

TELE LATEX

Breve corso telematico di LaTeX per fisici

Quest'opera è distribuita con licenza [Creative Commons «Attribuzione – Non commerciale – Condividi allo stesso modo 4.0 Internazionale»](#).



Indice

Premessa	1
Episodio 1: Introduzione e struttura del documento	2
1.1 Concetti preliminari	3
1.2 Struttura base	4
1.2.1 Preambolo	4
1.2.2 Documento	5
1.3 Formattazione	7
1.3.1 Dimensioni	7
1.4 Elenchi	9
1.5 Riferimenti ipertestuali	11
1.5.1 Nome	11
1.6 Bibliografia	12
Episodio 2: Ambienti matematici	13
2.1 Pacchetti utili per la matematica: introduzione	14
2.2 Ambienti matematici	14
2.2.1 Formule in linea con il testo	14
2.2.2 L'ambiente <i>equation</i> e simili	15
2.3 Simboli matematici	16
2.3.1 Apici, pedici, frazioni	17
2.3.2 Parentesi adattive	17
2.3.3 Funzioni particolari	18
2.3.4 Spaziatura nelle equazioni	18
2.3.5 Sommatorie e produttorie	18
2.3.6 Integrali	19
2.3.7 Matrici e sistemi	19
2.4 Altri strumenti utili	20
2.4.1 Il pacchetto <i>physics</i>	20
2.4.2 Labels e riferimenti incrociati	23
2.4.3 Equazioni "in scatola"	23
2.4.4 Equazioni con unità di misura	23
2.4.5 Teoremi ed oggetti simili	24
2.4.6 Comandi personalizzati	26
2.4.7 Cancellazioni nelle equazioni	27
2.4.8 Font di testo in ambiente matematico	28

Episodio 3: Figure e grafica	29
3.1 Perché <i>float</i> ?	30
3.2 Tabelle	30
3.2.1 Pacchetti essenziali	30
3.2.2 L'ambiente <i>tabular</i>	31
3.2.3 L'ambiente <i>table</i>	32
3.2.4 <i>multicolumn</i> e <i>multirow</i>	33
3.2.5 Tabelle circondate dal testo con <i>wraptable</i>	34
3.3 Figure e grafica	35
3.3.1 Immagini raster e vettoriali	35
3.3.2 Pacchetti essenziali e cartella immagini	36
3.3.3 Il comando <i>includegraphics</i> e l'ambiente <i>figure</i>	37
3.3.4 Figure circondate dal testo con <i>wrapfigure</i>	39
3.3.5 Sottofigure con <i>subfloat</i>	39
3.3.6 Grafica vettoriale con <i>TikZ</i>	40
Riferimenti	46

TELE LATEX

Premessa

TELE LATEX è un breve corso in tre episodi, prodotto in collaborazione tra AISF Palermo e AISF Bari, che non ha alcuna pretesa di completezza e, anzi, vuole essere essenzialmente un' ad un modo più efficiente e semplice per scrivere ciò che a noi (aspiranti) fisici più importa: report, relazioni di laboratorio, tesi e slide.

Lo scopo fondamentale degli incontri sarà dunque quello di presentare la struttura base di un documento e i modi per inserirvi equazioni, tabelle, immagini, bibliografia etc. entrando direttamente in contatto con l'editor di testo e tralasciando tutti gli aspetti tecnici che riguardano il funzionamento e la filosofia di LATEX. Per i più curiosi, rimandiamo alle risorse e ai testi presenti nei riferimenti bibliografici.

Prima di continuare, riteniamo doveroso ricordare a chiunque si accinga ad utilizzare per le prime volte questo fantastico strumento di non temere eventuali¹ errori di compilazione. In generale, crediamo che ogni fisico non dovrebbe mai aver paura di una certa situazione problematica, ma anzi dovrebbe sempre tentare, in ogni modo possibile, di risolverla superando tutti gli inevitabili ostacoli che via via si presentano.

I curatori,
Giovanni Cusimano, Enrico Di Benedetto, Francesco Schiavone

¹Leggasi "sicuri".

TELE LATEX

Episodio 1:

Introduzione e struttura del documento

A cura di²:

Giovanni Cusimano

giovanni.cusimano07@community.unipa.it

giovanni.cusimano07@you.unipa.it

5 dicembre 2020

Sommario

In questo primo incontro, come da titolo, ci si occuperà dei fondamenti di LATEX.

Dopo aver coperto alcuni CONCETTI PRELIMINARI, in cui si spiegherà cosa è, nella pratica, necessario e fondamentale per iniziare subito a scrivere, si introdurrà la STRUTTURA BASE di un documento. Il seguito è invece dedicato alla FORMATTAZIONE del testo, alle varie modalità che esistono per creare ELENCHI di varia natura, a come fare RIFERIMENTI IPERTESTUALI a parti del testo o a pagine web e alla BIBLIOGRAFIA.

²Sulla base della dispensa redatta da Dario Zarcone, Pietro Castronovo e Santi Macaluso, 27 aprile 2020, che ringrazio infinitamente.

1.1 Concetti preliminari

Cosa è L^AT_EX

L^AT_EX (pronunciato /'latek/, e non /'lateks/, perché la X è in realtà una χ (chi greca) maiuscola)[[Wikc](#)] è un sistema di composizione tipografica, ormai standard *de facto* per l'editoria scientifica per merito delle ampie possibilità di personalizzazione e, soprattutto, per merito della semplice gestione delle formule matematiche e della bibliografia, caratteristiche fondamentali di qualunque documento scientifico.

Ciò che rende L^AT_EX uno strumento molto potente è la capacità di produrre documenti di alta qualità tipografica, permettendo all'utente di concentrarsi solo sul contenuto.

Per raggiungere questo obiettivo, L^AT_EX risulta diverso alla radice da programmi di videoscrittura come *Microsoft Word* o *LibreOffice Writer*: mentre in questi quello che si scrive corrisponde al documento che viene prodotto (*wysiwyg*, *What You See Is What You Get*), in L^AT_EX si scrive un *codice sorgente* che verrà successivamente *compilato* ("L^AT_EX [...] compone in modo asincrono"[[PG17](#)]); l'output della compilazione è il documento in formato .pdf.

Strumenti necessari per scrivere in L^AT_EX

Per scrivere un documento in L^AT_EX sono quindi necessari almeno un **editor di testo**, con cui produrre il sorgente, e un **compilatore**. Esistono pacchetti software completi, che comprendono tutto quello che serve per scrivere in L^AT_EX, come mostrato in Figura 1.1 di seguito. Il software che utilizzeremo durante il corso è [Overleaf](#).



Linux

Check your Linux distributions software source for a TeX distribution including LaTeX. You can also install the current [TeX Live distribution](#) directly---in fact this may be advisable as many Linux distributions only contain older versions of TeX Live, see [Linux TeX Live package status](#) for details.



Mac OS

The [MacTeX](#) distribution contains everything you need, including a complete TeX system with LaTeX itself and editors to write documents.



Windows

Check out the [MiKTeX](#) or [proTeXt](#) or [TeX Live](#) distributions; they contain a complete TeX system with LaTeX itself and editors to write documents.



Online

LaTeX online services like [Papeeria](#), [Overleaf](#), [ShareLaTeX](#), [Datazar](#), and [LaTeX base](#) offer the ability to edit, view and download LaTeX files and resulting PDFs.

Figura 1.1: Alcune dei programmi disponibili per ogni sistema operativo. Fonte: [The L^AT_EX Project](#)

1.2 Struttura base

Ogni progetto L^AT_EX deve contenere almeno il file sorgente del documento che verrà compilato, il `main.tex`³.

Questo sorgente si compone di due parti principali, un *preambolo*, in cui si settano una serie di impostazioni del documento, e il *documento* vero e proprio.

All'interno del codice è possibile (ed è consigliato) *commentare*: tutto quello che segue % fino alla riga successiva non viene letto dal compilatore.

1.2.1 Preambolo

Un esempio di preambolo è riportato in seguito:

```
%%% INIZIO PREAMBOLO %%%
\documentclass[12pt, a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[italian]{babel}
\usepackage{xcolor}
\usepackage[colorlinks, linkcolor=blue]{hyperref}
%%% FINE PREAMBOLO %%%
\caption{Esempio minimale di preambolo di un documento \LaTeX{}.\}
%No, Einstein non ha mai scritto un articolo scientifico con questo sistema tipografico.
```

Document class

Il primo comando che va scritto è la dichiarazione della `\documentclass`, in cui si seleziona il tipo di documento da scrivere. Ci sono diverse classi tra cui scegliere⁴ e ognuna imposterà il documento in un determinato modo; con la classe `book`, ad esempio, è possibile scrivere libri: questa supporta la divisione in capitoli, la numerazione delle pagine da libro, la pagina del titolo. La classe usata nell'esempio (e in questo documento) è `article`, una delle classi incluse in ogni distribuzione di L^AT_EX, che considera una impaginazione e una organizzazione del testo da articolo scientifico.

Le opzioni della document class scelta, inserite in parentesi quadre, permettono di settare impostazioni del documento: in particolare, `12pt` setta la dimensione del font⁵ e `a4paper` imposta la dimensione del documento al formato A4.

Packages

Ogni funzionalità di L^AT_EX va quindi inserita nel preambolo sotto forma di *pacchetto*, usando il comando `\usepackage[opzioni_pacchetto]{nome_pacchetto}`. Nell'esempio di sopra si utilizzano:

- `\usepackage[utf8]{inputenc}`
per impostare la codifica UTF-8 dei caratteri e permettere quindi l'utilizzo di lettere accentedate

³ .tex è l'estensione dei file sorgente L^AT_EX: si tratta, comunque, di semplici file di testo.

⁴Oltre alle classi incluse in LaTe_X, su Overleaf è possibile scegliere una serie di classi aggiuntive che ne espandono le funzionalità, permettendo di scrivere libri di poesie, curriculum, presentazioni.

⁵La classe `article` supporta font di dimensione massima 12pt, e di default setta una dimensione di 10pt

- `\usepackage[italian]{babel}`
per impostare la lingua italiana nel documento e la corretta suddivisione in sillabe (necessaria quando il testo di una stessa parola deve essere suddiviso su più di una riga).
- `\usepackage{xcolor}`
per l'uso di colori nel testo.
- `\usepackage[colorlinks, linkcolor=blue]{hyperref}`
per l'utilizzo di riferimenti ipertestuali colorati in blu.

Esistono comunque tantissimi pacchetti per i più diversi scopi: uno dei più utilizzati in ambito scientifico, e oggetto della prossima puntata, è `amsmath`, che fornisce una serie di ambienti in cui scrivere matematica. Un altro pacchetto molto utile è `geometry`, che permette di settare i margini del testo manualmente⁶ attraverso, come ad esempio si è operato in questo documento:

```
\usepackage[right=20mm, left=20mm, top=20mm, bottom=20mm]{geometry}
```

1.2.2 Documento

Tutto quello che riguarda il documento sta all'interno del blocco

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

Questo genere di "blocchi" `begin/end` corrispondono a dei "blocchi" di testo (liste, equazioni, `abstract`...). Di seguito presento un esempio di document di cui ci occuperemo più in dettaglio:

```
%%% INIZIO DOCUMENTO %%%

\begin{document}

    \title{Zur Elektrodynamik bewegter Körper}
    \author{Albert Einstein}
    \date{Giugno 1905}

\begin{abstract}

    In realtà la relatività galileiana non funziona.

\end{abstract}

\tableofcontents
```

⁶È sconsigliato modificare i margini del testo: quelli scelti di default da L^AT_EX, infatti, sono stati studiati per garantire la leggibilità migliore possibile.[\[PG17\]](#)

```

\maketitle
\section{Questa è una sezione}
Un documento di tipo article si divide in sezioni, sottosezioni,
sottosottosezioni.
\subsection{Questa è una sottosezione}
\subsubsection{Questa è una sottosottosezione}

\end{document}
%%% FINE DOCUMENTO %%%

```

Esempio di codice 1: sorgente minimale di un documento L^AT_EX.

Titolo e abstract

Le impostazioni di titolo, autore e data sono impostate rispettivamente dai comandi `title`, `author`, `date`. Queste informazioni vengono usate per "scrivere" il titolo usando il comando `\maketitle`.

È possibile, seppur opzionale, mostrare un `abstract` usando il blocco

```

\begin{abstract}
...
\end{abstract}

```

Divisione del documento

Un documento viene in genere diviso in parti: se un `book` si divide in capitoli e sottocapitoli, un `article` si divide in sezioni e sottosezioni, come si vedrà di seguito. Queste vengono automaticamente referenziati e inseriti nell'indice, richiamato nel documento con il comando `\tableofcontents`.

È importante notare che questa divisione tra "*contenuto*" e "*contenitore*", e quindi tra le impostazioni del preambolo e il contenuto del documento, ha un grosso vantaggio: è sufficiente scrivere un preambolo una volta sola e copiarlo in ogni nuovo progetto per avere documenti con la stessa tipografia.

Un documento di tipo `article` si divide in sezioni, sottosezioni, sottosottosezioni, che automaticamente vengono inserite nell'indice. Una sezione, come questa, si crea con il comando:

```
\section{Questa è una sezione}
```

In modo analogo, con i comandi:

```

\subsection{Questa è una sottosezione}
%Contenuto della sottosezione
\subsubsection{Questa è una sottosottosezione}
%Contenuto della sottosottosezione

```

è possibile creare rispettivamente una sottosezione e una sottosottosezione, come visto nell'esempio di codice 1.

È importante notare che sezioni e sottosezioni vengono automaticamente numerate in maniera progressiva. Affinché questo non si verifichi, basta aggiungere un * alla fine del comando, ad esempio:

```
\section*{Questa è una sezione non numerata e non indicizzata}
```

1.3 Formattazione

Stili del testo

È possibile rendere il testo in diversi modi, tra cui⁷:

- `\textbf{grassetto}`
grassetto
- `\textit{corsivo}`
o, in modo equivalente, `\emph{corsivo}`
corsivo
- `\texttt{monospace}`
`monospace`
- `\underline{sottolineato}`
sottolineato

È inoltre sempre possibile combinare gli stili, ad esempio

```
\textbf{\textit{grassetto corsivo}}
```

grassetto corsivo

1.3.1 Dimensioni

È possibile modificare la dimensione del testo usando comandi come `\large`. Per modificare la dimensione di solo una porzione di testo, si utilizza una sintassi come la seguente

```
{\large Questa porzione di testo è grande,} questa no  
{\tiny e questa è molto piccola.}
```

Questa porzione di testo è grande, questa no e questa è molto piccola.

ovvero il comando da usare è di tipo `{\large testo}`. Si può trovare una lista delle dimensioni disponibili e dei relativi comandi in [PG17].

Colori

Usando il pacchetto `xcolor` è possibile rendere il testo colorato

```
\textcolor{blue}{testo in blu}
```

testo in blu

Per vedere i colori disponibili, e come usare il pacchetto `xcolor` con colori personalizzati, si rimanda a [Col].

⁷Per una lista completa delle possibilità, si veda [qui](#)

Paragrafi

Per cominciare un nuovo paragrafo è sufficiente lasciare una riga. Per andare a capo, invece, basta scrivere `\backslash`. La differenza tra i due comandi sta nell'indentazione che L^AT_EX aggiunge all'inizio di ogni paragrafo.

Paragrafo uno

Paragrafo due `\backslash` in cui vado a capo

da cui si ottiene:

Paragrafo uno

 Paragrafo due
in cui vado a capo

È possibile interrompere la pagina attuale per scrivere nella successiva usando il comando `\newpage` o il comando `\pagebreak`⁸.

Allineamento del testo

Il testo immesso in L^AT_EX viene automaticamente giustificato. A seconda delle esigenze⁹, si può allineare un blocco di testo al centro centro scrivendolo all'interno di un blocco `center`:

```
\begin{center}  
    Sono un blocco di testo allineato al centro.  
\end{center}
```

da cui si ottiene:

Sono un blocco di testo allineato al centro.

o anche, scrivendolo all'interno di un blocco `flushright`, allinearla al centro:

```
\begin{flushright}  
    Sono un blocco di testo allineato a destra.  
\end{flushright}
```

da cui si ottiene:

Sono un blocco di testo allineato a destra.

Note¹⁰

È possibile aggiungere al testo una nota a piè di pagina con il comando:

Questo testo ha una nota a piè di pagina`\footnote{E questa è la nota}`.

La nota viene automaticamente aggiunta nel testo e numerata. Il risultato è il seguente:

Questo testo ha una nota a piè di pagina¹¹.

⁸Per approfondimenti sulla differenza tra i due comandi, si veda [qui](#).

⁹Per un approfondimento in merito, si veda [qui](#)

¹⁰Per approfondimenti in merito, vedere [qui](#) per le note a piè di pagina e [qui](#) per le note a margine

¹¹E questa è la nota

Caratteri riservati

Alcuni caratteri sono riservati da L^AT_EX e non possono essere aggiunti nel documento semplicemente scrivendoli: è il caso, ad esempio, di "{", che, se scritta direttamente, verrà interpretato dal compilatore come una direttiva. La lista di tutti i caratteri riservati è riportata di seguito nella Tabella 1.1.

CARATTERE	COMANDO
%	\%
&	\&
\$	\\$
#	\#
-	_
{	\{
}	\}
\	\textbackslash
~	\textasciitilde
^	\textasciicircum

Tabella 1.1: Come scrivere i caratteri riservati da L^AT_EX nel sorgente.

1.4 Elenchi

Per scrivere elenchi sono disponibili gli ambienti `itemize` e `enumerate`. In particolare, per gli elenchi non numerati si utilizza

```
\begin{itemize}
    \item Elemento dell'elenco
    \item Altro elemento dell'elenco
\end{itemize}
```

- Elemento dell'elenco
- Altro elemento dell'elenco

Cioé ogni nuovo elemento si indica con `\item`.

Per gli elenchi numerati la sintassi è identica, ma si utilizza l'ambiente `enumerate`

```
\begin{enumerate}
    \item Elemento numero uno
    \item Elemento numero due
\end{enumerate}
```

1. Elemento numero uno
2. Elemento numero due

Gli elenchi possono essere semplicemente annidati, ovvero organizzati gerarchicamente, scrivendo in un `\item` un blocco `itemize` o `enumerate`:

```
\begin{enumerate}
    \item \begin{enumerate}
        \item Elemento numero uno
        \item Elemento numero due
    \end{enumerate}
    \item Elemento numero due
\end{enumerate}
```

1. (a) Elemento a
(b) Elemento b
2. Elemento numero due

In questo caso l'ambiente assegna, in automatico, una numerazione di tipo diverso all'interno della "sottolista".

Il comando `\item` accetta tra parentesi quadre un nome personalizzato dell'elemento. Ad esempio

```
\begin{enumerate}
    \item[Uno] Elemento numero uno
    \item[Due] Elemento numero due
\end{enumerate}
```

Uno Elemento numero uno

Due Elemento numero due

Tuttavia, se si vuole fare un elenco in cui ogni elemento comincia con un testo a piacere, ha più senso utilizzare `description`

```
\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}
```

itemize Per gli elenchi puntati.

enumerate Per gli elenchi numerati.

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

Per riferimenti ulteriori (cambiare gli indici degli elenchi, ad esempio) si rimanda a [Lis].

1.5 Riferimenti ipertestuali

Utilizzando il pacchetto hyperref è possibile fare riferimenti a parti diverse del testo. È sufficiente aggiungere, ad esempio all'interno di una sottosezione, il comando `\label{nomelabel}`, e riferirsi alla sezione utilizzando `\autoref{nomelabel}`, come nell'esempio di seguito:

```
.....  
\usepackage{hyperref}  
\begin{document}  
\subsection{Nome}\label{nomelabel}  
%%% Questa è una sottosezione a cui fare riferimento usando nomelabel %%%  
\end{document}
```

1.5.1 Nome

Questa è una sottosezione a cui fare riferimento usando `nomelabel`.

Si può quindi farvi riferimento usando il comando `autoref`:

Mi riferisco adesso a `\autoref{nomelabel}`.

Mi riferisco adesso a sottosezione 1.5.1.

.....

Esempio di codice 2: Riferimento a parti diverse del testo

In generale, `\label{}` si riferisce sempre al primo blocco o costrutto supportato alla sua sinistra. Le opzioni di hyperref impostate in precedenza attivano anche i riferimenti colorati (`colorlinks`) e i link in blu (`linkcolor=blue`).

È importante notare che babel fornisce la lingua italiana, quindi i riferimenti fatti con `\autoref{}` saranno tradotti in italiano ("sottosezione" invece di "subsection", "equazione" anziché "equation" e così via). Per questo è necessario, nel preambolo, caricare prima il pacchetto babel e poi il pacchetto hyperref.

Siti web e URL

Si possono creare dei collegamenti a pagine web sia tramite il comando `\url` per mostrare per esteso il link, sia tramite `\href` per inserire il collegamento in una parola o frase:

```
.....  
Per ulteriori informazioni clicca su  
\href{http://www.sharelatex.com}{Something Linky} o copia l'indirizzo seguente: \\  
\url{http://www.sharelatex.com}
```

Per ulteriori informazioni clicca [qui](#) o copia l'indirizzo seguente:

<http://www.sharelatex.com>

1.6 Bibliografia

Il modo sicuramente più moderno per gestire la bibliografia in L^AT_EX è tramite l'utilizzo del pacchetto `biblatex`. È necessario, a tal proposito, dichiarare nel preambolo il documento cui attingere per la bibliografia tramite il comando `\addbibresource{nomefile.bib}`. Il file in questione deve avere alcune caratteristiche particolari, illustrate [qui](#).

I comandi principali del pacchetto sono:

- `\cite{qualcosa}`

che di base inserisce un numero [1] per citare "qualcosa" dalla bibliografia.

- `\printbibliography`

che stampa la lista di elementi della bibliografia citati.

Il pacchetto in questione offre pressoché infinite possibilità di personalizzazione, tra cui modalità standard a seconda del formato richiesto da determinate istituzioni, che si preferiscono, per brevità, non riportare. Per ulteriori informazioni sulla bibliografia, due risorse estremamente utili sono [\[PG17\]](#) e [\[Oveb\]](#).

TELE LATEX

Episodio 2:

Ambienti matematici

A cura di¹²:

Enrico Di Benedetto

enrico.dibenedetto@community.unipa.it

enrico.dibenedetto@ai-sf.it

12 dicembre 2020

Sommario

In questa seconda lezione, si tratterà di come scrivere simboli matematici in un documento LATEX.
[Ovec] In particolare, si introdurranno gli ambienti principali per scrivere equazioni, matrici, integrali etc. Poi, si accennerà alla possibilità molto utile di introdurre comandi *user-defined* e si introdurrà il pacchetto che permette di introdurre diversi simboli propri del *fisichese*, come gli operatori vettoriali e la notazione di *Dirac*. [Wikb]

¹²Sulla base della dispensa redatta da Pietro Castronovo, 2 maggio 2020, che ringrazio un'infinità non numerabile di volte.

2.1 Pacchetti utili per la matematica: introduzione

Una delle principali caratteristiche che fanno di L^AT_EX lo strumento più usato per la scrittura di articoli e libri scientifici è la sua predisposizione all'inserimento nel testo di formule ed altri oggetti matematici.

I più importanti pacchetti utili all'inserimento di oggetti matematici in un documento L^AT_EX sono:

```
%Pacchetti dell'American Mathematical Society%
\usepackage{amsmath,amsthm,amssymb,amscd}
\usepackage{siunitx} %unità di misura
```

che permettono di inserire la maggior parte degli oggetti matematici che si utilizzano solitamente e permettono di scrivere decentemente le unità di misura¹³. A questi possono aggiungersene molti altri, a seconda delle evenienze. In particolare, noi sfrutteremo anche i pacchetti

```
\usepackage{cancel} %per eseguire le cancellazioni
\usepackage{xparse} %serve per poter utilizzare physics
\usepackage{physics} %pacchetto per simboli fisici
```

Alcuni pacchetti specifici per cambiare il font dei caratteri in speciali occasioni verranno introdotti nel seguito quando occorreranno.

2.2 Ambienti matematici

2.2.1 Formule in linea con il testo

Per inserire un carattere matematico o una formula *in mezzo al testo*, si usa il doppio dollaro (\$\$). Tale simbolo crea un *ambiente matematico* all'interno del paragrafo, e tra i due \$ si può digitare l'espressione che si desidera inserire. Ad esempio, il codice:

Un cerchio di raggio $\$r\$$ ha circonferenza pari a $\$2\pi r\$$

produce il testo:

Un cerchio di raggio r ha circonferenza pari a $2\pi r$

Tale funzione è utile per scrivere singoli simboli o brevi espressioni, ma risulta poco efficace per espressioni più voluminose, come:

Si dimostra che $\$e^x=\sum_{n=0}^{\infty}\frac{x^n}{n!}$

che produce il seguente testo, decisamente sgradevole:

Si dimostra che $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$

Si noti qui una caratteristica del tutto generale delle formule in linea col testo: quando si scrive un simbolo che possiede degli indici sopra o sotto, come ad esempio la sommatoria in questo caso, L^AT_EX pone questi indici accanto all'oggetto. Questo problema può essere risolto in due modi: semplicemente *non* scrivendo formule in linea col testo, utilizzando invece uno degli ambienti che verranno introdotti in sottosezione 2.2.2, oppure utilizzando il comando `\displaystyle`. [Ovea] Il comando ha la seguente sintassi

¹³Siete fisici, mettete *sempre* le unità di misura!

Si dimostra che $\sum_{n=0}^{\infty} \frac{x^n}{n!}$
e produce in output

$$Si dimostra che e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

2.2.2 L'ambiente *equation* e simili

Per scrivere equazioni su righe a parte, centrate, e se necessario numerate, si impiega un *ambiente* separato, che si apre e chiude con le istruzioni `\begin{}``` ed `\end{}```.

Il più comune tra questi ambienti è *equation*, che consente di scrivere un'unica equazione, la quale viene centrata e numerata (per evitare la numerazione basta aggiungere un asterisco alla dichiarazione dell'ambiente). Ad esempio, il codice:

La legge fondamentale della dinamica ha la forma:

```
\begin{equation}
\vec{F}=\frac{d \vec{p}}{dt}
\end{equation}
```

Per sistemi a massa costante, ciò equivale a:

```
\begin{equation*}
\vec{F}=m \vec{a}
\end{equation*}
```

produce il seguente testo:

La legge fondamentale della dinamica ha la forma:

$$\vec{F} = \frac{d\vec{p}}{dt} \tag{2.1}$$

Per sistemi a massa costante, ciò equivale a:

$$\vec{F} = m\vec{a}$$

È importante porre l'accento sulla struttura formale degli ambienti in L^AT_EX. Un ambiente si apre sempre con un `\begin{}``` e si chiude sempre con un `\end{}```, indipendentemente dal tipo specifico di ambiente. Si noti che anche il documento stesso è un tipo di ambiente, tant'è che inizia con `\begin{document}``` e termina con `\end{document}```.

Per inserire più equazioni nello stesso blocco si usa l'ambiente *gather*. Il comando che permette di andare a capo è `\`` e deve essere inserito alla fine di ogni riga tranne l'ultima. Ad esempio il codice:

```
\begin{gather}
f(x)=x^4+2x^2-5x\`\\
f'(x)=4x^3+4x-5\`\\
f''(x)=12x^2+4
\end{gather}
```

produce le equazioni

$$f(x) = x^4 + 2x^2 - 5x \quad (2.2)$$

$$f'(x) = 4x^3 + 4x - 5 \quad (2.3)$$

$$f''(x) = 12x^2 + 4 \quad (2.4)$$

Come si nota, ogni riga viene numerata a sé e, inoltre, esse non sono allineate l'una con l'altra, ma sono solamente centrate. Se si vuole che le equazioni siano allineate rispetto a un certo punto, come ad esempio rispetto ai caratteri = delle diverse equazioni, si può usare l'ambiente `align`. Scrivendo ad esempio:

```
\begin{align}
f(x) &= x^4+2x^2-5x \\
f'(x) &= 4x^3+4x-5 \\
f''(x) &= 12x^2+4
\end{align}
```

si produce

$$f(x) = x^4 + 2x^2 - 5x \quad (2.5)$$

$$f'(x) = 4x^3 + 4x - 5 \quad (2.6)$$

$$f''(x) = 12x^2 + 4 \quad (2.7)$$

dove il simbolo `&` indica all'ambiente di allineare le diverse righe rispetto al carattere immediatamente alla sua sinistra.

Per inserire equazioni molto lunghe, che vanno scritte su più righe e che si vuole siano allineate in qualche modo, si inserisce all'interno di `equation` l'ambiente `split`, nel modo seguente:

```
\begin{equation}
\begin{aligned}
(a+b)^3&=\\
&=a^3+3a^2b+3ab^2+b^3
\end{aligned}
\end{equation}
```

dove il simbolo `\backslash\backslash` consente di andare a capo, mentre `&` allinea tra loro i caratteri immediatamente seguenti. Il codice sopra riportato genera l'equazione:

$$\begin{aligned} (a+b)^3 &= \\ &= a^3 + 3a^2b + 3ab^2 + b^3 \end{aligned} \quad (2.8)$$

In questo caso, l'indice con cui le varie righe vengono numerate è unico in quanto si suppone che si stia scrivendo la stessa equazione, solamente su più righe.

2.3 Simboli matematici

Vediamo ora come scrivere in `LATeX` alcuni simboli ed oggetti matematici. Per una lista completa e di facile navigazione, si veda il documento `LaTeX_symbols.pdf` che contiene, tra le altre cose, tabelle esaustive delle lettere greche e dei diversi font matematici utilizzabili. Non listeremo tutti i comandi per tutti i simboli matematici. Una lista completa può essere trovata in bibliografia.
[\[Gia\]](#)

2.3.1 Apici, pedici, frazioni

Per apporre apici e pedici ad un simbolo, si usano `^` e `_`, rispettivamente. Se l'apice (o il pedice) contiene più di un carattere, il suo contenuto va messo tra parentesi graffe. Ad esempio, `a^2_{ij}` restituisce il simbolo a_{ij}^2 .

Il miglior comando per scrivere una frazione è `\frac{ }{ }`, i cui argomenti sono, in ordine, il numeratore ed il denominatore. Per esempio, il codice

```
\begin{equation*}
K=\frac{1}{2}mv^2=\frac{p^2}{2m}
\end{equation*}
```

restituisce

$$K = \frac{1}{2}mv^2 = \frac{p^2}{2m}$$

Poiché nel L^AT_EX standard non esiste un comando per scrivere la derivata, essa deve essere scritta come frazione. Ad esempio, il noto *teorema di Schwartz* sull'inversione dell'ordine delle derivate parziali verrebbe scritto come

```
\begin{equation*}
\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}
\end{equation*}
```

producendo in output

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}$$

dove il simbolo ∂ è introdotto dal comando `\partial`. Per la derivata ordinaria, basta inserire una semplicissima `d` al posto di `\partial`. Questo rappresenta soltanto un modo, poco efficiente, di scrivere la derivata, che viene inserito soltanto per ragioni didattiche. In seguito vedremo come scrivere più concisamente tale risultato tramite il pacchetto `physics`. [Bar]

2.3.2 Parentesi adattive

Un frammento di codice del tipo:

```
\begin{equation}
(x+\frac{1}{2})(x-\frac{1}{2})=x^2-\frac{1}{4}
\end{equation}
```

restituisce lo scadente risultato

$$(x + \frac{1}{2})(x - \frac{1}{2}) = x^2 - \frac{1}{4} \tag{2.9}$$

Per far sì che L^AT_EX adatti automaticamente le parentesi al loro contenuto, si utilizzano i comandi `\left` e `\right`, seguiti rispettivamente dalla parentesi sinistra e da quella destra. Ad esempio, il modo corretto di scrivere l'equazione (2.9) è:

```
\begin{equation}
\left(x+\frac{1}{2}\right)\left(x-\frac{1}{2}\right)=x^2-\frac{1}{4}
\end{equation}
```

che restituisce:

$$\left(x + \frac{1}{2}\right) \left(x - \frac{1}{2}\right) = x^2 - \frac{1}{4} \quad (2.10)$$

Tanti simboli possono essere resi adattivi tramite questa sintassi: tutti i tipi di parentesi (si ricorda che il comando per le graffe è `\{ \}`), il simbolo di valore assoluto e il simbolo di norma $\| \|$ (introdotto dal comando `\| \|`).

2.3.3 Funzioni particolari

Il pacchetto `\amsmath` prevede comandi speciali per scrivere in font rotondo le funzioni come `sin`, `cos` e `log` all'interno delle equazioni. Tali comandi sono costituiti semplicemente dal simbolo della funzione, preceduto da "`\`". Per esempio, il codice:

```
\begin{equation}
\cos(\alpha+\beta)=\cos\alpha\cos\beta-
\sin\alpha\sin\beta
\end{equation}
```

restituisce:

$$\cos(\alpha + \beta) = \cos \alpha \cos \beta - \sin \alpha \sin \beta \quad (2.11)$$

È possibile dichiarare le proprie funzioni tramite il comando `\DeclareMathOperator{ }{ }{ }`. Nelle prime parentesi va inserito il nome del comando che verrà poi richiamato, nelle seconde il testo che poi verrà stampato. Così ad esempio se si deve indicare spesso che un'applicazione è un endomorfismo di V , si può definire nel preambolo

```
\DeclareMathOperator{\End}{End}
```

e poi richiamare nel testo

```
f=\End V
```

ottenendo in output $f = \text{End } V$.

2.3.4 Spaziatura nelle equazioni

È possibile inserire manualmente degli spazi tra gli elementi di un'equazione. I più comuni sono `\,`, che inserisce un piccolo spazio e `\;` che inserisce un grande spazio. Altri comandi di spaziatura possono essere rintracciati nella documentazione allegata. [Gia]

2.3.5 Sommatorie e produttorie

I comandi corrispondenti ai simboli di sommatoria e produttoria sono `\sum` e `\prod`, e gli estremi della somma (o del prodotto) si inseriscono come pedici ed apici. Ad esempio, il codice:

```
\begin{gather*}
f(x)=\sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!} (x-x_0)^n \\
N! \not= \prod_{k=1}^N k^2
\end{gather*}
```

produce le due equazioni

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x_o)}{n!} (x - x_o)^n$$

$$N! \neq \prod_{k=1}^N k^2$$

2.3.6 Integrali

Il comando per stampare un integrale singolo è `\int`; gli eventuali estremi di integrazione si inseriscono come pedice ed apice di tale comando. Così, il codice:

```
\begin{equation*}
\int_a^b f(x) \, dx
\end{equation*}
```

stampa l'integrale

$$\int_a^b f(x) \, dx$$

Per stampare integrali doppi e tripli, si usano i comandi `\iint` ed `\iiint`, rispettivamente, mentre per ottenere il simbolo di integrale chiuso basta mettere una "o" all'inizio del comando. A titolo esemplificativo, scriviamo un frammento di codice che stampi l'espressione del teorema di Gauss per il campo elettrico:

```
\begin{equation}
\oint_{\Sigma} \mathbf{E} \cdot \hat{\mathbf{n}} \, dS = \iiint_V (\nabla \cdot \mathbf{E}) \, dV = \frac{Q_{int}}{\varepsilon_0}
\end{equation}
```

Tale codice restituisce l'equazione:

$$\iint_{\Sigma} \mathbf{E} \cdot \hat{\mathbf{n}} \, dS = \iiint_V (\nabla \cdot \mathbf{E}) \, dV = \frac{Q_{int}}{\varepsilon_0} \quad (2.12)$$

Ci sono diversi pacchetti che contengono definizioni degli integrali multipli chiusi; quello usato qui è `esint`.

2.3.7 Matrici e sistemi

Per scrivere le matrici, si usano ambienti specifici, che vanno inseriti all'interno dell'*equation*. Riassumiamo tali ambienti nella tabella seguente.

AMBIENTE	TIPO PARENTESI
<code>pmatrix</code>	parentesi tonde
<code>bmatrix</code>	parentesi quadre
<code>Bmatrix</code>	parentesi graffe
<code>vmatrix</code>	barre verticali
<code>Vmatrix</code>	doppie barre verticali

All'interno dell'ambiente, le entrate della matrice vanno inserite separando le colonne con "&" e le righe con "\\". Ad esempio, per costruire la matrice:

$$M = \begin{pmatrix} 1 & 3\pi & 7 \\ 5a & 4 & 37 \\ 42 & 115 & 0 \end{pmatrix}$$

occorre scrivere:

```
\begin{equation*}
M=
\begin{pmatrix}
1 & 3\pi & 7 \\
5a & 4 & 37 \\
42 & 115 & 0
\end{pmatrix}
\end{equation*}
```

Similmente, i sistemi di equazioni si costruiscono tramite l'ambiente `cases`, la cui sintassi è sostanzialmente identica a quella appena vista per le matrici. Così, il codice:

```
\begin{equation}
f(x)=
\begin{cases}
1 & x \in \mathbb{Q} \cap [0,1] \\
0 & x \in [0,1] \setminus \mathbb{Q}
\end{cases}
\end{equation}
```

consente di stampare la nota funzione di Dirichlet:

$$f(x) = \begin{cases} 1 & x \in \mathbb{Q} \cap [0,1] \\ 0 & x \in [0,1] \setminus \mathbb{Q} \end{cases} \quad (2.13)$$

2.4 Altri strumenti utili

2.4.1 Il pacchetto `physics`

Molto spesso in fisica si utilizzano notazioni abbastanza particolari, o comunque si seguono convenzioni tipografiche che si incontrano raramente fuori dall'ambiente fisico e che, per questo motivo, non sono ben implementate nel solo pacchetto `amsmath`. Per ovviare a queste mancanze, introduciamo il pacchetto `physics`. **Attenzione:** affinchè tale pacchetto funzioni, è necessario caricare anche i pacchetti `xparse` e `amsmath`.

Una lista completa dei comandi contenuti nel pacchetto è rintracciabile nella documentazione. [Bar] In questo breve paragrafo riassumeremo i comandi che servono per scrivere in notazione vettoriale e per scrivere le derivate.

Derivate

Come precedentemente notato, senza pacchetti specifici scrivere derivate in L^AT_EX sarebbe una bella faticaccia, in quanto dovrebbero essere scritte come frazioni. Il pacchetto physics semplifica enormemente la loro scrittura. Riassumiamo in tabella i comandi necessari a scrivere derivate. Tali differenziali sono utili per scrivere integrali ed oggetti simili.

CODICE	OUTPUT
\dd	d
\dd{x}	dx
\dd[n]{x}	$d^n x$

La sintassi dei comandi di derivata è simile: per la derivata ordinaria, il comando è `\dv[]{}{}` mentre per la derivata parziale il comando è `\pdv[2]{f}{x_i}{x_j} = \pdv[2]{f}{x_j}{x_i}`, dove l'argomento opzionale è l'ordine della derivata, se non specificato settato a 1, e gli altri due argomenti sono la funzione e la variabile rispetto a cui si deriva. Nel caso del comando `\pdv`, si possono aggiungere quante graffe si vuole, corrispondenti a tutte le variabili rispetto alle quali si deriva. Se il primo argomento tra graffe non è inserito, il codice stampa soltanto l'operatore di derivata. Così, il teorema di Schwartz viene scritto come

```
\begin{equation}
\pdv[2]{f}{x_i}{x_j} = \pdv[2]{f}{x_j}{x_i}
\end{equation}
```

che produce

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i} \quad (2.14)$$

Notazione Vettoriale

Ci sono diverse notazioni per scrivere un vettore: c'è chi preferisce indicarlo con una freccina sopra, chi semplicemente in grassetto, chi a queste due scelte aggiunge la richiesta che il vettore sia in corsivo. I comandi che corrispondono a queste quattro scelte possono essere stampate tramite il codice

```
\begin{equation*}
\text{\textbf{v}} \quad \text{\textit{v}} \quad \text{\textit{\textbf{v}}} \quad \text{\textit{\textit{v}}}
\end{equation*}
```

che stampa

$$\vec{v} \quad \vec{v} \quad \mathbf{v} \quad \mathbf{v}$$

Su questi vettori poi, come sappiamo, si possono eseguire le operazioni canoniche di prodotto interno ed esterno. I caratteri che vi corrispondono possono essere prodotti col codice

```
\begin{gather*}
\text{\textit{\textbf{v}}}\cdot\text{\textit{\textbf{v}}} \\
\text{\textit{\textbf{v}}}\times\text{\textit{\textbf{v}}}
\end{gather*}
```

che restituisce

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} \\ \mathbf{c} \times \mathbf{d} \end{aligned}$$

Infine, introduciamo il nostro migliore amico, ovvero il *Nabla* ∇ . Con esso si possono scrivere gli operatori di gradiente, divergenza, rotore e laplaciano. Prendiamo a prestito le equazioni di Maxwell e l'equazione di Schrödinger per dare un esempio di come scriverli. Tramite il codice

```
\begin{aligned}
\operatorname{div}\{\mathbf{E}\} &= \frac{\partial \mathbf{E}}{\partial t} \quad \& \\
\operatorname{curl}\{\mathbf{E}\} &= - \operatorname{pdv}\{\mathbf{B}\}\{t\} \quad \backslash \\
\operatorname{div}\{\mathbf{B}\} &= 0 \quad \& \\
\operatorname{curl}\{\mathbf{B}\} &= \mu_0 \mathbf{j} + \mu_0 \operatorname{varepsilon}_0 \operatorname{pdv}\{\mathbf{E}\}\{t\}
\end{aligned}
```

si scrivono le equazioni

$$\begin{aligned} \nabla \cdot \mathbf{E} &= \frac{\rho}{\epsilon_0} & \nabla \times \mathbf{E} &= - \frac{\partial \mathbf{B}}{\partial t} \\
\nabla \cdot \mathbf{B} &= 0 & \nabla \times \mathbf{B} &= \mu_0 \mathbf{j} + \mu_0 \epsilon_0 \frac{\partial \mathbf{E}}{\partial t}
\end{aligned}$$

Tramite invece il frammento di codice

```
\begin{aligned}
\operatorname{quad} \operatorname{ref}\{i\} \hbar \operatorname{pdv}\{t\} + \frac{\hbar^2}{2m} \operatorname{laplacian} \\
- \hat{U}(\mathbf{r}) \Psi(\mathbf{r}, t) = 0 \\
\operatorname{quad} \text{dove}; \operatorname{laplacian} = \operatorname{div}\{\operatorname{grad}\}
\end{aligned}
```

si scrive la versione tridimensionale dell'equazione di Schrödinger per sistemi scleronomi

$$\left[i\hbar \frac{\partial}{\partial t} + \frac{\hbar^2}{2m} \nabla^2 - \hat{U}(\mathbf{r}) \right] \Psi(\mathbf{r}, t) = 0 \quad \text{dove } \nabla^2 = \nabla \cdot \nabla$$

Notazione di Dirac

In meccanica quantistica si fa ampio uso della notazione di *Dirac*, o notazione *braket*. Esistono diversi pacchetti che implementano questa notazione in L^AT_EX. Uno di questi è *physics*. Nella tabella che segue sono raccolti i principali comandi del pacchetto.

CODICE	OUTPUT
<code>\ket{\psi}</code>	$ \psi\rangle$
<code>\bra{\psi}</code>	$\langle\psi $
<code>\braket{\varphi}{\psi}</code>	$\langle\varphi \psi\rangle$
<code>\dyad{a}{b}</code>	$ a\rangle\langle b $
<code>\expval{O}</code>	$\langle O \rangle$
<code>\ev{O}{\Psi}</code>	$\langle\Psi O \Psi\rangle$
<code>\mel{n}{O}{m}</code>	$\langle n O m\rangle$

2.4.2 Labels e riferimenti incrociati

Il pacchetto `amsmath` comprende il comando `\eqref{}`, che consente di citare le equazioni presenti all'interno del testo, a patto che a ciascuna di esse sia associata, tramite il comando `\label{}`, un'etichetta univoca. Ad esempio, il codice:

```
La pressione all'interno di un fluido dipende dalla profondità $h$ secondo la legge:  

\begin{equation}  

    p(h)=p_o+\rho gh  

    \label{Stevin}  

\end{equation}  

L'equazione \eqref{Stevin} è nota come Legge di Stevino.
```

restituisce il seguente testo:

La pressione all'interno di un fluido dipende dalla profondità h secondo la legge:

$$p(h) = p_o + \rho gh \quad (2.15)$$

L'equazione (2.15) è nota come Legge di Stevino.

2.4.3 Equazioni "in scatola"

Per mettere in evidenza una particolare equazione (ad esempio, il risultato di un lungo calcolo), si può usare il comando `\boxed{}`. Ad esempio, il codice:

```
\begin{equation}  

    \boxed{  

        p+\frac{1}{2}\rho v^2 + \rho gh = \text{costante}}  

    }  

\end{equation}
```

Corrisponde a:

$$\boxed{p + \frac{1}{2}\rho v^2 + \rho gh = \text{costante}} \quad (2.16)$$

2.4.4 Equazioni con unità di misura

Per inserire, ad esempio, risultati numerici corredati di unità di misura, è possibile utilizzare i comandi `\si` e `\SI`, caricati dal pacchetto `siunitx`.

Il comando `\SI{8.314}{\joule\per\kelvin\per\mole}` prende in input il valore numerico e le unità di misura. Ad esempio, per stampare:

$$R \simeq 8.314 \text{ J K}^{-1} \text{ mol}^{-1}$$

occorre scrivere:

```
\begin{equation*}  

    R \simeq \SI{8.314}{\joule\per\kelvin\per\mole}  

\end{equation*}
```

Un'altra caratteristica di \SI che lo rende particolarmente utile è la possibilità di scrivere facilmente i valori numerici in notazione scientifica. Supponiamo ad esempio di voler stampare il valore della costante di Planck:

$$h \simeq 6.626 \times 10^{-34} \text{ J s}$$

Allora, basta scrivere:

```
\begin{equation*}
h \simeq \SI{6.626e-34}{\joule\per\second}
\end{equation*}
```

Il comando \si, dalla sintassi più semplice, serve a stampare soltanto unità di misura, ed eventuali valori numerici devono essere scritti davanti ad esso. Tale comando risulta utile se si desidera scrivere l'unità di misura associata ad una certa grandezza fisica, senza dare un particolare valore numerico. Ad esempio, per stampare la frase:

La capacità termica di un corpo si misura in J K^{-1}

è sufficiente scrivere:

La capacità termica di un corpo si misura in $\text{\si{\joule\per\kelvin}}$

Un altro comando caricato da siunitx è \ang, che consente di stampare valori espressi in gradi sessualiimali, come in:

La latitudine di Palermo è $\text{\ang{38;6;43}N}$

che restituisce:

La latitudine di Palermo è $38^{\circ}6'43''N$

L'elenco di tutte le unità di misura supportate è rintracciabile nella documentazione del pacchetto. [Wri]

2.4.5 Teoremi ed oggetti simili

Per inserire nel testo teoremi, esempi o definizioni, opportunamente numerati, è possibile utilizzare ambienti dedicati, che vanno dichiarati all'interno del preambolo tramite il comando \newtheorem{}{}; tale comando prende in input un alias del tipo di ambiente da creare, ed il nome completo che verrà stampato nel testo. Ad esempio, scrivendo nel preambolo:

```
\newtheorem{teo}{Teorema}
\newtheorem{ex}{Esempio}
\newtheorem{df}{Definizione}
```

e nel testo:

```
\begin{teo}
Questo è un Teorema
\end{teo}
\begin{ex}
Questo è un Esempio
\end{ex}
\begin{df}
Questa è una Definizione
\end{df}
```

si ottiene:

Teorema 1. *Questo è un Teorema*

Esempio 1. *Questo è un Esempio*

Definizione 1. *Questa è una Definizione*

Per le dimostrazioni, è possibile utilizzare l'ambiente `proof`, predefinito nel pacchetto `amsmath`. Il risultato del comando

```
\begin{proof}  
    Questa è una dimostrazione  
\end{proof}
```

è il seguente

Dimostrazione. Questa è una dimostrazione □

Inserendo un asterisco, è possibile definire un teorema o un ambiente simile che non segue la numerazione. Questo serve ad esempio quando si deve evidenziare il nome di un teorema. Ad esempio, dichiarando nel preambolo

```
\newtheorem*{jordan}{Lemma di Jordan}
```

e inserendo nel testo

```
\begin{jordan}  
    Questo è il Lemma di Jordan.  
\end{jordan}
```

si ottiene

Lemma di Jordan. *Questo è il Lemma di Jordan.*

Anche in questo contesto, è utile conoscere come fare riferimenti incrociati a teoremi, proposizioni etc. Inserendo nel teorema 1 una label con `\label{teo1}`, è possibile farvi riferimento con il comando `\autoref{}`. Senza altro specificare, però, L^AT_EX farebbe riferimento al teorema solo con il suo numero, stampando solo "1". Per evitare questo, si deve avere l'accortezza di specificare nel preambolo come il comando `\autoref{}` debba riferirvisi. Tale specifica viene fatta tramite il comando `\def\<type>\autorefname{}`, dove si deve rimpiazzare `<type>` con il nome dell'ambiente di cui si vuole settare il nome e inserire tale nome tra parentesi. Così il codice

```
\def\teoautorefname{Teorema}
```

fa sì che il riferimento prodotto sia stampato come "Teorema 1" piuttosto che come "1".

2.4.6 Comandi personalizzati

Nella stesura di relazioni o lavori scientifici, può capitare di dover scrivere espressioni matematiche particolarmente lunghe o complesse; in questa situazione, il codice sorgente diventa spesso poco leggibile e, di conseguenza, diventa difficile trovare e correggere eventuali errori di battitura.

Un modo comodo di ovviare a questo problema è creare scorciatoie tramite la funzione `\newcommand{} [] {}`, che prende in input il nome del nuovo comando, il numero di argomenti (se ce ne sono), e l'espressione che il nuovo comando deve restituire. Come specifica aggiuntiva, mettendo un'altra coppia di parentesi quadre [] si può specificare il valore che il primo argomento #1 deve avere se non inserito in input dall'utente quando il comando viene richiamato nel codice. In tal caso, il primo argomento dovrà essere passato tra parentesi quadre. Facciamo qualche esempio.

Parentesi adattive

Se in un'espressione matematica compaiono più parentesi annidate occorre, per evitare risultati poco eleganti, utilizzare le parentesi adattive. Un trucco utile per evitare di scrivere molte volte `\left` e `\right` è dichiarare nel preambolo i comandi:

```
\newcommand{\graffe}[1]{\left\{#1\right\}}
\newcommand{\quadre}[1]{\left[#1\right]}
\newcommand{\tonde}[1]{\left(\#1\right)}
```

che consentono di scrivere un'espressione del tipo:

$$\left\{ 3x \left[5 \left(x - \frac{1}{2} \right)^2 \left(x + \frac{1}{2} \right)^2 \right] - 2(x^3 + 5) \right\}$$

tramite il seguente codice, abbastanza semplice e leggibile:

```
\begin{equation*}
\graffe{3x\quadre{5\tonde{x-\frac{1}{2}}^2
\tonde{x+\frac{1}{2}}^2}-2\tonde{x^3+5}}
\end{equation*}
```

Derivate

Come abbiamo visto, non usando il pacchetto `physics` le derivate si scrivono come frazioni, tramite il comando `\frac{}{}`. Tuttavia, questo metodo risulta pesante quando ci si trova a scrivere espressioni contenenti più derivate, come ad esempio l'equazione delle onde in una dimensione:

$$\frac{\partial^2 u}{\partial x^2} - \frac{1}{v^2} \frac{\partial^2 u}{\partial t^2} = 0 \tag{2.17}$$

Il codice "standard" per scrivere tale equazione è:

```
\begin{equation}
\frac{\partial^2 u}{\partial x^2}-
\frac{1}{v^2}\frac{\partial^2 u}{\partial t^2}=0
\end{equation}
```

Tuttavia, dichiarando nel preambolo il comando:

```
\newcommand{\derparquad}[2]{\frac{\partial^{#1}}{\partial^{#2}}}
```

tal codice diventa semplicemente:

```
\begin{equation}
\derparquad{u}{x} - \frac{1}{v^2} \derparquad{u}{t} = 0
\end{equation}
```

Se però si volesse definire un comando che in generale definisce la derivata di ogni ordine di una funzione di una variabile, si potrebbe inserire nel preambolo

```
\newcommand{\der}[3]{[\phantom{.}]\frac{d^{#1}}{d^{#2}}}
```

in maniera da poter stampare le derivate $\frac{df}{dx}$, $\frac{d^2f}{dx^2}$ e $\frac{d^3f}{dx^3}$ semplicemente utilizzando ripetutamente tale comando

$\$ \der{f}{x} \$$, $\$ \der[2]{f}{x} \$$ e $\$ \der[3]{f}{x} \$$

Notiamo che se il primo argomento tra parentesi quadre non è specificato, il comando `\der` assegna tale argomento a ``, che inserisce uno spazio bianco grande quanto il carattere che gli si passa come argomento. Così facendo, il primo argomento di `\der` è quindi settato come opzionale.

Naturalmente, questi sono esempi particolari, dettati da esigenze altrettanto particolari; è tuttavia utile sapere che L^AT_EX consente di semplificare notevolmente la scrittura di espressioni matematiche complicate, tramite la creazione di opportuni comandi "ad hoc".

2.4.7 Cancellazioni nelle equazioni

Tramite il pacchetto `cancel` è possibile effettuare le classiche cancellazioni che si usano fare quando si effettuano dei calcoli. La documentazione del pacchetto è consultabile in bibliografia. [Ars] I principali comandi del pacchetto sono esemplificati dal seguente codice. Scrivendo

```
\begin{gather*}
\cancel{x^2-x^2} \\
\cancelto{0}{x^2-x^2} \\
\bcancel{x^2-x^2}
\end{gather*}
```

si produce il seguente output

che cancella in diversi modi la stessa espressione.

2.4.8 Font di testo in ambiente matematico

Molto spesso è necessario inserire delle lettere in ambiente matematico, ad esempio per indicare gli insiemi numerici e non, le trasformate integrali etc. La semplice scrittura delle lettere comprese tra \$\$ produce come risultato $ABCDEFHIJKLMNOPQRSTUVWXYZ$. È possibile cambiare questo font utilizzando, all'interno dell'ambiente matematico i comandi `\mathbf{}`, `\mathcal{}` e `\mathbb{}`.

Per fare un esempio, il seguente codice

```
\begin{gather*}
    \mathbf{ABCDEFGHIJKLMNOPQRSTUVWXYZ} \\
    \mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ} \\
    \mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\end{gather*}
```

produce in output

$$\begin{aligned}
& ABCDEFHIJKLMNOPQRSTUVWXYZ \\
& \mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ} \\
& \mathbb{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\end{aligned}$$

È possibile poi introdurre il comando `\mathscr{}` aggiungendo nel preambolo alcuni pacchetti che ne definiscono il risultato. I pacchetti in questione sono listati in bibliografia. [Lee] Ad esempio, caricando il pacchetto `mathrsfs`, il risultato del codice

```
\begin{equation*}
    \mathscr{ABCDEFGHIJKLMNOPQRSTUVWXYZ}
\end{equation*}
```

è

$$\mathcal{ABCDEFGHIJKLMNOPQRSTUVWXYZ}$$

Questa è una caratteristica del tutto generale dei comandi che permettono di cambiare il font in ambiente matematico, ovvero esistono moltissimi pacchetti che cambiano il risultato che l'utilizzo di questi comandi produce.

TELE LATEX

Episodio 3:

Tabelle, figure e grafica

A cura di:

Francesco Schiavone
f.schiavone15@studenti.uniba.it

13 dicembre 2020

Sommario

L'argomento della terza e ultima lezione del corso è uno degli aspetti più importanti (e, secondo il parere di chi scrive, divertenti) della produzione di testi scientifici, vale a dire la presentazione di dati tramite tabelle e figure. Entrambe queste categorie sono generalmente trattate da LATEX come oggetti mobili o fluttuanti detti *float*, ed è quindi opportuno affrontarle insieme.

Cominceremo con una breve introduzione sugli oggetti *float*. Vedremo quindi nel dettaglio come produrre tabelle e come inserire e manipolare figure nel documento, e come riferirci a questi oggetti all'interno del testo. Un'ultima parte della lezione sarà dedicata a qualche cenno sulla vera e propria grafica di LATEX, ovvero la possibilità di produrre figure di alta qualità direttamente tramite qualche linea di codice.

3.1 Perché *float*?

Gli oggetti *float*, letteralmente “galleggianti”, sono in L^AT_EX quegli oggetti che non appaiono nel documento in una posizione corrispondente a quella in cui è stato inserito il codice. Generalmente sono caratterizzati da una didascalia che ne espone il contenuto e da un numero progressivo, che può essere utilizzato per richiamarli all’interno del testo (per esempio “Tabella 2” o “Figura 5”). A primo sguardo potrebbero sembrare piuttosto scomodi: cosa guadagniamo da tutta la faticaccia di scrivere un documento in codice, se poi le cose non stanno nemmeno al loro posto ma vanno a mettersi dove pare a loro? e soprattutto, se non noi, *chi* decide dove vanno?¹⁴

Il vantaggio sta nel fatto che, semplicemente, inserire una figura o una tabella nell’esatto punto del testo in cui essa viene citata può non essere possibile, per questioni di spazio, o può dar luogo a un risultato piuttosto disordinato. Pur essendo in linea di massima possibile seguire un approccio di questo tipo (ma tutto è possibile, se si ha voglia di passare un sufficiente numero di ore davanti a un computer), il metodo standard di L^AT_EX (e di gran parte della tipografia in generale) consiste nello “slegare” questi oggetti dal testo e posizionarli, secondo criteri stabiliti dal programma, nel modo più conveniente possibile. Questo fa anche sì che, nel momento in cui aggiungiamo, eliminiamo o riordiniamo parti di testo, il documento si “ristrutturi” automaticamente, risparmiandoci il più delle volte un tedioso riposizionamento manuale di figure e tabelle.

Non è un segreto che a volte il comportamento di questi oggetti possa risultare a dir poco capriccioso, ed è necessaria una buona dose di sano *trial and error* per ottenere il risultato desiderato. Tuttavia, sono e resto dell’idea che ne valga sempre la pena.

3.2 Tabelle

Le tabelle sono fra gli strumenti preferiti di un fisico quando si tratta di rappresentare e confrontare sinteticamente elenchi di dati. È quindi opportuno saperle utilizzare correttamente.

3.2.1 Pacchetti essenziali

Come breve preludio, segnalo qualche pacchetto a mio parere molto utile per produrre delle tabelle dall’aspetto professionale in L^AT_EX. Per ciascuno di essi consiglio di consultare la documentazione [CTA]. Inseriamo quindi nel preambolo del nostro documento:

- `\usepackage{booktabs}`, contenente comandi utili per la gestione delle tabelle;
- `\usepackage{multirow}`, che consente di realizzare tabelle con celle che coprono più righe;
- `\usepackage[font=small,labelfont=bf]{caption}`, che consente di personalizzare l’aspetto delle didascalie (*caption*, appunto) di tabelle e figure. Con le opzioni consigliate, il testo della didascalia sarà piccolo e la dicitura che specifica il numero della tabella o della figura sarà in **grassetto**;
- `\usepackage{wrapfig}`, che consente di avere tabelle e figure circondate dal testo.

¹⁴Molto spesso mi sono posto queste domande.

3.2.2 L'ambiente tabular

Per realizzare una tabella in L^AT_EX si usa un ambiente **tabular**, la cui sintassi è la seguente:

```
\begin{tabular}{ccr}
... & ... & ...
... & ... & ...
... & ... & ...
... & ... & ...
... & ... & ...
\end{tabular}
```

Quando si apre un ambiente **tabular**, è necessario specificare quante colonne conterrà la nostra tabella e come sarà allineato il testo all'interno di esse. Gli specificatori per l'allineamento sono *l*, *c*, *r* (testo allineato a sinistra, al centro e a destra, rispettivamente) e *p{...}* (testo sotto forma di paragrafo). Quest'ultimo specificatore ha un argomento obbligatorio dove va inserita la larghezza della colonna (ad esempio *p{0.5\textwidth}*). La tabella dell'esempio quindi conterrà tre colonne, con il testo allineato rispettivamente al centro nelle prime due e a destra nella terza. Non è invece necessario specificare il numero di righe.

I comandi *\toprule*, *\bottomrule* e *\midrule*, contenuti nel pacchetto *booktabs*, sono opzionali ma altamente consigliati (anche se talvolta possono causare un prolungamento dei tempi di compilazione su Overleaf)¹⁵. Essi producono linee orizzontali che segnalano l'inizio e la fine della tabella e che ne separano l'intestazione dal contenuto. Un comando simile ma decisamente più "spartano" è dato da *\hline*.

Le altre righe del codice esempio rappresentano il "cuore" della tabella, che è costruita riga per riga: i caratteri *&* separano le varie celle su una riga, mentre con *\backslash* si passa alla riga successiva. Il numero di celle su ciascuna riga dovrà essere esattamente pari al numero di colonne specificato all'inizio dell'ambiente; è però possibile lasciare alcune celle vuote, semplicemente non scrivendo nulla al posto dei puntini di sospensione.

Se si prevede di avere molte colonne con lo stesso allineamento, si può usare una sintassi abbreviata del tipo:

```
\begin{tabular}{*{3}{c}}
...

```

all'inizio dell'ambiente, lasciando invariato il resto del codice. Il primo argomento di **{...}{...}* rappresenta il numero di colonne, il secondo l'allineamento del testo: l'esempio produrrà quindi una tabella con tre colonne con testo centrale.

Una breve nota estetica è doverosa: di *default*, L^AT_EX non prevede linee verticali che separino le colonne di una tabella, ma solo linee orizzontali che ne separino le righe: le tabelle così composte risultano generalmente molto meno "pesanti" da leggere. Ad ogni modo, se proprio si volessero inserire linee verticali, basterebbe adoperare una sintassi come la seguente:

```
\begin{tabular}{l|cc}
...

```

che, ad esempio, produce una tabella di tre colonne (una allineata a sinistra, due al centro), dove la prima è separata dalle altre da una linea verticale.

¹⁵Nota di Giovanni Cusimano

3.2.3 L'ambiente table

Chi avrà provato a compilare qualcuno degli esempi precedenti si sarà reso conto che la tabella viene prodotta esattamente in corrispondenza del codice, proprio come *non* avevamo detto nella sezione 3.1. In effetti, per ottenere un ambiente *float* vero e proprio, è necessario “rinchiudere” l’ambiente *tabular* in un altro ambiente, ovvero *table*. La sintassi per farlo è la seguente:

```
\begin{table}
  \centering
  \begin{tabular}{...}
    ...
  \end{tabular}
  \caption{...}
  \label{...}
\end{table}
```

dove

- `\centering` allinea la tabella al centro;
- l’ambiente *tabular* si utilizza esattamente come abbiamo appena visto;
- l’argomento di `\caption {}` è la didascalia della tabella;
- l’argomento di `\label {}` è una “etichetta”, appunto, che ci consentirà di richiamare la tabella all’interno del testo.

Come già menzionato, il posizionamento di un oggetto *float* all’interno del documento sarà deciso da L^AT_EX in base a vari criteri di convenienza tipografica. Generalmente i *float* verranno collocati in alto nella pagina: volendo si può modificare questa opzione nel modo seguente:

```
\begin{table}[...]
  ...

```

dove l’argomento opzionale fra parentesi quadre può essere t (la tabella viene collocata preferibilmente in alto nella pagina), b (preferibilmente in basso), h (preferibilmente in corrispondenza del codice) o p (preferibilmente in una pagina a parte, contenente solo oggetti *float*). Personalmente consiglio di evitare queste opzioni a meno che non sia strettamente necessario ottenere un risultato particolare.

A questo punto possiamo vedere un primo esempio completo di tabella, che dovrebbe rendere esplicito quanto detto finora:

```
\begin{table}[t]
  \centering
  \begin{tabular}{lcp{0.4\textwidth}}
    \hline
    \textbf{Fenomeno} & \textbf{Forza} & \textbf{Causa} \\
    \hline
    Buco nero & Gravità & Collazzo gravitazionale \\
    Induzione magnetica & Elettromagnetismo & Variazione di flusso magnetico \\
    Decadimento $\beta$ & Interazione debole & Conversione di quark $d$ in $u$ \\
  \end{tabular}

```

Fenomeno	Forza	Causa
Buco nero	Gravità	Collasso gravitazionale
Induzione magnetica	Elettromagnetismo	Variazione di flusso magnetico
Decadimento β	Interazione debole	Conversione di quark d in u
Adronizzazione	Interazione forte	Legami fra quark

Tabella 3.1: Esempi di fenomeni dovuti alle interazioni fondamentali.

```

Adronizzazione & Interazione forte & Legami fra quark \\
\hline
\end{tabular}
\caption{Esempi di fenomeni dovuti alle interazioni fondamentali.}
\label{tab:interazioni}
\end{table}

```

Questo codice genera la tabella 3.1; notiamo che la terza colonna è più larga del testo che contiene, perché ne abbiamo esplicitato la larghezza con `p[0.4\textwidth]`. Per richiamare la tabella all'interno del testo utilizziamo il comando `\ref {tab:interazioni}`, che ha come argomento la `label` che abbiamo assegnato alla tabella e al posto del quale comparirà il numero della tabella in questione. Se il pacchetto `hyperref` è incluso nel documento, sarà anche possibile cliccare sul numero per essere portati direttamente alla pagina in cui è contenuta la tabella.

3.2.4 `multicolumn` e `multirow`

Ora che abbiamo visto in sufficiente dettaglio come è strutturata una tabella in L^AT_EX, possiamo presentare due piccoli esempi di tabelle che utilizzano i comandi `multicolumn` e `multirow` (contenuto nell'omonimo pacchetto) per produrre celle che si estendono per più colonne o righe adiacenti. Per restare in tema di particelle, vediamo due tabelle che classificano le famiglie di quark e leptoni. Utilizzando `multicolumn`:

```

\begin{tabular}{cccc}
\hline
\multicolumn{2}{c}{Quark} & \multicolumn{2}{c}{Leptoni} \\
\hline
$u$ & $d$ & $e$ & $\nu_e$ \\
$c$ & $s$ & $\mu$ & $\nu_\mu$ \\
$t$ & $b$ & $\tau$ & $\nu_\tau$ \\
\hline
\end{tabular}

```

Questo codice genera la tabella 3.2 (abbiamo ovviamente omesso l'ambiente `table` esterno). Il comando `\multicolumn {}{}{}` ha tre argomenti che specificano, rispettivamente, il numero di colonne su cui si estende la cella, l'allineamento del testo nella cella e il testo medesimo.

Possiamo organizzare la stessa tabella su righe, piuttosto che su colonne, utilizzando `multirow`:

```

\begin{tabular}{cccc}
\hline
\multirow{2}[2]{*}{Quark} & $u$ & $c$ & $t$ \\
\hline

```

Quark	Leptoni		
u	d	e	ν_e
c	s	μ	ν_μ
t	b	τ	ν_τ

Tabella 3.2: Quark e leptoni con `multicolumn`.

Quark	u	c	t
d	s	b	
Leptoni	e	μ	τ
	ν_e	ν_μ	ν_τ

Tabella 3.3: Quark e leptoni con `multirow`.

```

& $d\$      & $s\$      & $b\$      \\
\hline
\multirow[c]{2}{*}{Leptoni} & $e\$      & $\mu\$      & $\tau\$      \\
& $\nu_e\$ & $\nu_\mu\$ & $\nu_\tau\$ \\
\hline
\end{tabular}

```

Questo codice genera la tabella 3.3. La sintassi del comando `\multirow` [] {}{} prevede tre argomenti obbligatori, fra parentesi graffe, che specificano rispettivamente il numero di righe su cui si estende la cella, la larghezza della cella (che può essere resa automatica con *) e il testo contenuto in essa. L'argomento opzionale fra parentesi quadre può essere t, c o b e specifica l'allineamento verticale del testo (in alto, centrato o in basso). Ci sono altri argomenti opzionali, usati meno di frequente, descritti nella documentazione del pacchetto `multirow`. Notiamo infine che se si utilizza `multirow` per “unire” un certo numero di celle sulla stessa colonna, il comando andrà inserito solo nella prima di queste, mentre le altre andranno lasciate vuote.

3.2.5 Tabelle circondate dal testo con `wraptable`

Mostriamo infine la sintassi per realizzare tabelle circondate dal testo, cosa che può essere utile (anche se non troppo raccomandata da un punto di vista stilistico) se si ha a che fare con tabelle di dimensioni molto piccole. Ciò che bisogna fare è semplicemente, utilizzando l'usuale sintassi, sostituire all'ambiente `table` un ambiente `wraptable` (fornito dal pacchetto `wrapfig`). Questo ambiente viene introdotto con la sintassi seguente:

```

\begin{wraptable}[]{}[]{}
...
\end{wraptable}

```

dove il primo argomento, opzionale, indica il numero di righe da accorciare (automatico se non specificato); il secondo argomento, obbligatorio, indica il posizionamento della tabella (r, l, i, o per posizionare la tabella sul margine destro, sinistro, interno o esterno); il terzo argomento, opzionale, indica l'eventuale sporgenza nel margine; il quarto argomento, obbligatorio, indica la larghezza dell'ambiente (automatica se posta a 0pt).

\wedge	0	1
0	0	0
1	0	1

Tabella 3.4: Tabella di verità per l'operatore AND con `wraptable`.

Ad esempio, il codice seguente realizza la tabella 3.4:

```
\begin{wraptable}[r][0pt]{0.5\textwidth}
\centering
\begin{tabular}{c|cc}
\$ \land \$ & 0 & 1 \\
\hline
0 & 0 & 0 \\
1 & 0 & 1 \\
\end{tabular}
\caption{Tabella di verità per l'operatore AND con \texttt{wraptable}}
\label{tab:truth}
\end{wraptable}
```

C'è da dire che il comportamento e il posizionamento di oggetti di questo tipo può spesso essere imprevedibile e poco soddisfacente. Consiglio, nella maggior parte dei casi, di attenersi all'utilizzo dell'ambiente `table`.

3.3 Figure e grafica

Se è vero che un libro non si giudica dalla copertina (e che un articolo scientifico non si giudica da quanto sono belle e colorate le sue figure), è pure vero che anche l'occhio vuole la sua parte. Sapere organizzare bene il materiale visivo in un documento è un aspetto molto importante della sua stesura, ed è sicuramente quello che contribuisce maggiormente a dare un *look* professionale al tutto.

3.3.1 Immagini raster e vettoriali

Con buona pace degli informatici e dei grafici che potrebbero leggere questo testo, cercherò di spiegare in breve una distinzione fondamentale in grafica, ovvero quella fra immagini raster e vettoriali.

Le immagini *raster* (ovvero "griglia", "reticolo") sono costituite da un numero prefissato di pixel, organizzati appunto su una griglia e contenenti ciascuno le informazioni riguardanti colore e luminosità di uno specifico punto dell'immagine. Un'immagine da 500×500 pixel avrà sempre lo stesso numero di "quadratini fondamentali", e se proviamo a ingrandirla o rimpicciolirla saranno i pixel stessi a cambiare dimensione. Questo porta chiaramente a una perdita di qualità dell'immagine, che se ingrandita troppo risulterà, appunto, "pixelata". Praticamente tutte le immagini digitali di uso comune, ad esempio fotografie e scansioni nei formati JPEG e PNG, sono di questo tipo. Sono oggetti molto semplici da manipolare e, come vedremo, basta veramente poco per inserirli anche in L^AT_EX.

Per contro, le immagini *vettoriali* sono meglio definite da una serie di istruzioni matematiche, che generano l'immagine direttamente nel programma che si sta utilizzando. L'immagine è quindi perfettamente adattata al documento, non perde di qualità ingrandendola, occupa poco spazio e può essere facilmente convertita in raster (mentre il processo inverso non è affatto garantito). Lo svantaggio è chiaramente nel fatto che sono oggetti decisamente più complessi e meno immediati delle controparti raster.



(a) Immagine vettoriale.



(b) Immagine raster

Figura 3.1: Logo di AISF Bari in versione vettoriale e raster [grazie a Filippo Conte].

Provare per credere! In figura 3.1 è presente il logo del Comitato Locale AISF Bari, in versione vettoriale e raster. Zoomando sui dettagli del testo, del lampioncino, dell’onda o del cerchio esterno la differenza fra i due formati dovrebbe essere evidente. Notare che l’immagine sullo sfondo è raster in entrambi i casi.

3.3.2 Pacchetti essenziali e cartella immagini

Anche qui, prima di cominciare, è opportuno fare un breve elenco dei pacchetti essenziali per lavorare con le immagini in \LaTeX . I pacchetti `caption` e `wrapfig` introdotti nella sezione 3.2.1 sono validi in generale per tutti gli oggetti *float*, e quindi anche per le figure. Ad essi possiamo aggiungere:

- `\usepackage {graphicx}` integra il supporto per una varietà di formati grafici;
- `\usepackage {subfig}` consente di inserire sottofigure;
- `\usepackage {tikz}` è un potente pacchetto (in verità, quasi un motore grafico a parte) che consente la realizzazione di immagini vettoriali direttamente in \LaTeX ;
- `\usepackage {pgfplots}` è basato su TikZ e permette di realizzare plot in formato vettoriale;
- `\usepackage {circuitikz}`, anch’esso basato su TikZ, è un pacchetto specifico per i diagrammi circuituali.

Per assicurarsi di lavorare in maniera ordinata, è inoltre opportuno raccogliere tutte le figure che si utilizzeranno in una o più sottocartelle della propria cartella di lavoro. Per comunicare a \LaTeX la posizione di queste cartelle, inseriremo quindi nel preambolo il comando

```
\graphicspath{{figure1/}, {figure2/}, ...}
```

dove `figure1`, `figure2` etc. saranno i nomi delle sottocartelle in cui avremo inserito le immagini. In questo modo basterà specificare il nome del file, e non necessariamente tutto il suo percorso, per far sì che L^AT_EX lo cerchi automaticamente nella cartella di lavoro e nelle sottocartelle specificate con `graphicspath`.

3.3.3 Il comando `includegraphics` e l'ambiente `figure`

Se abbiamo immagini contenute in un file, che siano raster (.jpg, .png, .bmp...) o vettoriali (.eps, .pdf...), è particolarmente semplice includerle nel nostro documento L^AT_EX. Basterà infatti una singola linea di codice, costituita dal comando

```
\includegraphics[]{}
```

Il secondo argomento, obbligatorio, è costituito dal percorso dell'immagine che vogliamo inserire nel documento. Se però l'immagine si trova nella stessa cartella del file `.tex` su cui stiamo lavorando, oppure se avremo la premura di specificare tutte le sottocartelle in cui si trovano le nostre immagini con `graphicspath`, basterà semplicemente inserire il nome del file e L^AT_EX lo cercherà automaticamente all'interno di queste cartelle. Il primo argomento, opzionale, consente invece di applicare varie modifiche all'immagine:

- `width = ...` e `height = ...` specificano la larghezza e l'altezza dell'immagine. Dopo l'uguaglianza va inserito un numero con un'appropriata unità di misura¹⁶, come `cm` o `\textwidth`;
- `scale = ...` specifica di quanto scalare l'immagine rispetto alle dimensioni originali. Dopo l'uguaglianza va inserito un numero;
- `angle = ...` specifica di quanto ruotare l'immagine. Dopo l'uguaglianza va inserito un numero, che rappresenta l'angolo di rotazione di gradi;
- `trim = {...}` e `clip` tagliano l'immagine. La sintassi prevede che dopo l'uguaglianza siano specificati quattro numeri, fra parentesi graffe e separati da uno spazio, con appropriate unità di misura, che specificano di quanto tagliare la figura dal bordo sinistro, inferiore, destro e superiore (nell'ordine). `clip` applica quindi la modifica. Complessivamente avremo allora un codice del tipo:

```
\includegraphics[trim = {5cm 0 2cm 10cm}, clip]{...}
```

È inoltre possibile utilizzare più opzioni contemporaneamente, semplicemente separandole con una virgola.

Così come nel caso di `tabular`, il comando `includegraphics` piazza un'immagine esattamente nella posizione del testo in cui abbiamo scritto il codice. Per trasformarla in un oggetto `float` avremo bisogno di introdurre un altro ambiente, l'ambiente `figure`, che è esattamente l'analogico di `table` nel caso delle tabelle. Avremo quindi la possibilità di specificare una preferenza per il posizionamento della figura, di inserire una didascalia `caption` e di utilizzare una `label` per i riferimenti ipertestuali.

Includiamo ad esempio nel nostro documento la copertina dell'album *Surfin' USA* dei Beach Boys. Avremo quindi:

¹⁶Per un elenco completo, vedere ad esempio [\[Wikia\]](#).

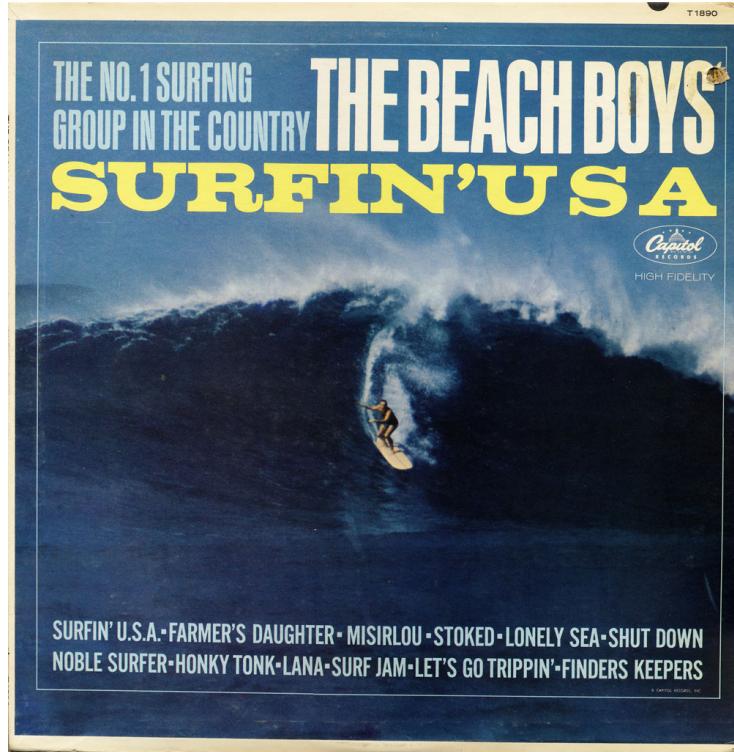


Figura 3.2: Everybody's gone surfin'...

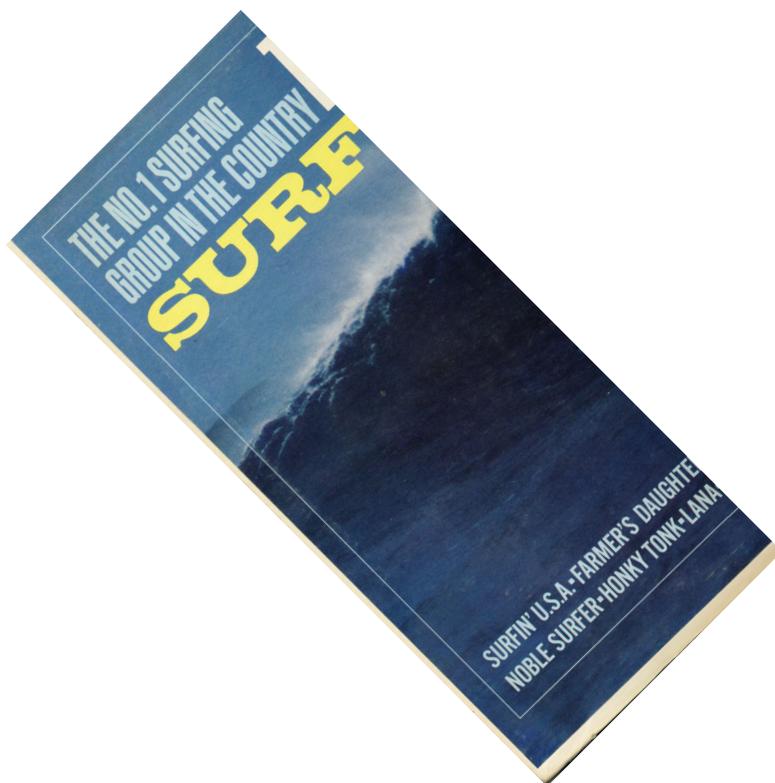


Figura 3.3: ... surfin' USA!

```
\begin{figure}
\centering
\includegraphics[width=0.55\textwidth]{surfin.jpg}
\caption{Everybody's gone surfin'...}
\label{fig:bb1}
\end{figure}
```

che produrrà la figura 3.2. Volendo, possiamo ruotare e ritagliare questa figura semplicemente modificando le opzioni del comando `includegraphics`. Ad esempio, scrivendo

```
\includegraphics[width=0.25\textwidth, angle=45, trim={5cm 0 20cm 1cm}, clip]{surfin.jpg}
```

otterremo la figura 3.3.

3.3.4 Figure circondate dal testo con `wrapfigure`

Il pacchetto `wrapfig` mette a disposizione un ambiente `wrapfigure` che è in tutto e per tutto analogo all'ambiente `wraptable` discusso nella sezione 3.2.5 e per il quale ripetiamo le stesse osservazioni. Per comprenderne la sintassi, possiamo limitarci a un breve esempio che utilizza un gattino. Il risultato è in figura 3.4.

```
\begin{wrapfigure}[r][0pt]{0pt}
\centering
\includegraphics[width=0.3\textwidth]{gattino.jpeg}
\caption{Un gattino inserito con \texttt{wrapfigure}.}
\label{fig:gattino}
\end{wrapfigure}
```



Figura 3.4: Un gattino inserito con `wrapfigure`.

3.3.5 Sottofigure con `subfloat`

Un'altra operazione che spesso si rende necessaria è quella di inserire sottofigure, come per esempio abbiamo fatto in figura 3.1. Per farlo utilizziamo il comando `subfloat` messo a disposizione dal pacchetto `subfig`. Come suggerito dal nome, in linea di principio questo comando potrebbe essere usato per un qualsiasi oggetto *float*, come una tabella: è però raro incontrare documenti che contengano sottotabelle, mentre invece le sottofigure sono piuttosto comuni. Vediamo direttamente un esempio pratico:

```
\begin{figure}
\subfloat[Episodio IV.\label{fig:starwarsIV}]{%
\includegraphics[width=0.3\textwidth]{starwars4.jpg}%
}
\hfill
\subfloat[Episodio V.\label{fig:starwarsV}]{%
\includegraphics[width=0.3\textwidth]{starwars5.jpg}%
}
\hfill
\subfloat[Episodio IV.\label{fig:starwarsVI}]{%
```



TWENTIETH CENTURY FOX PRESENTS A LUCAFILM PRODUCTION STAR WARS
Starring MARK HAMILL HARRISON FORD CARRIE FISHER
PETER CUSHING
ALEC GUINNESS
Written and Directed by GEORGE LUCAS
Produced by GARY KURTZ JOHN WILLIAMS
Music by RANASCIONE PINTS OF DE LUXE TECHNICOLOR
MOSAIC FILM SOUND SYSTEM INC.
Noise Reduction - High Fidelity

(a) Episodio IV.



(b) Episodio V.



(c) Episodio IV.

Figura 3.5: Tanto tempo fa, in una galassia lontana lontana...

```
\includegraphics[width=0.3\textwidth]{starwars6.jpg}
}
\caption{Tanto tempo fa, in una galassia lontana lontana...}
\label{fig:starwars}
\end{figure}
```

Questo codice produce la figura 3.5, suddivisa nelle sottofigure 3.5a, 3.5b e 3.5c. Il comando `\subfloat` [] {} ha quindi due argomenti: il primo, opzionale, contiene la didascalia e la label di ciascuna sottofigura; il secondo, obbligatorio, contiene l'immagine vera e propria. In questo caso l'argomento obbligatorio è un comando `includegraphics`, ma in generale potrebbe essere anche un ambiente `tabular` o il codice di un'immagine generata con TikZ (che sarà oggetto della prossima sezione). Notiamo che i vari comandi `subfloat` sono tutti compresi all'interno di un ambiente `figure`, che permette anche di specificare una didascalia e una label globali. Il comando `\hfill`, infine, agisce come una sorta di "colla" che riempie gli spazi orizzontali fra le sottofigure e assicura che queste siano correttamente spaziate e allineate.

3.3.6 Grafica vettoriale con TikZ

Finora ci siamo occupati esclusivamente di immagini raster, contenute in file .jpg. Come abbiamo visto, però, ci sono molte ragioni per preferire in certi casi una grafica di tipo vettoriale. Queste possono essere sia di natura estetica (come abbiamo già menzionato, le immagini vettoriali non perdono risoluzione) sia di natura pratica (soprattutto nella realizzazione di grafici e diagrammi, in cui è sicuramente più naturale pensare alle immagini in termini di relazioni matematiche e numeriche piuttosto che in termini di pixel). Presentiamo quindi qualche semplice esempio di applicazione del pacchetto TikZ che, così come tutto il resto del corso, *non* ha alcuna pretesa di completezza. Spero comunque che questa breve esposizione riesca a far intuire le po-

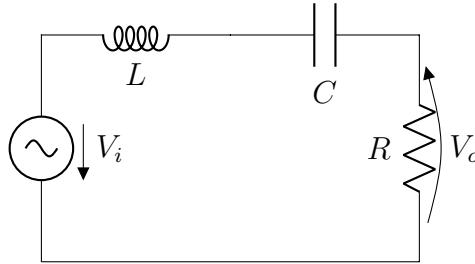


Figura 3.6: Un circuito RLC realizzato con `circuitikz`.

tenzialità di questo strumento e che possa essere un'utile introduzione all'enorme mondo della grafica vettoriale in \LaTeX .

Diagrammi circuituali con `circuitikz`

Un'applicazione particolarmente semplice di TikZ è data dalla realizzazione di diagrammi circuituali tramite il pacchetto `circuitikz`. Per conoscere le funzionalità complete di questo pacchetto invito a consultarne la documentazione [CTA]. Questa può rappresentare una buona introduzione a TikZ in generale, avendo una mole decisamente meno minacciosa rispetto al manuale completo.

Quando si disegna in TikZ, è sempre bene visualizzare l'immagine che si ha in mente su un piano cartesiano. Tutti i punti sono infatti definiti in termini di coordinate cartesiane, la cui unità di misura standard è il centimetro. Per comprendere meglio questo concetto, vediamo un semplicissimo codice che permette di realizzare il diagramma di un classico circuito RLC:

```
\begin{circuitikz}
\draw (0,0) -- (5,0) --
(5,0) to[R=$R$, v=$V_o$] (5,3) --
(5,3) to[C=$C$] (2.5,3) --
(2.5,3) to[L=$L$] (0,3) --
(0,3) to[sinusoidal voltage source=$\varepsilon$, v=$V_i$] (0,0);
\end{circuitikz}
```

Il risultato è in figura 3.6. Ovviamente, il codice è stato inserito in un ambiente `figure` al posto del comando `includegraphics`. Analizziamolo in maggiore dettaglio:

- il codice che produce il diagramma è tutto contenuto in un ambiente `circuitikz`;
- immaginiamo di partire dal punto $(0,0)$ e di percorrere il circuito in senso antiorario. Il comando `\draw` indica al programma di cominciare a disegnare finché non incontra un punto e virgola;
- i caratteri `--` tracciano un segmento semplice fra due punti;
- inserendo `to[...]` fra due coppie di coordinate, invece, si può specificare quali elementi di circuito inserire su quel tratto, eventualmente aggiungendo anche un'etichetta.

Come sempre, il lettore è invitato a zoomare sulla figura risultante e a constatare che con questo metodo non si verifica la minima perdita di risoluzione, anche a ingrandimenti elevati. Inoltre, essendo il testo già formattato secondo lo standard \LaTeX , la figura risulta molto meglio integrata nel documento rispetto ad una prodotta da un programma esterno.

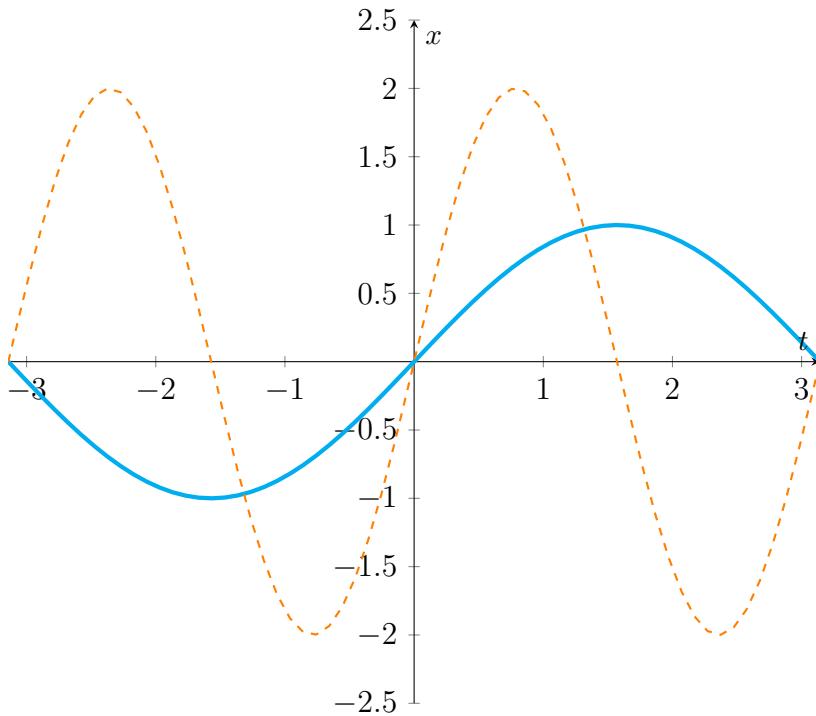


Figura 3.7: Due grafici di funzioni tracciati con pgfplots.

Grafici di funzioni con pgfplots

Un'altra applicazione semplificata di TikZ è nel disegno di grafici di funzioni, ed è fornita dal pacchetto pgfplots. Anche qui ci limitiamo a commentare un semplice esempio dimostrativo:

```
\begin{tikzpicture}
\begin{axis}[axis lines=middle,
xmin=-3.14,xmax=3.14,ymin=-2.5,ymax=2.5,
xlabel={\small $t$},
ylabel={\small $x$},
width=.7\textwidth]
\addplot [samples=100,cyan,ultra thick] {sin(deg(x))};
\addplot [samples=100,orange,dashed,thick] {2*sin(deg(2*x))};
\end{axis}
\end{tikzpicture}
```

Il risultato è in figura 3.7. In questo caso:

- tutto il codice è racchiuso in un ambiente `tikzpicture` e in un sottoambiente `axis`;
- in `axis` si possono specificare varie preferenze: qui ad esempio abbiamo richiesto che gli assi passassero per il centro della figura, abbiamo definito i valori massimi e minimi delle coordinate su ciascun asse, abbiamo dato delle etichette agli assi e abbiamo specificato la larghezza complessiva della figura;
- con il comando `\addplot [] {}` quindi si aggiungono i vari grafici. Nel secondo argomento, obbligatorio, va scritta l'espressione della funzione (notare che gli angoli vanno

convertiti da radianti in gradi tramite l'apposita funzione `deg()`, mentre nel primo argomento si possono dare varie preferenze, come colore e spessore della linea o numero di campionamenti della funzione.

È inoltre opportuno ricordare che alcuni programmi, come ROOT, consentono di esportare le figure direttamente in formato L^AT_EX. In questo caso, l'output sarà verosimilmente un file .tex contenente un codice analogo a quello appena illustrato. Per inserirlo nel documento principale, sarà sufficiente utilizzare il comando `\include {}` o `\input {}`.

Disegnare con TikZ

Oltre agli esempi specifici mostrati sopra, TikZ può essere usato per scopi molto più generali, realizzando disegni geometrici di ogni tipo. Va tenuto a mente che questo pacchetto è così ampio da richiedere un'ulteriore suddivisione in librerie che possono essere caricate a parte per usufruire di alcune funzionalità. In effetti, per compilare il codice seguente, sarà necessario caricare la libreria `calc` inserendo nel preambolo il comando

```
\usetikzlibrary{calc}
```

Il codice esempio, che andrà come al solito racchiuso in un ambiente `figure`, è il seguente:

```
\begin{tikzpicture}
\coordinate (S) at (0,1.5);
\coordinate (P) at (3,0);
\coordinate (Q) at (11,0);
\coordinate (R) at (7,6.93);
\coordinate (B) at (14,1.5);
\coordinate (T) at (14,3.5);

\coordinate (m) at ($(R)!0.5!(P)$);
\coordinate (b) at ($(Q)!0.4!(R)$);
\coordinate (t) at ($(Q)!0.6!(R)$);

\draw (P) -- (Q) -- (R) -- (P);
\draw (S) -- (m);
\draw (m) -- (t);
\draw (m) -- (b);

\coordinate (p1) at ($(b)!1/6!(t)$);
\coordinate (p2) at ($(b)!2/6!(t)$);
\coordinate (p3) at ($(b)!3/6!(t)$);
\coordinate (p4) at ($(b)!4/6!(t)$);
\coordinate (p5) at ($(b)!5/6!(t)$);
\coordinate (P1) at ($(B)!1/6!(T)$);
\coordinate (P2) at ($(B)!2/6!(T)$);
\coordinate (P3) at ($(B)!3/6!(T)$);
\coordinate (P4) at ($(B)!4/6!(T)$);
\coordinate (P5) at ($(B)!5/6!(T)$);
```

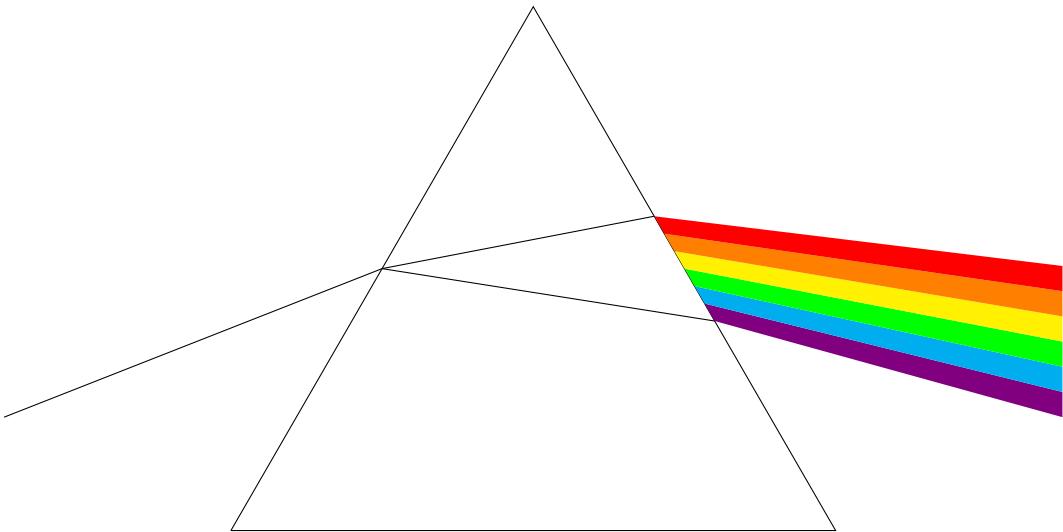


Figura 3.8: The Dark Side of Fisica II.

```
\fill[violet] (b) -- (B) -- (P1) -- (p1) -- (b);
\fill[cyan] (p1) -- (P1) -- (P2) -- (p2) -- (p1);
\fill[green] (p2) -- (P2) -- (P3) -- (p3) -- (p2);
\fill[yellow] (p3) -- (P3) -- (P4) -- (p4) -- (p3);
\fill[orange] (p4) -- (P4) -- (P5) -- (p5) -- (p4);
\fill[red] (p5) -- (P5) -- (T) -- (t) -- (p5);
\end{tikzpicture}
```

Il risultato, in figura 3.8, sarà sicuramente un'immagine cara agli appassionati di progressive rock e di ottica geometrica. Ma in cosa consiste questo codice? Andiamo per gradi:

- tutto il codice che produce la figura è contenuto in un ambiente `tikzpicture`;
- le prime righe definiscono le coordinate del punto da cui parte il raggio di luce (S), dei vertici del triangolo (P, Q, R) e delle estremità inferiore (B) e superiore (T) del fascio colorato. Per definire le coordinate in un'immagine TikZ è sempre bene visualizzarla su un reticolo cartesiano sul quale definire le coordinate dei vari punti. Notare inoltre che ogni linea termina sempre con un punto e virgola;
- le tre righe successive, per cui è necessaria la libreria `calc`, definiscono le coordinate dei punti `m`, `b` e `t` lungo i segmenti `RP` e `QR`. `m` si troverà a metà della lunghezza di `RP` e rappresenterà il punto in cui il raggio di luce incide sul lato sinistro del prisma, mentre `b` e `t` si troveranno rispettivamente al 40% e al 60% della lunghezza di `QR` e rappresenteranno i punti estremi in cui il fascio diffuso incide sul lato destro del prisma;
- seguono quindi alcune righe che, molto semplicemente, disegnano il prisma e il fascio di luce;
- le linee successive definiscono cinque punti `p1, ..., p5` che dividono in parti uguali il segmento `BT` e cinque punti `P1, ..., P5` che dividono in parti uguali il segmento `BT`;
- le ultime righe, infine, riempiono le bande colorate utilizzando i colori predefiniti di TikZ.

Sottolineo ancora una volta come queste piccole dispense non contengano affatto tutto ciò che c'è da sapere sulla grafica in L^AT_EX, soprattutto per quanto riguarda quella vettoriale. Ciononostante, mi auguro che possano essere un buon trampolino di lancio. Per il resto basta andare a tentativi, risolvere i problemi man mano che si presentano, divertirsi, e ricordare che Google è sempre vostro amico.

Riferimenti

- [PG17] Lorenzo Pantieri e Tommaso Gordini. *L'arte di scrivere con L^AT_EX*. 2017. URL: http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf.
- [Ars] Donald Arseneau. *The cancel package*. URL: <http://tug.ctan.org/macros/latex/contrib/cancel/cancel.pdf>.
- [Bar] Sergio C. de la Barrera. *The Physics Package*. URL: <http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/physics/physics.pdf>.
- [Col] Overleaf Colours. *Using colours in LaTeX*. URL: https://it.overleaf.com/learn/latex/Using_colours_in_LaTeX.
- [CTA] CTAN. *Comprehensive T_EXArchive Network*. URL: <https://www.ctan.org>.
- [Gia] Daniele Giacomini. *L^AT_EX: ambienti matematici*. URL: http://wwwcdf.infn.it/AppuntiLinux/latex_ambienti_matematici.htm.
- [Lee] John M. Lee. *Using script fonts in L^AT_EX*. URL: <https://sites.math.washington.edu/~lee/Writing/typesetting-script.pdf>.
- [Lis] Overleaf Lists. *Lists*. URL: <https://www.overleaf.com/learn/latex/lists>.
- [Ovea] Overleaf. *Display style in math mode*. URL: https://www.overleaf.com/learn/latex/display_style_in_math_mode.
- [Oveb] Overleaf. *Overleaf Documentation*. URL: https://www.overleaf.com/learn/latex/Bibliography_management_in_LaTeX.
- [Ovec] Overleaf. *Overleaf Documentation - Mathematics*. URL: https://www.overleaf.com/learn/latex/Mathematical_expressions.
- [Wika] Wikibooks. *LaTeX/Lengths*. URL: <https://en.wikibooks.org/wiki/LaTeX/Lengths>.
- [Wikb] Wikipedia. *Formule matematiche: Latex*. URL: https://it.m.wikipedia.org/wiki/Aiuto:Formule_matematiche_TeX.
- [Wikc] Wikipedia. *LaTeX*. URL: <https://it.wikipedia.org/wiki/LaTeX>.
- [Wri] Joseph Wright. *SiUnitx Package Documentation*. URL: <http://mirrors.ibiblio.org/CTAN/macros/latex/contrib/siunitx/siunitx.pdf>.