LaTeX Tutorial

Fabio Durastante*

Versione 1.2

Sommario

Un'introduzione estremamente informale e (inizialmente speravo più) breve all'uso di LaTeXQuesta è la Versione 1.1 di questo tutorial, quindi potrebbe esserci qualche inconsistenza in giro. Notificate i bug e saranno corretti.

1 Installazione

Prima di iniziare. Una buona fonte per tutte le cose TeX è: www.tug.org, funge un po' da crocevia per le informazioni che possono tornare comode per editare testi con questo sistema.

In generale, per essere in grado di trasformare un file `.tex` è necessario possedere un programma che sia in grado di "compilarlo". Questo di solito fa parte di un *ambiente* di pacchetti (ecosistema) che funge da libreria per avere accesso a funzionalità aggiuntive (comandi matematici estesi, gestione delle figure, inserimento di codice, *etc.*) e vi semplifica la vita nel costruire il vostro documento. L'installazione di questo ambiente (e la particolare distribuzione di pacchetti che viene con esso) dipende dal sistema operativo (OS) che state utilizzando.

Windows. Per Windows un ambiente di riferimento per lavorare con LaTeX è miktex.org. In realtà può essere utilizzato su tutti i sistemi operativi, ma in genere per Linux o Mac sono disponibili altre opzioni che sono meglio integrate con i rispettivi sistemi. Potete scaricare l'*installer* da miktex.org/download e seguire le istruzioni di installazione passo-passo su miktex.org/howto/install-miktex.

Linux. In genere LaTeX è direttamente disponibile attraverso il *package manager* della distribuzione. Ad esempio, su Ubuntu (o su una distribuzione Debian based) potete installare i pacchetti:

```
sudo apt-get install texlive-latex-base texlive-latex-extra
↪  texlive-latex-recommended
```

*Università di Pisa, fabio.durastante@unipi.it

Oppure consultare la documentazione specifica della vostra distribuzione (`texlive` è una distribuzione di pacchetti piuttosto universale, per cui dovrebbe essere inclusa più o meno ovunque), ad esempio su Fedora:

```
sudo dnf install texlive-scheme-basic
sudo dnf install texlive-scheme-medium
sudo dnf install texlive-scheme-full
```

oppure su openSUSE utilizzando il sistema di installazione 1-click per i pacchetti ad esempio da qui: software.opensuse.org/package/texlive-latex. Se usate distribuzioni più esotiche, probabilmente sapete cosa state facendo e non necessitate di ulteriori istruzioni.

Mac. Un'alternativa a MikTeX per Mac è www.tug.org/mactex/, per cui trovate le istruzioni di installazione su www.tug.org/mactex/mactex-download.html.

Hail, Mary. Se avete difficoltà ad installare software, pacchetti o la vostra macchina fa le bizze, un'alternativa online (disponibili gratuitamente) per utilizzare LaTeX è it.overleaf.com.

1.1 Editor

Questa parte dell'installazione vi ha procurato i "compilatori" e le librerie di comandi ("pacchetti") necessarie a trasformare un file `.tex` in un file `.pdf`. Per scrivere e modificare un file `.tex` è necessario utilizzare un qualche editor di testo. Se state usando Linux potete usare praticamente qualunque editor della vostra distribuzione, questo include anche scelte più esoteriche come `vi/vim`, `Emacs`, o `nano`. Per dare un po' di uniformità a questo documento supporrò che voi preferiate usare un editor con un supporto grafico (assurdo no?) e che mi permette di unificare la presentazione per tutti i sistemi operativi. Il suggerimento è quindi: installate TeXstudio da www.texstudio.org/ (oppure dal *package manager* della vosra distribuzione Linux). Se l'installazione della parte "motore" del punto precedente è stata completata con successo, questo dovrebbe essere in grado di trovare da solo quello che gli necessità per funzionare.

Nota: se siete su Windows e avete installato Miktex, questo viene con un suo editor di default che è già sufficiente agli scopi di produrre/compilare file TeX.

2 Il vostro primo documento

Aprite l'editor che avete installato, create e salvate un nuovo file che, come vuole la tradizione, chiameremo `helloworld.tex` (Su TeXstudio: File>Nuovo). In cui potete scrivere:

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

```
\title{There and Back Again: un racconto Hobbit}
\author{Bilbo Baggins}
\begin{document}
\maketitle
\end{document}
```

che potete poi compilare usando l'icona della doppia freccia verde (▶), oppure dal menù "Strumenti>Compila e visualizza." Questo genererà il file `helloworld.pdf` con la testata contenuta in Figura 1.

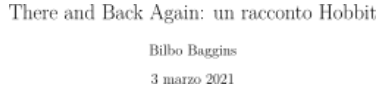


Figura 1: helloworld.pdf

Vediamo il contenuto del codice di esempio riga per riga.

`\documentclass{}` seleziona la *classe* del documento che volete scrivere. Con le distribuzioni usuali di LaTeX altre classi che sono disponibili e che potete provare sono `\documentclass{report}`, `\documentclass{book}` e le classi provviste dall'*American Mathematical Society* `\documentclass{amsart}`, `\documentclass{amsbook}`. Ad ognuna di queste classi possono essere passate delle opzioni cambiando il comando in

```
\documentclass[option1,option2]{}
```

dove, ad esempio, le opzioni possono essere `oneside`, `twoside`, `openany`, `openright`, `oonecolum`, `twocolumn`, `a4paper`, `a5paper`. Provate a vedere le differenze e quali combinazioni hanno senso.

`\usepackage[utf8]{inputenc}` scomponiamo il comando in due parti, in una questa è la prima istanza che vediamo di un comando `\usepackage[]{}`. Questa chiamata dice, in generale, a LaTeX di caricare un *pacchetto* che contiene un certo numero di comandi e funzioni ausiliarie che possono poi essere chiamate all'interno del file. Poi, in particolare, il pacchetto `inputenc` traduce vari standard e altre codifiche di input in un "linguaggio interno LaTeX". Il linguaggio interno è espresso interamente nella codifica di base di TeX. Praticamente, con l'opzione `utf8` vi permette di scrivere direttamente nel testo lettere accentate, senza dover ricorre a scappatoie come `\'e` per ottenere "é", che tuttavia possono essere usate per i caratteri per cui non avete accesso nella vostra tastiera. Ad esempio per scrivere il nome del matematico Erdős possiamo scrivere `Erd\H{o}s`.

`\usepackage[italian]{babel}` carica in maniera automatica le traduzioni nella lingua desiderata delle stringhe predeterminate di LaTeX(ad esempio i nomi dei mesi stampati dal comando `\maketitle` e la sillabazione automatica delle parole, esempio, dell'esempio, la sillabazione automatica comparsa alla riga precedente.)

`\title`,`\author`,`\date` Questi comandi sono piuttosto espliciti nel loro contenuto. In caso di un documento con più autori li si può inserire tutti nel comando `\author` come `\author{Bilbo Baggins \and Frodo Baggins}`. Sia il comando `\title`, sia il comando `\author` accettano un input opzionale che è, rispettivamente, una versione abbreviata del titolo e dei nomi degli autori che può essere usata nelle testatine delle pagine, e.g.,

```
\title[The exponential of a matrix]{Nineteen dubious ways to
↪  compute the exponential of a matrix}
```

Se il comando `\date` non viene inserito esplicitamente la data del giorno della compilazione del file viene usata.

Questi sono i comandi che terminano il cosiddetto *preambolo*, c'è poi la parte fondamentale del documento che è contenuta nell'ambiente

```
\begin{document}

\end{document}
```

in cui andrà tutto il testo del *report*, dell'articolo o del libro, ed in cui usiamo il comando `\maketitle` che stampa la testata del documento (si riveda la Figura 1).

3 Partizioni e riferimenti incrociati

Per dividere il testo in parti organiche si possono usare i comandi:

- `\part{}`, `\part*{}`
- `\chapter{}`, `\chapter*{}`
- `\section{}`, `\section*{}`
- `\subsection{}`, `\subsection*{}`
- `\subsubsection{}`, `\subsubsection*{}`
- `\paragraph{}`, `\paragraph*{}`

di cui le versioni "asteriscate" producono parti di testo non numerate. Si osserva che le parti al di sopra della sezione non sono definite per classi di tipo `article`. Ad ognuno di questi partizionamenti del testo si può aggangiare una etichetta, ad esempio

```
\section{Partizioni e riferimenti incrociati}
\label{sec:partizioni_e_riferimenti_incrociati}
```

a cui poi si può fare riferimento come:

```
Sezione~\ref{sec:partizioni_e_riferimenti_incrociati}
```

che produce "Sezione 3". È possibile stampare un indice utilizzando il comando `\tableofcontents`, ce ne è un esempio in fondo a questo *tutorial*.

4 Matematica

Per utilizzare a pieno le funzionalità matematica è buona norma caricare nel proprio preambolo i pacchetti:

```
\usepackage{amsmath}
\usepackage{amsfonts}
\usepackage{amssymb}
```

che permettono di usare diversi ambienti molto utili, ad esempio

```
\begin{equation}
\label{eq:asymptotics}
\int_{0}^{+\infty} e^{\left(x^2-2 x\right) (-\omega )} dx \approx
↪  \sqrt{\pi } e^{\omega } \sqrt{\frac{1}{\omega }} \text{ per }
↪  \omega \rightarrow +\infty
\end{equation}
```

restituisce

$$\int_0^{+\infty} e^{\left(x^2-2x\right)(-\omega)} dx \approx \sqrt{\pi} e^{\omega} \sqrt{\frac{1}{\omega}} \text{ per } \omega \to +\infty \tag{1}$$

a cui possiamo fare riferimento usando `Equazione~\eqref{eq:asymptotics}` che restituisce "Equazione (1)". Per cui un manuale completo delle opzioni può essere recuperato presso www.ctan.org/pkg/amsmath e che di da l'occasione di introdurre uno strumento fondamentale che è il "*Comprehensive TEX Archive Network*" (CTAN) su cui potete trovare le guide e le informazioni relative a tutti i pacchetti che formano LaTeX.

Un altro pacchetto estremamente utile di questa famiglia è il pacchetto `\usepackage{amsthm}` che ci permette di definire ambienti di tipo Teorema, Lemma, Corollario, Nota, Esempio. Inseriamo nel *preambolo* le definizioni degli ambienti di cui abbiamo bisogno

```
\theoremstyle{plain}
\newtheorem{theorem}{Teorema}
\newtheorem{proposition}{Proposizione}
\newtheorem{corollary}{Corollario}
\theoremstyle{definition}
\newtheorem{definition}{Definizione}
```

che possiamo poi chiamare nel corpo del documento, scrivendo

```
\begin{theorem}[Salomon, D.]\label{thm:mondaycode}
Sometimes it pays to stay in bed on Monday, rather than spending
↪  the rest of the week debugging Monday's code.
\end{theorem}
```

che ci restituisce

Teorema 1 (Salomon, D.)**.** *Sometimes it pays to stay in bed on Monday, rather than spending the rest of the week debugging Monday's code.*

A cui possiamo poi fare riferimento con `Teorema~\ref{thm:mondaycode}`, che restituisce Teorema 1. Allo stesso modo per una eventuale definizione

```
\begin{definition}[Densità di carica]\label{def:carica}
Definiamo \textbf{densità di carica} la funzione limite:
\begin{equation*}
\lim_{\Delta V \rightarrow 0} \frac{\Delta Q}{\Delta V} =
↪  \rho(x,y,z,t) = \frac{dq}{d^3x}
\end{equation*}
\end{definition}
```

ottenendo

Definizione 1 (Densità di carica)**.** Definiamo **densità di carica** la funzione limite:

$$\lim_{\Delta V \to 0} \frac{\Delta Q}{\Delta V} = \rho(x, y, z, t) = \frac{dq}{d^3x}$$

A cui possiamo poi fare riferimento con `Definizione~\ref{def:carica}`, che restituisce Definizione 1. Allo stesso modo per una eventuale definizione.

Gli stili disponibili possono essere trovati sulla documentazione relativa al pacchetto in CTAN. La struttura del comando `\newtheorem`, come avrete capito, è `\newtheorem{nomeambiente}{Nome Ambiente}`, in cui nella prima parentesi `nomeambiente` vi dice cosa dovete scrivere in `\begin{}\end{}`, mentre il secondo vi decide l'etichetta che verrà stampata in grassetto in testa all'ambiente.

5 Figure

Per inserire le figure in un documento LaTeX è necessario caricare il pacchetto `\usepackage{graphicx}`. Questo vi permette di costruire un ambiente del tipo:

```
\begin{figure}[htbp]
    \centering
    \includegraphics[width=0.8\columnwidth]{example-image-a}
    \caption{Didascalia della figura}
    \label{fig:mia_etichetta}
\end{figure}
```

che produce quello che vede in Figura 2 (ho ottenuto il riferimento con il comando: "`Figura~\ref{fig:mia_etichetta}`"). Questo è un cosiddetto ambiente di tipo *flottante*, per cui LaTeX cerca di determinare il miglior posizionamento all'interno del documento. Le opzioni passate all'ambiente cercano di suggerirgli il nostro ordine di preferenza `h` qui, `t` in cima alla pagina, `b` in fondo alla pagina, `p` se proprio questo fa danni ovunque, riservagli una pagina solo per la figura. Il comando `\includegraphics[]{}` è quello che fa la maggior parte del lavoro sporco. Questo comando prende un certo numero di opzioni che vi permette di regolare la dimensione della figura, il ritaglio e molto altro. Il ritornello è sempre lo stesso: guardate il manuale su CTAN per i dettagli. Un mio **suggerimento**

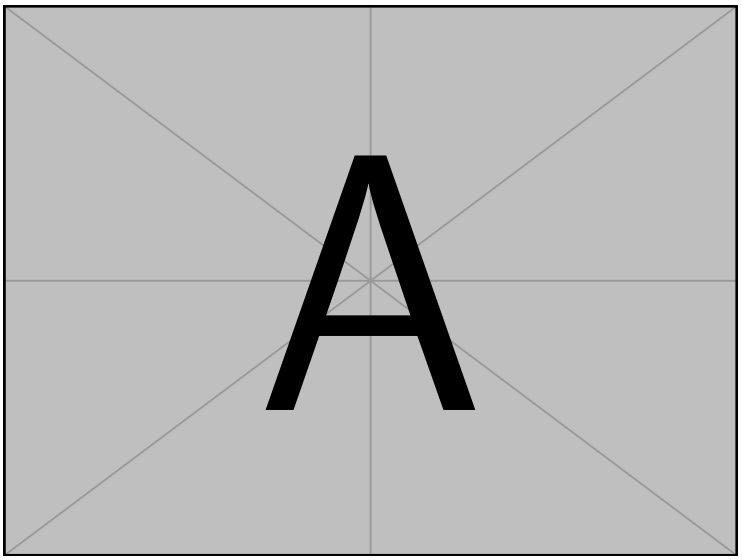


Figura 2: Didascalia della figura

per il massimo riciclo del codice LaTeX tra documenti, presentazioni e altro è di non usare mai dimensioni assolute, e.g., `width=2cm`, ma rapporti di dimensioni relativi all'ambiente, come c'è nell'esempio per cui usiamo il 60% dell'ampiezza di una colonna. Come pere il sezionamento generale, potete includere nel vostro documento una lista di tutte le figure usando il comando `\listoffigures`, di cui trovate un esempio in fondo a questo documento.

Ma **quali figure si possono inserire**? Il pacchetto `graphicx` carica un certo numero di *driver* che permettono di caricare figure con diverse estensioni (ed i corrispettivi formati). Da Matlab è conveniente utilizzare il formato di *output* `.eps`. Per cui salvate la figura come `ilmiografico.eps`, che può poi essere incluso come `\includegraphics{ilmiografico}` avendo messo la figura nella stessa cartella del sorgente, ed in caso aggiungendo le opzioni per il dimensionamento. Si **osserva** che non ho messo l'estensione, questo fa sì che il compilatore LaTeX si vada cercando il formato che lo soddisfa di più per l'output desiderato e usi quello. Potete sempre essere più precisi e richiedergli di usare l'immagine in un certo formato. Vi rimando al manuale per ulteriori dettagli.

Una maniera per sfruttare a pieno il motore LaTeX è quella di generare direttamente le figure da del codice. Per questo è possibile utilizzare i pacchetti `\usepackage{tikz}` e `\usepackage{pgfplots}`. Le possibilità di questi oggetti sono praticamente infinite, vi mostro qui giusto un esempio per stuzzicare la vostra curiosità (o farvi fuggire verso le colline):

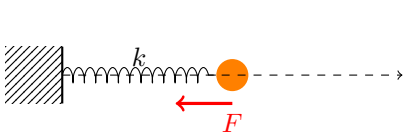
```
\begin{tikzpicture}[scale=1.5]
\node[circle,fill=orange,inner sep=1.5mm] (a) at (2,0.25) {};
```

```
\draw[decoration={aspect=0.3, segment length=1.5mm,
↪  amplitude=1mm,coil},decorate] (0.5,0.25) -- (a)
↪  node[midway,above] {$k$};
\fill [pattern = north east lines] (0,0) rectangle (0.5,0.5);
\draw[thick] (0.5,0) -- (0.5,0.5);
\draw[<-,very thick,red] (1.5,0.0) -- (2,0.0) node[below] {$F$};
\draw[->,dashed] (0.5,0.25) -- (3.5,0.25);
\node at (0.5,1) {$O$};
\node at (2,1) {$P$};
\end{tikzpicture}
```

che può essere compilato dopo aver aggiunto nel *preambolo*

```
\usepackage{tikz}
\usetikzlibrary{calc,patterns,angles,quotes,arrows,decorations.p⌋
↪  athmorphing}
```

e che vi restituisce la figura della molla che ho usato nelle *slide*:



Per utilizzare questi pacchetti un buon punto di partenza è guardare agli esempi su texample.net/tikz/examples. Osservate che potete mettere in generale questo codice all'interno di un ambiente `\begin{figure}\end{figure}`.

Un **trucco** per utilizzare questo modo di generare le figure, è di avere un codice che produce per noi (almeno in prima approssimazione) il codice da dare in pasto a LaTeX. Potete "divertirvi" guardando matlab2tikz per Matlab, oppure tikzplotlib per Python. Se siente inclini alla geometria euclidea, GeoGebra vi permette di esportare le figure in questa forma di codice. Di solito comunque il codice prodotto automaticamente da questi software necessita di aggiustamenti manuali.

6 Tabelle

Le tabelle possono essere create all'interno di ambienti flottanti come le figure, usando, ad esempio

```
\begin{table}[htbp]
    \centering
    \begin{tabular}{*{4}{c}}
        \toprule
         col1 & col2  &  \multicolumn{2}{c}{col3-4}\\
         \midrule
         a & b & c & d \\
         \cmidrule{3-4}
```

```
        a & b & c & d \\
        \bottomrule
    \end{tabular}
    \caption{Didascalia della mia tabella}
    \label{tab:lamiatabella}
\end{table}
```

Le opzioni di posizionamento restano le stesse delle figure, `h` qui, `t` in cima alla pagina, `b` in fondo alla pagina, `p` in una pagina riservata agli oggetti flottanti. In più sto usando dal pacchetto `\usepackage{booktabs}` i comandi `\toprule`, `\midrule`, `\bottomrule` che fanno esattamente quello che il nome suggerisce. In più il comando `\cmidrule{a-b}` vi permette di avere linee orizzontali che coprono solo dalla colonna a alla colonna b. Potete vedere il codice generato nella Tabella 1 (comando: `Tabella~\ref{tab:lamiatabella}`). L'altro punto fondamentale è

col1	col2	col3-4	
a	b	c	d
a	b	c	d

Tabella 1: Didascalia della mia tabella

rappresentato dalla necessità di dire a LaTeX di quante colonne è formata la vostra tabella. A questo serve la specificazione data da `*{4}{c}` che ci dice che vogliamo 4 colonne con allineamento centrato. Le lettere usate oltre `c`, sono (con poca fantasia), `l` per l'allineamento a sinistra e `r` per quello a destra. Potete specificare la stessa informazione ripetendo la lettera di allineamento corretto per ogni colonna, e.g., 4 colonne 2 centrate e 2 a sinistra, può essere scritto sia `*{2}{c}*{2}{l}` che `ccll`. Il comando `\multicolumn{}{}{}` vi permette di specificare una cella di più colonne, ovvero di tante colonne quante il numero nella prime graffe, con l'allineamento specificato nelle seconde e il testo nelle terze graffe.

Come per le figure, potete far stampare a LaTeX una lista delle tabelle contenute nel documento utilizzando il comando `\listoftables` di cui trovate un esempio in fondo a questo documento.

Un **suggerimento** di stile: non usate righe verticali nelle tabelle, in generale non servono a nulla e rendono solo la lettura più faticosa. Il *suggerimento* è talmente forte che, se ci fate caso, non vi ho nemmeno detto come farlo.

7 Codici e algoritmi

Il pacchetto che vi suggerisco per inserire dei listati, e d'altra parte quello che sto utilizzando per produrre queste pagine, in LaTeX è `\usepackage{minted}`. Questo vi permette di inserire in un documento LaTeXdel codice e di farlo apparire nella sua formattazione propria. Come per agli altri casi ne trovate la documentazione completa su CTAN. Il backend utilizzato da questo pacchetto

per evidenziare il codice è il pacchetto Python Pygment, che vi permette di avere accesso agli stili di evidenziazione di qualche centinaio tra linguaggi di programmazione, scripting e markdown.

Potete quindi inserire del codice facendo, ad esempio, per un codice Fortran,

```
\begin{minted}{fortran}
program hello
    print *, "Hello World!"
end program
\end{minted}
```

```
program hello
    print *, "Hello World!"
end program
```

oppure, per un codice c, in cui utilizziamo in più la possibilità di spezzare automatica il sorgente su più linee.

```
\begin{minted}[breakanywhere,b
↪  reaklines=true]{c}
#include <stdio.h>
int main() {
   // printf() displays the
   ↪  string inside quotation
   printf("Hello, World!");
   return 0;
}
\end{minted}
```

```
#include <stdio.h>
int main() {
   // printf() displays the
   ↪  string inside quotation
   printf("Hello, World!");
   return 0;
}
```

Altre varianti utili, sono quella per inserire il codice in modalità "inline":

```
\mintinline{Python}|num1 = input('Enter first number: ')|
```

che restituisce: `num1 = input('Enter first number: ')`. Ovvero, se si vuole che il codice prenda e stia su di un'intera riga con il comando:

```
\mint{linguaggio}|codice|
```

In ultima istanza è possibile caricare direttamente un file sorgente con:

```
\inputminted[breakanywhere,breaklines=true]{cpp}{nomesorgente.cpp}
```

Una possibile variante, molto utile per rappresentare i codice Matlab/Octave è quella del pacchetto `\usepackage{matlab-prettifier}`. Che permette ad esempio di scrivere

```
\begin{lstlisting}[style=Matlab-editor]
function y = PageRank(H, v, gamma, itmax)
n = size(H,1); usn = 1/n; e = ones(n,1);
d = (H*e)';       % Calcolo out-degree dei nodi
```

```
dang = (d == 0); % Ricerca dangling nodes
dh = d + dang*n;   dh = 1./dh; % Vettore dei pesi corretto
x = rand(1,n); x = x./sum(x); v = v./sum(v); % Norm. pesi
for it=1:itmax
 y = x.*dh;
 y = y*H + usn*sum(dang.*x);
 y = y*gamma+(1-gamma)*v;
 err = norm(x-y,'inf');
 x = y;
 fprintf('Iteration %d Error %1.2e\n',it,err);
 if err<1.e-13*max(x)
  break
 end % end if
end % end for
\end{lstlisting}
```

e vederlo formattato ed evidenziato esattamente come nell'editor di Matlab

```
function y = PageRank(H, v, gamma, itmax)
n = size(H,1); usn = 1/n; e = ones(n,1);
d = (H*e)';       % Calcolo out-degree dei nodi
dang = (d == 0); % Ricerca dangling nodes
dh = d + dang*n;   dh = 1./dh; % Vettore dei pesi
    corretto
x = rand(1,n); x = x./sum(x); v = v./sum(v); % Norm.
    pesi
for it=1:itmax
 y = x.*dh;
 y = y*H + usn*sum(dang.*x);
 y = y*gamma+(1-gamma)*v;
 err = norm(x-y,'inf');
 x = y;
 fprintf('Iteration %d Error %1.2e\n',it,err);
 if err<1.e-13*max(x)
  break
 end % end if
end % end for
```

Di nuovo, ulteriori informazioni su questo pacchetto possono essere trovate su CTAN. Sostanzialmente ha le stesse capacità di `minted`, ma ristrette al linguaggio e alla formattazione di codice Matlab/Octave (per cui potete mettere codice *inline*, caricarlo da un *file* sorgente, *etc.*).

Per inserire invece algoritmi in forma di *pseudocodice* all'interno di un documento TEX ci sono diverse alternative valide. La mie preferenze vanno per il pacchetto `\usepackage[]{algorithm2e}`, al solito manuale completo su CTAN, che vi permette di avere costrutti di questo tipo

```
\begin{algorithm}[H]
 \KwData{this text}
 \KwResult{how to write
 ↪  algorithm with \LaTeX2e }
 \label{alg:unesempiodialgorit
 ↪  mo}
 initialization\;
 \While{not at end of this
 ↪  document}{
  read current\;
  \eIf{understand}{
   go to next section\;
   current section becomes this
   ↪  one\;
   }{
   go back to the beginning of
   ↪  current section\;
  }
 }
 \caption{How to write
 ↪  algorithms}
\end{algorithm}
```

Data: this text
Result: how to write algorithm with LaTeX2e
initialization;
while *not at end of this document* **do**
  read current;
  if *understand* **then**
    go to next section;
    current section becomes this one;
  else
    go back to the beginning of current section;
  end
end

Algoritmo 1: How to write algorithms

che produce l'algoritmo che vedete, a cui potete fare riferimento al solito modo `Algoritmo~\ref{alg:unesempiodialgoritmo}`, ovvero Algoritmo 1. Come per i contenuti generali, le immagini e le tabelle potete far stampare a LaTeX una lista automatica degli algoritmi con il comando `\listofalgorithms`, di cui vedete un esempio in fondo a questo documento. In generale, il pacchetto `algorithm2e` non è localizzato in tutto le lingue, per cui se volete che compaia come qui con testate e informazioni in italiano, dovete aggiungere nel preambolo `\SetAlgorithmName{Algoritmo}{algoritmo}{Lista degli Algoritmi}`. Altre opzioni che possono essere utili per scrivere algoritmi con questo pacchetto sono

- **Opzioni di visualizzazione**, che possono essere passate sia come opzioni al singolo algoritmo, sia come opzioni generali quando si include il pacchetto (per cui vengono poi applicate uniformemente a tutte le istanze)

  `boxed` avere algoritmi racchiusi in una "scatola".

  `ruled` riga prima e dopo l'algoritmo, *caption* prima della riga in cima.

  `boxruled` combina i due precedenti, l'algoritmo è in una "scatola", la *caption* in cima e aggiunge una riga dopo di essa.

  `plain` il *default* che vedete nell'esempio.

- **Opzioni per la numerazione**, se volete che le righe dell'algoritmo siano numerate, è possibile farlo aggiungendo l'opzione `linesnumbered` al singolo

algoritmo, oppure a tutti passandola al pacchetto. Questo in genere è utile per poter fare dei riferimenti automatici ad una certa linea dello *pseudocodice* usando una coppia `\label{},\ref{}`.

Il **suggerimento** è di fissare l'aspetto per tutti gli algoritmi al caricamento del pacchetto e accendere la numerazione delle righe solamente nel caso in cui vogliate farci riferimento. Un po' secondo la filosofia dello scrivere solo cose utili.

Le possibili **varianti** e **alternative** di questo pacchetto sono ben spiegate su en.wikibooks.org/wiki/LaTeX/Algorithms.

8 Bibliografia

Se conoscete qualche umanista li avrete sicuramente sentiti lamentare del problema di gestire una bibliografia e gli stili di citazioni necessari. Questi seguono un certo numero di standard diversi e sono la stratificazione di necessità secolari per cui doverlo fare in maniera manuale è un bel grattacapo. LaTeX vi aiuta anche in questo. Per documenti brevi, come i report del laboratorio, è sufficiente utilizzare l'ambiente

```
\begin{thebibliography}{9}
\bibitem{latexcompanion}
Michel Goossens, Frank Mittelbach, and Alexander Samarin.
\textit{The \LaTeX\ Companion}.
Addison-Wesley, Reading, Massachusetts, 1993.

\bibitem{einstein}
Albert Einstein.
\textit{Zur Elektrodynamik bewegter K{\"o}rper}. (German)
[\textit{On the electrodynamics of moving bodies}].
Annalen der Physik, 322(10):891{921, 1905.

\bibitem{knuthwebsite}
Knuth: Computers and Typesetting,
\\\texttt{http://www-cs-faculty.stanford.edu/\~{}uno/abcde.html}
\end{thebibliography}
```

che vi permette poi di citare i lavori in bibliografia facendo `\cite{einstein}`, che vi restituisce [2]. Questo non necessità dell'uso di nessun pacchetto ausiliario, ma vi costa la fatica di dover formattare da voi la bibliografia... che è comunque una cosa estremamente noiosa da fare. Per gli articoli di matematica https://mathscinet.ams.org/mref vi produce già una versione TeX della citazione che potete copia-incollare nel vostro documento, quindi la cosa è gestibile.

Per un documento più corposo, come una tesi, costruire la bibliografia a questo modo è una maniera quasi sicura per esaurire le proprie forze. Il **suggerimento** è di utilizzare invece il pacchetto `\usepackage{biblatex}`. Questo vi permette di scegliere tra diversi stili di bibliografia semplicemente cambiando una delle

sue opzioni e di avere in realtà le informazioni conservate in un file "database". Concretamente, aggiungete nel preambolo:

```
\usepackage{biblatex}
\addbibresource{bibliografia.bib}
```

dove `bibliografia.bib` è il nome del file che contiene il database con gli articoli. Potete poi far apparire la bibliografia con il comando `\printbibliography`.

Il contenuto del file `bibliografia.bib` è fatto di una sequenza di entrate del tipo

```
@article{CitekeyArticle,
  author    = "P. J. Cohen",
  title     = "The independence of the continuum hypothesis",
  journal   = "Proceedings of the National Academy of Sciences",
  year      = 1963,
  volume    = "50",
  number    = "6",
  pages     = "1143--1148",
}
```

che possono poi essere citate con `\cite{CitekeyArticle}`, cioè in maniera identica al caso precedente. Informazioni relative ai formati delle entrate del file `.bib` possono essere trovate su www.bibtex.com. In generale è possibile scaricare direttamente per articoli e libri l'entrata di riferimento da scholar.google.it o di nuovo da https://mathscinet.ams.org/mref (la seconda vi riporta anche il nome abbreviato delle riviste, che è un retaggio di quando si doveva stampare tutto e quindi risparmiava carta e inchiostro).

Opzioni **utili** per il pacchetto sono lo stile di citazione `style=`, che nel vostro caso sarà quasi sempre `style=numeric-comp`. La scelta del software di *backend* che vi trasforma la bibliografia dal `.bib` a dei comandi LaTeX, per cui, `backend=biber` (che tuttavia dovete aver installato sulla vostra macchina... guardate il manuale!). Infine, quanti autori devono essere scritti nella bibliografia prima che compaia un "et Al.", `maxbibnames=`. Riassumendo,

```
\usepackage[backend=biber,style=numeric-comp,maxbibnames=10]{bib␣
↪ latex}
```

Per compilare un file con bibliografia è necessario fare più passaggi del compilatore LaTeX in genere un editor come TeXStudio è in grado di farlo in automatico. Se compilate da *shell* dovete farlo manualmente.

9 Varie ed eventuali

- Stili di testo *Enfasi* `\emph{Enfasi}`, **grassetto** `\textbf{Grassetto}`, *corsivo* `\textit{corsivo}`, Teletype `\texttt{Teletype}`,
- Link: `\usepackage{hyperref}`, il mio sito (cliccate è un link!) che si ottiene facendo:

```
\href{https://fdurastante.github.io/}{il mio sito}
```

A Appendici: dove cerco quello che non so dove trovare

Se volete creare un insieme di appendici per un documento che state scrivendo, questo può essere fatto utilizzando prima il comando `\appendix`, e poi utilizzando i normali comandi di sezionamento.

Alcuni riferimenti web utili

- Il luogo delle risposte: tex.stackexchange.com,
- Esempi d'uso: texample.net/
- Rubare dai maestri: arxiv.org se c'è un articolo che ha qualche utilizzo di LaTeX che vi piace potete scaricarne il file sorgente e vedere con che comandi hanno ottenuto il risultato che vi ha emozionato,
- Informazioni generali: www.latex-project.org
- La lista completa dei simboli LaTeX con riferimento ai pacchetti in cui sono contenuti symbols-a4.pdf.

L'ultima parte di questo documento è stata ottenuta usando:

```
\newpage
\thispagestyle{empty}

\addcontentsline{toc}{section}{\listfigurename}
\listoffigures
\addcontentsline{toc}{section}{\listtablename}
\listoftables
\addcontentsline{toc}{section}{Lista degli Algoritmi}
\listofalgorithms
\tableofcontents
```

Riferimenti bibliografici

[1] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

[2] Albert Einstein. *Zur Elektrodynamik bewegter Körper*. (German) [*On the electrodynamics of moving bodies*]. Annalen der Physik, 322(10):891–921, 1905.

[3] Knuth: Computers and Typesetting,
`http://www-cs-faculty.stanford.edu/~uno/abcde.html`

Elenco delle figure

1 helloworld.pdf 3
2 Didascalia della figura 7

Elenco delle tabelle

1 Didascalia della mia tabella 9

Lista degli Algoritmi

1 How to write algorithms 12

Indice

1 Installazione **1**
1.1 Editor 2

2 Il vostro primo documento **2**

3 Partizioni e riferimenti incrociati **4**

4 Matematica **5**

5 Figure **6**

6 Tabelle **8**

7 Codici e algoritmi **9**

8 Bibliografia **13**

9 Varie ed eventuali **14**

A Appendici: dove cerco quello che non so dove trovare **15**

Elenco delle figure **16**

Elenco delle tabelle **16**

Lista degli Algoritmi **16**