LaTeX-workshop (Manual)

De Leidsche Flesch

7th September 2022

1 Introduction

Welcome to the manual for the LaTeX-workshop by De Leidsche Flesch. First look through this manual, before having a go at the exercises. The exercises will require you to look up information both in this manual and on the internet.

You will probably make a few mistakes, because *somewhere in your code* will write something slightly **wrong**. If this happens to you, do not worry, as this will keep on happening through your studies (and even career). Making mistakes now will only help you to become better at solving them later! Also remember that there often are multiple ways of writing the same thing in LaTeX, allowing you to pick your own preference.

If you're struggling with parts of LaTeX, be sure to use online sources to help you out. The LaTeX-wikibook, which can be found on `https://en.wikibooks.org/wiki/LaTeX`, is an easily accessible manual with extensive descriptions of the most common issues. Another useful tool is the info page for the online LaTeX-editor Overleaf, which can be found on `https://www.overleaf.com`.

If you can not find the command for a specific symbol, you can use `https://detexify.kirlelabs.org`. This website allows you to simply draw the symbol, and returns the appropriate command[1]

1.1 Starting off

For this workshop we recommend using the online editor `https://www.overleaf.com`, which supports working in the cloud. In practice it works very similar to Google Docs, as you can access your files on every device and all documents are stored online. This also handily prevents you from losing your homework! Overleaf also allows you to work with multiple people on the same document.

Besides Overleaf there exist many other LaTeX-editors, like MiKTeX and TeXmaker. On university computers, you can find MiKTeX via the start menu → all programs → MiKTeX 2.9 → TeXworks. You can also download it on your own device via `https://miktex.org/download`

[1] For android there even exists an app: `https://play.google.com/store/apps/details?id=website.marty.detexify`

2 New Document

A LaTeX-document consists of two parts. The first part is called the *preamble*. Everything in the preamble holds for the entire document. Examples of what to put here are LaTeX-packages that you would like to use, as well as the font and font size that you would like to use. The second part starts at `\begin{document}` and ends at `\end{document}`. In between is the content of your document.

You can also put comments to be read only by you or your colleagues in your .tex file, by writing a % in front of your comment.

Note that an 'empty' Overleaf document will automatically contain some basic commands, to have your document look like an actual document.

The exercises in this workshop will already have all commands mentioned in this section within them, so that you do not have to copy them from here. Be sure however to still read through them here, so that you roughly understand what each command does.

2.1 Preamble

1. At the top of the document put `\documentclass{article}`. This command tells the editor what type of document you will be working on, in this case an article. Besides this, it also determines the font size. In almost any situation, the article class is best suited one

2. Immediately below you put the packages that you want to use. These make it possible to add additional functions to LaTeX besides the standard ones. Add a package via the `\usepackage{packagename}` command.

   Below is a list of some of the most used packages:

   1). `\usepackage[a4paper]{geometry}` This assures that your file is formatted to A4 size.

   2). `\usepackage[english, dutch]{babel}` This makes sure that line breaking and translations are handled properly. An example is having 'Hoofdstuk' instead of 'Chapter'. English is also loaded, so that you can for example use English citations. The last language mentioned within square brackets is the main language.

   3). `\usepackage{parskip}` This makes sure paragraphs are handled nicely.

   4). `\usepackage{amsmath, amssymb}` These packages make formulas and other mathematical symbols look prettier.

   5). `\usepackage{graphicx}` This package allows you to add individual images.

   6). `\usepackage{enumerate}` This package makes it easier to create numbered lists.

   7). `\usepackage{url}` This makes your URLs readable.

8). `\usepackage{float}` This package allows you to smoothly insert images.

9). `\usepackage[hidelinks]{hyperref}` To create an index where when you click on a particular section, you are taken there immediately.

10). `\usepackage{color}` This allows you to write text in colour.

3. Next up, put the following commands in the preamble:

   `\title{your_title}`
   `\author{A. Uthor}`
   `\date{\today}`, where the command `\today` automatically adds the current date

4. Finally, you put down `\begin{document}` and `\end{document}`. In between these commands is where you write your actual text

5. Now compile the document

2.2 Text in document

2.2.1 Title and Author

Immediately after the `\begin{document}` command, write `\maketitle`. This makes sure that the information that you wrote in the preamble is put in the correct place. In this case, the title, author and date are put in the right spot.

2.2.2 Index

An index is easily made with the `\tableofcontents` command. After the index you usually want to start a new page, which is done via `\newpage`

2.2.3 Headers

A document of the 'article'-class makes use of sections, subsections, subsubsections and paragraphs. As an example, in the article that you are reading right now, the header 'New Document' is a section. As one would expect, it is created by writing `\section{New Document}`. Similarly, the 'Text in Document' header was made via `\subsection{Text in Document}`, and the 'Headers' header was made via `\subsubsection{Headers}`.

Paragraph Example Finally there is the paragraph, which is made using the command `\paragraph{Paragraph Example}`

2.2.4 Indentation

You can make indentations by writing an empty line in between text.

2.2.5 Enumerations and Lists

You can make an unnumbered enumeration using the `\begin{itemize}` command. A single item in the enumeration is created by typing `\item`. You then

simply type a space followed by the text you want to have for that item. You finish the enumeration using `\end{itemize}`.

- This is the first item in an unnumbered enumeration
- This the second
- This the last

You can also choose for a numbered enumeration, as was used prior in this manual, via the `\begin{enumerate}` command. Using the same structure as before, we then get

1. example 1
2. example 2

If you would like a different format, for example using letters instead of numbers, you can write `\begin{enumerate}[a.]`. This gives you:

a. example 1

b. example 2

In a similar fashion, you can use roman numerals via `\begin{enumerate}[i.]`

i. example 1

ii. example 2

3 Text

Just like in Microsoft Word, LaTeX allows you to present your text in different ways. Below are a few examples:

- **Bold**, type `\textbf{your_text}`
- *Italics*, type `\textit{your_text}`
- Colour, type `\textcolor{new_colour}{text}` or `{\color{new_colour}text}`
- Small, type `\small` followed by your text
- Large, type `\large`
- Larger, type `\Large`
- Even larger, type `\huge`
- Largest, type `\Huge`

After this, type `\normalsize` to continue in the default size.

4 Maths environment

4.1 Displaying equations

If you want to have any equations in your report, you need to use a "maths environment". There are several ways to enable one, the three most used ones are *math*, *displaymath* en *align*. *math* puts the mathematical symbols in the line[2], e.g. $a^n + b^n = c^n$. You can enter this enable in several ways, the most used one is to surround your formulas with `$`-symbols.

The *displaymath* environment is used to show maths symbols in the centre of a separate line, e.g.

$$\frac{\hbar^2}{2m}\nabla^2\Psi + V(\mathbf{r})\Psi = -i\hbar\frac{\partial\Psi}{\partial t}$$

You can enable this environment by surrounding your equations between square brackets: `\[...\]`.
The last of the three is the *align* environment, which is used to put maths centred on a separate line and also insert a reference to the equation.

$$\int_{-\infty}^{\infty} \frac{\cos x}{x^2+1} = \frac{\pi}{e} \tag{1}$$

$$\int_{-\infty}^{\infty} \frac{\cos x}{x^2+1} = \frac{\pi}{e} \tag{2}$$

To show your equations like this, you put `\begin{align` and `\end{align}` around your maths. If you want to remove the reference symbols at the end, you can use `\begin{align}` and `\end{align*}`. To put multiple equations on multiple lines, you type `\\` behind your equation. By using the `&`-sign you can align these different equations, most of the time you put it in front of the `=`-sign. If you'd like to experiment, try to find out what happens if you put multiple `&` in the same line.

4.2 Superscript and subscript

In the maths environment the subscript (`_`) and superscript (`^`) become available and are used frequently. We can for example display the sequence $a_1, a_2, \ldots, a_{n-1}$ by using the code `a_1, a_2, \dots, a_{n-1}` and the equation $a^b \cdot a^c = a^{b+c}$ by using `a^b \cdot a^c = a^{b+c}`. Note that you have to put { and } around the super- or subscript if it consists of more than one symbol.

4.3 Brackets

If an equation has brackets around for example a fraction, it's important to scale the brackets to the height of the term inside. This can be done by adding the `\left` and `\right` commands in front of the brackets:

$$\left|-\frac{1}{2}\right|^3 = \left(\frac{1}{8}\right)$$

[2]For technical reasons, this doesn't always work in headings. In that case you need to add the package *fixltx2e* to make it work.

is produced by the code:
`\[\left| -\frac{1}{2} \right| ^3 = \left( \frac{1}{8} \right)\]`

If you want to match the left bracket with a right bracket on a different line, you can use the invisible `\right.` command. This is useful for long definitions of sets:

$$\left\{ \text{s.t. } f^2(x) < \frac{\epsilon}{2} \right.$$

which is produced by
`\[\left.\text{s.t. } f^2(x) < \frac{\epsilon}{2} \right\} \]`
(Beware to escape the curly braces).

4.4 Roots and fractions

Roots and fractions are of course frequently used when displaying maths. To display the fraction $\frac{a}{b}$, you can type in a maths environment the code `\frac{a}{b}` and to display the root $\sqrt{a}$ you use the code `\sqrt{a}`. Cube roots and higher order roots like $\sqrt[n]{a}$ can be produced by `\sqrt[n]{a}`.

4.5 Special symbols

There are of course many different symbols used for maths. In this section we will discuss some, but there are of course a lot more. To find out quickly which command produces the symbol you want, you can use Detexify[3]. You can think of integrals, but also α, β or even ∂. The following table lists the most used symbols from the Greek alphabet.

	code		code		code		code		code		code
α	`\alpha`	β	`\beta`	γ	`\gamma`	δ	`\delta`	ϵ	`\epsilon`	ε	`\varepsilon`
ζ	`\zeta`	θ	`\theta`	λ	`\lambda`	μ	`\mu`	ξ	`\xi`	π	`\pi`
ρ	`\rho`	σ	`\sigma`	τ	`\tau`	ϕ	`\phi`	φ	`\varphi`	χ	`\chi`
ψ	`\psi`	Γ	`\Gamma`	Δ	`\Delta`	Λ	`\Lambda`	Φ	`\Phi`	Ψ	`\Psi`

Alongside the Greek alphabet there are also a lot of standard functions for which LaTeX commands exist. These standard functions are displayed correctly if the command is used. The most used ones are:

	code		code		code		code		code		code
arccos	`\arccos`	arcsin	`\arcsin`	arctan	`\arctan`	arg	`\arg`	sinh	`\sinh`	cos	`\cos`
cosh	`\cosh`	deg	`\deg`	det	`\det`	dim	`\dim`	tan	`\tan`	ker	`\ker`
lim	`\lim`	log	`\log`	max	`\max`	min	`\min`	sin	`\sin`		

Next there are also sums, products, integrals, etc. These symbols can adapt to the environment they are in. They become bigger when displayed in *dispaymath* than in *math*. These are the most used symbols:

	code		code		code		code
$\sum$	`\sum`	$\bigcap$	`\bigcap`	$\iint$	`\iint`	$\bigcup$	`\bigcup`
$\prod$	`\prod`	$\int$	`\int`	$\iiint$	`\iiint`	$\oint$	`\oint`

[3] `https://detexify.kirelabs.org/` or download the app

For $\sum$ as well as $\prod$ and lim it looks neat if when using *displaymath* the upper and lower bounds are directly above and below the quantifier. This is done using subscript and superscript. So you display the following sum

$$\sum_{n=1}^{\infty} \frac{1}{n}$$

by typing the following code `\sum_{n=1}^{\infty} \frac{1}{n}`. When using multiple integrals you should use the command `\limits`. This places the the bounds at the right position. Compare

$$\iint_D \text{ with } \iint\limits_D$$

which was produced using `\[\iint_D \text{ with } \iint\limits_D\]`. The second one is the correct way.

To end this section there are some more useful to know signs:

	code		code		code		code
$\pm$	`\pm`	$\times$	`\times`	$\setminus$	`\setminus`	$\cdot$	`\cdot`
$\mp$	`\mp`	$\gg$	`\gg`	$\ll$	`\ll`	$\top$	`\top`
$\leq$	`\leq`	$\in$	`\in`	$\subset$	`\subset`	$\subseteq$	`\subseteq`
$\cong$	`\cong`	$\geq$	`\geq`	$\approx$	`\approx`	$\mid$	`\mid`
$\equiv$	`\equiv`	$\sim$	`\sim`	$\neq$	`\neq`	$\leftarrow$	`\leftarrow`
$\Leftarrow$	`\Leftarrow`	$\leftrightarrow$	`\leftrightarrow`	$\Leftrightarrow$	`\Leftrightarrow`	$\mapsto$	`\mapsto`
$\hookleftarrow$	`\hookleftarrow`	$\rightarrow$	`\rightarrow`	$\Rightarrow$	`\Rightarrow`	$\updownarrow$	`\updownarrow`
$\Updownarrow$	`\Updownarrow`	$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`	$\downarrow$	`\downarrow`
$\Downarrow$	`\Downarrow`	$\cdots$	`\cdots`	$\vdots$	`\vdots`	$\ddots$	`\ddots`
$\forall$	`\forall`	$\exists$	`\exists`	∞	`\infty`	∇	`\nabla`
∂	`\partial`	$\ni$	`\ni`	$\supseteq$	`\supseteq`	$\Longrightarrow$	`\Longrightarrow`

For the comparing operators like $\leq$, $\in$ and $\subset$ you can produce the not version by putting `\not` in front of the operator. Then you get $\not\leq$, $\notin$ and $\not\subset$.

4.6 Matrices

There are different kinds of matrices in LaTeX. The matrix you'll use the most is the matrix between brackets. Other matrices which are also commonly used are the matrix between straight lines (for calculating a determinant) or a matrix without any lines. There are of course more variations, but you'll encounter them less often.

		LaTeX-code
Normal	$\begin{matrix} a & b \\ c & d \end{matrix}$	`\begin{matrix} a & b \\ c & d \end{matrix}`
Between brackets	$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$	`\begin{pmatrix} a & b \\ c & d \end{pmatrix}`
Between straight lines	$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$	`\begin{vmatrix} a & b \\ c & d \end{vmatrix}`

5 Tables

There are multiple ways to create tables in LaTeX. The most commonly used ones are the table (`\begin{tabular}[position]{layout}`) and the array (`\begin{array}[position]{layout}`). An array can only be created inside a maths environment.

Just like matrices the elements in a row are separated by a `&` and the rows are separated by a `\\`. To get a horizontal line in the table, you use `\hline` in the table code.

To determine the position, the following commands can be used.

Command	Result
`t`	The top of the table is aligned with the bottom of the line.
`b`	The bottom table is aligned with the bottom of the line.
`c`	The table is centred around the bottom of the line.

When determining the layout, you also determine the amount of columns the table consists of. To determine how these will look, you can use the following:

Command	Result
`l`	Adds a column where the text is left aligned.
`r`	Adds a column where the text is right aligned.
`c`	Adds a column where the text is centred.
`p{x}`	Adds a column where the text is stretched to fill the width `x`.
`*{x}{layout}`	Repeats the structure in `layout` `x` times.
|	Adds a vertical line.
||	Adds a double vertical line.
`@{text}`	Uses `text` as vertical line.

Example

Time in months	Amount of rabbits	Calculation
0	1	1
1	2	1+1
2	3	1+2
3	5	2+3
4	8	3+5
5	13	5+8
6	21	8+13
⋮	⋮	⋮

This table can be created using:

```
\begin{tabular}[c]{| l | c | r |}
\hline
Time in months & Amount of rabbits & Calculation \\
\hline
0 & 1 & 1 \\
1 & 2 & 1+1  \\
2 & 3 & 1+2 \\
```

```
3 & 5 & 2+3 \\
4 & 8 & 3+5 \\
5 & 13 & 5+8 \\
6 & 21 & 8+13 \\
$\vdots$ & $\vdots$ & $\vdots$ \\
\hline
\end{tabular}
```

6 Include images

It's also possible to include images using LaTeX. First you need to include `\usepackage{graphicx}` at the top of your code. This package let's you include the images. Next you put `\includegraphics{image_name}` to actually show the image. The image needs to be in a (sub)folder where you also keep you LaTeX file. Note: LaTeX can only read `.pdf`, `.png` or `.jpg` files.

To adjust the image size, you can use `\includegraphics[scale=0.5]{image_name}`. Instead of scaling the image, you can also set a specific width using `[width=0.9\textwidth]`.

6.1 Position the image

It's often complicated to get an image where you want it to appear. To accomplish this, we'll put the image in a floating environment. We do this by typing:

```
\begin{figure}[]
\centering
\caption{Image caption}
\includegraphics[width=0.5\textwidth]{image_name}
\end{figure}
```

You can add square brackets at `\begin{figure}[]` to adjust the placement of the figure. The following options are good to know:

command	result
h	Try to place the environment here.
t	Try to place the environment at the top of a page.
b	Try to place the environment at the top of a page.
!	Force one of the previous options.

As you can see in the code above, it's also possible to add a caption to your image. To do so, you use the command `\caption{description}` in the floating environment. The caption will then appear underneath the image. You can also put the caption text next to the image by using the following code:

```
\begin{SCfigure}
\caption{Short description of the image}
\includegraphics[width=0.5\textwidth]{image_name}
\end{SCfigure}
```

You can also wrap the text around your image. That way the image appears in the text itself. First you need to import the package: `\usepackage{wrapfig}`. Then to actually show the image, you can use the code:

```
\begin{wrapfigure}{r}{0.5\textwidth}
\includegraphics[width=0.45\textwidth]{image_name}
\caption{Some information about the image}
\end{wrapfigure}
```

In this example you can control the position of the figure within the two curly braces, the example uses a `r`. The most common options for the placement are:

command	result
r	Align to the right side of the page.
l	Align to the left side of the page.
R	Align right while floating, to avoid splitting on a page break.
L	Align left while floating.

Then there is also the possibility to display several images next to or underneath each other. This can be useful for instance when comparing different graphs. First you need to import the following package: `\usepackage{subfig}`. Then you can use the following code to create a grid of four images, where the images will appear in two rows of two images.

```
\begin{figure}
\centering
\subfigure[Description 1]{\includegraphics[width=0.4\textwidth]{graph1}
~
\subfigure[Description 2]{\includegraphics[width=0.4\textwidth]{graph2} \\
\subfigure[Description 3]{\includegraphics[width=0.4\textwidth]{graph3}
~
\subfigure[Description 4]{\includegraphics[width=0.4\textwidth]{graph4}
\caption{Graphs of ...}
\end{figure}
```

7 Include code

The easiest way to include code is by using the commands from the package *listings*. This provides the environment *lstlisting*, so you can put your script or code between `\begin{lstlisting}` and `\end{lstlisting}`. To get the correct syntax highlighting, you need to specify which programming language you are using; e.g. when using C++ you have to put `\lstset{language=C++}` in you preamble (*above* `\begin{document}`).

The example LaTeX-code

```
\begin{lstlisting}
  int main (int argc, char** argv) {
    std::cout << "Hello world" << std::endl;
    return 0;
  }
\end{lstlisting}
```

gives the following output

```
int main (int argc, char** argv) {
  std::cout << "Hello world" << std::endl;
```

```
    return 0;
}
```

You can also just include the code file itself by typing:

`\lstinputlisting{file.extension}`

8 References

For almost all numbered things (sections, equations, figures, etc.) you can use the command `\label{aNameHere}`. Once you have put your `\label{...}` at the right place, you can refer to it using `\ref{aNameHere}`. You could also use `\pageref{aNameHere}` to refer to the page the label is on. To create a footnote, you can write in your text `\footnote{This is a footnote}`[4].

For more extensive references/citations, you can write

```
\begin{thebibliography}{99}
\bibitem{images} Wikibooks, \emph{LaTeX/Floats, Figures and Captions
--- Wikibooks{,} The Free Textbook Project}.
\url{http://en.wikibooks.org/wiki/LaTeX/Floats,_Figures_and_Captions}
\end{thebibliography}
```

Now you can refer to e.g. a url with more info on images using `\cite{images}` [2].

For references to work properly it's important to compile the code twice. Otherwise there will be question marks or old numbering at the reference.

9 Other interesting packages

There are lots of other LaTeX packages that can make your life easier. Look them up on the internet (`https://ctan.org/pkg`).

- `\usepackage{fancyhdr}` For intelligent headers and footers on your page.
- `\usepackage[section]{placeins}` Stops floating objects from moving to another section. You can also define a barrier yourself for all floats using `\FloatBarrier`
- `latexmk` Can automate the whole compiling process for you.
- `\usepackage{sidecap}` To have captions next to your floats instead of above or below.
- `\usepackage{beamer}` To create beamer presentations (PowerPoint).
- Did you know `\\`, the command for a new line, has an optional argument? Try for instance `\\[5cm]`.

[4] This is a footnote

- The LION has its own LaTeX class for writing theses [3]. This does a lot of work for you, e.g. the layout of the front page.

References

[1] http://en.wikibooks.org/wiki/LaTeX

[2] Wikibooks, *LaTeX/Floats, Figures and Captions — Wikibooks, The Free Textbook Project*. http://en.wikibooks.org/wiki/LaTeX/Floats,_Figures_and_Captions

[3] https://www.physics.leidenuniv.nl/bachelor/80-education/606-education-thesis-templates