \csc, 88
 \cup, 82
 \cdoubleitem, 279
 \cventry, 279
 \cvitem, 279
 \cyr, 301

D

\d, 60
 \dagger, 92
 \dashv, 92
 \date, 62, 268, 283, 284
 \dategreek, 298
 \ddagger, 92
 \ddigamma, 296
 \ddot, 84
 \ddots, 91
 \DeclareMathOperator, 88, 353
 \defbibheading, 232, 233
 \deg, 88
 \Delta, 83
 \delta, 83
 \det, 88
 \dfrac, 97, 133
 \DH, 60
 \dh, 60
 \diamond, 92
 \Digamma, 296
 \dim, 88
 \displaystyle, 96
 \div, 92
 \DJ, 60
 \dj, 60
 \documentclass, 25, 26, 28, 32, 346
 \dot, 84
 \dots, 25, 30, 58, 59, 85, 339, 352
 \dottedcircle, 306
 \doublespacing, 35

`\Downarrow`, 85
`\downarrow`, 85
`\draw`, 157, 179–181

E

`\ell`, 92
`\em`, 61, 249
`\email`, 279
`\emph`, 25, 26, 61
`\emptyset`, 82
`\encl`, 283
`\enclname`, 253
`\endfirsthead`, 139
`\endfoot`, 139
`\endhead`, 139
`\endlastfoot`, 139
`\epsilon`, 83
`\eqref`, 79, 352
`\equiv`, 93
`\eta`, 83
`\euro`, 59, 298
`\exists`, 86
`\exp`, 88

F

`\fax`, 279
`\figurename`, 253
`\fill`, 179–181
`\floatname`, 149
`\floatstyle`, 149, 150
`\footcite`, 228
`\footnote`, 43, 137
`\footnotesize`, 62
`\forall`, 86
`\foreach`, 184
`\foreignlanguage`, 249
`\frametitle`, 266
`\frenchspacing`, 57
`\frontmatter`, 36, 39
`\fullcite`, 228

G

`\Gamma`, 83
`\gamma`, 83
`\gb`, 246
`\gcd`, 88
`\ge`, 93
`\gets`, 85
`\gg`, 93
`\glossaryname`, 253
`\graphicspath`, 49
`\greco`, 295
`\Greeknumeral`, 296
`\greeknumeral`, 296
`\Grtoday`, 298
`\GuIT`, 59, 72
`\GuIT*`, 59

H

`\H`, 60
`\hat`, 84
`\hbar`, 92
`\headtoname`, 253
`\heartpar`, 250
`\hline`, 124, 349
`\hom`, 88
`\homepage`, 279
`\hookleftarrow`, 85
`\hookrightarrow`, 85
`\href`, 44
`\hspace`, 92
`\hspace`, 133
`\Huge`, 62
`\huge`, 62
`\hypersetup`, 44
`\hyphenation`, 56, 347

I

`\i`, 184
`\idotsint`, 81
`\iff`, 86
`\iiiint`, 81
`\iiint`, 81
`\iint`, 81
`\Im`, 92
`\imath`, 92
`\implies`, 86
`\in`, 82
`\include`, 50
`\includegraphics`, 117, 118, 142, 148, 185, 270, 350
`\includeonly`, 50
`\index`, 240
`\indexname`, 240, 253
`\inf`, 88
`\infty`, 81
`\input`, 49, 50
`\institute`, 268
`\int`, 81
`\iota`, 83
`\item`, 26, 65, 266, 271
`\itshape`, 26, 61

J

`\jmath`, 92

K

`\k`, 60
`\kappa`, 83
`\katwtonos`, 296
`\ker`, 88
`\Ket`, 90

L

`\L`, 60
`\l`, 60
`\label`, 43, 79, 99, 105, 120, 145, 218, 349, 350
`\Lambda`, 83
`\lambda`, 83
`\land`, 86
`\angle`, 101, 339
`\LARGE`, 62
`\Large`, 62
`\large`, 62
`\LaTeX`, 21, 25, 26, 59
`\LaTeXe`, 59
`\latino`, 292, 295
`\le`, 93
`\left`, 89, 353, 354
`\Leftarrow`, 85
`\leftarrow`, 85
`\leftleftarrows`, 85
`\Leftrightarrow`, 85
`\leftrightarrows`, 85
`\legend`, 200
`\lettrine`, 250, 311, 314, 316
`\LettrineFont`, 314, 315
`\LettrineFontEPS`, 315
`\LettrineFontHook`, 315
`\LettrineTextFont`, 314
`\lg`, 88
`\lim`, 88
`\liminf`, 88
`\limsup`, 88
`\linespread`, 26
`\listfigurename`, 253
`\listof`, 149
`\listoffigures`, 37, 105, 145, 149
`\listoftables`, 37, 105, 145, 149
`\listtablename`, 253
`\ll`, 93
`\ln`, 88
`\location`, 284
`\log`, 87, 88
`\logo`, 268
`\Longleftarrow`, 85
`\longleftarrow`, 85
`\Longleftrightarrow`, 85
`\longleftrightarrow`, 85
`\longmapsto`, 85
`\Longrightarrow`, 85
`\longrightarrow`, 85
`\lor`, 86
`\lstinline`, 103, 104, 106

`\lstinputlisting`, 104, 106
`\lstlistoflistings`, 105
`\lstnewenvironment`, 111, 112
`\lstset`, 106
`\lVert`, 84
`\lvert`, 84

M

`\mail`, 45
`\mainmatter`, 36
`\makecvtitle`, 279
`\makeindex`, 239, 332
`\maketitle`, 30, 63
`\MakeUppercase`, 40
`\mapsto`, 85
`\markboth`, 39, 40
`\markright`, 39
`\mathbb`, 82, 95
`\mathbf`, 90, 95, 96
`\mathcal`, 95, 354
`\mathfrak`, 95
`\mathit`, 95
`\mathrm`, 95
`\mathscr`, 95
`\mathsf`, 95
`\mathtt`, 95
`\max`, 88
`\mbox`, 258
`\medskip`, 55
`\MF`, 59
`\mid`, 84, 93
`\midrule`, 124
`\MiKTeX`, 59
`\min`, 88
`\mobile`, 279
`\models`, 86
`\MP`, 59
`\mp`, 79
`\mu`, 83
`\multicolumn`, 26, 126

N

`\nabla`, 92
`\name`, 284
`\ne`, 93
`\nearrow`, 85
`\neg`, 86
`\newblock`, 273
`\newcolumntype`, 134
`\newcommand`, 245, 246
`\newenvironment`, 111, 246, 247
`\newfloat`, 148, 150
`\newline`, 55, 127
`\newpage`, 261

`\newtheorem`, 43, 97, 148, 329
`\newtheorem*`, 98
`\newtheoremstyle`, 98, 329
`\nexists`, 86
`\ni`, 82
`\nobreakspace`, 296
`\nocite`, 228, 237
`\node`, 166, 167
`\noindent`, 55
`\normalsize`, 61, 62
`\notag`, 94
`\notin`, 82
`\nu`, 83
`\num`, 342
`\narrow`, 85

O

`\O`, 60
`\o`, 60
`\oddsidemargin`, 35, 346
`\odot`, 92
`\OE`, 60, 292
`\oe`, 60, 292
`\oint`, 81
`\Omega`, 83
`\omega`, 83
`\omissis`, 59, 339
`\onehalfspacing`, 35
`\onslide`, 271
`\opening`, 283
`\oplus`, 92
`\otimes`, 92
`\overbrace`, 89
`\overline`, 83, 353
`\overrightarrow`, 83
`\owns`, 82

P

Pacchetto

acronym, 70–72
 afterpage, 40, 253
 amsmath, 43, 59, 77, 88, 89, 92, 93, 126, 133, 265, 299, 329, 351–356
 amssymb, 77, 265, 329
 amsthm, 97, 98, 265, 329, 356
 appendix, 38
 arabtex, 305, 307
 array, 127, 134, 136
 ArsClassica, 249, 335, 336

babel, 23, 24, 46, 56, 60, 105, 219, 229, 253, 291–296, 298, 338, 347
 bchart, 187, 189, 213
 beamer, 275
 biblatex, 37, 217, 219–226, 228–230, 233, 235, 318
 bmpsize, 141
 bodegraph, 187, 189
 bookmarks, 44
 booktabs, 117, 124, 317, 336, 349
 braket, 90
 caption, 117, 121, 122, 145, 147
 chemfig, 101, 187, 188
 chemformula, 101, 187
 chemmacros, 187, 188
 circuitikz, 187, 189
 cjhebrew, 305
 ClassicThesis, 249, 317
 classicthesis, 319, 320, 328, 331
 crop, 35
 csquotes, 68, 219
 dblfloatfix, 252
 draftwatermark, 253
 dtklogos, 59
 emptypage, 40, 324
 enumerate, 348
 epigraph, 250
 etoolbox, 254
 eurosym, 59
 float, 123, 148, 150, 252
 fontenc, 22, 46, 250, 298, 311, 345
 fontspec, 298
 footmisc, 64
 forest, 187, 189
 fourier, 250, 311
 frontespizio, 63
 geometry, 35, 279, 327, 346
 ghsystem, 187
 glossaries, 70

- graphicx, 117, 140, 142, 144, 152, 265, 315
- guit, 59, 72
- hf-tikz, 187, 189
- holologo, 59
- hyperref, 41, 44, 45, 72, 219, 265, 347
- imakeidx, 239, 240
- indentfirst, 55
- index, 235
- inputenc, 14, 22, 46, 298, 345
- keyval, 152
- LayAureo, 34, 46, 346
- layaureo, 327
- lettrine, 250, 311
- listings, 70, 103–107, 110–113, 320
- longtable, 138
- makeidx, 37, 235, 332
- mathpazo, 250, 311
- mathrsfs, 95
- mathtools, 84, 355
- mflogo, 59
- mhchem, 187
- microtype, 55, 298, 319, 328
- MinionPro, 319
- minitoc, 40
- modiagram, 187, 188
- mparhack, 64
- multicol, 251
- natbib, 229
- pdfscape, 140
- pgfplots, 212
- pgf-blur, 187–189
- pgfpages, 275
- pgfplots, 141, 187, 188, 191, 192, 194, 196–198, 201, 205, 206, 208–210, 336, 383
- polyglossia, 219, 298
- PSTricks, 151
- pxfonts, 250, 311
- quoting, 68
- ragged2e, 136
- rotating, 138
- schemabloc, 187, 189
- setspace, 35
- shapepar, 250
- shorttoc, 40, 42
- showidx, 261
- showkeys, 44
- sidecap, 43, 144
- siunitx, 100, 129, 131, 342, 343
- soul, 328
- steroid, 45
- subfig, 43, 145
- supertabular, 138
- tabularx, 126–128
- tcolorbox, 187, 189
- teubner, 299
- textcomp, 101
- TikZ, 141, 151–157, 162, 164, 166, 173, 174, 176, 177, 179, 181, 184, 186, 191, 192, 336, 381, 384
- tikz-cd, 187, 188
- tkz-euclide, 187, 189
- tkz-graph, 187, 189
- tocbibind, 254
- txfonts, 250, 311
- type1ec, 250, 311
- typearea, 327
- url, 45, 347
- varioref, 43
- verse, 69
- wrapfig, 147
- wrapfloat, 43
- xcolor, 110, 152, 176, 177, 192, 198, 252, 315
- xtab, 138
- xwatermark, 253
- yfonts, 316
- \pagebreak, 259, 261
- \pagename, 253
- \pageref, 43
- \par, 54, 61
- \paragraph, 36, 37, 42, 43
- \parallel, 84, 93
- \parencite, 228
- \part, 36, 37, 42, 43
- \partial, 92
- \partname, 253
- \path, 156, 157, 166, 170, 179–181
- \pattern, 181
- \pause, 271
- \pdfbookmark, 26
- \ped, 60
- \permill, 298
- \perp, 93
- \pgfplotsset, 192
- \phantomsection, 41, 219
- \Phi, 83
- \phi, 83
- \phone, 279
- \photo, 279
- \Pi, 83
- \pi, 83
- \pianta, 245, 246
- \place, 287
- \pm, 79
- \pmod, 88
- \Pr, 88
- \prec, 93
- \preceq, 93
- \prefacename, 253, 292
- \printbibliography, 37, 228, 229, 232, 233
- \printindex, 37, 239, 241
- \prod, 80
- \proofname, 253
- \propto, 93
- \ProsodicMarksOn, 293
- \PS, 287
- \ps, 283
- \Psi, 83
- \psi, 83
- Q**
- \qedhere, 100
- \qoppa, 296
- \qqquad, 48, 80, 86, 87, 133
- \quad, 48, 86, 87, 133
- R**
- \r, 60
- \raggedbottom, 35
- \RaggedLeft, 136
- \raggedleft, 135
- \RaggedRight, 136
- \raggedright, 135
- \rangle, 101, 339
- \Re, 88, 92
- \ref, 43, 350
- \refname, 219, 253
- \renewcommand, 246, 247, 314
- \renewenvironment, 247
- \restylefloat, 150
- \rho, 83, 331
- \right, 89, 353, 354
- \Rightarrow, 85
- \rightarrow, 85
- \rightrightarrows, 85

- \RL, 307
- \rmfamily, 61
- \rVert, 84
- \rvert, 84
- S
- \sampi, 296
- \scriptscriptstyle, 96
- \scriptsize, 62
- \scriptstyle, 96
- \scshape, 61
- \searrow, 85
- \sec, 88
- \section, 30, 36, 37, 42, 43, 64, 279
- \section*, 232, 346
- \seename, 253
- \Set, 90
- \setbeamercovered, 270
- \setmainfont, 298
- \setmainlanguage, 298
- \setminus, 82, 84
- \setotherlanguage, 298
- \setotherlanguages, 298
- \sffamily, 61
- \sgn, 88
- \shade, 180
- \shorttoc, 42
- \Sigma, 83
- \sigma, 83, 331
- \signature, 283, 284
- \sim, 93
- \simeq, 93
- \sin, 87, 88
- \singlespacing, 35
- \sinh, 88
- \sisetup, 129
- \slshape, 61
- \small, 26, 62
- \smallskip, 55
- \spacedallcaps, 321
- \spacedlowsmallcaps, 321
- \sqrt, 80
- \square, 92
- \sigma, 296
- \subfloat, 26, 145
- \subparagraph, 37, 42, 43
- \subsection, 37, 42, 43, 279
- \subsection*, 232
- \subset, 82
- \subseteq, 82
- \subsubsection, 37, 42, 43
- \succ, 93
- \succeq, 93
- \sum, 80
- \sup, 88
- \supercite, 228
- \supset, 82
- \supseteq, 82
- \surd, 92
- \swarrow, 85
- T
- \t, 60
- \tablename, 253
- \tableofcontents, 30, 37, 120
- \tabularnewline, 135
- \tan, 88
- \tanh, 88
- \tau, 83
- \telephone, 284
- \TeX, 25, 59
- \text, 79, 94, 126
- \textbackslash, 28
- \textbf, 61, 127, 245
- \textcite, 228
- \textcyr, 301
- \textheight, 143
- \textit, 26, 61, 65, 245, 321
- \textrm, 61
- \textsc, 61, 321
- \textsf, 61
- \textsl, 61
- \textstyle, 96
- \textsuperscript, 60
- \texttt, 61
- \textup, 96
- \textwidth, 35, 118, 127, 143, 251, 346
- \tfrac, 97
- \TH, 60
- \th, 60
- \thanks, 62
- \thechapter, 233
- \theoremstyle, 98
- \thesection, 233
- \Theta, 83
- \theta, 83
- \tikz, 153, 154
- \tikzset, 178
- \tilde, 84
- \tiny, 62
- \title, 29, 279
- \titlegraphic, 270
- \to, 85
- \today, 25, 26, 293
- \toprule, 124
- \triangle, 92
- \triangleleft, 92
- \triangleright, 92
- \ttfamily, 61
- U
- \u, 60
- \underbrace, 89
- \underline, 83
- \underset, 83
- \Uparrow, 85
- \uparrow, 85
- \Updownarrow, 85
- \updownarrow, 85
- \Upsilon, 83
- \upsilon, 83
- \url, 45, 347
- \useoutertheme, 270
- \usepackage, 46
- \usetheorem, 270
- V
- \v, 60
- \vardigamma, 296
- \varepsilon, 83
- \varphi, 83
- \varpi, 82, 83, 329
- \varrho, 83, 331
- \varsigma, 82, 83, 329, 331
- \vartheta, 83
- \vdash, 92
- \vdots, 91
- \vec, 83, 84, 90
- \vee, 92
- \verb, 69, 103
- \Vert, 84
- \vert, 84
- \vref, 43, 44
- \vspace, 55
- \vspace*, 55
- W
- \wedge, 92
- \widehat, 84
- \widetilde, 84
- \wp, 92
- X
- \x, 164
- \Xi, 83
- \xi, 83
- \xleftarrow, 85
- \xrightarrow, 85
- Y
- \yinipar, 316
- Z
- \zeta, 83