Ulteriori Manuali e video guide sul software libero e/o free al sito:

http://www.istitutomajorana.it

Presentazione

L'amica e brava ***Nadia Garbellini***, autrice di questa bella e semplice guida per LATEX, ha voluto aiutare quanti vogliono avvicinarsi a questo fantastico strumento, in grado di creare documenti di grande qualità e senza limitazioni grafiche di sorta. L'idea nasce quasi per caso, mentre si discuteva del manuale "Ubuntu facile" che stavo realizzando. In un primo momento, Nadia, pensava di realizzare un capitolo, del detto manuale su Ubuntu, dedicato a LATEX. Però, andando avanti con la realizzazione dell'opera, mi resi conto, leggendo le varie revisioni, che sarebbe stato restrittivo relegare al rango di capitolo, una così completa e ed autonoma guida. Ma proprio perchè in origine doveva essere un capitolo del manuale su Ubuntu, pregai Nadia di utilizzare una scrittura facile, immediata ed alla portata di tutti, anche di coloro che sono alle prime armi, proprio per conformarsi a criteri di semplicità del manuale. In definitiva abbiamo ritenuto di dare identità propria a questa bella e facile guida su LATEX.

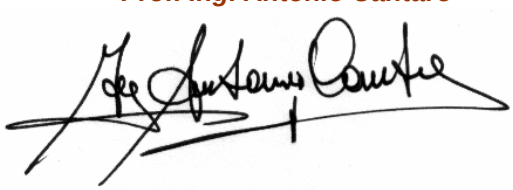
LATEX è un linguaggio utilizzato per la composizione di testi, soprattutto scientifici ma anche letterari, che permette di ottenere risultati professionali: impaginazione, tabelle, note, bibliografa, e tutto quello che comunemente trovate in qualsiasi libro di testo può essere realizzato con il vostro computer. . . e un po' di voglia di imparare qualcosa di nuovo. Il secondo requisito è fondamentale. LATEX, infatti, è molto diverso da ciò che siete abituati ad usare, probabilmente Microsoft Word o OpenOffice, sia nell'utilizzo che nella filosofia. Contrariamente ad altri word-processors, che si basano sul paradigma WYSIWYG (What you See Is What You Get, 'ciò che vedi è ciò che ottieni, ossia quello che vedete sul monitor è esattamente quello che ottenete stampando la pagina), con LATEX si lavora in modo WYSIWYW (What you See Is What You Want, ossia, ciò che vedi è ciò che vuoi). State utilizzando una sorta di linguaggio di programmazione, quindi ciò che vedete sul monitor è il 'codice' che immettete, grazie al quale potete controllare tutti gli aspetti tipografici del testo che state componendo.

Eccovi alcuni validi motivi per passare a LATEX.

1. LATEX è un software completamente gratuito. Se siete degli utenti di Linux, questo dovrebbe essere già un buon motivo, in quanto si inserisce, perfettamente, nella filosofia di questo sistema operativo. Non solo: LATEX è stato sviluppato, in origine, proprio per Unix, che è il 'papà' di Linux (e di Mac).
2. LATEX è multipiattaforma, cioè funziona benissimo con qualunque sistema operativo stiate usando. Questo significa anche che i documenti scritti su Linux potranno essere tranquillamente letti su Windows.
3. i documenti scritti in LATEX si possono velocissimamente convertire in pdf.
4. LATEX è un ottimo strumento non solo per scrivere, ma anche per disegnare, fare tabelle e grafici, fare presentazioni, ecc…
5. LATEX vi permetterà di imparare a ragionare con la stessa logica che serve a costruire un programma informatico.

Addì, 18 settembre 2008

Prof. Ing. Antonio Cantaro



Ulteriori Manuali e video guide sul software libero e/o free al sito:

http://www.istitutomajorana.it

Indice

Capitolo 1

Che cos'è LaTeX?

1.1 Introduzione

LaTeX è un linguaggio utilizzato per la composizione di testi, soprattutto scientifici ma anche letterari, che permette di ottenere risultati professionali: impaginazione, tabelle, note, bibliografia, e tutto quello che comunemente trovate in qualsiasi libro di testo può essere realizzato con il vostro computer... e un po' di voglia di imparare qualcosa di nuovo.

Il secondo requisito è fondamentale: LaTeXinfatti è molto diverso da ciò che siete abituati ad usare – probabilmente Microsoft Word o OpenOffice WordProcessor – sia nell'utilizzo che nella *filosofia*.

1.1.1 Un po' di storia

LaTeX è basato su un motore di tipocomposizione chiamato TeX. I due programmi sono stati sviluppati inizialmente in parallelo, sebbene il primo sia basato sul secondo.

Donald Ervin Knuth, docente di informatica all'Università di Stanford, cominciò a lavorare a TeX nel 1977, principalmente allo scopo di disporre di uno strumento per la composizione di formule matematiche e grafici. All'epoca infatti tutto doveva essere realizzato con la macchina da scrivere, in maniera estremamente laboriosa, e sia i grafici che molti simboli dovevano essere aggiunti a mano alla fine.

Il progetto di LaTeX invece è stato iniziato, alla fine degli anni '70, da Leslie Lamport, che nel frattempo collaborava con Knuth.

Ecco perchè i due progetti hanno seguito per un po' di tempo strade parallele. Nel corso del tempo, dagli anni '80 ad oggi, entrambi i programmi sono stato cpstantemente migliorati e arricchiti di nuove funzionalità, fino a diventare quello che sono oggi. Naturalmente, i progetti sono costantemente

in progress, e molti sviluppatori concorrono ogni giorno alla creazione di nuovi strumenti.

1.1.2 L'etimologia

La parola TeX deriva da $\tau\grave{\epsilon}\chi\nu\eta$, parola greca dal duplice significato di 'arte' e 'tecnica'. La 'X' di TeX– e di LaTeX– infatti, non si pronuncia alla latina, ma alla greca. Si tratta di un suono che non esiste nella nostra lingua, ma che si pronuncia come il 'ch' in tedesco o la 'j' in spagnolo.

Conoscere l'etimologia della parola dovrebbe aiutarvia capire qual è lo scopo di LaTeX: fornire uno strumento per l'arte della scrittura, che possa essere utilizzato anche per comporre testi tecnici.

Come abbiamo detto, LaTeX è nato soprattutto per scritti scientifici, ma nel corso del tempo sono stati introdotti paccheti adatti a scrivere poesie, sceneggiature, versi in greco e latino (e moltissime altre lingue, compreso il cinese!!) e persino partiture musicali.

1.2 Perché LaTeX è diverso?

La differenza principale, come abbiamo appena detto, risiede nella filosofia del programma: contrariamente ad altri word processors, che si basano sul paradigma WYSIWYG (What you See Is What You Get, 'ciò che vedi è ciò che ottieni': quello che vedete sul monitor è esattamente quello che ottenete stampando la pagina), con LaTeX si lavora in modo WYSIWYW – What you See Is What You Want, 'ciò che vedi è ciò che vuoi'. State utilizzando una sorta di linguaggio di programmazione, quindi ciò che vedete sul monitor è il 'codice' che immettete, grazie al quale potete controllare tutti gli aspetti tipografici del testo che state componendo.

Probabilmente si tratta di un concetto un po' difficile da capire all'inizio. Andando avanti a leggere questa guida tutto dovrebbe diventare più chiaro, ma comunque è meglio fare subito un **esempio**.

Supponete di voler scrivere questa frase:

Non è *poi* **così** difficile

Per ottenere questo risultato, il codice è:

```
\textcolor{red}{Non} \textcolor{green}{\`{e}} {\em poi} {\bf
cos\`{i}} {\huge difficile}
```

Non spaventatevi!! Il carattere `\` è semplicemente ciò che va messo davanti ad ogni comando per fare in modo che LaTeX lo consideri come tale.

`\textcolor{}{}`, come avrete capito, è il comando che stabilisce il colore con cui scrivere determinate parole. Il colore va messo nelle prime parentesi, le parole da colorare nelle seconde. Questo è qualcosa che funziona con quasi tutti i comandi: si riferiscono alla parte di testo che è racchiusa tra le parentesi graffe che li seguono. Ma ne parleremo più diffusamente in seguito.

`{\em }` serve a scrivere parole in corsivo (em=*emphasized*, enfatizzato); `{\bf }` per scriverle in grassetto (bf=*bold face*) e `{\huge }` per scriverle in grande, che è appunto il significato di *bold*.

1.3 Non è poi così difficile...

...anche se forse, a prima vista, può sembrarlo.

Le potenzialità di questo software sono enormi, e sfruttarle tutte richiede sicuramente una conoscenza molto, molto approfondita del suo linguaggio.

Tuttavia, tale conoscenza si acquisisce piuttosto facilmente con la pratica, e per iniziare, vi assicuro, non occorre sapere molte cose. Inoltre, quasi tutti gli editor di testo[1] esistenti hanno dei menù a tendina che permettono di selezionare tutte, o quasi, le opzioni di base senza dover per forza conoscere il codice. Ma dopo qualche ora di utilizzo viene naturale inserirlo a mano!!

Lo sforzo iniziale necessario a imparare ad usare questo software verrà presto ripagato dalle soddisfazioni che vi darà! Inoltre, il linguaggio di LaTeX è di fatto molto simile all'html. Questo significa che:

- per quanti di voi già lo usano un po', imparare il LaTeX sarà una passeggiata!!
- per quanti di voi vorrebbero impararlo, LaTeX sarà un'ottima palestra.

1.4 Se non siete ancora convinti...

Se non siete ancora convinti che usare LaTeX sia una buona idea, proverò a fornirvi una serie di ragioni per cui, invece, dovreste esserlo, e assolutamente!

1. LaTeX è un software completamente gratuito. Se siete degli utenti di Linux, questo dovrebbe essere già un buon motivo, in quanto si inserisce

[1] Gli editor di testo sono semplicemente dei programmi, come il blocco note per intenderci, che ci permettono di utilizzare LaTeX, cioè all'interno dei quali si scrive il codice. Ma anche di questo avremo modo di parlare in seguito...

perfettamente nella filosofia di questo sistema operativo. Non solo: LaTeX è stato sviluppato, in origine, proprio per Unix, che è il 'papà' di Linux (e di Mac).

2. LaTeX è multipiattaforma, cioè funziona benissimo con qualunque sistema operativo stiate usando. Questo significa anche che i documenti scritti su Linux potranno essere tranquillamente letti su Windows.

3. i documenti scritti in LaTeX si possono velocissimamente convertire in pdf.

4. LaTeX è un ottimo strumento non solo per scrivere, ma anche per disegnare, fare tabelle e grafici, fare presentazioni, ecc ecc.

5. LaTeX vi permetterà di imparare a *ragionare* con la stessa logica che serve a costruire un programma informatico.

Capitolo 2

Come cominciare

Come abbiamo già avuto modo di accennare alla fine del capitolo **??**, per cominciare ad utilizzare LaTeX, oltre al software stesso, serve un editor di testo, chiè l'interfaccia tramite la quale possiamo comporre i nostri testi.

Abbiamo anche detto che LaTeX è multipiattaforma. Questo però non significa che la stessa versione del programma, o lo stesso editor, funzioni con ogni sistema operativo.

Vediamo quindi nel dettaglio che cosa è necessario fare per avere sul nostro pc tutto ciò che ci occorre per metterci all'opera.

2.1 Procurarsi LaTeX

2.1.1 Linux

Se lavoriamo in ambiente Linux (voi avete probabilmente già cominciato ad usare, sicuramente con successo, Ubuntu 8.04), le cose sono estremamente semplici. le istruzioni che seguono si applicano proprio ad Ubuntu, la distro che voi conoscete meglio.

La procedura è davvero semplice. Aprite Synaptic; cercate, e installate, texlive, texlive-base e texlive-common. Se avete qualche mega di spazio libero, potete installare anche texlive-full. La differenza tra la versione base e quella full è che la seconda contiene già tutti i pacchetti presenti per LaTeX. La prima invece contiene solo i pacchetti, appunto, di base. Tutti gli altri pacchetti che potrebbero servirvi, andranno quindi installati di volta in volta, con una procedure molto semplice, che vedremo a tempo debito.

2.1.2 Mac OS X

Tutto quello che vi occorre è liberamente scaricabile a questo indirizzo.

Una volta completato il download, non dovrete fare altro che installare tutto sul vostro Mac!

2.1.3 Windows

La distribuzione di LaTeX più diffusa per Windows è senz'altro Miktex, che potete scaricare liberamente direttamente dal sito di Miktex.

Anche qui, trovate due versioni: quella 'di base' e quella completa.

2.2 Procurarsi un buon editor di testo

Esistono moltissimi editor di testo adatti a lavorare con LaTeX, alcuni specifici per LaTeX stesso, altri che possono essere utilizzati anche per altri linguaggi di programmazione. In quanto segue, vi consigliarò un editor per ciascun sistema operativo, limitandomi a quelli liberamente scaricabili dalla rete.

2.2.1 Ubuntu

Aprendo 'Aggiungi/Rimuovi Applicazioni' dal menú, potete trovare moltissimi editor adatti a lavorare con LaTeX. Alcuni però sono piuttosto difficili da usare e richiedono molta esperienza. Io vi consiglio invece di installare TexMaker (quello che sto usando io in questo momento!) che è davvero semplice ed intuitivo da utilizzare.

2.2.2 Mac

Il pacchetto che scaricate dal link fornito nel paragrafo 2.1.2 è già comprensivo di un buon editor di testo. Quindi, se avete un Mac, avete già tutto quello che vi occorre.

2.2.3 Windows

In ambiente Windows l'editor più usato è probabilmente WinEdit. Tuttavia non si tratta di un software open, quindi il mio consiglio è di utilizzare LeD (LaTeX Editor), che è ottimo, e può essere scaricato cliccando qui.

2.3 Ultimi preparativi

Ora che avete tutto il software necessario per poter lavorare, ciò che vi resta da fare è organizzare il lavoro. Scrivere un documento in LaTeX, infatti,

significa soprattutto creare un progetto. Seguite un paio di consigli e non avrete problemi.

In primo luogo, create una cartella in cui terrete tutti i vostri lavori in LaTeX. All'interno di questa cartella, creerete una subdirectory per ogni specifico progetto.

Ogni documento infatti, in genere si compone di più files (a meno che non si tratti di un documento davvero minimale). Ad esempio per documenti molto complessi solitamente si crea un file per ogni capitolo; inoltre, potreste voler inserire delle figure. In questo caso, tutti i files in uso devono stare nella medesima cartella.

I files sorgente, cioè quelli contenenti il codice, sono files di testo con estensione .tex, che potete creare semplicemente aprendo un file vuoto con l'editor di testo che avete scelto e salvandolo con questa estensione.

2.4 Ora è tutto pronto!!

A questo punto, installato LaTeX e l'editor di testo che abbiamo scelto di utilizzare, non ci resta che imparare a comporre un documento.

Le parti fondamentali di un documento LaTeX sono il *preambolo* e il corpo del testo.

Il preambolo é la parte iniziale del documento, all'interno della quale si scelgono le caratteristiche tipografiche principali che LaTeX deve utilizzare nella realizzazione del documento finito.

In primo luogo, si sceglie la classe, cioé il tipo, di documento che vogliamo realizzare: può essere un libro (*book*), un articolo (*article*), e tanto altro ancora. Tale scelta si effettua tramite il comando `\documentclass`:

```
\documentclass[12pt,a4paper]{book}
```

Quello che vedete è lo stile con cui è stato scritto questo documento. La classe scelta va in parentesi graffe, mentre quelle in parentesi quadre sono delle *opzioni*: nel nostro caso, si è scelto un carattere di 12pt di dimensione e una formattazione del testo adatta ad un foglio A4.

Subito dopo la dichiarazione della classe del documento, bisogna indicare i paccheti che si vogliono usare. Il comando da dare in questo caso è `\usepackage`:

```
\usepackage[italian]{babel}
\usepackage[T1]{fontenc}
```

Come vedete, anche con questo comando è possibile scegliere delle opzioni. I due pacchetti *fontenc* e *babel* servono a dichiarare la lingua in cui sarà scritto

il documento, in modo che LaTeX utilizzi le corrispondenti regole per andare a capo, dia i nomi giusti a capitoli e paragrafi (Capitolo, e non Chapter, che è quello che otterremmo se non di chiarassimo nulla: LaTeX, di default, parla inglese...).

Inoltre il preambolo è il luogo deputato alla difinizione di nuovi comandi e nuovi ambienti tramite i comandi `\newcommand` e `\newenvironment`.

Queste però sono funzioni avanzate, per cui potete consultare manuali di LaTeX più avanzati (troverete tutti i riferimenti necessari in bibliografia).

Per il momento questo dovrebbe bastare. Man mano che ci si trova di fronte alla necessità di utilizzare comandi specifici di altri pacchetti, è sufficiente aggiungerli al preambolo e continuare a scrivere. Se il pacchetto non fosse presente nella distribuzione di LaTeX che abbiamo scaricato, sarà necessario, come abbiamo accennato in precedenza, installarlo. A questo tema dedicheremo una sezione apposita nel capitolo **??**, paragrafo 3.1.

A questo punto, diamo il comando che segnala la fine del preambolo e l'inizio del documento vero e proprio:

```
\begin{document}
```

Quando avremo finito di comporre il nostro testo, ci basterà scrivere, in fondo a tutto:

```
\end{document}
```

Tutto ciò che verrà scritto dopo tale comando verrìgnorato, quindi state bene attenti!!

Subito dopo aver dato inizio al documento, potete cominciare a scrivere. Però, soprattutto se state scrivendo in stile *book*, potreste desiderare una prima pagina in cui compaia il titolo del vostro lavoro e il nome dell'autore, cioè voi. In questo caso vi basta utilizzare i comandi che seguono:

```
\title{Il titolo del libro}
\maketitle
```

Con `\title` dite a LaTeX qual è il titolo del vostro libro, il quale verrà scritto nel punto in cui avete inserito la stringa `\maketitle`. Potete poi anche aggiungere:

```
\author{Il vostro nome}
\date{la data che volete}
```

Se non specificate una data precisa, LaTeX metterà quella del giorno corrente. Se non volete che la data compaia, vi basterà lasciare vuote le parentesi graffe.

L'ultima cosa che generalmente trovate all'inizio di un libro è l'indice. Naturalmente, anche questo si può ottenere con un semplice comando:

```
\tableofcontents
```

L'indice verrà generato automaticamente: ogni volta che inserirete un nuovo capitolo, o un nuovo paragrafo, LaTeX lo aggiornerà.

2.5 Classi di documenti

Prima di pricedere oltre, vediamo una breve panoramica di quali sono le classi di documenti dispinibili in LaTeX e le varie opzioni che possiamo utilizzare.

2.5.1 Classi principali

Il nome della classe scelta, come sapete, va inserito all'interno di parentesi graffe dopo il comando `\documentclass`.

article: è una classe progettata per scrivere articoli – non articoli di giornale, ma su riviste specializzate! – e quindi non prevede l'uso del comando `\chapter`, in quanto la suddivisione è solo in *sections* e sottogruppi.

book: serve a scrivere documenti più lunghi e complessi, che contengano una suddivisione in sezioni più articolata e comprensiva di parti e capitoli.

letter: è una classe pensata appositamente per la composizione di lettere.

beamer è la classe per le presentazioni – slides – di cui però parleremo più diffusamente in seguito, avendo delle caratteristiche molto particolari.

Ce ne sono molte altre, come *thesis* – per le tesi di laurea – e *report*, simile a *book*, la cui trattazione però va oltre gli scopi di questa guida, che vuole essere introduttiva.

2.5.2 Opzioni principali

Le opzioni vanno inserite tra parentesi quadre dopo `\documentclass` e prima della dichiarazione della classe del documento. Se ne utilizzate più d'una, dovete separarle con delle virgole (niente spazi!!).

Detto questo, passiamo in rassegna le principali:

- Xpt: dove X è la dimensione del carattere (10pt se non si specifica altro).
- Xpaper: dove aX è la dimensione del foglio e puàssumere i valori a4, a5, letter, b5, ecc...
- twocolumn: ordina a LaTeX di formattare il testo in due colonne.

- titlepage (notitlepage) indica a LaTeX se deve o meno iniziare una nuova pagina dopo il titolo del documento. La classe *article* ha di default la seconda; le classi *book* e *report* la seconda.

Anche in questo caso, documentandovi su internet, troverete molte altre opzioni utili che potrete utilizzare nei vostri documenti.

2.6 Caratteri speciali

Come avrete notato dalla lettura del paragrafo precedente, LaTeX utilizza alcuni caratteri speciali:

- `^` *hat*, introduce gli esponenti in modalità matematica;
- `{}` racchiudono i *gruppi*;
- `%` *percent*, inizia i commenti;
- `$` *dollar*, delimita le formule matematiche;
- `_` *underscore*, indica i pedici nelle formule matematiche;
- `&` *ampersand*, funziona da separatore di colonne nelle tabulazioni;
- `#` *hash*, indica l'argomento all'interno della definizione di nuovi comandi;
- `~` *tilde*, produce uno spazio insecabile o un accento circumflesso
- `\` *backslash*, introduce i comandi.

I caratteri speciali possono essere inseriti in vari modi a seconda della tastiera o del sistema operativo. Con Linux, ad esempio, tilde è ottenuto con la combinazione AltGr+ì, mentre sotto Windows tramite la combinazione Alt+126 (cifre da tastierino!!)

Dal momento che hanno funzioni specifiche, i caratteri speciali non possono essere ottenuti in stampa semplicemente digitandoli: devono essere preceduti da `\`. Questo ad eccezione di tilde, hat e backslash.

Tilde e hat servono a produrre accenti; per ottenerli in stampa si utilizzano quindi i codici `\~{}` e `\^{}`.

Per quanto riguarda backslash, non è possibile farlo precedere da un altro backslash per il semplice fatto che il codice `\\` ha già un altro significato: serve per le interruzioni di riga. Per ottenerlo in stampa si usa pertanto `\textbackslash`.

2.7 Capitoli, paragrafi e sottoparagrafi

Eccoci quindi alla fase di scrittura vera e propria. La prima cosa che dovete imparare è come inserire un capitolo, un paragrafo, un sottoparagrafo, e così via.

Anche in questo caso, la sintassi è semplice:

```
\chapter{Titolo del capitolo}
\section{Titolo del paragrafo}
\section{Titolo del sotoparagrafo}
```

iniziano, rispettivamente, un capitolo, un paragrafo, un sottoparagrafo, e così via.

Comunque, se usate TexMaker, potete fare tutto questo semplicemente selezionando l'opzione che vi interessa dal menù a tendina (vedi figura 2.1), alla voce LaTeX->Sectioning .

Un'ultima cosa, molto importante: se volete iniziare un nuovo capoverso, dovete lasciare una riga vuota. Se vi limitate semplicemente ad andare a capo, LaTeX interpreterà la cosa come un semplice spazio e quindi *non* inizierà un nuovo capoverso!!

2.8 La compilazione

La compilazione è il passaggio fondamentale da compiere per ottenere il file di output (in formato .dvi, .ps o .pdf a seconda del tipo di compilazione utilizzato). Per eseguire tale passaggio, potete utilizzare i pulsanti presenti nell'editor di testo che state utilizzando – generalmente situati nel menù in alto – o, se usate linux, digitare da terminale

```
latex percorsodocumento.tex
```

Per documenti senza indice, è sufficiente compilare una volta sola. Per documenti con indice, bisogna compilare due volte: con la prima si creano tutti i files ausiliari, e con la seconda l'indice viene inserito nel documento. Infine, per documenti con indice analitico e/o bibliografia occorrono tre compilazioni: la seconda crea l'indice – la bibliografia – e la terza lo inserisce nel documento. Nel caso in cui dovessero esserci degli errori – come dei pacchetti mancanti o dei comandi errati – la compilazione vi restituirà un messaggio di errore, generalmente specificandone il tipo e la riga in cui l'errore stesso ètato fatto. Una volta corretto, ripetete l'operazione fino a che la compilazione non andrà a buon fine.

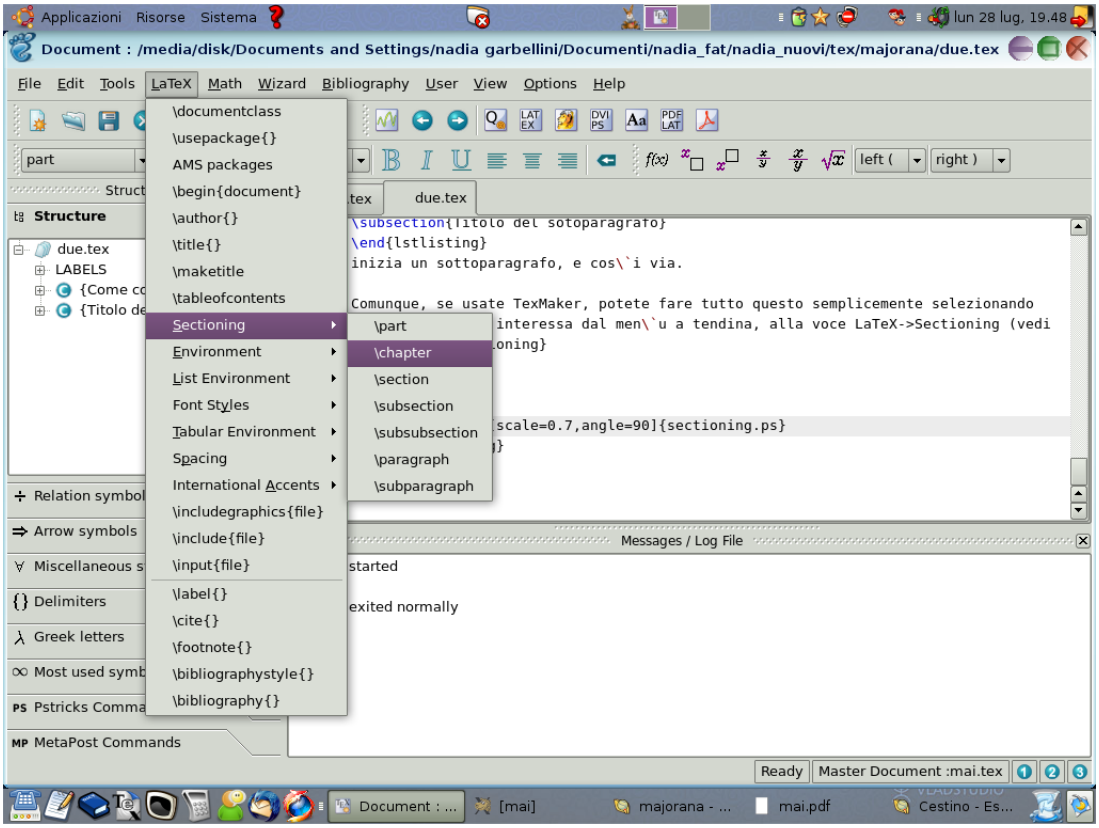


Figura 2.1: Impostare capitoli e paragrafi dal menù di TexMaker

2.9 Un piccolo esempio

Vediamo quindi un esempio di documento, veramente minimo, che faccia sfoggio di alcune delle cose viste fin qui. Il codice è:

```
\documentclass [a4paper,12pt]{book}
\usepackage[italian]{babel}
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
\author{Nadia}
\title{Esempio}
\begin{document}
\maketitle
\tableofcontents
\chapter{Un piccolo esempio}
\section{Una {\em section}\ldots}
\subsection{\ldots e una {\em subsection}}
\section{Secondo paragrafo}
con un po' di testo
\subsection{come vedete\ldots}
\subsubsection{\ldots le {\em subsubsections}}
non compaiono nell'indice!
\end{document}
```

mentre potete vedere il risultato nella figura 2.2.

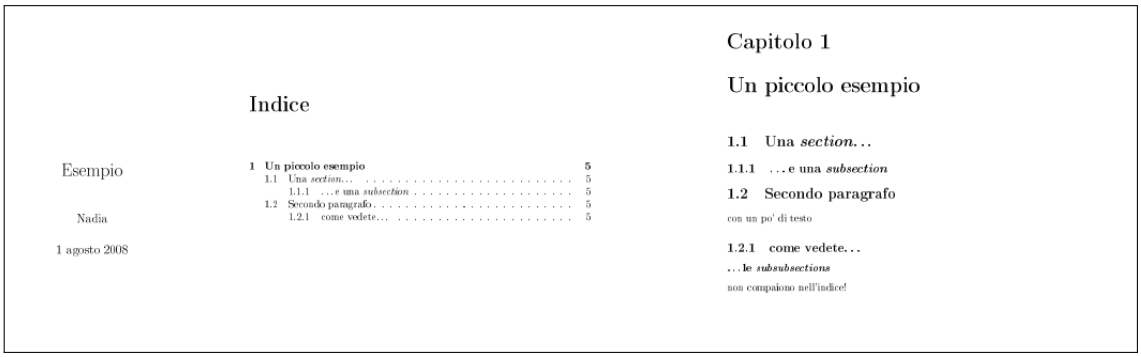


Figura 2.2: Esempio di documento minimo

Capitolo 3

Qualcosa in più

Adesso che sappiamo come comporre un documento che contenga il minimo indispensabile, possiamo passare a qualcosa di più elaborato. Vi assicuro che le possibilità offerte da LaTeX sono praticamente illimitate, per cui saremo costretti a passare in rassegna solo le principali. Come ho già avuto modo di dire in precedenza, tuttavia, la rete abbonda di guide, anche molto avanzate, e quindi, quando vi sarete impratichiti, vi potrete sbizzarrire!!

3.1 Premessa: pacchetti aggiuntivi

In ambiente Linux Ubuntu, installare pacchetti aggiuntivi è veramente molto semplice. La prima cosa che dovete fare è scaricare l'archivio del pacchetto che vi interessa da CTAN (basta cercare in google il nome del pacchetto, e troverete immediatamente la pagina che vi interessa. Cercherò comunque di fornirvi tutti i link alle pagine dove trovare i pacchetti di cui parleremo all'interno di questo manuale).

Tutte le distribuzioni di LaTeX hanno tre cartelle in cui posizionare i files di stile, quindi i pacchetti:

1. albero principale: contiene i files installati con la distribuzione, disponibili per tutti gli utenti.

2. albero locale: contiene files, disponibili a tutti gli utenti, che si possono modificare senza alterare l'albero principale.

3. albero personale: contiene i files personali, appunto, di ogni singolo utente, che la può modificare inserendo i pacchetti che utilizza di norma.

Per conoscere il percorso di ognuno dei tre alberi, vi basterà aprire un terminale e digitare, rispettivamente:

```
kpsexpand '$TEXMFDIST'
kpsexpand '$TEXMFLOCAL'
kpsexpand '$TEXMFHOME'
```

Per quanto riguarda l'albero locale, generalmente il suo percorso è /texmf. Per aggiungere pacchetti bisogna copiare la relativa cartella nella subdirectory /texmf/tex/latex, che potete creare digitando da un terminale:

```
sudo mkdir ~/texmf/tex/latex
```

La stessa procedura si utilizza per posizionare i pacchetti negli altri due alberi. Tuttavia, in questo caso, una volta copiata la cartella è necessario digitare da terminale:

```
sudo texhash
```

Quest'ultimo comando serve ad aggiornare l'albero delle dipendenze di LaTeX e quindi a fargli riconoscere ed utilizzare il pacchetto appena aggiunto.

In alternativa, si può semplicemente copiare il file di stile nella cartella in cui avete salvato il file .tex su cui state lavorando.

Alcuni pacchetti richiedono di essere installati in modo diverso. Se vi dovesse capitare di incontrarne uno, leggete semplicemente la documentazione che vi viene fornita all'interno dell'archivio, che in genere riporta le istruzioni per l'installazione nel caso sia necessario utilizzare delle procedure particolari.

3.2 Espressioni matematiche

LaTeX, come abbiamo già avuto modo di dire, è particolarmente usato per scrivere formule matematiche, la cui resa grafica è ottima. Potete scrivere qualunque tipo di formula, dalla più semplice alla più complessa, in maniera estremamente semplice. Naturalmente, anche in questo caso dovete utilizzare dei comandi specici, che però trovate anche nel menù 'Math'– per quanto riguarda le funzioni e gli operatori matematici – e nel menù laterale – per quanto riguarda i simboli – di TexMaker (vedere figura 3.1).

Ricordate che ci sono vari tipi di 'ambienti' matematici:

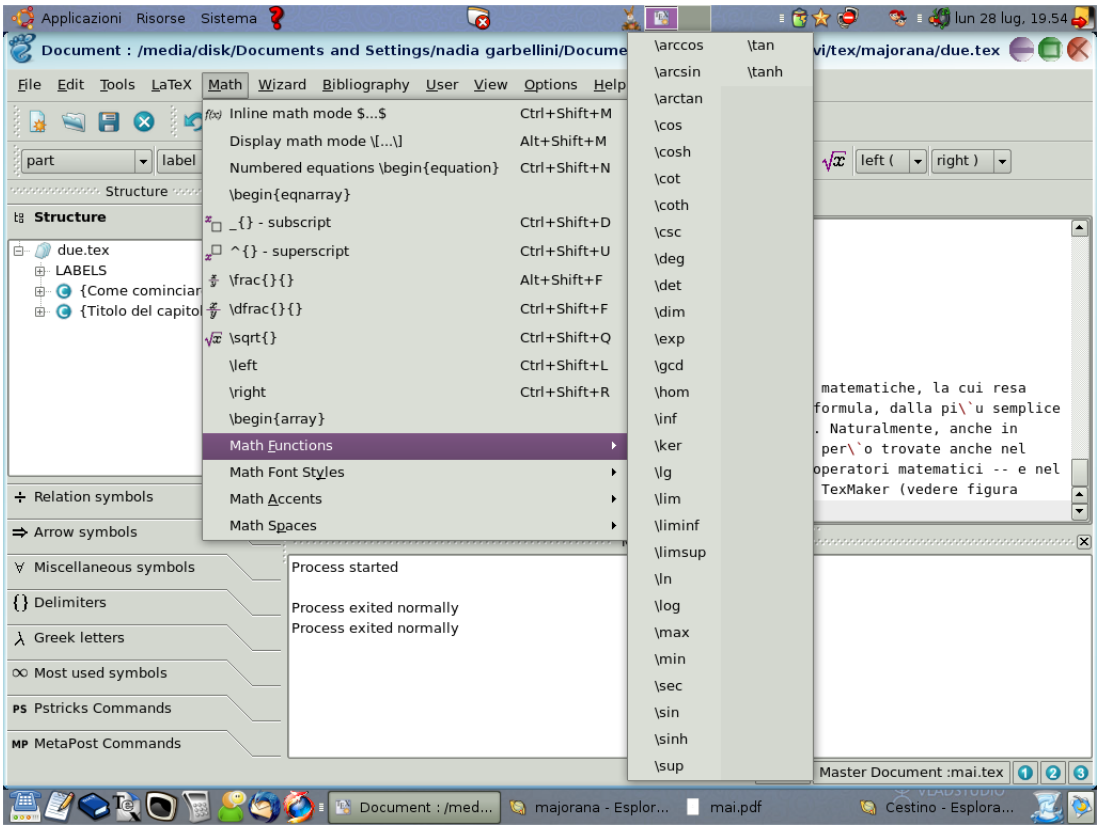


Figura 3.1: Scrivere espressioni matematiche usando il menù 'Math' di TexMaker

1. Brevi formule matematiche all'interno di una linea ('in corpo'), ad esempio:

La formula per ottenere l'area del quadrato è $\sqrt{A}$	`La formula per ottenere l'area del quadrato è $\sqrt{A}$`

   Quindi, prima e dopo la formula bisogna inserire il simbolo '`$`', in modo da far entrare LaTeX in modalità matematica.

2. Formule 'fuori corpo':

$f(x) = x^2 + \dfrac{2\sqrt{x}}{34} + \dfrac{25}{x}$	`$f(x)=x^2+ \dfrac{2\sqrt{x}}{34}+ \dfrac{25}{x}$`

   In questo caso c'è una distinzione da fare. Se volete che la formula non sia numerata, dovete inserire prima e dopo la formula una linea contenente `$$`. Se invece volete che sia numerata, utilizzate `\begin{equation}` e `\end{equation}` rispettivamente prima e dopo la formula.

   Se volete essere in grado di fare riferimento, più avanti, alla formula utilizzando la numerazione che le è stata assegnata, aggiungete, subito dopo `\begin{equation}` e senza spazi, il comando `\label{nomeformula}`. Per richiamarla utilizzate il comando `\eqref{nomeformula}`: il numero verrà messo in parentesi. Se non le volete, al posto di `eqref` utilizzate semplicemente `ref`.

   In questo modo, LaTeX gestisce autonomamente tutti i riferimenti incrociati. Se decideste di inserire un'altra equazione prima di quella richiamata, la numerazione si aggiornerebbe automaticamente.

Ora vediamo un esempio (scritto a caso, senza alcun significato matematico!!) di ciò che potete fare con l'ambiente matematico di LaTeX:

$\underbrace{2\sin x - x_{\lambda}^{\gamma\delta}}_{primaparte} + \underbrace{\int_0^{\infty} \log x^3 + \arctan x_{\omega}}_{secondaparte}$	`\underbrace{2\sin x-x_{\lambda}^{\gamma\delta}}_{prima parte}+ \underbrace{\int_0^{\infty} \log x^3+ \arctan x_{\omega}}_{seconda parte}`

Leggendo il codice, vedete come si creano esponenti e pedici. In entrambi i casi, le parentesi graffe sono necessarie solo se l'argomento è composto da più di un carattere, altrimenti potete anche non inserirle.

3.3 Ambienti

Gli ambienti, in LaTeX, sono particolari porzioni di testi racchiusi tra i comandi `\begin{nomeambiente}` e `\end{nomeambiente}`. Ne abbiamo già incontrati alcuni (*equation* ad esempio). Vediamo ora quali sono i principali.

3.3.1 Elenchi puntati e numerati

Per quanto riguarda gli elenchi puntati e numerati, i due ambienti principali sono *enumerate* ed *itemize*. La differenza risiede nel fatto che il primo crea un elenco numerato, il secondo un elenco puntato. In entrambi i casi, ogni nuovo punto dell'elenco viene introdotto dal comando `\item`.

Ad esempio:

1. primo
2. secondo
3. ...

```
\begin{enumerate}
\item primo
\item secondo
\item \ldots
\end{enumerate}
```

- primo
- secondo
- ...

```
\begin{itemize}
\item primo
\item secondo
\item \ldots
\end{itemize}
```

Per quanto riguarda l'ambiente *itemize*, può essere facilmente personalizzato come segue:

& primo
$ secondo
abc insomma come volete
...

```
\begin{itemize}
\item[\&] primo
\item[\$] secondo
\item[\#]
\item[abc] insomma come
volete
\ldots
\end{itemize}
```

3.3.2 Tabelle

L'ambiente più utilizzato nella creazione di tabelle è *tabular*. Come abbiamo già avuto modo di dire parlando dei caratteri speciali,

Vediamo subito un esempio, che ci aiuterà nella comprensione del funzionamento di questo ambiente:

Titolo tabella		
2 colonne		3
1	2 colonne	
1	2	3
a	b	c

```
begin{tabular}{ccc}
\hline
\multicolumn{3}{c}{Titolo tabella}\\
\hline
\multicolumn{2}{c}{2 colonne}& 3\\
\hline
1 &\multicolumn{2}{c}{2 colonne}\\
\hline
1& 2& 3\\
a& b& c\\
\end{tabular}
```

Cominciamo dalla prima linea di codice: `{ccc}` significa: 'voglio che la mia tabella sia composta da tre<colonne, in cui il testo deve essere posizionato al centro'. Quindi, subito dopo aver dichiarato l'inizio della tabella, occorre specificare, per ogni colonna, dove vogliamo che il teso venga posizionato: al centro ('c': *centre*), a destra ('r': *right*) o a sinistra ('l': *left*). Se avessimo voluto quattro colonne, di cui la prima e l'ultima con il testo posizionato, rispettivamente, e sinistra e a destra e le due centrali con il testo, appunto, centrato, avremmo dovuto scrivere: `{lccr}`.

Passiamo ora alla seconda linea: con il comando `\hline` tracciamo una linea orizzontale (*horizontal line*).

Il comando `\multicolumn` serve a scrivere del testo disposto inmodo tale da occupare piú colonne. In questo caso il testo 'Titolo tabella' occupa tre colonne ed è centrato.

Come abbiamo giàvuto modo di dire parlando dei caratteri speciali, il carattere `&` serve a dividere le colonne all'interno delle tabulazioni.

Inoltre, alla fine di ogni colonna bisogna aggiungere `\\`, comando con il quale dichiariamo la fine delle righe della nostra tabella.

Naturalmente ci sono molte altre cose che si possono fare con questo ambiente, come scrivere testo su più righe, colorare le celle, ecc. Anhe in questo caso, però, si tratta di comandi avanzati per i quali rimanderemo a guide più complete.

3.4 Inserire figure e tabelle

Abbiamo già visto come creare delle tabelle.

Tuttavia, per fare in modo che tabelle e immagini, siano numerate, abbiano una didascalia e possano essere richiamate in seguito dobbiamo utilizzare due ambienti particolari: *table* e *figure*. Per spiegare nella maniera più semplice possibile di che si tratta, diciamo che questi due ambienti sono delle 'scatole' dentro le quali inserire tabelle e immagini a nostro piacimento. LaTeX le riconosce e le utilizza per la gestione di numerazioni, indici, riferimenti e quant'altro.

Ambiente *table*

```
\begin{table}\label{nome}
Tabella creata usando tabular
\caption{didascalia}
\end{table}
```

Ambiente *figure*

```
\begin{figure}\label{nome}
\includegraphics{file.formato}
\caption{didascalia}
\end{figure}
```

Come vedete, le etichette si assegnano come per le equazioni. Qui in più abbiamo le didascalie (*caption*).

Per quanto riguarda le figure, come vedete si inseriscono creando la 'scatola' appropriata e poi inserendo il nome (o il percorso se non si trova nella cartella in cui stiamo salvando tutto il nostro lavoro) del file interessato. Qui occorre stare attenti: se stiamo compilando utilizzando LATEX, allora le immagini andranno inserite nei formati .ps o .eps. se invece usiamo PDFLATEX, dovremo utilizzare i file di immagini cui siamo abituati, come .png, .jpeg, eccetera.

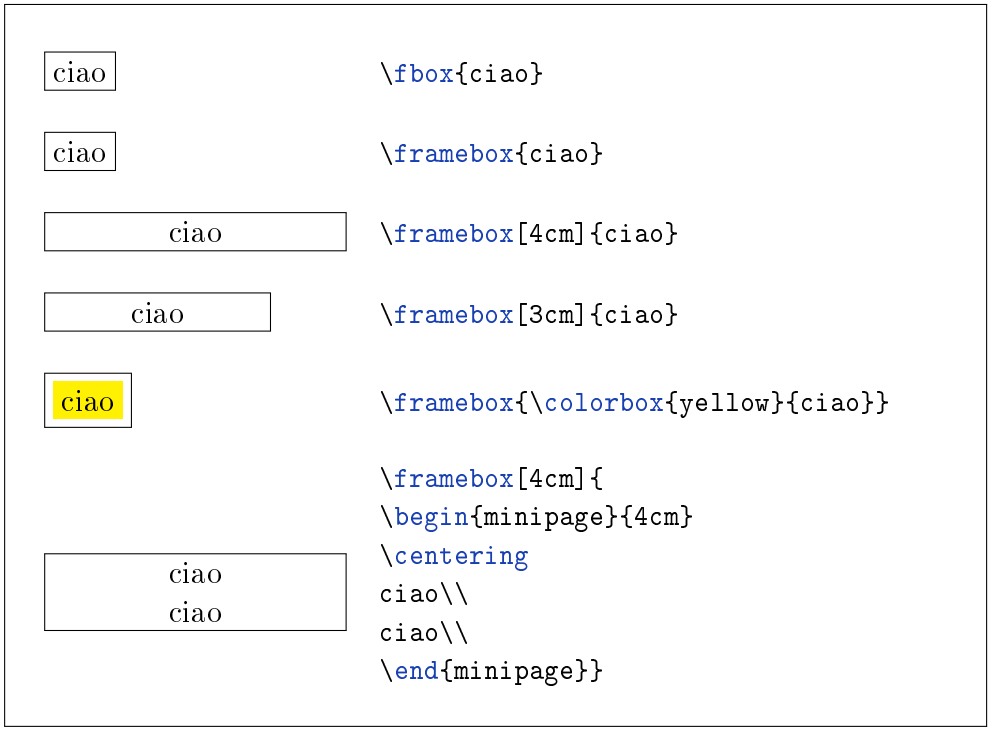
Questo é molto importante: se cercate di compilare un sorgente contenente immagini in formati non 'riconoscibili' con il particolare tipo di compilazione che state utilizzando, otterrete un errore e molto probabilmente non sarete nemmeno in grado di creare un file di output.

3.4.1 Figure e testo riquadrati

Un'ulteriore possibilità che LaTeX vi fornisce è quella di inserire figure e testo in una 'scatola' (box), incorniciata da un riquadro ed eventualmente colorata.

L'ambiente adatto allo scopo è *framebox*, o *fbox*. Il primo, a differenza del secondo, ci permette di utilizzare delle opzioni.

Vediamo qualche esempio del funzionamento di tali pacchetti.



Come vedete, `\fbox{testo}` permette di riquadrare solo parti di testo lunghe non più di una riga. Lo stesso vale per `\framebox{testo}`, a meno che non utilizziamo degli 'espedienti' per poter introdurre più righe di testo.

Uno di questi consiste nell'utilizzo dell'ambiente *minipage*, che richiede di specificare la larghezza sia del riquadro (tra parentesi quadre) che della minipage stessa (tra parentesi graffe). Quest'ultima può essere minore della prima, se vogliamo un po' di spazio ai lati. Aggiungiamo che per andare a capo dobbiamo aggiungere `\\` alla fine della riga, proprio come per le tabelle.

Arriviamo così al secondo espediente che possiamo utilizzare, cioÃÍ quello di utilizzare l'ambiente *tabular* all'interno di *framebox*, per ottenere lo stesso risultato.

All'inizio tutto questo può sembrare molto complesso, ma vi assicuro che un po' di esperienza – e di pazienza – permette di ottenere ottimi risultati in poco tempo.

3.5 Gestire documenti complessi

Uno degli usi più diffusi di LaTeX è probabilmente la stesura di tesine e tesi di laurea, che spesso sono formate da molti capitoli. Quando ci ci strova

di fronte a documenti di questo genere, risulta molto utile disporre di una struttura portante, il cosiddetto *master document*, sul quale innestare i vari capitoli. Con LaTeX questo risultato è molto semplice da ottenere.

In primo luogo, creiamo un documento, che chiameremo *master.tex*, in cui inseriremo il preambolo, quindi tutti i pacchetti che ci possono servire, e gli eventuali nuovi comandi che riteniamo possano servirci.

In secondo luogo, creiamo altri documenti, tutti con estensione .tex, uno per ogni capitolo del nostro lavoro. Possiamo chiamarli *capuno.tex*, *capdue.tex*, e così via.

A questo punto riprendiamo il nostro master document e includiamo i vari capitoli utilizzando il comando `\include{nomefile}` (nomefile *senza* estensione!!). In figura 3.3 vediamo un esempio di master document.

Noterete alcune novità rispetto a quanto detto fin qui. In primo luogo, il comando `\bibliographystyle{stile}`. Con questo comando si dichiara lo stile della bibliografia. Noi abbiamo scelto lo stile *Harvard*, di cui notate in preambolo il corrispondente pacchetto, ma e ne sono moltissimi altri, ed esiste anche la possibilità di creare degli stili bibliografici propri (si tratta ovviamente di una funzione avanzata, di cui non è il caso di parlare in questa sede).

In secondo luogo, il comando `\bibliography{bibliografia}`. Qusto comando crea la bibliografia utilizzando un file bibliografico da noi creato, con estensione .bib – in questo caso il file sarà *bibliografia.bib* – e salvato nella nostra directory di lavoro. All'interno di questo file salveremo i riferimenti bibliografici a tutti i libri, articoli, riviste, siti web, eccetera, che abbiamo citato nel corso del nostro lavoro.

> IMPORTANTE: si devono *sempre* citare le fonti che utilizziamo nel nostro lavoro. Non farlo significa appropriarsi indebitamente di qualcosa che nostro non è. Se questo discorso vi sembra contrario alla filosofia *open source*, vi sbagliate: gli autori di software e documentazione libera non vi impediscono di utilizzare il frutto del loro lavoro, né vi impongono di pagare per farlo. Tuttavia, si tratta sempre di qualcosa che è stato da loro pensato, studiato e realizzato: citarli significa semplicemente attribuire loro la paternità di quanto hanno fatto.

Chiusa questa piccola raccomandazione, torniamo alla spiegazione. Ogni editor di testo, o quasi, ha un'apposita sezione del menù dedicata alla bibliografia. Con TexMaker, In alto potete vedere la voce *Bibliography*, da

```
% file bibliografico: bibliografia.bib
@Article{etichetta,
author = {Autore articolo},
title = {Titolo articolo},
journal = {Nome rivista},
year = {Anno pubblicazione},
OPTvolume = {},
OPTnumber = {},
OPTpages = {},
}

@Book{etichetta,
author = {Autore libro},
editor = {Curatore edizione},
title = {Titolo libro},
publisher = {Casa editrice},
year = {},
}
```

Figura 3.2: Esempio di file di bibliografia: *bibliografia.bib*

cui potete selezionare il tipo di fonte che state utilizzando (libro, articolo, intervento tenuto d una conferenza, ecc).

Naturalmente, dovete prima di tutto aver creato il file di bibliografia. Da lì utilizzate il sopra citato menù: selezionando una voce, vi compaiono i relativi campi: titolo, autore, editore, curatore,[1] eccetera. Compilateli con le relative informazioni, e LaTeX penserà a produrre la bibliografia.

La figura 3.2 vi mostra un piccolo esempio di file bibliografico. L'etichetta è quella che richiameremo nel testo quando avremo bisogno di citare una fonte, usando il comando `\cite{etichetta}`. I campi preceduti dalla stringa *OPT* sono campi *opzionali* – gli altri sono obbligatori – e, nel caso in cui dovessimo decidere di riempirli, dobbiamo cancellare queste tre lettere.

Una volta terminata la produzione del file, dovete utilizare una precisa procedura di compilazione: una volta LATEX, una volta BIBTEX, tre volte di nuovo LATEX.

Tornando per un momento allo stile della bibliografia, bisogna sottolineare che la maggior parte di quelli preesistenti crea bibliografie in inglese.

[1] I campi sono in inglese. In inglese *editor* significa curatore, mentre la parola italiana 'editore' si traduce con *publisher*. Attenzione a non fare concfusione!!

Tuttavia, questo sito fornisce degli stili in italiano (a meno che non vogliate cimentarvi nel'impresa di crearne uno tutto vostro!!)

```
\documentclass[12pt,a4paper]{book}

\usepackage[italian]{babel}
\usepackage[T1]{fontenc}
\usepackage{amssymb,amsmath,multicol,tabularx,makeidx,color}
\usepackage{fancyhdr,amsfonts,graphicx,listings}
\usepackage[dcucite]{harvard}
\usepackage{pstricks,pstricks-add}
\usepackage{pst-plot}
\usepackage{pst-node}
\usepackage{hyperref}

\begin{document}

\title{\LaTeX}

\maketitle

\tableofcontents

\include{capuno}

\include{capdue}

\include{capconclusioni}

\bibliographystyle{harvard}

\bibliography{bibliografia}

\end{document}
```

Figura 3.3: Un esempio di *master document*

Capitolo 4

Il pacchetto Pstricks

Pstricks è un pacchetto di grafica estremamente interessante, che permette di disegnare non solo grafici, ma anche forme molto complesse. Qui vedremo solo alcune applicazioni, molto semplici. Nel caso in cui voleste vedere una serie di esempi (alcuni sorprendenti!!) di figure realizzate con Pstricks, vi consiglio di visitare il sito.

4.1 Installare il pacchetto

Se avete installato solo la versione base di LaTeX, il pacchetto non sarà presente. Potete installarlo utilizzando la procedura descritta nel paragrafo 3.1 ma, se usate Ubuntu, lo trovate anche in Synaptic. Vi basterà quindi selezionarlo per l'installazione ed applicare le modifiche.

Vedremo inoltre in seguito che per alcune funzionalità occorrono degli ulteriori pacchetti. Queste necessità comunque vengono rese di volta in volta note in fase di compilazione (segnalazione di eventuale mancanza di pacchetti – files di stile), e si risolvono semplicemente utilizzando la già citata procedura di installazione, resa ancora più semplice che dal sito potete scaricare tutto ciò che vi occorre, direttamente o attraverso i link riportati.

4.2 Cominciamo a disegnare!!

Prima di partire con la descrizione degli esempi riportati, un paio di precisazioni.

Ci sono due modi di inserire un grafico realizzato con pstricks in un documento LaTeX, a seconda del tipo di compilazione cheintendete utilizzare.

Se usate LATEX, l'immagine puÃš essere inserita direttamente come codice sorgente. Tuttavia ciò non può essere fatto allinterno del documen-

to: dovete aprire un file a parte – chiamiamolo, ad esempio, *figura.tex*; nel documento principale scriviamo:

```
\begin{figure}
\centering
\input{figura.tex}
\caption{Didascalia}
\label{etichetta}
\end{figure}
```

Se utilizzate questa modalità, per ottenere il pdf dovete fare il passaggio da .dvi a .ps e poi da .ps a .pdf. Inoltre, dal momento che la figura viene inclusa inun documento LaTeX principale, il file i cui la componente non necessita di preambolo né dei comandi `\begin{document}` e `\end{document}`.

Se invece volete compilare direttamente in PDFLATEX, la figura deve essere prima trasformata in un formato consono. Quindi il file in cui la componete deve essere completo di preambolo. Compilate in LATEX, trasformate il risultato da .dvi a .ps, e con un programma di grafica – tipo Gimp – trasformate la figura che avete ottenuto in .png, .jpeg, o quello che preferite. A questo punto potete includere la figura con il comando `\includegraphics`.

Adesso possiamo iniziare con un paio di esempi, che ci permetteranno di illustrare le peculiarità più interessanti di questo pacchetto.

I codici per ottenere le due figure sono riportati, rispettivamente, nelle figure 4.3 e

Partiamo dalla figura 4.1, e quindi dal codice in figura 4.3.

Innanzitutto dobbiamo inserire la figura nell'ambiente appropriato, utilizzando all'inizio e alla fine `\begin{pspicture}` e `\end{pspicture}`. Le coppie di valori in parentesi tonda immediatamente dopo la dichiarazione di inizio della figura ne delimitano l'area: rappresentano le coordinate del punto in basso a sinistra e del punto in alto a destra. Tutti gli elementi che inseriremo in seguito dovranno stare all'inetrno di questi limiti.

Il comando `\pscircle` serve, come immagino sia chiaro, a disegnare circonferenze. In parentesi quadre vanno inserite le eventuali opzioni (colore, spessore, stile della linea). In parentesi tonde si inseriscono le coordinate del centro della circonferenza e subito dopo, in parentesi graffe, la lunghezza del raggio.

Il comando `\psplot` serve invece a tracciare il grafico di una funzione matematica. Oltre alle opzioni, come sempre in parentesi quadre, vediamo tre coppie di parentesi graffe: nella prime due si inseriscono il valore iniziale

e finale dell'ascissa, vale a dire si specifica l'eare della figura entro la quale la funzione verrà rappresentata. Nell'ultima si specifica la funzione vera e propria. Il codice potrà apparire piuttosto strano. In realtà non si tratta di una cosa complicata, ma ne parleremo diffusamente nel paragrafo 4.3.

Il comando `\rput{gradi}(x,y){oggetto}` serve a ruotare un oggetto (ultime graffe) di un certo numero di gradi (prime graffe) attorno ad un punto a nostra scelta (parentesi tonde). In questo caso noi abbiamo scelto di ruotare la funzione plottata in precedenza attorno al punto di coordinate $(0, 0)$. Il codice riportato in figura 4.3 non è completo: il comando si ripeteva molte volte, e ci siamo limitati a riportarne alcune.

Il comando `\pstextpath{path}{testo}` serve a disporre un testo lungo una curva (*path*: sentiero). Attenzione: normalmente verrebbero tracciati sia il testo che la curva stessa; se non vogliamo che quest'ultima appaia, dobbiamo richiederlo esplicitamente tramite l'opzione `[linestyle=none]`.

Veniamo ora alla figura 4.2, e quindi al codice in figura 4.4 (tralasceremo ovviamente i comandi già spiegati in precedenza).

Troviamo due nuovi comandi.

Il primo è `\psaxes`, che serve a tracciare le assi cartesiane. Le opzioni sono come al solito dichiarate in parentesi quadre – in questo caso noi vogliamo che le assi non siano graduate, e ne specifichiamo lo spessore. Subito dopo, in parentesi graffe, specifichiamo il tipo di freccia che desideriamo ottenere (si tratta di un argomento opzionale, senza il quale non comparirebbe alcun tipo di freccia). Vediamo poi tre coppie di parentesi tonde, che specificano, rispettivamente, il centro del sistema cartesiano, il punto in basso a sinistra, e quello in alto a destra.

Il secondo comando è `\uput[posizione](x,y){etichetta}`, che serve a posizionare un'etichetta nei pressi di un punto. L'argomento in parentesi quadre indica il posizionamento dell'etichetta stessa rispetto al punto:

- c L'etichetta si posiziona esattamente sul punto (c: *center*);
- l L'etichetta si posiziona alla sinistra del punto (l: *left*);
- r L'etichetta si posiziona alla destra del punto (r: *right*);
- u L'etichetta si posiziona sopra il punto (u: *up*);
- d L'etichetta si posiziona sotto il punto (d: *down*).

Queste istruzioni non hanno alcuna pretesa di completezza, in quanto le possibilità offerte dal pacchetto Pstricks sono infinitamente più numerose. per conoscerle tutte, il rimando é di nuovo al sito e al manuale del pacchetto presente nella documentazione inclusa nella cartella di installazione.

4.3 Come si inseriscono le funzioni matematiche

Il linguaggio utilizzato per inserire funzioni matematiche tramite il comando `\psplot` è il inguaggio *PostScript* (o meglio, un'infinitesimale parte di esso...).

Le principali funzioni matematiche incluse sono:

`add` addizione;

`sub` sottrazione;

`mul` moltiplicazione;

`div` divisione;

`sqrt` radice quadrata;

`exp` esponenziale;

`sin` sen;

`cos` coseno.

E molti altri ancora (ma per iniziare, questi dovrebbero bastare).

Tutti questo comandi vanno utilizzati seguendo una logica ben precisa: ognuno di essi agisce sui due valori (o sul valore) che li precedono, trasformandoli in un unico valore su cui è possibile far agire i comandi successivi.

Facciamo alcuni esempi per maggiore chiarezza:

$y = x^2$	`x 2 exp`
$y = \sqrt{x}$	`x sqrt` *oppure* `x 1 2 div exp`
$y = 1000 + x^2$	`1000 x 2 exp add`
$y = \frac{x^3}{x^4}$	`x 3 exp x 4 exp div`
$y = 1000 + \frac{x^3}{x^4}$	`1000 x 3 exp x 4 exp div add`
$y = 1000 + \frac{x^3}{x^4} - \sqrt{x}$	`1000 x 3 exp x 4 exp div add x sqrt sub`

Una trattazione completa del pacchetti pstricks e di tutti i pacchetti che ruotano intorno ad esso richiederebbe un intero manuale, e va molto oltre gli scopi che ci prefiggiamo all'interno di questa piccola guida, il cui intento è semplicemente quello di fornirvi l'intuizione delle potenzialità di questo ed altri strumenti.

Vi invito quindi a visitare il sito del progetto e a seguire uno dei moltissimi tutorials presenti in rete (e ovviamente a leggere la documentazione ufficiale).

Figura 4.1: Cerchi, rotazioni, colori

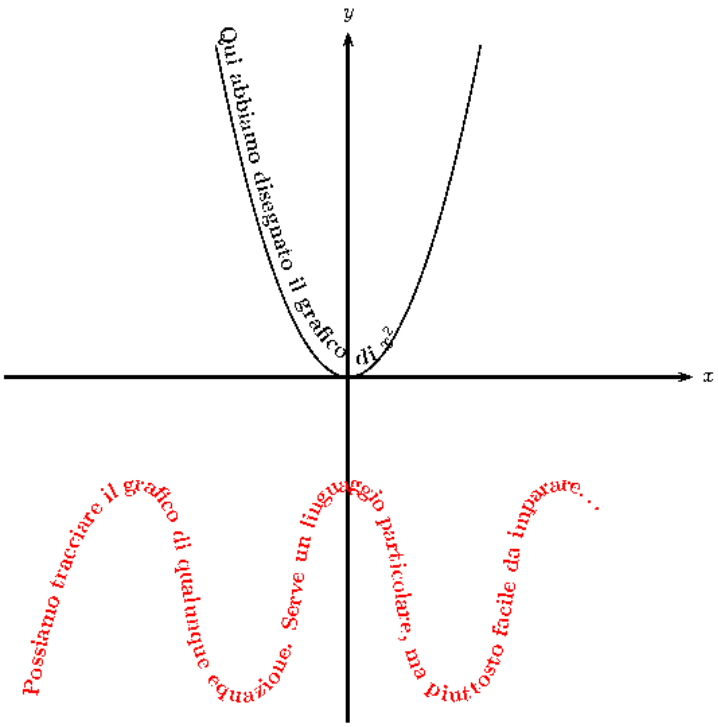


Figura 4.2: Rappresentazione di funzioni

```
\begin{pspicture}(-7,-7)(7,7)
\pscircle[linecolor=blue](0,0){3}
\psplot[linecolor=red,linewidth=1pt]{0.01}{2.6}{x 2 exp x ln mul
2 x mul sub}
\rput{15}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{30}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{45}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{60}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{75}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
⋮
\rput{300}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{315}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{330}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{345}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\rput{360}(0,0){\psplot[linecolor=red,linewidth=1pt]{0.01}{3}{x 2
exp x ln mul 2 x mul sub -1 mul}}
\pstextpath{\pspolygon[linestyle=none](-7,-7)(-7,7)(7,7)(7,-7)}
{\textcolor{nverdolino}{Decidiamo quale sar\`a l'area della
figura:  tutto ci\`o che disegneremo dovr\`a stare dentro questi
confini.  Questa volta abbiamo scelto un quadrato.  Come vedete,
possiamo tracciare linee, curve, cerchi, poligoni,\ldots usando
le lettere dell'alfabeto!  E naturalmente, possiamo fare molte,
moltissime altre cose ancora\ldots}}
\pstextpath{\pspolygon[linestyle=none](-6,-6)(-6,6)(6,6)(6,-6)}
{\textcolor{nazzurro}{Possiamo disegnare una figura e ruotarla,
intorno ad un punto a nostra scelta, di quanti gradi vogliamo,
quante volte vogliamo.  Oppure possiamo spostarla in qualunque
direzione.}}
\end{pspicture}
```

Figura 4.3: Sorgente della figura 4.1.

```
\begin{pspicture}(-7,-7)(7,7)
\psaxes[ticks=none,linewidth=2pt]{->}(0,0)(-6.5,-6.5)(6.5,6.5)
\uput[r](6.5,0){$x$}
\uput[u](0,6.5){$y$}
\pstextpath{\uput[u](0.2,0.2){\psplot[linestyle=none]{-2.5}{2.5}
{x 2 exp}}}
{\textbf{Qui abbiamo disegnato il grafico di $x^2$}}
\psplot{-2.5}{2.5}{x 2 exp}
\pstextpath{\pscurve[linestyle=none](-6,-6)(-4,-2)(-2,-6)
(0,-2)(2,-6)(4,-2)(6,-6)}
{\textbf{\textcolor{red}{Possiamo tracciare il grafico di
qualunque equazione.  Serve un linguaggio particolare, ma
piuttosto facile da imparare\ldots}}}
\end{pspicture}
```

Figura 4.4: Sorgente della figura 4.2.