Using LaTeX

Scott Harper

Abstract

LaTeX is the standard means of preparing a mathematical document. This guide is intended to provide an introduction to using LaTeX and highlight some of its key features. A miscellany of various tips and tricks has also been included. Once some basic familiarity with LaTeX has been obtained, it is recommended that the reader consult the associated `.tex` file in parallel with reading this guide.

1 Introduction

LaTeX is the standard means of preparing a mathematical document. This is for some very good reasons: it produces attractive looking documents; it is free; and it makes some potentially difficult tasks very easy. However, if you have never used LaTeX before, then you will have to allow a little time to get used to using it. As with many things, LaTeX is best learned through practice. As such, I believe that the best approach to LaTeX is to start writing a document and look up the things you don't know how to do. However, there are two main problems with this approach alone. How do you learn the very basics? And how do you handle some more obscure issues when you become more advanced? This guide is aimed at these two ends of the learning process.

I hope that by reading the main body of this guide, in parallel with looking at the associated `.tex` file and trying out the features highlighted, you should quite quickly be comfortable with how LaTeX works. After this, when you come to use a particular command and cannot remember the details you can quickly look these details up in this guide or (probably more quickly) online. The features set off by a black vertical line are not intended to be read on the first reading, but rather these are advanced features or typesetting tips, which are the result of personal experience. I hope that collecting these titbits together in one document will serve as a useful resource for anyone who comes across similar problems which are less easily solved by a quick internet search, or for someone who wants to help polish their LaTeX or typesetting technique.

Unlike producing working LaTeX code, typesetting is necessarily a more subjective topic; however, at times it is difficult to separate these two ideas. Where comments on typesetting are made, these reflect my personal taste and I expect some to disagree with me. I have attempted to maintain consistency with the *New Oxford Style Manual* [3] (which incorporates the current incarnation of *Hart's Rules*).

Three caveats: *I do not seek to be authoritative.* To the best of my knowledge the content of this guide is correct, but I do not doubt that it may contain errors or that I may have missed more efficient or elegant methods. Any errors or suggestions would be most welcome. *I do not seek to be comprehensive.* Even if a LaTeX guide could be comprehensive it would be to long to be useful. I have aimed to introduce the core features of LaTeX but this will have undoubtedly been skewed by personal interests. The selection of titbits is a product of my (pretty short) experience of LaTeX. *I do not seek to be technical.* I am not a computer scientist, nor do I claim to have a great knowledge of the inner workings of LaTeX. This guide unashamedly reflects this and, thus, is written from a user-focussed perspective.

2 Getting Started

2.1 Getting LaTeX

LaTeX is a mechanism by which one writes a *source file* – a plain text file with file extension `.tex` – and transforms this into a high quality document (usually a PDF). The source file contains both the content of your document and instructions on how this content should be presented. Therefore, you will need an *editor* for writing the `.tex` file and a *compiler* to create final document.

The fastest way to begin doing this – but one which is not free of disadvantages – is the online resource Overleaf. In your browser there will be two panels: in one you write your `.tex` file and the other will show the compiled PDF when you click "Recompile". Overleaf has capabilities for multiple users sharing files and working on the same document at once.

In the long run, working on your desktop, rather than the browser, is probably recommended. In this case, you should download a LaTeX editor – for example, TeXstudio – and a compiler. For the latter, I recommend that Mac users download MacTeX [10] and Windows users download MiKTeX [11]. Of course, if you prefer you can work in the terminal. Here you can use your favourite text editor to write the `.tex` file, then you can compile using `pdflatex`.

2.2 Your first document

Let us begin by explaining how to produce your first LaTeX file. Open your LaTeX editor (for example, TeXstudio, Overleaf or vim) and write the following basic LaTeX file.

```
\documentclass{article}

\begin{document}
This is a very simple document.
\end{document}
```

Save this file as `first.tex`, then compile this document. Your LaTeX editor's manual should make it clear how to do this: in Overleaf click the "Recompile" button and in TeXstudio click the play button. More files will have appeared in the folder where you saved the file `first.tex`. One of these files will be `first.pdf`. This compiled document should include "This is a very simple document." and nothing else (other than a page number).

If you managed this, then you can now write any document, provided that it consists only of text and provided you don't want to change the formatting of the document. In Section 3, we look at how to do much more exciting things. However, we will conclude this section by explaining some of the essential background to LaTeX.

2.3 The basics

2.3.1 The outline of a LaTeX document

Every LaTeX file begins with the line `\documentclass{?}` where `?` is your choice of the *document class*. Possible document classes include `article`, `report` and `book`. For the purpose of this guide we may assume that the document class is `article`. Other document classes, unsurprisingly, provide features suitable for producing other classes of documents (see [14, Document Structure/Document classes]).

Every LaTeX file contains the line `\begin{document}` and later `\end{document}`. All lines between these two, contribute directly to output file. The lines before these are known as the *preamble*, where general commands are placed which affect the entire document.

Some features of LaTeX require extra packages to be loaded. To include a package named `?` include `\usepackage{?}` in the preamble. Multiple packages can be loaded in the same use of the command `\usepackage`. For example, I recommend loading the following AMS packages in all documents:

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

2.3.2 Comments

To make *comments* (notes in the `.tex` file which do not influence the compiled file) preface the text with `%`. All text on a line after a `%` will not be taken into account when the document is compiled. For example, the following LaTeX file would produce the same compiled document as the one in Section 2.2.

```
\documentclass{article}

\begin{document}
This is a very simple document. % A comment
% Another comment
\end{document}
```

2.3.3 Spaces and paragraphs

LaTeX considers multiple spaces the same as one. To manually add an extra space, use `\`. For example:

```
Spot the  extra space \ in this.
```

Spot the extra space in this.

LaTeX will not take into account the starting of a new line. To begin a new paragraph leave a blank line. LaTeX considers several blank lines the same as one, so you can use blank lines to make your LaTeX file easy to read.

Typesetting tip There are places where you will want to avoid a line breaking. For example, it would be undesirable for "10 km" or "Dr Frankenstein" or "Theorem 42" to be broken over two lines. In these situations, use a non-breaking space given by `~`. For example, `10~km` will not break over two lines.

Typesetting tip By default LaTeX inserts a larger space between sentences than between words. This is the traditional method of spacing but has been abandoned by many authors. (For example, it is the Oxford style to have the same size of space between words and sentences.) To make interword spaces and intersentence spaces equal, include the command `\frenchspacing` in the preamble.

LaTeX believes that a sentence has ended when a lower case letter then a full stop appear. (It chooses not to believe that `D.` ends a sentence to ensure that `John D. Smith` has interword spacing between `D.` and `Smith`.) Therefore, if you choose to have longer intersentence spaces, then you must take the following precautions. First, if a sentence ends with a character that is not a lower case letter, then an intersentence space must be manually added using the command `\@`. For example, "I want a PhD. I am a student." should be typeset as `I want a PhD.\@ I am a student.` Second, if a lower case letter then a full stop occurs in the middle of a sentence, then a interword space must be manually added using the command `\`. For example, the sentence "I want a doctorate, e.g. a DPhil." should be typeset as `I want a doctorate e.g.\ a DPhil.`

2.4 The title

The *title* section of a LaTeX document is generated in two steps. First, include the information about the document in the preamble. For example

```
\title{Alice's Adventures in Wonderland}
\author{Lewis Carroll}
\date{\today}
```

sets the title as "Alice's Adventures in Wonderland", sets the author as "Lewis Carroll" and sets the date as today's date. Second, include the command `\maketitle` after the `\begin{document}` line to print the title information.

2.4.1 Symbols

To insert *inverted commas* use ` to open and ' to close. For example, `` `single' `` gives 'single' and ``` ``double'' ``` gives "double".

To insert a *hyphen* use -; for example, `ice-cream` gives "ice-cream". To insert a *dash* use --; for example, `9am--5pm` gives "9am–5pm" and `John -- my dad -- likes cats` gives "John – my dad – likes cats".

Typesetting tip For pauses and parenthesis, the *em dash* (— given by ---) was traditionally used instead of the *en dash* (– given by --). Most US writers and many UK writers now use the en dash for all dashes. However many UK writers still use em dashes but leave no space either side of the dash (the latter is done automatically in LaTeX). This is the Oxford style. For example, `John---my brother---likes cats` gives "John—my brother—likes cats".

As we have already explained, the symbol % has a special meaning in LaTeX. Therefore, simply typing % will not lead to a percentage sign in the compiled file. Instead we must type \%. For this reason, we say that % is a *reserved symbol*. The reserved symbols and how to write them are given below.

%	^	_	~	$	&	#	{	}	\
`\%`	`\^{}`	`\_`	`\~`	`\$`	`\&`	`\#`	`\{`	`\}`	`\textbackslash{}`

Other symbols, in addition to these reserved ones, can be included in a LaTeX document by writing an appropriate *command* beginning by with a backwards slash. For example `\dots` provides an ellipsis . . . , `\LaTeX` provides the LaTeX symbol and `\pounds` provides the pounds sterling symbol £. To find out the command required for a particular symbol, use the Detexify app [9] or consult *The Comprehensive LaTeX Symbol List* [4]. Observe that most symbols when they are included *gobble* the space that follows them, so an additional space must be manually added after them. Therefore, in the middle of a sentence the LaTeX symbol is actually typeset as `\LaTeX\`.

There are many commands for including letters from *other alphabets* and *diacritics*. Examples of these are given below.

ł	ø
`\l{}`	`\o`

à	é	û	š	ç	ñ	ö	ő
``\`{a}``	`\'{e}`	`\^{u}`	`\v{s}`	`\c{c}`	`\~{n}`	`\"{o}`	`\H{o}`

Note the difference between the final two diacritics. The latter, the Hungarian double acute, is the diacritic used in the name Paul Erdős.

To place a diacritic above an i or j, use the dotless versions of these letters provided by `\i` and `\j`; for example `na\"{\i}ve` gives "naïve".

3 How to Use LaTeX

3.1 Text formatting

To format text in LaTeX, call a particular command then enclose the text you want to format in braces `{...}`. The following commands are available:

- `\textbf{bold font}` gives **bold font**
- `\textit{italic font}` gives *italic font*
- `\texttt{teletype font}` gives `teletype font`
- `\textsc{small caps}` gives SMALL CAPS

To emphasise a word, use the command `\emph`; for example, `\emph{maths}` gives *maths*. This is preferable to using `\textit{maths}` since it separates the content from the appearance. By default `\emph` applies `\textit`, but if you later decide you want to emphasise your text by emboldening it rather than italicising it, then you can change the definition of `\emph` and all your emphasised text will change at once.

The document's *text size* can be altered by passing an optional argument to the the `\documentclass` command. For example, `\documentclass[14pt]{article}` will produce a document where the text has size 14pt. Size of text within the document can be changed relative to this default size in the following way. Notice that the arrangement of the command and braces differs from above.

- `{\tiny far too small}` gives far too small
- `{\normalsize standard text}` gives standard text
- `{\Huge far too big}` gives far too big

See [14, Fonts/Font styles] for a full list of available sizes. There are not many occasions for manually changing the size of text. For example, section headings are bigger than body text and footnotes are smaller.

Different *colours* can be used in a LaTeX document. However, in general, there are better methods of emphasis since you cannot guarantee that your document will be read in colour. To use colour in LaTeX load the `xcolor` package. Then the `\textcolor` command can be used to provide coloured text. For example, `\textcolor{red}{cats}` will give cats ["cats" is in red]. The first argument of `\textcolor` specifies the colour and the second argument is the text to be coloured. There are several ways of specifying a colour. Many basic colours are available by name (such as `red` in this example), and more colours can be called in this way by loading the `xcolor` package with the optional argument `dvipsnames`. Alternatively, specific colours can be defined in the preamble, according to a choice of colour models, then used later in the document. For more information see [14, Colors].

3.2 Sections and cross-referencing

To begin a new *section* of a document entitled "Proof of the Riemann Hypothesis", include the line `\section{Proof of the Riemann Hypothesis}`. Sections are automatically numbered. To create an unnumbered section use the command `\section*`. Sections can have subsections and subsections can have subsubsections. These are included using the `\subsection` and `\subsubsection` commands, and these are numbered according to the section (and subsection, if appropriate) they are contained in. For example, observe the numbering in this document.

To include a *table of contents* include the command `\tableofcontents` in the appropriate place in the document. Yes, it is that easy!

LaTeX makes light work of labelling components of the document and referring back to them. To label a section, a figure or a table (see Section 3.6), or theorem (see Section 3.3.2) use the command `\label`. For example `\section{Cats} \label{sec:cats}` creates a section entitled "Cats", then gives that section the label `sec:cats`. Labelling a section does not print anything in the document; rather, it gives the section an internal name to allow us to refer to it later. To refer back to this section use the command `\ref`. For example, `Section~\ref{sec:cats}` gives "Section 6", if the section on cats is the sixth section. If we later reorder our sections and the section on cats becomes the fourth section, then `Section~\ref{sec:cats}` will update and change to "Section 4". (Observe that we use a non-breaking space between "Section" and the number, see Section 2.3.3.)

Advanced The `imakeidx` package allows you to make an *index*. First, include the command `\makeindex` in the preamble, and include the command `\printindex` where you wish the index to appear. Whenever you want a position in the document to be indexed use the command `\index`. For example, `\dots we study cats\index{cat} \dots` will produce an item in the index called "cats" and will give the page number where that except appears. Observe that the command `\index` does not print anything in the main body of the document.

You can create subentries in an index, "see also" entries, main entries and multiple page entries. You can also also format the index in many ways. For further details see [14, Indexing].

Unless your LaTeX compiler has been set up especially, an extra step is required at the compilation stage. Compile the document; this will produce a `.idx` file. Now run the `makeindex` on the `.idx` file to produce a `.igl` file. Compile the document again; the index will appear.

3.3 Environments

3.3.1 Lists

The `itemize` environment allows bulleted lists; for example:

```
\begin{itemize}
\item First item
\item[*] Second item
\item Third item
\end{itemize}
```

- First item

* Second item

- Third item

Note that the `itemize` environment begins with `\begin{itemize}` and ends with `\end{itemize}`. This is the general pattern for an environment. Each item in the list is preceded by the command `\item`. This is the general pattern for a list environment.

In an `itemize` environment, by default, the symbol is a round black bullet, but the symbol for each of the items can be changed by using the optional argument of the `\item` command.

The `enumerate` environment allows numbered lists.

```
\begin{enumerate}
\item First item
\item Second item
\item Third item
\end{enumerate}
```

1. First item
2. Second item
3. Third item

By default the numbering is in Arabic numerals, but the numbering system can be changed by using the optional argument of the `enumerate` environment (if the `enumitem` package is loaded with the optional argument `shortlabels`). The argument `(i)` provides lower case roman numerals in brackets and `A.` provides upper case letters followed by a full stop. See Section 6 of the `enumitem` package information for more details.

3.3.2 Mathematical environments

One can easily construct environments for typesetting definitions, the statements of theorems, proofs and much beyond.

First, set up, in the preamble, the environments which will be used. A typical setup is the following:

```
\theoremstyle{definition}
\newtheorem{definition}{Definition}[section]
\newtheorem{remark}[definition]{Remark}
\newtheorem{example}[definition]{Example}
\newtheorem*{example*}{Example}

\theoremstyle{plain}
\newtheorem{lemma}[definition]{Lemma}
\newtheorem{proposition}[definition]{Proposition}
\newtheorem{theorem}[definition]{Theorem}
\newtheorem*{theorem*}{Theorem}
\newtheorem{corollary}[definition]{Corollary}
```

Before explaining how this setup works, observe that we can now include, for example, a definition, lemma and example as follows by using the now-available `definition`, `lemma` and `example*` environments.

```
\begin{definition}
A positive integer is \emph{prime} if it has no proper non-trivial divisors.
\end{definition}

\begin{lemma}
Every positive integer is a product of primes.
\end{lemma}

\begin{example*}
$60 = 2^2 \cdot 3 \cdot 5$
\end{example*}
```

Definition 3.1. A positive integer is *prime* if it has no proper non-trivial divisors.

Lemma 3.2. *Every positive integer is a product of primes.*

Example. $60 = 2^2 \cdot 3 \cdot 5$

Note that the text in the body of the definition and example are in upright font whereas the lemma is italicised. This is a consequence of the `definition` and `example*` environments being defined after the line `\theoremstyle{definition}` was called and `lemma` after `\theoremestyle{theorem}`.

Note that the definition and lemma are numbered as the section number then a full stop and then the result within that section. This is because the `definition` environment was defined to be numbered according to the section (this explains the optional argument `[section]` in the definition of the `definition` environment), and the `lemma` environment was defined to be numbered in the same way as the `definition` environment (this explains the optional argument `[definition]` in the definition of the `lemma` environment). As with sections, mathematical environments can be labelled and referred to later (see Section 3.2).

Note that the example is not numbered. This is because the `example*` environment was defined using the `\newtheorem*` command, which creates unnumbered theoremlike environments.

For more information on creating theoremlike environments see [14, Theorems].

Typesetting tip If you begin a theorem (or a similar environment) with a list (such as `itemize`), then the first item will be placed on the same line as the theorem heading. However, this is not particularly attractive. To force the items to begin on the following line, insert `\quad` after the `\begin{theorem}` line. For example, this code

```
\begin{theorem*} \quad
\begin{enumerate}[(i)]
\item $0+0=0$
\item $1\cdot1=1$
\end{enumerate}
\end{theorem*}
```

produces

Theorem.

(i) $0 + 0 = 0$

(ii) $1 \cdot 1 = 1$

The `proof` environment provides an environment which begins with the word proof and ends with a QED square; for example:

```
\begin{proof}
$1+1=2$.
\end{proof}
```

Proof. $1 + 1 = 2$. □

The optional argument of the `proof` environment allows the introduction to be something other than "Proof."; for example:

```
\begin{proof}[Proof of Fermat's Last Theorem]
I have discovered a truly marvelous proof of this,
which this margin is too narrow to contain.
\end{proof}
```

Proof of Fermat's Last Theorem. I have discovered a truly marvelous proof of this, which this margin is too narrow to contain. □

3.4 Macros

To avoid repeatedly typing the same commands to include symbols or format text, you can create *macros*. Macros are written in the preamble and apply to the entire document. To create a macro use the `\newcommand` command which takes two arguments: the name of the command and the definition of the command.

For example, if the macro

```
\newcommand{\atlas}{\textsc{Atlas}}
```

is included in preamble, then throughout the document the command `\atlas` produces the ATLAS.

Macros can also take arguments. Here `\newcommand` takes an optional argument between the name and the definition which is the number of arguments the new command will have. Then each of these arguments are referred to as `#1`, `#2`, etc. in the definition. For example,

```
\newcommand{\bi}[1]{\textbf{\textit{{#1}}}}
```

creates a command which takes one argument and returns that argument in bold italics. For example, throughout the document, `\bi{group theory}` will produce ***group theory***.

One cannot create a new command called `\cats` using `\newcommand` if there is already a command called `\cats`. In this case, one should use the command `\renewcommand`, whose syntax is otherwise the same as `\newcommand`. However, one should be very careful not to overwrite commands which are already being made use of in the document.

3.5 Mathematics

3.5.1 Basic input

Normal writing, as we have discussed so far, is carried out in *text mode*. To include mathematical notation, use *math mode*. There are two ways to enter math mode, which lead to two different styles of presentation: *inline* and *display*. For inline math mode surround the expressions by `$` and `$`, and for display math mode surround the expressions by `\[` and `\]`. For example, `$z$` gives z inline whereas `\[ y = ax+b \]` gives

$$y = ax + b$$

displayed. Notice that letters, numbers and keyboard symbols work as expected in math mode.

Advanced The old TeX mechanism for including maths inline was `$...$` and for display was `$$...$$`. The LaTeX mechanism for inline is `\(...\)` and for display is `\[...\]`. There are problems associated with using `$$...$$` for display mathematics, but many people still use `$...$` for inline mathematics.

Use `\sqrt` to write roots. For example, `$\sqrt{9}=3$` gives $\sqrt{9} = 3$ and `$\sqrt[3]{9-1}=2$` gives $\sqrt[3]{9-1} = 2$. Use `\frac` for fractions. For example, `\[ \frac{1 + \sqrt{5}}{2} \]` gives

$$\frac{1+\sqrt{5}}{2}$$

and `$\frac{1}{2} = 0.5$` gives $\frac{1}{2} = 0.5$. Use `\binom` for binomial coefficients. For example,

```
\[ \binom{n}{k} = \binom{n}{n-k} \]
```

gives

$$\binom{n}{k} = \binom{n}{n-k}$$

Advanced Notice the size of `\frac{a}{b}` differs between inline maths and display maths. To provide a display maths sized fraction in inline maths (or in display maths) use `\dfrac{a}{b}`, and for an inline maths sized fraction use `\tfrac{a}{b}`. For example, here is the inline and display output for `\frac`, `\tfrac` and `\dfrac`: $\frac{1}{2}, \tfrac{1}{2}, \dfrac{1}{2}$ and

$$\frac{1}{2}, \tfrac{1}{2}, \dfrac{1}{2}.$$

Use `^` to insert *superscripts* and `_` for *subscripts*. For example, `$x_n=n^2$` gives $x_n = n^2$. To include longer expressions in a superscript or subscript, enclose the expression in braces `{...}`. For example, `$y_{ij}=2x^{10}$` gives $y_{ij} = 2x^{10}$.

Advanced There may be occasion for superscripts and subscripts to the *left* of the main letter. The makeshift way to achieve this is as follows: $_GM$ written as `$_GM$`. However, if the presuperscript or presubscript follows another symbol one must include empty braces first to avoid the superscript attaching itself to the previous symbol; compare the correct result $G = {}^2F_4$ of `$G={}^2F_4$` with the incorrect result $G =^2 F_4$ of `$G=^2F_4$`.

This issue can be avoided by, instead, using the `\prescript` command from the `mathtools` package. This bespoke method has the added advantage of ensuring the prescripts are arranged appropriately around the main symbol. Here `\prescript{a}{b}{X}` gives $\prescript{a}{b}{X}$ and can be used to write expressions like the following

$$\prescript{A}{Z}{X} \to \prescript{A-4}{Z-2}{Y} + \prescript{4}{2}{\alpha}.$$

To include some text in math mode, use the `\text` command. For example,

```
$\{ n^2 \mid n \ \text{is odd} \}$
```

gives $\{n^2 \mid n \ \text{is odd}\}$. Observe that a manual space using `\` was inserted before the `\text` command since all space around text in math mode is gobbled.

To display a *numbered equation*, use the `equation` environment rather than using display math mode. As with sections, equations can be labelled then referred to later, see Section 3.2. When an equation is referred to use the command `\eqref`, rather than `\ref`, to ensure that the reference is enclosed in brackets. For example:

```
\begin{equation} \label{eq:euler}
1^{i+\pi} = e^{0}
\end{equation}

By Euler's equation \eqref{eq:euler}, \dots
```

$$1^{i+\pi} = e^0 \tag{1}$$

By Euler's equation (1), ...

To label an equation by a symbol rather than a number, use the `\tag` command. For example:

```
\begin{equation} \label{eq:fermat}
3987^{12} + 4365^{12} = 4472^{12} \tag{$\dagger$}
\end{equation}

By Homer's theorem \eqref{eq:fermat}, \dots
```

$$3987^{12} + 4365^{12} = 4472^{12} \tag{†}$$

By Homer's theorem (†), ...

To print a series of displayed equations which are *aligned* at the equals sign (or similar), use the `align` environment. Each row is ended with `\\`, and on each row `&` is placed to indicate where the equations should align. For example:

```
\begin{align}
\int \sec x \, dx &= \int \frac{\sec x(\sec x + \tan x)}{\tan x + \sec x}\,dx \\
&= \int \frac{(\sec x)^2 + \sec x \tan x}{\tan x + \sec x}\,dx \label{eq:ex} \\
&= \log|\tan x + \sec x| + C
\end{align}
```

$$\begin{aligned}
\int \sec x \, dx &= \int \frac{\sec x(\sec x + \tan x)}{\tan x + \sec x}\,dx && (2)\\
&= \int \frac{(\sec x)^2 + \sec x \tan x}{\tan x + \sec x}\,dx && (3)\\
&= \log|\tan x + \sec x| + C && (4)
\end{aligned}$$

By default, the lines in an `align` environment are numbered. Use the `align*` environment to omit the numbering. As with the `equation` environment, a row can be labelled to be referred to later.

Typesetting tip If a proof ends at the end of an `align` environment then the QED square will not appear on the final line of this environment but on the next blank line. This is not appealing. Therefore, in this case, one should place the command `\qedhere` within the `align` environment at the end of the proof to ensure that the QED square appears on the correct line. We present an example of this.

```
\begin{proof}
Let $x, y \in k$. For all $1 \leq i \leq p$ the binomial coefficient
$\binom{p}{i}$ is divisible by $p$. Therefore,

\begin{align*}
(x+y)^p &= \sum_{i=0}^{p}\binom{p}{i}x^iy^{p-i} \\
        &= x^p + y^p. \qedhere
\end{align*}

\end{proof}
```

Proof. Let $x, y \in k$. For all $1 \leq i \leq p$ the binomial coefficient $\binom{p}{i}$ is divisible by p. Therefore,

$$\begin{aligned}
(x+y)^p &= \sum_{i=0}^{p}\binom{p}{i}x^iy^{p-i} \\
&= x^p + y^p. \qquad \square
\end{aligned}$$

Advanced On occasions one might want to number just one line of an aligned display. This can be done in an `align*` environment, by using the `\tag` command together with a manual manipulation of the LaTeX numbering system. We present an example of this without explanation. See [14, Counters] for more information on numbering and LaTeX counters.

```
\begin{align*}
g^{-1}hg &= ghg \\
        &= ghghh \stepcounter{equation}\tag{\theequation}\label{eq:trick} \\
        &= h
\end{align*}
```

$$\begin{aligned} g^{-1}hg &= ghg \\ &= ghghh \qquad (5) \\ &= h \end{aligned}$$

3.5.2 Common symbols

Writing mathematics requires many more symbols than those available on the keyboard. These can be included with special commands in math mode. For example `\gamma` and `\Gamma` give γ and Γ. All Greek upper and lower case letters can be provided in this way; however, if an upper upper case Greek letter matches an upper case Latin letter, then the command for the Greek letter is not defined (e.g. `\Alpha` is not defined since upper case alpha is identical to the Latin A). For some Greek letters there are optional variants. For example I prefer ε and φ provided by `\varepsilon` and `\varphi` over the default ϵ and ϕ provided by `\epsilon` and `\phi`.

Standard *mathematical symbols* can be inserted in a similar way. (More obscure symbols may require extra packages to be loaded.) For example:

```
\[ |x-a| \leq \delta \implies |f(x)-f(a)| \leq \varepsilon \]
```

$$|x-a| \leq \delta \implies |f(x)-f(a)| \leq \varepsilon$$

The negation of a symbol can (usually) by achieved by preceding the symbol by `\not`. For example, `\not\in` gives $\not\in$. If you cannot guess the command for your required symbol, then you can use the Detexify app [9] or consult *The Comprehensive LaTeX Symbol List* [4].

Big mathematical symbols – for example for integrals, sums, products, unions, intersections – take the form `\symbol_{lower}^{upper}` and these commands require a value for their lower and upper bounds (even if you wish for one of these to be empty). For example, `$\bigcup_{i \in I}^{} U_i$` gives $\bigcup_{i \in I}^{} U_i$ and

```
\[ \sum_{n=1}^{\infty}\frac{\pi^2}{6}\]
```

gives

$$\sum_{n=1}^{\infty}\frac{\pi^2}{6}$$

Note the difference between display and inline. For more see [14, Mathematics/Sums and integrals].

Advanced To include multiple lines of text under a large operator, use the `\substack` command. For example, `\[ \sum_{\substack{i=0 \\ i \not\in A}}^{n} i^2 \]` gives

$$\sum_{\substack{i=0 \\ i \not\in A}}^{n} i^2$$

Typesetting tip When defining a function use the correct arrows and colon. The expression

```
\[ f \colon \mathbb{R} \to \mathbb{R} \quad x \mapsto x^2 \]
```

$$f \colon \mathbb{R} \to \mathbb{R} \quad x \mapsto x^2$$

should be typeset using the command `\to` for the arrow $\to$ between sets and the command `\mapsto` for the arrow $\mapsto$ defining how the function acts on elements. In addition, the colon between the function name and the first set should be given using `\colon` and not `:` since the latter will provide the incorrect space around the colon.

Typesetting tip When writing a set using set-builder notation be sure to use the correct commands for the colon (use `:` and not `\colon`) or pipe (use `\mid` and not `\div`). (Whether one uses a colon or a pipe is a matter of personal taste.) For example, both of the following sets are correctly typeset:

```
\[ \{ x \in \mathbb{R} \mid x^2 > 2 \} \]
```

$$\{ x \in \mathbb{R} \mid x^2 > 2 \}$$

```
\[ \{ x \in X : xg=x \text{ for all } g \in G \} \]
```

$$\{ x \in X : xg=x \text{ for all } g \in G \}$$

Some mathematical symbols such as $\mathbb{R}$ are simply letters in a particular style. There are several commands to provide these styles in math mode. For example:

- roman maths given by `\mathrm` allows the algebraic group GL_n
- fraktur maths given by `\mathfrak` allows the Lie algebra $\mathfrak{gl}_n$

and for upper case letters:

- blackboard maths given by `\mathbb` allows the real numbers $\mathbb{R}$
- calligraphic maths given by `\mathcal` allows the power set $\mathcal{P}$
- script maths given by `\mathscr` allows an alternative for the power set $\mathscr{P}$. (The command `\mathscr` requires the `mathrsfs` package.)

The text mode commands `\textsf` and `\texttt` which provide sans serif and typewriter fonts have analogues in math mode: `\mathsf` and `\mathtt`.

3.5.3 Brackets and arrays

There are many types of *brackets* used in mathematics:

- standard brackets given by `(...)` allow the expression $(x+y)^2$
- square brackets given by `[...]` allow the interval $[0,1]$
- braces (or curly brackets) given by `\{...\}` allow the set $\{0,1,2,\dots\}$
- angled brackets given by `\langle...\rangle` allow inner products $\langle v,w \rangle$
- single bars given by `|...|` allow the absolute value $|x-y|$
- double bars given by `\|...\|` allow norms $\|f\|$
- floor and ceiling given by `\lfloor...\rfloor` and `\lceil...\rceil` allow $\lfloor x \rfloor$ and $\lceil y \rceil$

When enclosing a taller expression in brackets preface the left bracket by `\left` and the right bracket `\right`. For example:

```
\[ \left| \frac{x}{y}-\frac{1}{2} \right| \leq \frac{|x|}{|y|}+\frac{1}{2}. \]
```

$$\left| \frac{x}{y}-\frac{1}{2} \right| \leq \frac{|x|}{|y|}+\frac{1}{2}.$$

Typesetting tip When using using square brackets to denote an interval as part of a longer expression, enclose the entire interval in braces to ensure the spacing is correct. Moreover, if the lower bound in a closed interval is negative then enclose the negative number in braces. For example `x \in [-\pi, \pi]` gives $x \in [-\pi, \pi]$ whereas `x \in {[{-\pi}, \pi]}` gives $x \in {[{-\pi}, \pi]}$ . . . yes, I promise the spacing is different!

The `array` environment is a very useful one: it allows us to write matrices, vectors, two-line permutations, case definitions, tables and much beyond. We begin with an example of a matrix where we produce an array and enclose it in the appropriate brackets:

```
\[
A = \left(
\begin{array}{rl}
\cos\theta  & \sin\theta \\
-\sin\theta & \cos\theta \\
\end{array}
\right)
\]
```

$$A = \left(
\begin{array}{rl}
\cos\theta & \sin\theta \\
-\sin\theta & \cos\theta \\
\end{array}
\right)$$

The `array` environment takes one argument which can be a string of letters `c`, `l`, `r`, one for each column. The text in each column will be justified depending on this letter: `c` for centred, `l` for left aligned and `r` for right aligned. We write the entries of the matrix reading along the rows, separating entries by `&` and rows by `\\`.

We now give an example of a case definition:

```
\[
\delta_{ij} = \left\{
\begin{array}{ll}
1 & \text{if $i=j$} \\
0 & \text{if $i \neq j$} \\
\end{array}
\right.
\]
```

$$\delta_{ij} = \left\{
\begin{array}{ll}
1 & \text{if $i=j$} \\
0 & \text{if $i \neq j$} \\
\end{array}
\right.$$

Although we do not need to close the brace in this example (that is, there is a left `\{` but no right `\}`), LaTeX insists that we still match the `\left` with a `\right`. For this reason we include the *empty delimiter* `\right.` which does not print anything but handles the LaTeX bookkeeping.

Now we turn to tables. Vertical lines are achieved by inserting a pipe `|` between the columns in the argument, and horizontal lines are achieved by inserting `\hline` between the rows. Here is an example of a multiplication table:

```
\[
\begin{array}{c|cccc}
  & e & a & b & c \\
\hline
e & e & a & b & c \\
a & a & e & c & b \\
b & b & c & e & a \\
c & c & b & a & e \\
\end{array}
\]
```

	e	a	b	c
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

and here is an example of a standard table:

```
\[
\begin{array}{ccc}
\hline
x                    & |x^G| & C_G(x)                                        \\
\hline
\mathrm{id}          &  1    & G                                             \\
(1\, 2)(3\, 4)       & 15    & \langle (1\, 2)(3\, 4), (1\, 3)(2\, 4) \rangle \\
(1\, 2\, 3)          & 20    & \langle (1\, 2\, 3) \rangle                   \\
(1\, 2\, 3\, 4\, 5) & 12    & \langle (1\, 2\, 3\, 4\, 5) \rangle           \\
(1\, 2\, 3\, 5\, 4) & 12    & \langle (1\, 2\, 3\, 5\, 4) \rangle           \\
\hline
\end{array}
\]
```

x	$\lvert x^G \rvert$	$C_G(x)$
id	1	G
$(1\,2)(3\,4)$	15	$\langle (1\,2)(3\,4), (1\,3)(2\,4) \rangle$
$(1\,2\,3)$	20	$\langle (1\,2\,3) \rangle$
$(1\,2\,3\,4\,5)$	12	$\langle (1\,2\,3\,4\,5) \rangle$
$(1\,2\,3\,5\,4)$	12	$\langle (1\,2\,3\,5\,4) \rangle$

To include tables in text mode rather than math mode, use the `tabular` environment instead of `array`. The above guidance is valid for the `tabular` environment (see Table 1).

Advanced There are many further packages designed specifically for typesetting tables. For example, the `array` package allows formatting of columns, `booktabs` allows one to produce very attractive tables with appropriate vertical spacing and variable line widths, `tabularx` automatically adjusts column widths and `longtable` allows tables which span more than one page. (See Table 1 for an example using `booktabs` for variable line thickness.)

Typesetting tip Avoid vertical lines in tables and minimise the number of horizontal lines.

3.5.4 Dots and spacing

Ellipsis, three dots with appropriate spacing, is used to indicate omitted items. In text mode `\dots` provided the standard ellipsis ...but in math mode there are some subtleties. In some contexts the dots should be on the base line (for example, when the items are separated by commas or are the missing digits of a number), but in other contexts the dots should be raised (for example, when the items are separated by addition signs or are part of a product).

The command `\ldots` provides the lower dots ... and `\cdots` provides the raised dots $\cdots$. The command `\dots` tries to guess which of `\ldots` and `\cdots` is appropriate; this will *not* always be correct. (In general, `\dots` will correctly provide `\cdots` when followed by a binary relation, but in the cases where `\cdots` are desired to indicate a product or concatenation `\cdots` will need be used explicitly.) For example:

```
\[ 1, 1, 2, 5, 8, \dots \quad 1 + 8 + 27 + \dots + n^3 \quad \pi = 3.14159\dots
\quad 1 \cdot 2 \cdot 3 \cdots 9 = 362880 \quad x = x_1 x_2 \cdots x_k \]
```

$$1, 1, 2, 5, 8, \dots \quad 1 + 8 + 27 + \dots + n^3 \quad \pi = 3.14159\dots \quad 1 \cdot 2 \cdot 3 \cdots 9 = 362880 \quad x = x_1 x_2 \cdots x_k$$

There are contexts when vertical or diagonal dots are required. One can use `\vdots`, `\ddots` and (with the `mathdots` package) `\iddots`. This is particularly useful in matrices:

```
\[
A =
\left(
\begin{array}{cccc}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \cdots & a_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mn} \\
\end{array}
\right)
\]
```

$$A = \left(\begin{array}{cccc} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \\ \end{array} \right)$$

In general, LaTeX does *spacing* very well. However, there can be occasions where one needs to manually add space in math mode. LaTeX ignores any typed spaces in math mode, so the following commands are useful for inserting and removing horizontal space. Here we use the unit mu which is equal to $\frac{1}{18}$ em. In turn, the unit em was historically equal to the width of a letter M in the font being used but is now usually set at the current font size (see [14, Lengths]).

- `\ ` provides a 6 mu space, which is a roughly normal space in text mode
- `\;` provides a 5 mu space
- `\:` provides a 4 mu space
- `\,` provides a 3 mu space, which is a *thin space*
- `\,` provides a *negative* 3 mu space

- `\quad` provides a 1 em space
- `\qquad` provides a 2 em space

Typesetting tip There are several important uses of the the thin space provided by `\,`.

- When typesetting units, be sure to insert a thin space between the number and the unit; for example `$3 \, \mathrm{mu}$` gives $3\,\mathrm{mu}$. (If units will be used frequently then the one should use a bespoke units package such as `siunitx`.)
- When writing permutations in cycle notation, if one leaves spaces between the points rather commas then use a thin space; for example `$(1 \, 2)(3 \, 4 \, 5)$` gives $(1\,2)(3\,4\,5)$.
- Insert a thin space between the product of two factorials. For example:

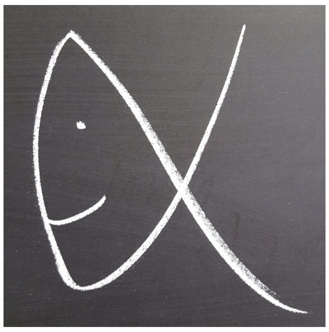
```
\[ \binom{n}{k} = \frac{n!}{(n-k)!\,k!}. \]
```

$$\binom{n}{k} = \frac{n!}{(n-k)!\,k!}.$$

3.6 Images, floats and captions

To insert an *image* into a LaTeX file, save the image file in the same folder as your `.tex` file and ensure that the image is a `.png`, `.jpg` or `.pdf` file. If the image is named `fish.png`, then the command `\includegraphics{fish.jpg}` (from the `graphicx` package) will insert the image at the corresponding place in the document.

By default, the image will be included at its original size. The command `\includegraphics` has optional arguments which allow you to adjust the size. To specify the width of the image, use the argument `width=?` (where `?` is the desired width) and for height use `height=?`. If only one of the width and height are specified, then the aspect ratio will be maintained from the original image (this is generally recommended). The dimensions can be specified in any of the units LaTeX understands (for example, pt, cm, in, ex, em, see [14, Lengths]), but `\textwidth`, which is set to be the width of the text in the document, is often very useful in this case. For example, the following image is centred, is a quarter of the text width and has the same aspect ratio as the original image.



Often one does not wish images to appear in a particular position on the page but rather somewhere nearby in the document which ensures that it fits on one page and does not interrupt the flow of the document too much. LaTeX has a way of doing exactly this: *floats*. By enclosing an image in a `figure` environment the image will appear at an appropriate place nearby in the document, according to LaTeX's internal principles. This is a great example of LaTeX separating content (you inserting the figure) and appearance (LaTeX deciding the best place to put it).

Often one wants to float tables as well as images. This is achieved by using the `table` environment, which is analogous to the `figure` environment introduced above.

animal	collective noun
baboon	troop
eagle	convocation
hippo	bloat
owl	parliament
rhino	crash
wombat	wisdom

Table 1: Some collective nouns.

Advanced The `figure` (and `table`) environment take an optional argument to allow one to have some say over where LaTeX decides to place the figure: `t` and `b` encourage LaTeX to place the figure at the top or bottom a page; `p` encourages the figures to be placed on a page dedicated to figures (and tables); `h` explains that, after all, you would actually quite like the figure to be where you placed it in the file; and `H` (which requires the `float` package) explains, in no uncertain terms, that you really want the figure where you placed it in the file. In all cases, following the argument by `!` attempts to make your opinion override LaTeX's.

It is useful to *caption* floating figures and tables. This can be done by placing the `\caption` command above or below the image or table *inside* the float environment as shown in Table 1.

Advanced If the caption is placed above (or below) the image or table in the float environment, then the caption will appear above (or below, respectively) the image or table. The standard typesetting convention is that captions be placed below figures but above tables. However, the default LaTeX spacing around floats and captions anticipates that caption the will be placed below all floats. To customise the caption spacing – in addition to various other caption parameters such as font, style and justification – use the `caption` package. In particular, to adjust the spacing around captions to suit the standard convention, load the `caption` package with the optional arguments `[tableposition=top, figureposition=bottom]`.

3.7 Citations

As with cross-referencing, *citations* are something which could potentially be difficult but which LaTeX makes very easy. Citing documents comes in two parts. First, at the end of the document, one must create a list of all of the references which are referred to in the document with an identifier (for mathematical documents this is usually just a number) associated to each one. Second, one must include a marker in the text whenever you want to cite a reference in the list.

The reference list can be produced in one of two ways: manually or automatically. To manually produce a reference list, create a `thebibliography` environment wherever you wish the list of references to appear (this will typically be at the end of the document just before `\end{document}`). This environment has one argument which tells LaTeX how long the identifying number might be for each reference. Unless you plan to have at least 100 references, make this argument `99`. Then, for each reference, include something of the following form:

```
\bibitem{ref:wilson_09} R. A. Wilson, \textit{The Finite Simple Groups},
Graduate Texts in Mathematics vol. 251, Springer, 2009.
```

In this example, `ref:wilson09` is the *cite key*, which we use to refer to this reference when we cite it, and the details which follow are the bibliographic data of that reference. Note that the bibliographic data will appear exactly as written and the references will be listed in the order they are included in the file.

To create an inline citation of the above reference, simply include `\cite{ref:wilson_09}` wherever you wish the citation to appear in the text; it appears as [7]. To refer to a particular result or section of a text, use the optional argument. For example, `\cite[Theorem~2.4]{ref:wilson_09}` appears as [7, Theorem 2.4]. To include several citations, include them in one instance of `\cite{}`. For example, write `\cite{ref:cameron_81,ref:wilson_09}` which gives [1, 7] rather than writing `\cite{ref:cameron_81} \cite{ref:wilson_09}` which gives [1] [7].

Typesetting tip There are many different referencing styles and there is not a complete consensus in mathematics. You should check the conventions of the relevant journal or university. However, the following points are worth bearing in mind.

(i) The style of the items on the reference list of this document is common. For example, see [7] for books, [1] for papers and [2] for chapters.

(ii) In mathematics journals, the inline marker for a citation is usually a number in square brackets as in this document. This is the default in LaTeX.

(iii) The standard ordering of references is: order alphabetically by the surname of the first author; to break ties, place single author papers before joint papers; to break ties on joint papers, pass to the second author, third author, . . . ; to break further ties, order by date of publication; to break ties still, order alphabetically by the name of the work.

(iv) Journal names are usually abbreviated in a reference list. For a list of the standard journal abbreviations, see [8].

Advanced BibTeX provides a more automated way of producing reference lists. With this, one produces a separate BibTeX file, named say `references.bib`, which contains the bibliographic data, then includes the commands `\bibliographystyle{plain}` and `\bibliography{references.bib}` at the end of the main LaTeX file. Inline markers are still included using the `\cite`. To customise the style of the references, replace `plain` by an alternative style or a bespoke `.bst` file. There are style files for most major referencing styles. There are also programs for producing a `.bst` file based on particular preferences. Alternatively, one can write one's own in reverse Polish notation. For more, see [14, Bibliography Management].

3.8 Page layout and paragraphing

The many optional arguments of the `geometry` package allow one to control the *page layout*. For example, this document has `geometry` loaded with the arguments `a4paper` and `margin=2cm` to set the page size to A4 (the default is US Letter) and set all margins to 2 cm. Lengths can be expressed in many units such as pt, in, cm, ex, em (see [14, Lengths]). Each of the four margins can be set separately by including `top=3cm` and similar. The orientation of the page can be changed by including `landscape`. For more details, including how to set up asymmetric book style margins, see [14, Page Layout] or the `geometry` package documentation.

Advanced By default, there is no vertical space between paragraphs and each paragraph (except the first of each section) is indented. To change the paragraph indentation, include in the preamble `\setlength{\parindent}{?}` where `?` is the desired paragraph indentation width. A standard typographical alternative is to have vertical space between paragraphs instead of paragraph indentations. To change the interparagraph vertical spacing, use `\setlength{\parskip}{?}`. However, changing the value of `\parskip` changes the spacing in lists in an often undesirable way. To compensate for this, after loading the `enumitem` package in the preamble call `\setlist{topsep=0pt}`.

4 Looking Ahead

4.1 Ti*k*Z

Ti*k*Z ist *kein* zeichenprogramm or Ti*k*Z, for short, is *not* a drawing programme. Rather, Ti*k*Z is a *language* for producing high quality graphics. Therefore, one can use Ti*k*Z to produce graphics in a LaTeX document by including additional code in the `.tex` file. This allows the graphics to be consistent with the rest of the document in quality and style, and also allows the full LaTeX functionality (math mode, text formatting, cross-references) to be used in the production of the graphics. Drawing basic geometric graphics is straightforward, as is drawing graphs (in the graph theoretic sense). There are also libraries for drawing automata, fractals, knots and much else. Many complex graphics can be produced by exploiting for loops, if statements and the ability to carry out arithmetic operations. The Ti*k*Z and PGF manual [6] is the definitive resource for learning and using Ti*k*Z.

4.2 Beamer

In this document we have described how to produce a LaTeX document using the document class `article` or similar (see Section 2.3.1). To produce presentation slides, use the BEAMER document class. With this document class, the `.tex` file is a collection of `frame` environments in which each slide is written. Producing slides in BEAMER allows the full LaTeX functionality to be used. See the BEAMER manual for further details [5].

For more advanced BEAMER users we provide some useful tips.

Typesetting tip By default, text in BEAMER is typeset in sans serif font. Therefore, in math mode use `\mathsf` rather than `\mathrm` for upright text. For example, use `\mathsf{GL}_n(q)`, which gives GL, rather than `$\mathrm{GL}_n(q)$`, which gives GL.

On a similar note, the default commas and full stops in BEAMER's math mode are serif and do not match the sans serif commas and full stops in text mode. (In particular, if the projector has a poor resolution, in math mode, the commas look like full stops because the tails are too faint to be seen.) To make the commas and full stops in math mode match those in text mode, add the following two lines to the preamble:

```
\DeclareMathSymbol{,}{\mathpunct}{operators}{"2C}
\DeclareMathSymbol{.}{\mathpunct}{operators}{"2E}
```

Typesetting tip By default, a collection of navigation symbols appears in the bottom-right of BEAMER slides. If you wish to remove these – which you should – then include the following in the preamble:

```
\usenavigationsymbolstemplate{}
```

Typesetting tip In many styles, there is a space for the title, author and date in the footer of the slides. If any of these are too long to fit in this space, or you otherwise wish to include a different version of any of these in the footer, then you should make use of the optional arguments in the `\title`, etc. commands. For example, if `\date[14.03.2017]{14th March 2017}` is included in the preamble, then the date on the title slide will be "14th March 2017", but the date in the footer will be "14.03.2017".

4.3 Sources of reference

The LaTeX wikibook [14] is an excellent resource and is a good go-to reference. There are several good LaTeX tutorials online; Overleaf [12] is particularly good. The Detexify app (or website [9]) is an invaluable resource for determining the command for a particular symbol. The BEAMER manual [5] and the Ti*k*Z and PGF manual [6] are good references on these topics.

However, a quick internet search is usually the fastest way to solve a problem in LaTeX. In particular, most standard (and many very non-standard) questions have been asked and answered (in great detail) on StackExchange [13].

Happy TeXing.

References

[1] P. J. Cameron, *Finite permutation groups and finite simple groups*, Bull. London Math. Soc. **13** (1981), 1–22.

[2] J. Fulman and R. M. Guralnick, *Derangements in simple and primitive groups*, in Groups, Combinatorics & Geometry: Durham 2001, World Sci. Publ., (2003), 99—121.

[3] Oxford University Press, *New Oxford Style Manual*, Oxford University Press, Oxford, 2016.

[4] S. Pakin, *The Comprehensive LaTeX Symbol List*, 2017, `mirror.ox.ac.uk/sites/ctan.org/info/symbols/comprehensive/symbols-a4.pdf`

[5] V. Miletić, T. Tantau, J. Wright, *The* BEAMER *class*, 2017, `tug.ctan.org/macros/latex/contrib/beamer/doc/beameruserguide.pdf`

[6] T. Tantau, *The TikZ and PGF Packages*, 2013, `mirror.ox.ac.uk/sites/ctan.org/graphics/pgf/base/doc/pgfmanual.pdf`

[7] R. A. Wilson, *The Finite Simple Groups*, Graduate Texts in Mathematics, vol. 251, Springer, 2009.

[8] Abbreviations of Names of Serials, `msc2010.org/MSC2010-CD/extras/serials.pdf`

[9] Detexify, `detexify.kirelabs.org/classify.html`

[10] MacTeX, `tug.org/mactex/mactex-download`

[11] MiKTeX, `miktex.org/howto/install-miktex`

[12] Overleaf, `www.overleaf.com`

[13] TeX StackExchange, `tex.stackexchange.com`

[14] LaTeX wikibook, `en.wikibooks.org/wiki/LaTeX`

Index