Scrivere in TEX e LATEX

Leonardo Robol <leonardo.robol@unipi.it>

—

Sergio Steffè <steffe@dm.unipi.it>

Laboratorio di Comunicazione mediante Calcolatore

Anno Accademico 2020 – 2021

1 TEX e LATEX

Il LATEX è, in un certo senso, un evoluzione del TEX. Più precisamente, è un insieme di macro (ovvero, comandi) e convenzioni su come strutturare un documento, che permettono all'utente finale di non preoccuparsi dell'impaginazione del documento, ma solo della sua struttura.

Queste note sono strutturate come una serie di esercizi che si possono svolgere per creare pagine in LATEX. Seguendo gli esercizi in ordine si dovrebbe riuscire ad acquisire le competenze necessarie per scrivere un file contenente degli appunti (come questo, oppure quelli di un corso), o delle slides per una presentazione.

Prima di partire con in LATEX, faremo qualche prova con il linguaggo TEX, per renderci conto delle differenze.

2 Primi passi con il TEX

Il linguaggio TEX è quasi sempre utilizzato in combinzione con il pacchetto di macro LATEX, e quasi mai da solo. In ogni caso, è opportuno farne la conoscenza perché può essere utile per estendere e programmare nostri comandi all'interno di documenti LATEX complessi.

Esercizio 1

Si cominci a produrre un documento TEX minimale: si crei un file di testo dal nome `esempio.tex` e si inserisca il seguente contenuto:

```
Questo file e' una prova di TeX.
\end
```

Il comando `\end` serve a terminare l'input, e fa in modo che il compilatore TEX termini di processare il file. Si può utilizzare il comando `tex esempio.tex` per compilare questo file.

Il comando dato nel precedente esercizio creerà due file:

- Il file `esempio.log` contiene l'output della compilazione.
- Il file `esempio.dvi` è invece l'output della nostra compilazione.

Per aprire il file `.dvi` si utilizzi il comando `xdvi esempio.dvi`, oppure `evince esempio.dvi`. In alternativa è possibile aprire il file-manager e cliccarci sopra.

Il formato DVI, che sta per "DeVice Independent", ed è un file che contiene una rappresentazione grafica del testo che abbiamo prodotto, in maniera indipendente dal supporto su cui andrà stampato (i.e., schermo oppure una stampante vera e propria).

Questo formato è stato nel tempo quasi del tutto rimpiazzato, nell'uso comune, dai formati Postscript (`.ps`) ed in particolare PDF (`.pdf`)

Esercizio 2

Si provi a ricompilare l'esempio precedente utilizzando `pdftex` invece che il comando `tex`. Questo creerà un file `.pdf`, che è possibile aprire con il comando `evince file.pdf`.

Attenzione! L'estensione di tutti i file che vengono considerati in queste note è sempre `.tex`, indipendentemente dal compilatore scelto.

2.1 Qualche formula matematica

Si può ora procedere a modificare l'esempio precedente inserendo del contenuto matematico. Si provi a modificare il file sorgente (`esempio.tex`) perché contenga il seguente testo:

```
Ora vogliamo scrivere della formule in linea, ad
esempio $y = x + 1$, anche con esponenti e/o
pedici: $x_{k+1} = A x_k + q$.

\end
```

Esercizio 3

Si compili il file precedente utilizzando `pdftex` e si controlli l'output. Modificarlo poi per ottenere la seguente equazione:

$$x_k^{(j)} = \gamma^{k-j}, \qquad \gamma \neq 0.$$

Suggerimento: le lettere greche si possono ottenere con i comandi `\alpha`, `\beta`, ..., e il simbolo $\neq$ con il comando `\neq`. Per centrare l'equazione, si usi il modo matematico speciale che si ottiene con `$$`. Per gli spazi, all'interno

delle equazioni, si provino[1] il simbolo `~` oppure il comando `\qquad`.

2.2 Formattare il testo

In alcuni casi, è desiderabile cambiare la formattazione del testo. Ad esempio, come è possibile ottenere del testo in grassetto, oppure in italico? Il TEX ha dei comandi apposta per cambiare modalità di inserimento del font. Ad esempio, il comando `\bf` passa dal font normale a quello in grassetto (`\bf` sta per Bold Format). Una volta attivato, però, quest'impostazione rimane sempre attiva!

Per evidenziare solo una parola o una frase è uso comune racchiudere la parte da evidenziare fra parentesi graffe, in modo che la modalità diversa sia attiva solo in quel blocco. Ad esempio, il codice:

```
Come inserire una parola in {\bf grassetto}?
```

produce il seguente output:

Come inserire una parola in **grassetto**?

In maniera simile si pùo usare il comando `\it` per attivare il carattere italico (`\it` sta per *italic*). In matematica, è comune usare il carattere italico per le definizioni.

Esercizio 4

Si aggiunga, utilizzando i comandi specificati sopra, la seguente frase al documento creato precedentemente:

> Sia $f : X \to Y$, con X, Y spazi topologici. Allora diremo che $f(x)$ è una *funzione continua* se, per ogni insieme aperto A si Y, $f^{-1}(A)$ è un insieme aperto in X.

Notiamo che nella definizione sopra solo le parole *funzione continua* sono in italico, mentre tutte le formule sono semplicemente in modo matematico! Per realizzare il simbolo $\to$, si può utilizzare il comando `\to`.

Attenzione, vin LATEX, i comandi `\bf`, `\it` vanno, se possibile, evitati; si preferisce sostituirli con i loro equivalenti LATEX (`\textbf` e `\emph`, che hanno un comportamento diverso, si veda la sezione 3.1).

[1] Attenzione, questi comandi speciali per la spaziatura sono da utilizzarsi solo nelle equazioni, e solo dove necessario. Nel testo, lasciare uno spazio lascerà il normale spazio fra le parole, e nelle equazioni non c'è bisogno di inserire spazi particolari fra i simboli, a meno di non volerli forzatamente separare.

3 Primi passi con LaTeX

Un documento LaTeX è costituito da uno (o più, come vedremo in seguito) file di testo, solitamente con estensione `.tex`. Supponiamo, in queste note, di voler creare un file di appunti, e di chiamarlo `appunti.tex`.

Esercizio 5

Si crei un file `appunti.tex` con il seguente contenuto:

```
\documentclass[a4paper]{article}
\title{Appunti di Analisi}
\author{Me stesso}
\begin{document}
  \maketitle
  Testo di prova
\end{document}
```

A questo punto, si apra un terminale nella cartella dove abbiamo creato questo file. Poi, si utilizzi il comando `pdflatex appunti.tex` per compilarlo, ed ottenere un file PDF.

Dopo aver completato l'Esercizio 5 avremo compilato il nostro primo documento LaTeX! Una cosa da notare è che non sono stati creati solamente il file `appunti.pdf`, ma anche dei file ausiliari come `appunti.log`, e `appunti.aux`. Per il momento non li discuteremo.

Analizziamo invece i vari comandi che abbiamo dato nel documento. Un comando (o, più precisamente, una macro) in LaTeX ha la forma:

```
\nomecomando[opt1,opt2][opt3]{arg1}{arg2}{arg3}
```

dove il carattere `\` indica l'inizio di un comando, i parametri `opt1`, `opt2` e `opt3` sono parametri opzionali, che potrebbero essere omessi, mentre `arg1`, `arg2` e `arg3` sono parametri obbligatori. Il numero di comandi opzionali e/o obbligatori dipende dal comando in considerazione. Analizziamo il caso del documento minimale da noi prodotto.

`\documentclass` serve a specificare le caratteristiche del documento. Ad esempio, nel nostro caso abbiamo usato `\documentclass[a4paper]{article}` per impostare la dimensione della pagina ad A4 (lo standard usuale in Europa), e il tipo di documento ad `article`. Esistono altri tipi di documenti, come `minimal`, `book`, ...

`\title` **e** `\author` servono ad indicare titolo ed autore del documento. Sono entrambi comandi che prendono esattamente un parametro. Attenzione: questi comandi non producono il testo, ma impostano solamente questi parametri all'interno del documento, per poter essere utilizzati in seguito.

`\begin{document}` **ed** `\end{document}` si utilizzano per iniziare e concludere la parte del documento che contiene il documento vero e proprio. Tutto quello che precede il comando `\begin{document}` non viene riprodotto sul documento (serve ad impostare parametri globali), e tutto quello che segue `\end{document}` viene ignorato. La parte che precede il comando `\begin{document}` si chiama *preambolo*.

`\maketitle` è un comando che stampa titolo, autore e data. L'output preciso dipende dal tipo di documento che è stato scelto (`book`, `article`, ecc.).

Il testo inserito dopo il comando `\maketitle` è semplice testo che viene inserito nel documento. Questo testo verrà formattato con il carattere standard all'interno del documento.

I comandi del tipo `\begin{...}`, `\end{...}` servono a delimitare gli *ambienti*.

Un ambiente è una porzione di codice dove, solitamente, vengono applicati degli stili particolari, e che può essere utilizzato per entrare in qualche altra modalità del TEX. Ad esempio, ci sono ambienti per creare liste puntate e numerate, o per allineare il testo. Ci sono anche ambienti che permettono di inserire figure, tabelle, ecc., come si vedrà in Sezione 6.

Esercizio 6

Si modifichi il testo precedente racchiudendo la stringa "Testo di prova" all'interno di un ambiente di tipo `center`, ovvero ottenendo:

```
\begin{center}
Testo di prova
\end{center}
```

Cosa succede (non troppo inaspettatamente)?

LATEX ignora gli spazi ripetuti: uno spazio o molti sono rappresentati come un unico spazio, e così vale anche per le nuove linee. L'unica eccezione a questa regola è il caso in cui si lascia una riga vuota, che viene interpretato come l'istruzione "Comincia un nuovo paragrafo".

L'usuale convenzione tipografica è di iniziare un nuovo paragrafo per separare le varie parti logiche del discorso. Questo crea automaticamente un rientro a sinistra per la prima riga. È possibile forzare una nuova linea con il comando `\\`, e questo non crea un nuovo paragrafo (ovvero: nessun rientro per la prima riga); è però sconsigliato a meno che non ci siano buone ragioni per farlo.

3.1 Formattare il testo

Si è visto nella sezione 2.2 come formattare il testo passando al grassetto o all'italico in TEX. I comandi `\bf` e `\it` che abbiamo usato stanno per "Bold Face" e "Italic", che si usano in tipografia per indicare il grassetto ed il corsivo (nel senso di font inclinati, indicati anche come *slanted* in inglese).

Questa sintassi è di solito evitata in LaTeX, e si preferisce utilizzare dei comandi che prendono come argomento il testo da modificare, invece che costringersi a racchiudere il testo all'interno di blocchi.

Ad esempio, invece di scrivere:

```
Questa {\bf parola} è in grassetto.
```

si utilizza:

```
Questa \textbf{parola} è in grassetto.
```

Similmente, si utilizza `\emph` invece di `\it`. Esiste anche il comando `\textit`, che si allinea più fedelmente al comando `\it`, mentre il comando `\emph` ha più precisamente lo scopo di enfatizzare il testo. Renderà dunque corsivo del testo all'interno di una regione con font normale, e invece rimuoverà il corsivo ad una parola in una zona dove il font è già corsivo (ad esempio all'interno dell'enunciato di un teorema). Questo ha lo scopo di far spiccare il testo rispetto a quello che lo circonda, e si usa spesso nelle definizioni.

Esercizio 7

Provate a modificare l'esercizio 4 in modo da essere codice LaTeXche segue le convenzioni sopra.

3.2 Suddividere il testo

Durante la stesura di un documento, è naturale suddividerlo in sezioni. Questa suddivisione viene gestita automaticamente da LaTeX: è sufficiente indicare la struttura del documento, e titoli di sezioni, indici, ecc., verranno gestiti in maniera completamente automatica.

Ad esempio, si provi a dare il comando `\section{Nuova sezione}`. Questo si occuperà di stampare sul testo il titolo di una nuova sezione, con la corrispondente numerazione.

Nel caso volessimo creare una nuova sezione senza numero, potremmo usare il modificatore `*`; il comando `\section*{Sezione senza numero}` creerà una sezione con il nome specificato, ma senza incrementare e stampare il numero della sezione corrente.

Esercizio 8

Si aggiungano sezioni, sottosezioni e paragrafi al vostro documento utilizzando i comandi `\section`, `\subsection`, `\subsubsection` e `\paragraph`. Provate anche il modificatore `*` per creare alcune sezioni senza numero.

In alcuni tipi di documenti può essere utile produrre un sommario all'inizio che lista le sezioni presenti nel documento. Questo può essere ottenuto tramite il comando `\tableofcontents`, che funziona in modo simile a `\maketitle`.

Attenzione: Perché il comando `\tableofcontents` funzioni nel modo corretto, potrebbe essere necessario compilare il documento due volte. Questo succede perché il compilatore LaTeX salva la lista delle sezioni all'interno del file `appunti.aux`, che verrà letto solo durante la successiva compilazione.

3.3 Riferimenti nel testo

Una volta create le sezioni, ci si potrebbe trovare nel caso in cui sia necessario riferirsi ad una sezione precedente, tramite il suo numero. Ad esempio, in una frase del tipo: "Come visto nella Sezione 2.1". È immediatamente chiaro che scrivere direttamente il numero della sezione non è una soluzione robusta a questo problema: nel caso aggiungessimo un'altra sezione precedente dovremmo andare a cercare e sostituire tutti i riferimenti precedentemente inseriti.

LaTeX propone la seguente soluzione: ad ogni elemento citabile si assegna un'etichetta tramite il comando `\label`, e poi si utilizza il comando `\ref` per citarla. Ad esempio:

```
\section{Introduzione} \label{sec:intro}

[...]

Come abbiamo visto nella Sezione~\ref{sec:intro}, [...]
```

Qualche osservazione:

- Il label viene assegnato all'ultimo comando che ha creato un elemento "citabile". Per il momento abbiamo visto solamente le sezioni, ma altri esempi sono le equazioni, tabelle, figure, ecc. Volendo, persino gli elementi di una lista numerata!
- È importante che `\label` segua il comando a cui si vuole assegnare l'etichetta: il LaTeX processa il documento dall'inizio alla fine, e dunque può etichettare solo l'output di comandi precedenti.
- È buona norma, come nell'esempio sopra, separare il nome dell'oggetto citato (nel nostro caso "Sezione") e il numero ottenuto dal comando `\ref` con il simbolo `~`. Questo rappresenta uno "spazio indistruttibile", che il LaTeX non potrà mai separare. Possiamo quindi garantire che non avremo nel testo una nuova linea fra "Sezione" e "2.1".
- Il nome del label può essere una qualunque stringa di testo. In questo caso abbiamo scelto di utilizzare il prefisso `sec:` per sottolineare che si tratta di una sezione, ma non è obbligatorio.

3.4 Sistema di pacchetti

Una delle grosse novità del LaTeX rispetto al TeX è la possibilità di ottenere funzionalità aggiuntive tramite un sistema di pacchetti ben organizzato. Sebbene

anche in TEX sia possibile caricare estensioni aggiuntive (in un certo senso il LATEX stesso lo è), i pacchetti LATEX sono organizzati in modo coerente e disponibili all'interno della distribuzione standard.

Ad esempio, possiamo caricare i pacchetti dell' AMS (American Mathematical Society) utilizzando il comando

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

Il precedente comando carica quattro pacchetti, con i nomi riportati nella lista sopra. Questi contengono varie funzionalità, come ad esempio il carattere per indicare i campi o anelli (ad es., $\mathbb{C}$ per i complessi o $\mathbb{R}$ per i reali).

Il comando `\usepackage` va utilizzato nel preambolo, ovvero prima del comando `\begin{document}`.

3.5 Lettere accentate e caratteri speciali

Dovendo scrivere in italiano, vi capiterà spesso di dover inserire caratteri accentati. Noterete quasi subito che inserirli direttamente nel testo però non ha nessun effetto: questo caratteri non appaiono nel documento finale!

Per ovviare a questo problema ci sono (almeno) due soluzioni:

1. La prima consiste nell'inserire le lettere accentate utilizzando dei comandi appositi. Ad esempio, per scrivere é possiamo utilizzare il comando `\'e`, mentre per inserire è utilizziamo `\`e`. Il comando `\'` significa "metti l'accento alla prossima lettera".

2. Possiamo caricare un pacchetto apposito, per "insegnare" a LATEX a leggere questi caratteri speciali. Nel nostro caso, questo va fatto con il comando `\usepackage[utf8x]{inputenc}`.

Storicamente, la prima strategia è stata sempre considerata preferibile, perché nello scambio di file non richiede che mittente e destinatario siano in grado di interpretare caratteri speciali. Tuttavia, nei sistemi operativi moderni questo è sempre verificato, e dunque anche la soluzione 2. può essere considerata senza troppi problemi. Notiamo però che la soluzione 1. ci permette di inserire anche lettere accentate con accenti che non abbiamo sulla tastiera. Ad esempio, se volessi scrivere ä potrei digitare `\"a`, oppure se volessi scrivere ǒ potrei inserire `\v{o}`, e così via. Attenzione, non sempre questi caratteri vengono riconosciuti nel modo corretto quando si effettua una ricerca nel testo!

Un altro pacchetto rilevante nel caso si voglia scrivere in italiano è il pacchetto `babel`, che traduce la maggior parte dei termini in italiano (ad esempio, il comando `\tableofcontents` produrrà un "Indice", invece di una sezione "Contents"); in aggiunta, implementa anche la sillabazione corretta che permette a LATEX di spezzare correttamente le parole quando deve andare a capo.

4 Scrivere equazioni

La ragione principale per cui il LaTeX è così diffuso nell'ambiente matematico è la facilità con cui si possono scrivere equazioni. In effetti, questo è il motivo principale per cui il software è stato inventato.

4.1 Modi matematici

Per poter realizzare equazioni bisogna passare dal modo testuale normale al modo matematico, in cui il testo che scriviamo viene interpretato in modo diverso. Ci sono due diversi modi matematici che si possono usare:

1. Il modo *in linea*, che permette di inserire matematica all'interno del testo, e nel quale si entra e si esce utilizzando il singolo `$`.

2. Il modo *display*, nel quale le equazioni vengono centrate all'interno della pagina. Si può entrare in questo modo con il comando `\[`, ed uscire con `\]`. In alternativa è possibile rimpiazzare questi due comandi con il doppio dollaro `$$`, che però può produrre spaziature meno precise.

3. L'ambiente `equation`, quasi equivalente alla modalità *display*, ma nel quale ad ogni equazione viene assegnato un numero. Invece che delimitare l'equazione con `\[` e `\]`, si utilizzano i comandi `\begin{equation}` ed `\end{equation}`. Si può utilizzare `\label` (ad esempio subito dopo `\begin{equation}`, in ogni caso all'interno dell'ambiente) per assegnare un'etichetta all'equazione, da citarsi nel seguito con `\eqref`; attenzione, per quest'ultimo comando serve includere i pacchetti `amsmath`.

Consideriamo un primo esempio di matematica, che contiene un misto di equazioni in linea ed equazioni centrate.

> Il prodotto dei polinomi $x^2 + x + 1$ e $x - 1$ è dato da:
>
> $$(x^2 + x + 1)(x - 1) = x^3 - 1.$$

Questo esempio si può ottenere con il seguente codice:

```
Il prodotto dei polinomi $x^2 + x + 1$ e $x - 1$ è dato da:
\[
  (x^2 + x + 1) (x - 1) = x^3 - 1.
\]
```

Osserviamo che viene utilizzata la sintassi `x^k` per scrivere un esponente. In maniera simile, potremmo utilizzare `x_k` per ottenere il pedice x_k.

Nel caso sia necessario scrivere esponenti e/o pedici (o in generale, argomenti di comandi) composti da più caratteri, sarà necessario utilizzare le parentesi graffe per raggrupparli. Ad esempio, supponiamo di voler ottenere il seguente testo:

Le radici dell'unità, che risolvono $x^n - 1 = 0$, possono essere descritte da ξ_n^j, dove:

$$\xi_n = e^{\frac{2\pi i}{n}},$$

e il simbolo i denota l'unità immaginaria, ovvero $i^2 = -1$.

Possiamo utilizzare i seguenti comandi:

```
Le radici dell'unità, che risolvono $x^n - 1 = 0$,
possono essere descritte da $\xi_n^j$, dove:
\[
  \xi_n = e^{\frac{2 \pi i}{n}},
\]
e il simbolo $i$ denota l'unità immaginaria, ovvero $i^2 = -1$.
```

Nell'esempio precedente possiamo imparare molti nuovi comandi. Ad esempio, `\xi` produce la lettera greca ξ. In maniera analoga, sono disponibili `\alpha`, `\beta`, `\gamma`, ...

Il comando `\frac` produce una frazione tipo $\frac{a}{b}$. Prende due argomenti, racchiusi fra graffe se composti da più di un carattere, che sono rispettivamente numeratore e denominatore.

Supponiamo ora di voler essere più precisi, e dire che le radici ξ_n^j sono quelle di $x^n - 1$ sul campo $\mathbb{C}$. Possiamo ottenere il simbolo che usualmente denota i complessi con il comando `\mathbb C`. Il comando `\mathbb`, che permette di utilizzare il font denominato `bb`, si utilizza solitamente per anelli e campi, come $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{F}_p$, ecc. Questo comando non è disponibile nel LaTeX standard, e per poterlo utilizzare dovremo caricare un pacchetto ausiliari, che contiene le definizioni di nuovi comandi. In questo caso specifico, si tratta del pacchetto `amsfonts`, dove `ams` sta per American Mathematical Society. Per caricare il pacchetto, dobbiamo inserire il comando `\usepackage{amsfonts}` nel preambolo del documento (prima del comando `\begin{document}`), come spiegato prima all'interno della Section 3.4.

Esercizio 9

Modificare l'esempio di sopra inserendo la nota che le radici stanno all'interno del campo $\mathbb{C}$. Se preferite, invece che scriverlo come testo, potete anche utilizzare la notazione $x \in \mathbb{C}$, utilizzando con il comando `\in`.

I comandi matematici sono moltissimi. Fra i più utilizzati possiamo considerare quelli per creare il simbolo di integrale. Qualche esempio per impararne qualcuno:

$$f(z) = \frac{1}{2\pi i} \int_\Gamma \frac{f(w)}{w - z}\, dw$$

```
\[
```

```
  f(z) = \frac{1}{2\pi i} \int_\Gamma \frac{f(w)}{w - z}\ dw
\]
```

$$f(z) = \frac{1}{2\pi i} \int_\Gamma \frac{f(w)}{w - z}\ dw$$

```
\[
  \int_a^b f(x)\ dx = F(b) - F(a).
\]
```

$$\int_a^b f(x)\ dx = F(b) - F(a).$$

```
\[
  \sum_{j = 1}^\infty \frac{1}{j} = \infty, \qquad
  \sum_{j = 1}^n j = \frac{j(j+1)}{2}.
\]
```

$$\sum_{j = 1}^\infty \frac{1}{j} = \infty, \qquad \sum_{j = 1}^n j = \frac{j(j+1)}{2}.$$

```
\[
  \sum_{j = 1}^\infty \frac{1}{j} = \infty, \qquad
  \sum_{j = 1}^n j = \frac{j(j+1)}{2}.
\]
```

$$\frac{\partial}{\partial x} \sqrt{xy} = \frac{\sqrt{y}}{2\sqrt{x}}$$

```
\[
  \frac{\partial}{\partial x}  \sqrt{xy} =
  \frac{\sqrt{y}}{2 \sqrt{x}}
\]
```

$$7 \equiv 1 \mod 3$$

```
\[
  7 \equiv 1 \ \mathrm{mod}\ 3
\]
```

Alcuni comandi sono abbastanza autoesplicativi. Notiamo solo che `\` e `\qquad` possono essere utilizzati per produrre spazi piccoli e grandi all'interno di equazioni. Se questi non vengono utilizzati, gli spazi saranno completamente ignorati.

Il comando `\mathrm`, invece, permette di utilizzare il font roman all'interno di un'equazione, e può essere utile per inserire nome di funzioni / operatori (come per mod sopra). Alcune funzioni dispongono già di un comando con il loro nome, come ad esempio quelle trigonometriche (si provino i comandi `\sin`, `\cos`, ...)

4.2 Matrici e vettori

Come possiamo inserire una matrice? Anche per questa domanda non esiste una risposta univoca. Partiamo dalla soluzione più semplice, che richiede di utilizzare il pacchetto `amsmath`. Assicuratevi dunque di avere il comando

```
\usepackage{amsmath}
```

nel preambolo. A questo punto, potete utilizzare gli ambienti `pmatrix` e/o `bmatrix` per inserire delle matrici, come nel seguente esempio

```
A = \begin{bmatrix}
  1 & 2 & 3 \\
  4 & 5 & 6 \\
\end{bmatrix} = \begin{pmatrix}
1 & 2 & 3 \\
4 & 5 & 6 \\
\end{pmatrix}.
```

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}.$$

Gli stessi comandi si possono usare per vettori colonna (come matrici $m \times 1$) e vettori riga (matrici $1 \times n$). Il comando `&` separa le varie colonne, e `\\` inizia una nuova riga.

Potrebbe capitare di volere un controllo più fine sul risultato finale della matrice che stiamo inserendo, ad esempio, come inserire una matrice di questo tipo?

$$A := \left[\begin{array}{cc|c} 1 & 2 & 3 \\ 4 & 5 & 6 \end{array}\right]$$

Il primo passo è fare uso dell'ambiente array, che permette di disporre del contenuto in forma di tabella. Richiede un argomento aggiuntivo che gli dica quante colonne ci sono, e come deve venire allineato il contenuto al loro interno. Noi scegliamo 3 colonne centrate, identificate da `ccc`:

```
\begin{array}{ccc}
  1 & 2 & 3 \\
  4 & 5 & 6 \\
\end{array}
```

$$\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \end{array}$$

Ora, modifichiamo questo esempio per inserire la riga verticale che separa la seconda e la terza colonna. Per farlo, basta modificare l'indicazione delle colonne con `cc|c`.

```
\begin{array}{cc|c}
1 & 2 & 3 \\
4 & 5 & 6 \\
\end{array}
```

$$\begin{array}{cc|c} 1 & 2 & 3 \\ 4 & 5 & 6 \end{array}$$

Adesso non ci rimane che racchiudere questo array fra parentesi quadre. Se mettessimo semplicemente i caratteri `[` e `]` a destra e sinistra questi sarebbero troppo piccoli. Possiamo usare i comandi `\left` e `\right` per generare dei delimitatori che si ingrandiscano abbastanza da racchiudere tutto quello che sta fra di loro:

```
\left[ \begin{array}{cc|c}
1 & 2 & 3 \\
4 & 5 & 6 \\
\end{array} \right]
```

$$\left[\begin{array}{cc|c} 1 & 2 & 3 \\ 4 & 5 & 6 \end{array}\right]$$

Ecco il nostro esempio completo. Osserviamo che non c'è bisogno che i delimitatori per `\left` e `right` siano coerenti. In particolare, è possibile specificare un delimitatore invisibile scrivendo `\left.` oppure `\right.`; questo può tornare utile nel caso si voglia definire una funzione in modo diverso su diverse parti del dominio:

```
f(x) = \left\{
  \begin{array}{cc}
    1 & x = 0 \\
    \frac{\sin(x)}{x} & x \neq 0
  \end{array}
\right.
```

$$f(x) = \left\{\begin{array}{cc} 1 & x = 0 \\ \frac{\sin(x)}{x} & x \neq 0 \end{array}\right.$$

In realtà, per questo caso particolare il pacchetto `amsmath` fornisce già l'ambiente `cases` (provatelo!) che permette di ottenere un risultato migliore.

Esercizio 10

Si provi a realizzare una pagina partendo da un modello, senza seguire una procedura dettagliata. Si scarichi il file `pagina-test.pdf` da Moodle, e si provi a replicarlo nel modo più fedele possibile utilizzando LaTeX.

4.3 Teoremi ed enunciati

Esercizio 11

Scegliete un teorema a piacere, e riportartelo nel vostro file `appunti.tex`. Inserite nel preambolo il comando `\newtheorem{theorem}{Theorem}`, che creerà un nuovo ambiente chiamato `theorem`, e poi racchiudete il vostro teorema dentro uno di questi:

```
\begin{theorem}
Sia ... . Allora ...
\end{theorem}
```

Nella maggior parte dei documenti matematici si trovano svariati enunciati, come teoremi, proposizioni, lemmi, ecc., ma anche definizioni, osservazioni, ecc. Il pacchetto `amsthm`, che abbiamo caricato in precedenza, ci permette di strutturarli con una numerazione all'interno del documento. In particolare, potremo usare anche i comandi `\label` e `\ref` per inserire riferimenti in punti successivi del testo.

Per poter utilizzare questi ambienti è necessario definirli nel preambolo, ovviamente dopo aver caricato il pacchetto `amsthm`. La sintassi da utilizzare è la seguente:

```
\newtheorem{theorem}{Theorem}[section]
\newtheorem{lemma}[theorem]{Lemma}
\newtheorem{proposition}[theorem]{Proposition}

\theoremstyle{remark}
\newtheorem{remark}[theorem]{Remark}

\theoremstyle{definition}
\newtheorem{definition}[theorem]{Definition}
```

Soffermiamoci ora sui vari comandi dati sopra:

`\newtheorem{theorem}{Theorem}[section]` indica di creare un nuovo ambiente di nome `theorem`, che creerà degli enunciati con nome "Theorem". La numerazione di questi enunciati seguirà la numerazione delle sezioni. Ad esempio, nella Sezione 2 avremo Theorem 2.1, Theorem 2.2, ecc. Se il parametro opzionale non viene dato, allora verrà utilizzata una numerazione incrementale con 1, 2, ...

`\newtheorem{lemma}[theorem]{Lemma}` indica invece di creare un nuovo ambiente `lemma`, in maniera simile a `theorem`. In questo caso però, il parametro opzionale cambia posizione e dice a questo nuovo ambiente di condividere la numerazione con l'ambiente `theorem` creato in precedenza In questo modo, se aggiungiamo 2 teoremi con un lemma intermedio questi si chiameranno Theorem 2.1, Lemma 2.2, e Theorem 2.3. Senza quest'accortezza, tutti questi oggetti avrebbero numerazioni indipendenti.

`\theoremstyle` serve a cambiare stile degli ambienti teoremi. La scelta di default è *theorem*, che utilizza il testo in italico, e sono disponibili anche *definition* e *remark*. Il loro comportamento esatto dipende anche dal tipo di documento che si sta compilando.

Esercizio 12

Si seguano le istruzioni sopra aggiungendo almeno un Lemma agli appunti, condividendo la numerazione con il teorema che aggiunto prima. Si inserisca poi una dimostrazione utilizzando i comandi `\begin{proof}` ed `\end{proof}`.

5 Realizzare delle slide

La classe di documento più spesso utilizzata è `article`. Nel caso si vogliano realizzare delle slide, ad esempio per un seminario, è possibile però specificare la classe `beamer`.

La struttura del documento cambia: all'interno di `\begin{document}` ed `\end{document}` sarà necessario inserire uno o più ambienti `frame`; ognuno di questi rappresenta una slide, ed è comune utilizzare il primo semplicemente per il comando `\maketitle`.

Un documento minimale potrebbe essere il seguente:

```
\documentclass{beamer}

\title{Titolo}
\author{Io}

\begin{document}
  \begin{frame}
    \maketitle
  \end{frame}

  \begin{frame}{Prima slide}
    Contenuto della prima slide.
  \end{frame}
\end{document}
```

Esercizio 13

Si provi a compilare il documento sopra, e ad effettuare qualche modifica, in modo da ottenere tre slide contenenti almeno un'equazione.

5.1 Transizioni

All'interno della classe `beamer` ci sono alcuni comandi a disposizione che sono particolarmente utili, e che permettono di gestire le transizioni:

- Il comando `\pause` permette di far apparire il contenuto incrementalmente. Potete inserire un numero arbitrario di questi in una slide, e l'effetto durante la presentazione sarà come se la slide apparisse passo passo.
- In maniera simile, potete utilizzare il comando `\only<n>{ ... }` per inserire del contenuto solo nell'n-esima transizione di una slide. Oltre a specificare un intero, è anche possibile specificare un intervallo come `n-m`, eventualmente anche aperto a destra o sinistra scrivendo `n-` oppure `-m`.

- Il comando `\alt<n>{a}{b}` funziona in modo simile al comando `\only`, e mostra `a` quando ci troviamo alla transizione n-esima, e `b` altrimenti. Come per `\only`, è possibile specificare anche intervalli, e non solo una singola transizione.

Esercizio 14

Sperimentate con i comandi precedenti in modo da ottenere una lista numerata di 4 elementi, che appaiono nella sequenza:

- 1 e 2
- solo 2
- 3 e 4
- solo 3.

Dovreste riuscire ad ottenere questo effetto facilmente con il comando `\only`.

5.2 Temi per beamer

Il tema di default di beamer non è particolarmente accattivante. Se ne possono utilizzare altri tramite il comando

```
\usetheme{nome}
```

dove `nome` è il nome del tema scelto. In maniera simile, è possibile cambiare lo schema di colori con il comando

```
\usecolortheme{nome}
```

Per vedere i temi e gli schemi di colori disponibili di default, si può consultare il sito `https://hartwork.org/beamer-theme-matrix/`.

Questo non esaurisce la lista dei temi disponibili. In rete se ne possono trovarne altri, oppure è possibile anche scriverne uno da soli!

È anche possibile personalizzare lo sfondo delle slide alterando la proprietà `background canvas` della presentazione, con il seguente comando:

```
\setbeamertemplate{background canvas}{
  [ ... Codice LaTeX qui ... ]
}
```

dove si può inserire del codice LaTeX personalizzato. Questo si rivelerà particolarmente utile dopo aver visto la prossima sezione (che descrive come inserire immagini), o quella su TikZ, che vi potrebbe dare un'idea di come disegnare sullo sfondo.

6 Grafici, immagini e tabelle

LaTeX gestisce in modo particolare l'inserimento di elementi come grafici e tabelle. In generale, esiste un ambiente utilizzato per creare gli elementi da inserire (ad esempio, `tabular` per le tabelle), ed un altro per posizionarli nella pagina con una didascalia ed i riferimenti corretti (nel caso delle tabelle, l'ambiente `table`).

Questo può causare un po' di frustrazione iniziale perché LaTeX è dotato di un algoritmo per posizionare gli elementi nella pagina in modo "ottimale", secondo opportune regole tipografiche, ed è molto difficile da condizionare.

Sebbene sia teoricamente possibile specificare dei suggerimenti per il posizionamento, come vedremo in seguito, questi vengono tenuti molto poco di conto.

6.1 Tabular e table

Per inserire una tabella possiamo usare l'ambiente `tabular`. Si può notare che ci sono molte similitudini con l'ambiente `array` e quelli per inserire matrici, come ad esempio le parole chiave per l'allineamento all'interno delle celle e il carattere `&` per separare le colonne. Ad esempio, il seguente codice:

```
\begin{tabular}{rc|c}
  Colonna 1 & Colonna 2 & Colonna 3 \\ \hline
    1 & $x - y$ & oggi \\
    2 & $e^x$   & domani
\end{tabular}
```

genera il seguente output:

Colonna 1	Colonna 2	Colonna 3
1	$x - y$	oggi
2	e^x	domani

Si può osservare in particolare:

- La specifica delle colonne viene effettuata con la stringa `rcc`, che significa che la prima colonna dev'essere allineata a destra, la seconda e la terza al centro. È possibile specificare anche l'allineamento a sinistra (con la keyword `l`), oppure colonne di larghezza fissata.
- Il comando `\hline` può essere utilizzato per generare una riga divisoria, ad esempio dopo l'intestazione. È possibile utilizzare una riga doppia ripetendolo due volte.
- È possibile utilizzare equazioni all'interno delle tabelle, racchiudendolo in opportuni dollari come nel resto del documento.

Nella maggior parte dei casi è necessario inserire tabelle a cui fare riferimento nel testo, ad esempio con espressione del tipo "Come visto in Tabella 2

...". Per ottenere questo risultato, si preferisce includere l'ambiente `tabular` all'interno di un ambiente più esterno `table`, che può contenere anche una didascalia (tramite il comando `\caption`) e un `\label`. La sintassi per creare una table è la seguente:

```
\begin{table}
  \begin{tabular}{...}
    [...]
  \end{tabular}
  \caption{La mia tabella}
  \label{tab:mia}
\end{table}
```

È possibile inserire riferimenti a una tale tabelle all'interno del documento utilizzando il comando `\ref{tab:mia}`, esattamente come fatto per le sezioni. Come in quel caso, è preferibile inserire uno spazio non interrompibile fra la parola "Tabella" e il riferimento, in modo che LaTeX non vada a capo. Ad esempio, si può scrivere `Tabella~\ref{tab:mia}`.

Quando viene creato un ambiente `table`, non ci sono garanzie che appaia nel punto in cui è stato inserito il codice corrispondente. Anzi, questo succederà molto di rado. LaTeX tenterà di posizionarlo nel posto "migliore possibile". È possibile, tuttavia, specificare una propria preferenza, con un parametro opzionale dell'ambiente `table`, da inserire fra parentesi quadre.

Ad esempio, il comando

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

genera una tabella chiedendo a LaTeX di inserirla dove è stato scritto il codice, o comunque non troppo lontano. In maniera simile, è possibile utilizzare la lettera `t` per dire che si preferisce averla in cima alla pagina, oppure `b` per il fondo della pagina. Si può utilizzare `h!` per insistere sul fatto che la tabella dovrebbe essere inserita nella posizione corrente. In ogni caso, LaTeX avrà sempre l'ultima parola in proposito.

6.2 Includere immagini esterne

LaTeX è abbastanza pignolo sui formati delle immagini esterne da inserire. Quelli supportati da `pdflatex`, che è il compilatore utilizzato fino ad ora, sono:

- PNG, per la grafica non vettoriale. Le immagini in questa categoria includono le foto scattate con una macchina fotografica, che sono rappresentate sul computer come una matrice di punti colorati (detti pixel).
- PDF, per la grafica vettoriale. Queste immagini, al contrario delle precedenti, sono composte da una loro descrizione geometrica. Questo implica che è possibile scalarle senza alcuna perdita di qualità.

Per inserire una qualsiasi immagine nel documento, è possibile utilizzare il comando

```
\includegraphics{immagine}
```

che caricherà automaticamente un file `immagine.pdf` oppure `immagine.png`. Attenzione, perché il comando sia disponibile è necessario includere il pacchetto `graphicx`, tramite il comando `\usepackage{graphicx}`.

In maniera simile a quanto succede per la tabelle, anche le immagini vengono solitamente posizionate all'interno di un riquadro con una didascalia e un label per riferirsi ad esse in seguito. Questo si ottiene con l'ambiente `figure`, da usare nel seguente modo:

```
\begin{figure}
  \includegraphics{immagine}
  \caption{Una mia fotografia}
  \label{fig:miafoto}
\end{figure}
```

Esercizio 15

Si scelga una fotografia e la si inserisca nel documento, posizionandola all'interno di una ambiente `figure`.

Le stesse considerazioni fatte per l'ambiente `table` sul posizionamento valgono anche per l'ambiente `figure`.

6.3 Grafici di funzioni e dati

Consideriamo ora un altro task piuttosto comune. Supponiamo di voler produrre un grafico di una funzione, oppure di alcuni dati. Ad esempio, potremmo voler fare un grafico di $f(x) = x \log x$ su $[0, 2]$, oppure plottare dei dati calcolati da un altro programma.

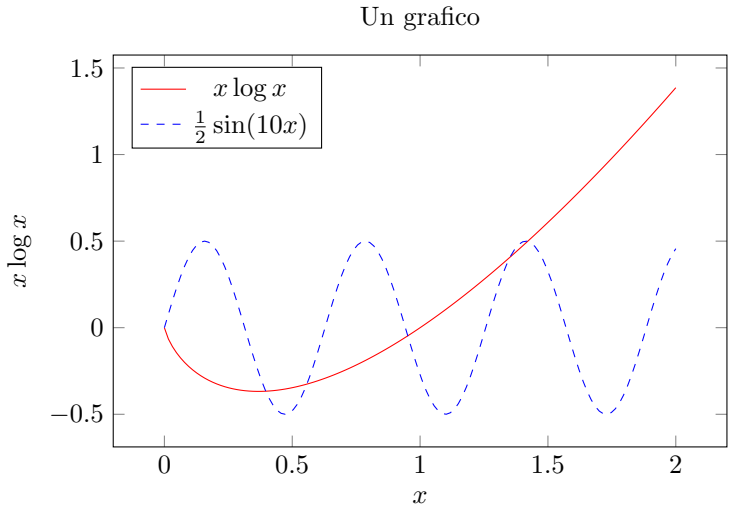
Per questo scopo, è possibile utilizzare il pacchetto `pgfplots`. Come gli altri pacchetti, va incluso con il comando `\usepackage{tikz,pgfplots}`. Questo comando carica il pacchetto `tikz`, che si può utilizzare per produrre grafica di vario tipo, e il pacchetto `pgfplots` che è basato su TikZ e permette di realizzare facilmente grafici di funzione.

Ad esempio, si provi ad inserire il seguente codice nel documento:

```
\begin{tikzpicture}
\begin{axis}[xlabel = $x$, ylabel = $x \log x$,
             title = {Un grafico}, legend pos = north west,
             width = .8\linewidth, height = .35\textheight]
  \addplot[domain = 0:2, red, samples = 128] { x * ln(x)};
  \addplot[domain = 0:2, blue, dashed, samples = 128] {sin(deg(x)*10)/2};
  \legend{$x \log x$, $\frac{1}{2}\sin(10x)$};
```

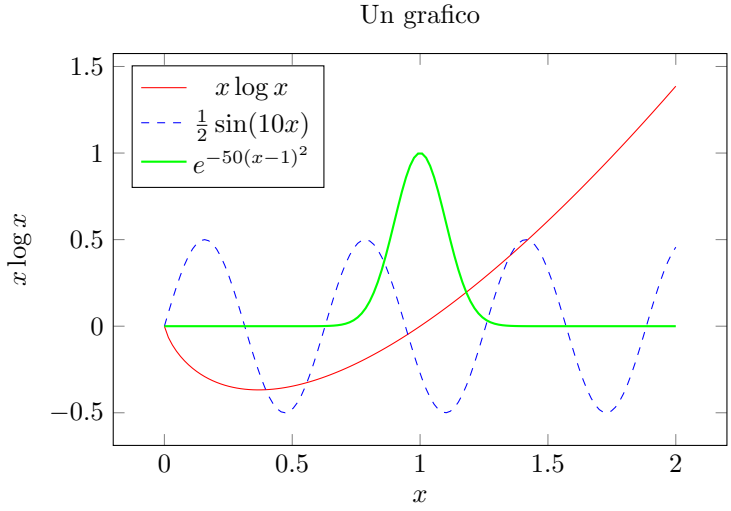
```
\end{axis}
\end{tikzpicture}
```

L'output ottenuto dovrebbe essere simile al seguente:



Esercizio 16

Si provi a modificare il codice sopra per plottare anche il grafico della funzione $e^{-50(x-1)^2}$ (si utilizzi `exp` per indicare l'esponenziale). Utilizzare un tratto molto marcato verde (ottenibile con l'opzione `thick`). Il risultato dovrebbe essere il seguente:



Nel caso si avessero a disposizione dei dati generati da un programma, potremmo volerli plottare come punti. Ad esempio, consideriamo un file `punti.dat` ottenuto calcolando per dei punti fra 0 ed 1 il loro quadrato; potete crearlo copiando i seguenti valori:

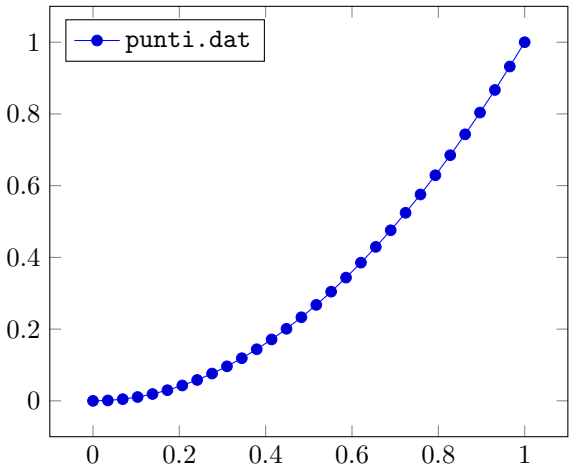
```
0.00000   0.00000
0.03448   0.00119
0.06897   0.00476
0.10345   0.01070
0.13793   0.01902
0.17241   0.02973
0.20690   0.04281
0.24138   0.05826
0.27586   0.07610
0.31034   0.09631
0.34483   0.11891
0.37931   0.14388
0.41379   0.17122
0.44828   0.20095
0.48276   0.23306
0.51724   0.26754
0.55172   0.30440
0.58621   0.34364
0.62069   0.38526
0.65517   0.42925
0.68966   0.47562
0.72414   0.52438
0.75862   0.57551
0.79310   0.62901
0.82759   0.68490
0.86207   0.74316
0.89655   0.80380
0.93103   0.86683
0.96552   0.93222
1.00000   1.00000
```

Esercizio 17

Utilizzare il comando, da inserire all'interno di un ambiente `axis`,

```
\addplot table {punti.dat};
```

per ottenere un plot dei punti contenuti nel file `punti.dat` appena creato. Il risultato dovrebbe essere simile al seguente:



Si cerchi in rete il manuale di `pgfplots`, e si modifichi il codice in modo da cambiare il tipo di punti in da pallini a triangoli.

Commento: Alcuni programmi che possono essere d'aiuto per generare questo tipo di dati sono MATLAB, GNU/Octave, Maple, oppure Mathematica o Maxima. Fra questi Octave e Maxima sono open source e gratuiti, e di MATLAB esiste una licenza d'ateneo. Una versione vetusta ma funzionante di Maple è disponibile sulle macchine dell'Aula 4. Alcuni di questi software possono anche produrre direttamente dei grafici da poi importare con `\includegraphics`, ma che spesso hanno una più bassa qualità tipografica ed integrazione in LaTeX.

7 Definire nuovi comandi

Spesso ci si trova a dover usare un simbolo, o qualche funzionalità evoluta di LaTeX ripetutamente all'interno di un documento. Supponiamo ad esempio di voler definire un operatore:

$$\begin{array}{rccl} \mathfrak{R}: & [0,1] & \to & [0,1] \\ & x & \mapsto & \sqrt{2}\cdot x \mod 1, \end{array}$$

ma di non essere ancora sicuri del simbolo da usare per denotarlo. Vorremmo evitare di trovarci più avanti a dover sostituire all'interno di tutto il documento la notazione usata precedentemente.

Per ovviare a quetso inconveniente, si può utilizzare il comando LaTeX `\newcommand`, che permette di definire nuovo comando. Ad esempio, nel caso precedentemente avremmo potuto inserire nel preambolo:

```
% Operatore R(x)
\newcommand{\myop}{\mathfrak{R}}
```

Questo ci permette di scrivere l'operatore precedente semplicemente utilizzando il comando `\myop(x)`. Se nel futuro cambiassimo idea, e volessimo denotare questo operatore con, ad esempio, $\Gamma(x)$, sarebbe sufficiente cambiare la definizione del comando nel preambolo. È possibile anche definire comandi che accettino un certo numero di parametri. Ad esempio,

```
\newcommand{\mycommand}[2]{Ciao #1, io mi chiamo #2}
```

si può chiamare con la sintassi `\mycommand{Giovanni}{Piero}`, e produrrebbe l'output "Ciao Giovanni, io mi chiamo Piero".

Esercizio 18

Si scriva un comando `\norm` che accetti due parametri: un vettore x ed un intero p, e che produca il simbolo della norma p del vettore x:

```
\norm{x}{2}
```

$\|x\|_2$

Si possono usare i comandi `\lVert` e `\rVert` per produrre le stanghettte verticali (rispettivamente a sinistra e a destra del singolo). Si modifichi poi il comando per adattarsi all'altezza del contenuto, e testarlo sul seguente esempio:

```
\norm{
  \begin{bmatrix}
    x_1 \\ x_2
  \end{bmatrix}
}{2}
```

$$\left\| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right\|_2$$

8 Utilizzare la bibliografia

Quando si realizza un documento come un articolo scientifico, oppure una tesi di laurea, è necessario includere nel documento i riferimenti bibliografici che si sono utilizzati. Le distribuzioni LaTeX includono un software, chiamato BibTeX, che permette di realizzarla con poco sforzo.

L'uso di BibTeX si organizza in questo modo:

- Si crea un file che contiene il database della bibliografia da utilizzare, con estensione `.bib`; ad esempio, potremmo creare un file `biblio.bib`.
- In questo file, si inseriscono i riferimenti da utilizzare. Questi vanno scritti in un formato particolare compreso da BibTeX, di cui forniremo dettagli in seguito. Ogni elemento bibliografico è identificato da un nome univico.
- Nel documento, si citano i lavori con il comando `\cite{ID}`, dove ID è l'identificatore menzionato sopra.

- Alla fine del documento, si sceglie lo stile da usare per la nostra bibliografia, e si specifica il nome del file "database" da utilizzare. Ad esempio:

  ```
  \bibliographystyle{plain}
  \bibliography{biblio}
  ```

  Lo stile `plain` è abbstanza comune, ma ne esistono molti altri disponibile (provare ad esempio `alpha`).

Gli elementi da inserire all'interno del file `biblio.bib` (che è un semplice file di testo, hanno il seguente formato:

```
@book{knuth1997art,
  title={The art of computer programming},
  author={Knuth, Donald Ervin},
  volume={3},
  year={1997},
  publisher={Pearson Education}
}
```

Nell'esempi sopra, per citare il lavoro di Knuth, potremmo usare nel documento LaTeX il comando `\cite{knuth1997art}`. Gli elementi da inserire nel database non si scrivono (usualmente) a mano, ma si possono scaricare da alcuni portali. Uno di questi è offerto dell'AMS (American Mathematica Society) e si chiama MathSciNet. Sfortunatamente, è accessibile solamente dai computer collegati alla rete dell'Ateneo. In alternativa, è possibile utilizzare il sito web `https://scholar.google.com/`. Su quest'ultimo è possibile cercare libri e articoli scientifico, e citarli cliccando sul simbolo delle virgolette, che permette di scaricare la bibliografia anche in formato BibTeX.

Per utilizzare BibTeX, è necessario compilare il documento seguendo la procedura (supponiamo il documenti si chiami `miofile.tex`)

1. Compilare il documento con `pdflatex miofile.tex`;
2. Compilare la bibliografia con `miofile.aux`; il file `.aux` generato da `pdflatex` contiene le informazioni necessarie per sapere quali elementi estrarre (solo quelli citati).
3. Ricompilare con `pdflatex miofile.tex`, per risolvere i riferimenti.

Se utilizzate un editor come TeXworks, TeXstudio, o l'interfaccia web basata su Overleaf, questi step verranno eseguiti in automatico. Altrimenti, dovete ricordare di farli sempre. In realtà, gli step 2 e 3 sono necessari solamente se la bibliografia cambia.

Esercizio 19

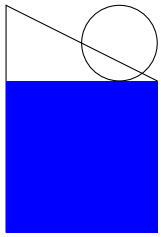
Si inserisca una bibliografia, e si citino il lavoro di Knuth riportato sopra e il LaTeX companion (libro da trovarsi su Google Scholar).

9 Disegnare in LaTeX

Abbiamo già visto nelle sezioni precedenti come generare dei grafici direttamente in LaTeX. Questo è solo un piccolo esempio di uso di un pacchetto molto potente, chiamato TikZ, che permette di disegnare all'interno del nostro documento. In ogni momento, è possibile iniziare un disegno tramite l'ambiente `tikzpicture`, e produrre elementi grafici con appositi comandi. Per usare il pacchetto, occorre inserire il comando `\usepackage{tikz}`.

Si consideri il seguente esempio, utilizzando i comandi `\draw` e `\fill` che permettono di disegnare linee e riempire delle regioni.

```
\begin{tikzpicture}
  \draw (0,0) -- (2,0) --
     (2,2) -- (0,3) -- cycle;
  \fill[blue] (0,0) rectangle (2,2);
  \draw (1.5,2.5) circle (.5);
\end{tikzpicture}
```



È spesso utile anche il comando `\filldraw`, che permette di disegnare contemporaneamente il contorno ed il riempimento di una regione. Tutti questi comandi possono accettare molte opzioni specificate fra parentesi quadre, ad esempio per produrre una freccia:

```
\begin{tikzpicture}
  \draw[->] (0,0) -- (2,0);
\end{tikzpicture}
```


E se volessimo disegnare una freccia curva, ad esempio per denotare una mappa fra due insiemi? Possiamo creare una linea di questo tipo usando i cosiddetti punti di controllo. L'idea è di connettere due punti P_0 e P_1 definendo una curva

$$P(t) := (P_x(t), P_y(t)), \qquad \deg P_x, P_y \leq 3,$$

tali che $P(0) = P_1$, $P(1) = P_1$, e siano assegnate le derivate $P'(0)$ e $P'(1)$. Lo spazio dei polinomi di grado al più 3 ha dimensione 4, e le condizioni citate sono 4 condizioni lineari affini, e linearmente indipendenti. Dunque, la curva polinomiale che soddisfa questi requisiti è unica[2].

Per specificare le derivate si utilizzano dei punti di controllo, $\hat{P}_0$ e $\hat{P}_1$, tali che la derivata in 0 sia $\hat{P}_0 - P_0$, ed in 1 sia $P_1 - \hat{P}_1$. Ad esempio:

```
\begin{tikzpicture}
  \draw[->] (0,0) ..
    controls (0.5,0.5) and (1.5,0.5)
    .. (2,0);
\end{tikzpicture}
```

[2] a meno di casi degeneri.

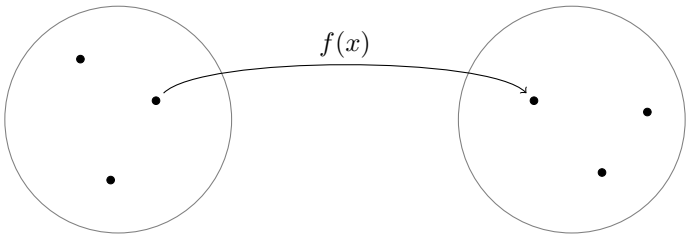
Nella figura sopra sono riportate anche le tangenti alla curva nei punti P_0, P_1 per facilitare la comprensione. Per aggiungere del testo possiamo creare dei nodi, che ci permettono di "attaccare del testo" ad alcuni punti. Ad esempio, potremmo modificare l'esempio precedente per indicare P_0 e P_1:

```
\begin{tikzpicture}
  \draw[->] (0,0) ..
    controls (0.5,0.5) and (1.5,0.5)
    .. (2,0);
    \node at (0,-0.25) {$P_0$};
    \node at (2,-0.25) {$P_1$};
\end{tikzpicture}
```

A questo punto abbiamo tutti gli strumenti necessari per realizzare il seguente disegno.

Esercizio 20

Si realizzi il seguente disegno utilizzando TikZ:



Ovviamente non è necessario che sia esattamente uguale, ma solo che siano presenti gli elementi corretti.

Una caratteristica utile di TikZ è che può utilizzare le capacità del TEX come linguaggio di programmazione. Ad esempio, supponisamo di voler disegnare 20 cerchi di raggio piccolo, uno di fianco all'altro. Potremmo ripetere lo stesso codice più volte, oppure utilizzare il seguente codice:

```
\begin{tikzpicture}
  \foreach \i in {0, 0.1, ..., 1.9} {
    \filldraw (\i, 0) circle (0.025);
  }
\end{tikzpicture}
```

Si utilizzi un ciclo per disegnare delle spirali come negli esempi seguenti; a questo scopo, potrebbe essere utile conoscere il comando `\pgfmathparse` che permette di valutare il risultato di un'operazione aritmetica, e `\pgfmathresult` che permette di ritornare il risultato dell'ultima operazione fatta. Si può combinare queste istruzioni con `\let`, che permette di definire una variabile uguale al risultato di un'operazione, ad esempio:

```
\pgfmathparse(1 + 3)
\let\w\pgfmathresult % Ora il comando \w vale 4
```

Ecco alcune spirali da provare a riprodurre: