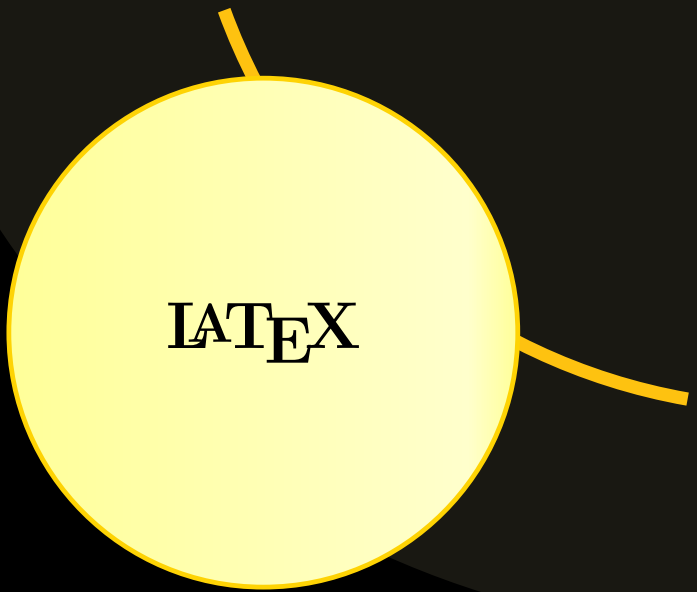
HANSRAJ COLLEGE, UNIVERSITY OF DELHI

LaTeX Practicals

Research Methodology

SHIV SHANKAR SARAN



Download Code!

Submitted to: Prof. Jyoti Bhola

Shiv Shankar Saran

The Practicals of

LaTeX

RESEARCH METHODOLOGY

HANSRAJ COLLEGE
UNIVERSITY OF DELHI

This page is intentionally left blank.

Dedication

This book is dedicated to my family and teachers — thank you for your constant support and guidance which made this work possible and helped me grow academically and personally.

Contents

List of Tables

List of Figures

Part I

Basics

1 Basics of LaTeX

Introduction

LaTeX is a typesetting software. It is a standard format for writing research articles, journals, or any kind of publication. One may wonder why we use LaTeX when we have other software like MS Word, Google Docs, or Libre Office. First, we need to understand the difference between them. While LaTeX is a markup language, the others are word processors. Each has a different use.

When our focus is on the presentation of content, we use word processors, where we can format our content using various options. On the other hand, when our main focus is on the content itself and not its presentation, we use LaTeX.

Since LaTeX is a markup language, we simply define elements like section or chapter names, and LaTeXautomatically formats them according to predefined rules. It is also independent of the application or operating system you are using — it will render the same output regardless, unlike word processors. This makes it highly portable.

1.1 Starting LaTeX

- **Choose a LaTeXenvironment**: online editors like Overleaf or offline installations such as TeX Live or MiKTeX.
- **Create a new `.tex` file**.
- Write the basic structure of a LaTeXdocument, including the document class and content.
- **Compile the `.tex` file**.

Here is a minimal example to start with:

```
\documentclass{book}
\begin{document}
    Hello, world!
\end{document}
```

We can compile this file to produce first LaTeX document.
Output:
Hello, world!

Remark:

- We prefer Overleaf in this book.

1.2 Preparing an input file

To prepare an input file:

1. Start with the document class declaration, for example, `\documentclass{article}`.
2. Write title of document, author name, and date by using `\title{name}, \author{name}, \date{date}`.
3. Add packages (optional) with `\usepackage{}`.
4. Write content between `\begin{document}`...`\end{document}`.

Example of a basic input file:

```
\documentclass{article}
\usepackage{amsmath}

\title{My First \LaTeX Document}
\author{Your Name}
\date{\today}

\begin{document}
\maketitle
This is a sample document.
\end{document}
```

Remark:

- There are different types of document class like book, article, poster, etc.
- `\documentclass[a4paper,11pt]{article}`
- In `\date{}`, if we use `\today` instead of writing any date, then it will always reflect the date when the document is viewed.
- If we write `\date{}` only, then it will not show anything in document.
- We have to use `\maketitle` after `\begin{document}` to get title, author name, and date reflected in document.
- All the things written outside `\begin{document}...\end{document}` are not shown in document.

1.3 Sequences and paragraphs

In LaTeX, the term sequence refers to the order in which content is written and processed. LaTeX reads the document linearly, interpreting commands and text in the order they appear. This sequence determines how the final output is rendered.

- Commands are processed sequentially: For example, `\section{}` must come before `\subsection{}`.
- Packages and preamble must be declared before `\begin{document}`.
- Environment nesting must follow logical order: e.g., `\begin{itemize}` must be closed with `\end{itemize}` before starting a new environment.

Paragraphs are separated by one or more blank lines in `.tex` file. LaTeX treats a blank line as the end of one paragraph and the start of another.
By default, LaTeX indents the first line of each paragraph. To change the amount of indentation globally, we can use `\setlength{\parindent}{0pt}`. Here 0pt refers to zero indentation.
To increase space between paragraphs, we use `\setlength{\parskip}{1em}`.
Paragraphs are usually justified (aligned both left and right) by default.
To change alignment we can use flushleft/flushright/center environment to align it left/right/center.

```
\begin{flushright}
This paragraph is right aligned.
\end{flushright}
```

Output:

This paragraph is right aligned.

Remark:

- To remove indentation for a specific paragraph, we can use `\noindent` before starting a paragraph.
- We can use `\\` or `\newline` to break the line but stay in the same paragraph.

1.4 Quotation marks

Quotation marks are essential in typesetting for clarity and proper punctuation.
We can use `` or ` for starting quotation. To close quotation we can use ' or ''.

Remark:

- These quotation marks are not displayed as in keyboard. For example, to start the quotation, we have to use the symbol on keyboard which is generally on top left corner alongside tilde (~).
- For closing, we can simply use apostrophe symbol in keyboard.

1.5 Dashes

There are three main types of dashes in LaTeX:

- Hyphen (-): Used for hyphenating words or combining them.
- En Dash (–): Used to represent a range of values (like numbers, dates, or time spans).
- Em Dash (—): Used to indicate a break in thought, an interruption, or to add emphasis.

Remark:

- For producing a hyphen, just type a single dash.
- For producing an en-dash, type two dashes.
- For producing an em-dash, type three dashes.

1.6 Space after a period

In LaTeX, when we type a period (".") followed by a space, LaTeX automatically adjusts the space based on its typesetting rules. This is part of what makes LaTeX a superior typesetting system for documents like books, research papers, and articles.

Remark:

- `This is the first sentence. This is second sentence.` We can notice the difference between space after a full-stop and space between any two random words. There is slightly more space between the full-stop and the word "This".
- However, in actual, we gave only one space after the period, but LaTeXautomatically adjusted it according to its algorithm.

1.7 Special symbols

LaTeXhas built-in commands for common symbols (like Greek letters, math symbols, and currency signs), and it allows us to create custom symbols if necessary.
There are different categories of symbols:

Mathematical Symbols: Symbols used in equations and formulas (e.g., `\alpha, \beta, \sum, \int`).
Currency Symbols: Symbols like `\$, €, ¥` for currencies.
Punctuation Symbols: Special punctuation marks (e.g., `\%, \&, \#`).
Accents and Diacritical Marks: Add accents on letters (e.g., `\acute{e}, \tilde{n}`).
Greek Letters: Widely used in scientific notation and math (e.g., `\alpha, \beta`).
Miscellaneous Symbols: Arrows, set notation, and logic symbols (e.g., `\forall, \Rightarrow`).

Remark:

- Some symbols have pre-defined meanings in LaTeX. For example, the percentage symbol requires a backslash: `\%`.
- To use a backslash itself, we use `\textbackslash`.

1.8 Simple text generating commands

These are commands used to format and display basic text in a LaTeXdocument — such as making text bold, italic, or underlined, and changing things like font size, alignment, and more.

Formatting	Command	Example
Bold text	`\textbf{...}`	**Bold Text**
Italic text	`\textit{...}`	*Italic Text*
Underlined text	`\underline{...}`	Underline this
Typewriter/monospace	`\texttt{...}`	`Code-like Text`
Small caps	`\textsc{...}`	Small Caps Text
Emphasized text	`\emph{...}`	*Important word*

Table 1.1: Basic Text Commands

Remark:

- All these commands can be nested and combined for more complex formatting.

1.9 Emphasizing text

We use `\emph` to give more emphasis on certain words or to make them stand out from the rest.

- In general, it's hard to distinguish `\emph` and `\textit`. `\textit` just italicizes the text, while `\emph` gives semantic emphasis and may behave differently in nested use.
- `\textit{This is italicised text}`
  Output: *This is italicised text*
- `This is outer text \emph{emphasized text} outer text`
  Output: This is outer text *emphasized text* outer text
- `\textit{This is outer text \emph{This is emphasized text} This outer text}.`

  Output: *This is outer text* This is emphasized text *This outer text*.

Remark:

- If you nest `\emph` inside `\textit`, it switches back to normal font instead of italic. This is by design to emphasize within already italicized text.

1.10 Preventing line breaks

LaTeX automatically breaks lines to fit text within margins. Sometimes, we want to prevent unwanted line breaks to keep certain text together. Automatic line breaks may split important units like names, URLs, or numbers.

Some methods to prevent line breaks:

- Use `~` instead of a regular space to tie words together.
  Example: `Mr.~Smith went to New~York on July~4th.`
  Output: Mr. Smith went to New York on July 4th.
- Use `\nolinebreak[n]`, where `n = 0 to 4` (4 is strongest).
  Example: `www.\nolinebreak[4]shivedit.com` will force the URL into a single line.

Remark:

- Preventing line breaks is especially important when working with figures, tables, or technical content.

1.11 Footnotes

Footnotes help us provide additional information, citations, or clarifications without cluttering the main text.
Example: `Hi, I\footnote{Shiv is Great} am Shiv!`
Output:

Hi, I[a] am Shiv!

[a] Shiv is Great

Remark:

- Footnotes are numbered automatically.
- They reset per chapter in book or report class.
- Manual numbering: `\footnote[num]{text}`.
- In tables or figures, use `\usepackage{tablefootnote}` and `\tablefootnote`.

1.12 Ignorable input

Ignorable input refers to any code or text that LaTeX parses but does not typeset or execute in the output.

Comments: Notes to self, not processed by LaTeX. Use `%` to start a comment.

Whitespace: Extra spaces, tabs, or blank lines. LaTeX ignores multiple spaces and treats them as one.

Example (comment):

```
% This is a comment. It will not appear in the output.
Hello, World!% This comment is ignored too.
```

Output: Hello, World!

Example (whitespace):

```
Hello,          World!
```

Output: Hello, World!

Remark:

- Use comments to annotate and organize code — they don't appear in final document.
- Use `\%` to display the percent symbol in output.

Part II

Advanced

2 Basic Structure of a Page

Introduction

Advanced LaTeX document structuring and content organization. The main topics include understanding the document class, managing the title page, sectioning documents, and incorporating displayed material such as quotations, lists, displayed formulas, and declarations. These skills are essential for creating well-structured and professional documents using TeX distribution software.

2.1 The Document Class

The `\documentclass` command determines the fundamental layout of the document. It takes two arguments:

1. **Optional parameters** in brackets `[]`
2. **The class name** in curly braces `{}`

Some of the popular classes available to us are:

- `article`: Ideal for assignments, essays, and short papers.
- `report`: Suitable for theses and multi-chapter documents.
- `book`: Used for published books; includes front matter, chapters, and indexing features.

Remark:

- There are some useful options available to us which we can include in our parameter.
  - `fontsize` – changes base font size for readability, maximum upto 12pt.
  - `a4paper` – defines page size.
  - `twocolumn` – Creates a two-column layout.
  - `oneside` / `twoside` – The oneside option formats the document as if it will be printed on only one side of each sheet of paper (or stapled like a report). Default Classes: The article and report classes use oneside by default. twoside The twoside option prepares the document for professional double-sided printing, much like a published book.

2.2 The Title Page

A good looking title page can quickly be created using `\maketitle` as we did in previous chapter. Alternatively, we could use a `titlepage` environment to freely design its layout. So, let's design a nice title page for our book.

```
\begin{titlepage}
\raggedleft
{\Shiv Shankar Saran\\[1in]}
{\The Practicals of\\}
{\Huge\LATEX\\[.2in]}
{\RESEARCH METHODOLOGY\\}
\vfill
{\itshape submitted to\\Prof. JYOTI BHOLA}
\end{titlepage}
```

So this is the exact code of our title page, which we can see on the first page.

Remark:

- When we create title page by using `\maketitle`, then in the `article` class, the title appears on the first page and in `report` and `book`, a dedicated title page is generated by default.
- When we create title page using `titlepage` environment, then it takes entire page regardless of its class.

2.3 Sectioning

Sectioning commands create structure, improve readability, and automatically generate numbering and table-of-contents entries. These are the commands which we can use:

- `\section{Introduction}`
- `\subsection{Background}`
- `\subsubsection{Details}`
- `\paragraph{A further point}`
- `\subparagraph{Subpoint}`

Remark:

- There is an another command namely, `\chapter{}`. Only `report` and `book` support the `\chapter` command.
- We should avoid skipping levels (e.g., using `\subsubsection` without a `\subsection`).

2.4 Displaying quotes

Imagine our text contains a quotation of another author. It might be hard to read if it's just embedded in our words. A common way to improve the readability is displaying: setting the text off by indenting on both margins.

```
\begin{quote}
This is quote by Shiv.
\end{quote}
```

> This is quote by Shiv.

Longer quotations use the `quotation` environment. As in normal text, blank lines separate the paragraphs. They are left-indented at their beginning just like in all our body text.

```
\begin{quotation}
First paragraph of the quotation which is given by me.
And it should be given by everyone.

Second paragraph of quotation which is not given by me
and shouldn't be given by anyone else also.
\end{quotation}
```

> First paragraph of the quotation which is given by me. And it should be given by everyone.
>
> Second paragraph of quotation which is not given by me and shouldn't be given by anyone else also.

Remark:

- We do not begin a new paragraph for it, because the quotation is already set a bit off. That's the reason we don't use a blank line before and after the environment.
- Quotations preserve line breaks and formatting, making them ideal for research papers or literature assignments.

2.5 Lists

Arranging text in the form of a list can be very reader-friendly. We can present several ideas by a clear structure which is easy to survey. Commonly, three types of lists are used:
1. Unordered Lists
2. Ordered Lists
3. Definition lists

2.5.1 Unordered Lists (Bullets)

We use an environment called `itemize`. The command `\item` inside it tells LaTeX that a new item to the list follows.
Syntax:

```
\begin{itemize}
  \item item1
  \item item2
\end{itemize}
```

- item1
- item2

2.5.2 Ordered Lists (Numbers)

We use an environment called `enumerate`. The command `\item` inside it tells LaTeX that a new item to the list follows.
Syntax:

```
\begin{enumerate}
  \item item1
  \item item2
\end{enumerate}
```

1. item1
2. item2

2.5.3 Description Lists

Syntax: `\item [label text] Text`

```
\begin{description}
  \item[Dog] Domestic animal, man's best friend.
  \item[#] Feline pet, very independent.
\end{description}
```

Dog Domestic animal, man's best friend.
\# Feline pet, very independent.

2.5.4 Customizing Lists

To follow language specific habits or certain requirements, we might wish to enumerate by Roman numbers or alphabetically; parentheses or dots might be required. Some may prefer dashes instead of bullets.
So,to achieve this we can use `\usepackage{enumitem}`.
After including the package, when we create a list, then after `\begin{enumerate}` or `\begin{itemize}`, use square brackets, and inside square brackets, include label name. One example is given here:

```
\begin{enumerate}[label=S~\Roman*.]
    \item Shiv
    \item
    \begin{enumerate}[label=\(\star\)]
        \item Shankar
        \item Saran
    \end{enumerate}
\end{enumerate}
```

S I. Shiv
S II. ⋆ Shankar
⋆ Saran

Remark:

- We can also crate a nested list, i.e., list within list.

```
\begin{enumerate}
  \item Item One
    \begin{itemize}
      \item[] Subitem One
      \item[$\heartsuit$] Subitem Two
    \end{itemize}
  \item Item Two
\end{enumerate}
```

Output:

1. Item One
   Subitem One
   ♡ Subitem Two
2. Item Two

- Up to four levels are possible, otherwise LaTeX would stop and print out the error message ! LaTeX Error: Too deeply nested.
- When we create a list, it contains too much space in between.So to remove them we can use this command `\setlist{nolistsep}` in preamble(before `\begin{document}`).
- If we do not want any number or bullet to be printed before an item, then we can use square bracket after `\item`.
- We can also make custom Label for individual items only. For this, in square brackets after `\item`, we need to enter the required label.
- Some labels name which we can try are: `\(\Rightarrow\)`, `\checkmark`, `\times`, `$\star$`, `$\ast$`, `$\diamond$`, `$\bigcirc$`, `$\bullet$`, `$\triangleright$`, `$\heartsuit$`, `$\clubsuit$`.

2.6 Typing Maths Formulas

LaTeX knows three general **modes**:

- **The paragraph mode:** The text is typeset as a sequence of words in lines, paragraphs, and pages.
- **The left-to-right mode:** The text is also considered to be a sequence of words, but LaTeX typesets it from left to right without breaking the line. For instance, the argument of `\mbox` will be typeset in this mode; that's why `\mbox` prevents hyphenation.
- **The math mode:** Letters are treated as math symbols. That's why they're typeset in italic shape, which is common for variables. A lot of symbols can be used, most of them exclusively in this mode. Such symbols are roots, sum signs, relation signs, math accents, arrows, and various delimiters like brackets and braces. Space characters between letters and symbols are ignored. Instead, the spacing depends on the type of symbols—distances to relation signs are different from distances to opening or closing delimiters. This mode is required for all math expressions.

2.6.1 Embedding math expressions within text

Inline math mode

We use `$ ... $` or `\(...\)` to embed math inside running text.

Display math mode (for equations on their own line)

The effect of this environment is that the paragraph will be ended, some vertical space follows, then the centered formula plus the following vertical space. As this math environment takes care of the spacing, don't leave empty lines before and after it! This would cause additional vertical space because of the superfluous paragraph breaks. We use `\[ ... \]` or `equation` environment for displayed equations.

Using math inside text with `amsmath` package

We can use `\text{...}` to include text within math environment.

2.6.2 NumberingEquations

Equations and formulas in general may be numbered. However, this applies only to displayed formulas. The equation environment is responsible for this:

```
\begin{equation}
 \label{key}
 expression
\end{equation}
```

$$expression \tag{2.1}$$

It looks similar to displaymath but numbered this time. The number will be displayed in parentheses on the right side of the equation.
There are two options that modify the way the formulas are displayed:

- `fleqn` This causes all displayed formulas to be aligned at the left margin.
- `leqno` All numbered formulas would get the numbers on the left side instead of the right.

Remark:

- It is advisable to always load `amsmath` package whenever we are working with mathematics.
- Writing formulas in-line saves space and allows fluent explanations. This is recommendable for short math expressions within text. Formulas in the displayed style are outstanding; they are centered and require more space. Furthermore, they can be numbered and we may refer to them using the `\label` and `\ref` which we will learn in upcoming chapters.
- `leqno` and `fleqn` must appear only in `\documentclass[...]{...}` as it is a global option.

2.7 Declarations

Declarations in LaTeX are commands that modify formatting or behavior from the point where they are issued until the end of the current group or environment. Unlike commands with explicit arguments (such as `\textbf{...}`), declarations apply automatically and continuously. When the group ends, the effect of the declaration also ends.

2.7.1 Font Declarations

Font declarations change the type style without requiring arguments:

```
\bfseries    % boldface
\itshape     % italics
\ttfamily    % typewriter
\sffamily    % sans-serif
\mdseries    % medium weight
\upshape     % upright shape
```

Example:

```
{\bfseries This entire sentence is bold.}
```

2.7.2 Size Declarations

Font size can be changed using:

```
\large
\Large
\LARGE
\huge
\Huge
```

Example:

```
{\Large This line appears in a larger font.}
```

2.7.3 Math Style Declarations

Math style declarations control the appearance of mathematical expressions:

```
\displaystyle
\textstyle
\scriptstyle
\scriptscriptstyle
```

Example:

$$\frac{a}{b} \qquad \frac{a}{b}$$

2.7.4 Page and Paragraph Declarations

These influence alignment and spacing:

```
\centering
\raggedright
\raggedleft
\sloppy
\frenchspacing
\nonfrenchspacing
```

Example:

```
\begin{center}
\centering
This paragraph is centered.
\end{center}
```

2.7.5 Math Spacing Declarations

Advanced spacing declarations include:

```
\thinmuskip
\medmuskip
\thickmuskip
```

These adjust the spacing around mathematical operators.

2.7.6 Scope of Declarations

Declarations remain in effect until the current group ends. For example:

```
{\bfseries Bold here} but not here.
```

Remark:

- Outside the group, the document returns to its previous formatting.

3 Math Elements

Introduction

Essential mathematical components required for typesetting in LaTeX are discussed here. This chapter introduces methods for applying accents to characters, using mathematical symbols, and writing subscripts and superscripts accurately. It also covers the construction of fractions and roots, along with the correct usage of ellipsis in mathematical notation. Together, these topics equip learners with the foundational tools needed to produce clear and well-structured mathematical expressions in LaTeX.

3.1 Changing the Type style

LaTeX offer extensive control over text styling through various font commands. Here's a comprehensive list of type styles with examples:

Command	Declaration	Meaning
`\textrm{...}`	`\rmfamily`	roman family
`\textsf{...}`	`\sffamily`	sans-serif family
`\texttt{...}`	`\ttfamily`	typewriter family
`\textbf{...}`	`\bfseries`	**bold-face**
`\textmd{...}`	`\mdseries`	medium
`\textit{...}`	`\itshape`	*italic shape*
`\textsl{...}`	`\slshape`	slanted shape
`\textsc{...}`	`\scshape`	Small Caps shape
`\textup{...}`	`\upshape`	upright shape
`\textnormal{...}`	`\normalfont`	default font

Table 3.1: Type styles

The Roman style is the default serif font of LaTeX. When writing technical documents, we might use sans-serif for heading or `typewriting` font for code snippets. **Bold text** helps emphasize important concepts, while *italic* are perfect for **slighted slanted** alternatives. The Small Caps style works well for Proper Names or Acronyms. Finally we can underline key points when absolute necessary , though underling is used less frequently in professional typesetting that **bold** or *italic* emphasis.

Remark:

- LaTeX provides several predefined font size commands that can be used to changed the size of text within a document.

```
{\tiny tiny text}
{\scriptsize scriptsize }
{\footnotesize footnotesize}
{\small small}
{\normalsize Normal}
{\large large}
{\Large Large}
{\LARGE LARGE}
{\huge huge}
{\Huge Huge}
```

Output:

3.2 Accents

Some languages have letters with accents that you can't simply type with your editor. Suppose we need to write some words having letters with accents in Portuguese and in French like:
Não compreendo. Há aqui alguémque fale inglês? Comment ça va? Où se trouve l'aéroport?
So, LaTeX provides some special commands to produce a variety of accents. They may be combined with any letter. Such a command consists of a backslash followed by one character. The accent will be put above or below the letter that has to follow in curly braces. The following table will list these commands and their effect:

Command	Output	Command	Output
\'{a}	á	\u{a}	ă
\`{o}	ò	\v{e}	ě
\^{e}	ê	\H{a}	a̋
\"{u}	ü	\t{oo}	o͡o
\~{n}	ñ	\c{c}	ç
\={o}	ō	\d{n}	ṇ
\.{a}	ȧ	\b{i}	i̲

Table 3.2: Accents in text mode

Remark:

- Above accents are only text based, so it will only work in text mode. For math mode, we have different accents which we will look in upcoming chapter.

3.3 Subscripts and Superscripts

Subscripts and superscripts are created using _ and ^ respectively: Use braces to mark the concerned expression. So, the common forms are as follows:

```
{expression}_{subscript}
{expression}^{superscript}
```

This may be nested. Let's look at an example:

```
\[ x_1^2 + x_2^2 = 1, \quad 2^{2^x} = 64 \]
```

$$x_1^2 + x_2^2 = 1, \quad 2^{2^x} = 64$$

Remark:

- They work in math mode only.
- If we use subscripts and superscripts to the same expression, the order of ^ and _ is not important.
- In the case of single letters, numerals, or symbols, we can omit the braces.

3.4 Fractions in LaTeX

Syntax: `\frac{numerator}{denominator}`

Example: `\frac{a}{b}` gives output $\frac{a}{b}$

3.4.1 Inline Style

The `amsmath` package provides two commands which we can use in inline mode:

```
\dfrac{1}{2}   % big fraction in text
\tfrac{1}{2}   % small fraction
```

$\frac{1}{2}$ $\frac{1}{2}$

Remark:

- We can also create a nested fractions: `\frac{1}{1 + \frac{1}{2}}`

$$\frac{\frac{1}{2}+\frac{3}{4}}{\frac{5}{6}+\frac{7}{8}}$$

- There is a special command by `amsmath` in LaTeX for continued fractions: `\cfrac`.

```
\cfrac{1}{a + \cfrac{1}{b + \cfrac{1}{c}}}
```

$$\cfrac{1}{a+\cfrac{1}{b+\cfrac{1}{c}}}$$

3.5 Roots

Roots are fundamental elements in mathematical expressions, particularly in algebra and calculus. LaTeX provides clear commands for typesetting roots with precise scaling and positioning.

3.5.1 Square Root

Syntax: `\sqrt{expression}`

```
$\sqrt{2}$
```

Output: $\sqrt{2}$

3.5.2 n-th Root

Syntax: `\sqrt[n]{expression}`

```
$\sqrt[3]{n} = x$
```

Output: $\sqrt[3]{n} = x$

Remark:

- Multiple Roots

  ```
  $\sqrt{1 + \sqrt{1 + x}}$
  ```

  Output: $\sqrt{1+\sqrt{1+x}}$

3.6 Ellipsis

Ellipses in LaTeX are symbols that represent omitted terms, continuations, or patterns in mathematical expressions. Different types of ellipses are used depending on the direction and context of the continuation.

3.6.1 Text Ellipsis

Syntax: `\ldots`

```
1,\, 2,\, 3,\, \ldots
```

$$1, 2, 3, \ldots$$

3.6.2 Centered Ellipsis

Syntax: `\cdots`

```
x_1 + x_2 + \cdots + x_n
```

$$x_1 + x_2 + \cdots + x_n$$

3.6.3 Vertical and Diagonal Ellipsis

Syntax: `\vdots or \ddots`

```
\vdots \qquad \ddots
```

$$\vdots \qquad \ddots$$

Remark:

-
```
\begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1n} \\
a_{21} & a_{22} & \ddots & \vdots \\
\vdots & \ddots & \ddots & a_{n-1,n} \\
a_{n1} & \cdots & a_{n,n-1} & a_{nn}
\end{bmatrix}
```

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_{n-1,n} \\ a_{n1} & \cdots & a_{n,n-1} & a_{nn} \end{bmatrix}$$

4 Mathematical Typesetting

Introduction

Mathematical typesetting tools are used in LaTeX to create clear, precise, and well-structured mathematical documents. It introduces a range of fundamental components, including mathematical symbols, Greek and calligraphic letters, log-like functions, arrays, delimiters, and methods for vertical alignment. The chapter also explains how to construct arrays and multiline formulas, providing the foundational techniques required for producing professional mathematical expressions in LaTeX.

4.1 Mathematical Symbols

We will now look at some math symbols and the commands for producing them. The package `amssymb` provides a large collection of additional mathematical symbols beyond the standard ones in `amsmath`. Hence, it is advisable to apply `\usepackage{amssymb}`.

4.1.1 Binary Operation Symbols

Besides plus and minus, there are a few more operations:

$\amalg$	\amalg	$*$	\ast	$\bigcirc$	\bigcirc
$\bigtriangleup$	\bigtriangleup	$\bigtriangledown$	\bigtriangledown	$\bullet$	\bullet
$\cap$	\cap	$\cdot$	\cdot	$\circ$	\circ
$\cup$	\cup	$\dagger$	\dagger	$\ddagger$	\ddagger
$\diamond$	\diamond	$\div$	\div	$\mp$	\mp
$\odot$	\odot	$\ominus$	\ominus	$\star$	\star
$\times$	\times	$\triangleleft$	\triangleleft	$\triangleright$	\triangleright
$\sqcap$	\sqcap	$\sqcup$	\sqcup	$\setminus$	\setminus
$\vee$	\vee	$\wedge$	\wedge	$\wr$	\wr
$\oplus$	\oplus	$\oslash$	\oslash	$\otimes$	\otimes
$\pm$	\pm	$\uplus$	\uplus		

Table 4.1: Binary Operation Symbols

4.1.2 Binary Relation Symbols

Values of expressions might be equal, but there are more possible relations, for example, they may be congruent, parallel, or they might stand in any other relation:

$\approx$	\approx	$\equiv$	\equiv	$\prec$	\prec
$\asymp$	\asymp	$\frown$	\frown	$\preceq$	\preceq
$\bowtie$	\bowtie	$\mid$	\mid	$\propto$	\propto
$\cong$	\cong	$\models$	\models	$\sim$	\sim
$\dashv$	\dashv	$\parallel$	\parallel	$\simeq$	\simeq
$\doteq$	\doteq	$\perp$	\perp	$\smile$	\smile
$\succ$	\succ	$\succeq$	\succeq	$\vdash$	\vdash

Table 4.2: Binary Relation Symbols

4.1.3 Inequality Relation Symbols

If expressions are not equal, the inequality might be expressed in different ways:

$\geqq$	\geqq	$\gg$	\gg	$\leqq$	\leqq	$\ll$	\ll	$\neq$	\neq

Table 4.3: Inequality Relation Symbols

4.1.4 Subset and Superset Symbols

For comparing sets and expressing relations between them, there are many symbols:

$\sqsubseteq$	\sqsubseteq	$\subset$	\subset	$\supset$	\supset
$\sqsupseteq$	\sqsupseteq	$\subseteq$	\subseteq	$\supseteq$	\supseteq

Table 4.4: Subset and Superset Symbols

4.1.5 Variable Sized Operators

For sums, products, and set operations, for example, we can use operator symbols which are variable in size: bigger in display style and smaller in text style

$\bigcap$	\bigcap	$\bigotimes$	\bigotimes	$\bigwedge$	\bigwedge
$\bigcup$	\bigcup	$\bigodot$	\bigodot	$\coprod$	\coprod
$\bigcup$	\bigcup	$\biguplus$	\biguplus	$\int$	\int
$\bigoplus$	\bigoplus	$\bigvee$	\bigvee	$\oint$	\oint
$\prod$	\prod	$\sum$	\sum		

Table 4.5: Variable Sized Operators

4.1.6 Arrows

Arrows are used for implications, maps, or descriptive expressions:

$\leftarrow$	\leftarrow	$\longleftarrow$	\longleftarrow	$\rightarrow$	\rightarrow
$\Leftarrow$	\Leftarrow	$\Longleftarrow$	\Longleftarrow	$\Rightarrow$	\Rightarrow
$\leftrightarrow$	\leftrightarrow	$\longleftrightarrow$	\longleftrightarrow	$\Leftrightarrow$	\Leftrightarrow
$\mapsto$	\mapsto	$\longmapsto$	\longmapsto	$\hookleftarrow$	\hookleftarrow
$\hookrightarrow$	\hookrightarrow	$\uparrow$	\uparrow	$\downarrow$	\downarrow
$\updownarrow$	\updownarrow	$\Uparrow$	\Uparrow	$\Downarrow$	\Downarrow
$\Updownarrow$	\Updownarrow	$\nwarrow$	\nwarrow	$\searrow$	\searrow
$\swarrow$	\swarrow				

Table 4.6: Arrows

4.1.7 Harpoons

There are special arrows called harpoons:

$\leftharpoonup$	\leftharpoonup	$\leftharpoondown$	\leftharpoondown	$\rightharpoonup$	\rightharpoonup
$\rightharpoondown$	\rightharpoondown	$\rightleftharpoons$	\rightleftharpoons		

Table 4.7: Harpoons

4.1.8 Symbols Derived from Letters

Some letter-like symbols are used in math:

$\bot$	\bot	$\forall$	\forall	$\imath$	\imath
$\top$	\top	ℓ	\ell	$\hbar$	\hbar
$\in$	\in	$\ni$	\ni	∂	\partial
$\wp$	\wp	$\exists$	\exists	$\Im$	\Im
$\jmath$	\jmath	$\Re$	\Re		

Table 4.8: Symbols Derived from Letters

Remark:

- Here are more Miscellaneous symbols LaTeX symbols :

$\aleph$	`\aleph`	$\angle$	`\angle`	$\clubsuit$	`\clubsuit`
$\diamondsuit$	`\diamondsuit`	$\emptyset$	`\emptyset`	$\flat$	`\flat`
$\heartsuit$	`\heartsuit`	∞	`\infty`	∇	`\nabla`
$\natural$	`\natural`	$\neg$	`\neg`	$\prime$	`\prime`
$\sharp$	`\sharp`	$\spadesuit$	`\spadesuit`	$\surd$	`\surd`
$\triangle$	`\triangle`				

4.2 Greek Letters

Mathematicians like to use Greek letters, for instance, to denote constants. To get a lowercase Greek letter, we just write the name with a backslash for the command.

α	\alpha	ζ	\zeta	λ	\lambda	ϕ	\phi
β	\beta	η	\eta	μ	\mu	ρ	\rho
γ	\gamma	θ	\theta	ν	\nu	χ	\chi
δ	\delta	ι	\iota	ξ	\xi	ψ	\psi
ϵ	\epsilon	κ	\kappa	o	o	σ	\sigma
τ	\tau	υ	\upsilon	ω	\omega		

Table 4.9: Greek Letters

We can also produce remaining uppercase Greek letters as follows:

Γ	\Gamma	Λ	\Lambda	Σ	\Sigma	Ψ	\Psi
Δ	\Delta	Ξ	\Xi	Υ	\Upsilon	Ω	\Omega
Θ	\Theta	Π	\Pi	Φ	\Phi		

Remark:

- As the omicron just looks like an o, there's no command for it. It's similar for most uppercase Greek letters, which are equal to Roman letters.

4.3 Calligraphic Letters

For the twenty-six uppercase letters A, B, C, … , Z, there's a calligraphic shape, produced by `\mathcal`:

`\mathcal{A}` $\mathcal{A}$, `\mathcal{F}` $\mathcal{F}$

Remark:

- There are some Bold math symbols (vectors, matrices) which we can use in LaTeX by using `bm` package :

`\bm{v}` $\bm{\Gamma}$, `\bm{A}` $\bm{\Gamma}$

4.4 Mathematical Operators

Trigonometric functions, logarithm functions, and other analytic and algebraic functions are commonly written with upright Roman letters. Simply typing `log` would otherwise look like a product of the three variables, namely, l, o, and g. To ease the input, there are commands for many common functions or so-called **operators**. Here's an alphabetical list of the predefined ones:

arccos	`\arccos`	arcsin	`\arcsin`	arctan	`\arctan`
arg	`\arg`	cos	`\cos`	cosh	`\cosh`
cot	`\cot`	coth	`\coth`	csc	`\csc`
deg	`\deg`	det	`\det`	dim	`\dim`
exp	`\exp`	gcd	`\gcd`	hom	`\hom`
inf	`\inf`	ker	`\ker`	lg	`\lg`
lim	`\lim`	lim inf	`\liminf`	lim sup	`\limsup`
ln	`\ln`	log	`\log`	max	`\max`
min	`\min`	Pr	`\Pr`	sec	`\sec`
sin	`\sin`	sinh	`\sinh`	sup	`\sup`
tan	`\tan`	tanh	`\tanh`		

Table 4.10: Mathematical Operators

Remark:

- The modulo function may be written in two ways, either by using `\bmod` for a binary relation or by using `\pmod{argument}` for a modulo expression in parentheses.
- There are some functions with subscripts and limits also:

```
\max_{1\leq i\leq n} x_i     \log_{10}(x+y)
\lim_{x\to 0} f(x)           \sup_{x\in X} g(x)
```

Output:

$$\max_{1\leq i\leq n} x_i \quad \log_{10}(x+y)$$
$$\lim_{x\to 0} f(x) \quad \sup_{x\in X} g(x)$$

4.5 Arrays and the array environment

For arranging math expressions within a surrounding expression, there's the `array` environment. We use it exactly like a tabular environment. However, it requires math mode and all of its entries are made using the math mode as well.
Example:

```
\[
  \begin{array}{cc}
    a & b\\[4pt]
    c & d
  \end{array}
\]
```

Output:

$$\begin{array}{cc} a & b \\ c & d \end{array}$$

4.5.1 Matrices

The `amsmath` package provides commands to typeset matrices with different delimiters. Once we load `\usepackage{amsmath}` in our preamble, we can use the following environments in our math environments:

Type	LaTeX markup	Renders as

Plain	`\begin{matrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{matrix}`
		$\begin{matrix} 1 & 2 & 3 \\ a & b & c \end{matrix}$

Parentheses (round brackets)	`\begin{pmatrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{pmatrix}`
		$\begin{pmatrix} 1 & 2 & 3 \\ a & b & c \end{pmatrix}$

Brackets (square)	`\begin{bmatrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{bmatrix}`
		$\begin{bmatrix} 1 & 2 & 3 \\ a & b & c \end{bmatrix}$

Braces (curly)	`\begin{Bmatrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{Bmatrix}`
		$\begin{Bmatrix} 1 & 2 & 3 \\ a & b & c \end{Bmatrix}$

Pipes	`\begin{vmatrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{vmatrix}`
		$\begin{vmatrix} 1 & 2 & 3 \\ a & b & c \end{vmatrix}$

Type	LaTeX markup	Renders as

Double pipes	`\begin{Vmatrix}`
	`1 & 2 & 3\\`
	`a & b & c`
	`\end{Vmatrix}`
		$\begin{Vmatrix} 1 & 2 & 3 \\ a & b & c \end{Vmatrix}$

Table 4.11: Matrices

Remark:

- If we need to create matrices with different delimiters, we can add them manually to a plain matrix like this:

```
\left\langle
\begin{matrix}
1 & 2 & 3\\
a & b & c
\end{matrix}
\right\rangle
```

$$\left\langle \begin{matrix} 1 & 2 & 3 \\ a & b & c \end{matrix} \right\rangle$$

4.6 Align environment

The `align` environment in LaTeX (from the `amsmath` package) is one of the most useful tools for typesetting multi-line equations that need to look clean and professional. Inside an align block, we use the ampersand & to mark how things should align. Everything placed after & aligns in a vertical column. We can have multiple & alignment points:
Example:

```
\begin{align}
f(x) &= (x+1)(x-1)\\
&= x^2 - x + x - 1 \nonumber \\
&= x^2 - 1
\end{align}
```

Output:

$$f(x) = (x+1)(x-1) \tag{4.1}$$
$$= x^2 - x + x - 1$$
$$= x^2 - 1 \tag{4.2}$$

Remark:

- We can see that equtions are numbered. To remove these numbers, we can use starred version of align i.e. `align*`.
- To remove numbering from specific line, we can use `\nonumber` before ending line.

4.7 Delimiters

Delimiters in LaTeX are symbols used to enclose mathematical expressions, such as parentheses, brackets, braces, vertical bars, and angle brackets. They help represent mathematical structure clearly. LaTeX offers several methods for typesetting delimiters with correct size and alignment.
Below are frequently used delimiters:

Delimiter	*Code*	*Output*
Parentheses	`( )`	( )
Squarebrackets	`[ ]`	[ ]
Braces	`\{ \}`	{ }
Anglebrackets	`\langle \rangle`	⟨ ⟩
Verticalbars	`| |`	| |
Doublebars	`\\|  \\|`	‖ ‖
Floor	`\lfloor \rfloor`	⌊ ⌋
Ceiling	`\lceil \rceil`	⌈ ⌉

Table 4.12: Delimiters

Remark:

- In LaTeX we can resize our delimiters.
  Syntax:

  ```
  \left <delimiter> ... \right<delimiter>
  ```

4.8 Multi-line formulas

Through `amsmath` package we can write multi-line formulas. We will use `multiline` environment for it. Here, the first line is left-aligned, the last line right-aligned, and all other lines in between are centered.
Example:

```
\begin{multline}
\sum = a + b + c + d + e \\
 + f + g + h + i + j \\
 + k + l + m + n
\end{multline}
```

Output:

$$\sum_{n=1}^{30} n = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 + 16 + 17 + 18 + 19 + 20 + 21 + 22 + 23 + 24 + 25 + 26 + 27 + 28 + 29 + 30 \quad (4.3)$$

Remark:

- To numbering from the line use starrerd version of the environment, like `multiline*`.
- Wr can also use `gather` environment. All lines are centered here.
- Each line in all above environment is ended by \\, except the last one.

5 Math formatting

Introduction

Mathematical writing relies on clear and consistent notation to communicate ideas precisely. LaTeX offers a rich set of commands for arranging symbols, expressions, and accents in ways that reflect mathematical meaning—from stacking elements like limits and fractions, to placing bars and arrows over variables, and underlining or boxing key expressions. This chapter introduces essential LaTeX techniques for vertical and horizontal alignment, mastering accents, and symbol stacking.

5.1 Putting a symbol above another

In mathematical typesetting, we often need to place symbols or expressions above or below others. LaTeX provides several methods for this. One of it is through `amsmath`.

- `\underset{expression below}{expression}` puts an expression below another, using the subscript size below.
- `\overset{expression above}{expression}` puts an expression above another, using the subscript size above.

```
\overset{\alpha}{x} \qquad \underset{\beta}{x}
```

$$\overset{\alpha}{x} \qquad \underset{\beta}{x}$$

5.1.1 {\atop }command

This stacks two expressions without a dividing line.

```
{a \atop b}
```

$$\begin{Bmatrix} a \\ b \end{Bmatrix}$$

Remark:

- The standard way to insert upright text inside math mode is `\text{...}` (from amsmath).
- To write Binomial coefficient, we can use command like this:

```
\[
\binom{n}{k} \quad \dbinom{m}{r} \quad \tbinom{p}{q}
\]
```

$$\binom{n}{k} \quad \dbinom{m}{r} \quad \tbinom{p}{q}$$

5.2 Underlining and Overlining

`\overline` puts a line above its argument, which may be nested.
The counterpart is `\underline`.

```
sss = \overline{SHIV\underline{SHANKAR}}
```

$$sss = \overline{SHIV\underline{SHANKAR}}$$

Remark:

- It doesn't have to be always a line. Braces are popular too. The commands are `\underbrace` and `\overbrace`.

```
SSS = \underbrace{S + H + I + V}_{SARAN}^{SHANKAR}
```

$$SSS = \underbrace{S + H + I + V}_{SARAN}^{SHANKAR}$$

5.3 Setting accents

We've already seen accents in text mode. For the math mode, we need different commands. They may be applied to any letter. Here's the list of **math accents** using the lowercase letter 'a' as an example:

$\acute{a}$ \acute{a}	$\check{a}$ \check{a}	$\grave{a}$ \grave{a}	$\tilde{a}$ \tilde{a}
$\bar{a}$ \bar{a}	$\ddot{a}$ \ddot{a}	$\hat{a}$ \hat{a}	$\vec{a}$ \vec{a}
$\breve{a}$ \breve{a}	$\dot{a}$ \dot{a}	$\mathring{a}$ \mathring{a}	

Table 5.1: Accents

Remark:

- There are extensible accents also *wide accents*; they try to fit the width of their argument.

$\widehat{SHIV}$ \widehat{ABC} $\widetilde{SHANKAR}$ \widetilde{abc}

5.4 Stacking symbols

The LaTeX command `\stackrel` is used to place one symbol or piece of text directly above another symbol in math mode, typically for relational operators and arrows. It is especially useful for annotating symbols with definitions, conditions, or labels. **Syntax:** `\stackrel{above}{main}`.
It places "above" directly over "main", often used for relational symbols or arrows. It is similar to `\overset{above}{main}`, but it doesn't requires any package.

Remark:

- When we want to create something like piecewise function, then we can use `cases` command.

```
$\begin{cases}
f(x)=
x^2 & \text{if } x \ge 0 \\
e^x - 1 & \text{if } x < 0
\end{cases}$
```

Output:

$$f(x) = \begin{cases} x^2 & \text{if } x \geq 0 \\ e^x - 1 & \text{if } x < 0 \end{cases}$$

5.5 Spacing in Math Mode

When typing mathematics, LaTeX automatically inserts most of the spacing we need. Operators, relations, fractions, and superscripts are spaced according to standard mathematical conventions. However, there are times when we may want to adjust the spacing manually.
We can fine-tune spacing using explicit spacing commands inside math mode:

\,	thin space
\ :	medium space
\;	thick space
\quad	1 em space
\qquad	2 em space

Example:

```
a\!b, \qquad a\,b, \qquad a\;b, \qquad a\quad b
```

$ab, \qquad a\,b, \qquad a\;b, \qquad a\quad b$

Remark:

- We can also use negative spacing `\!` which decreases spacing.

  ```
  $SHIV\!\!\!\!\!\!\!\!\!SHANKAR$
  ```

  Output:
  $SHIVSHANKAR$

5.6 Type style in Math Mode

There are somr commands for changing the font style in math mode:

Command	Used package	Example
\mathrm{...}		roman 123
\mathit{...}		*italic 123*
\mathsf{...}		sans-serif 123
\mathbb{...}	amsfonts, bbm	$\mathbb{ABC}$
\mathds{...}	dsfont	$\mathbb{CRQZ1}$
\mathfrak{...}	eufrak	$\mathfrak{ABC}$
\mathnormal{...}		normal

Table 5.2: Type style in Math mode

Remark:

- Though letters in math mode are italic, they are considered to be separate symbols, which results in a different spacing than that of an italic word.
  `\textit{Definition}, \(Definition\)`
  Definition, $Definition$
- Also, `\mathit` treats the argument as text in italic math font. So, for text within formulas, use a text or math font command, or even better: use `\text{...}` of amsmath.

5.7 Math Styles Math Mode

There are four *math styles* available to us, determining the way of typesetting and the font size:

Style	Command	Meaning
display	`\displaystyle`	Default for letters and symbols in displayed formulas
text	`\textstyle`	Default for letters and symbols within in-text formulas
script	`\scriptstyle`	Smaller font size, used for subscripts and for superscripts
scriptscript	`\scriptscriptstyle`	Even smaller font size, for nested script style

Table 5.3: Math styles in Math mode

$a+b$ `\scriptstyle`

$a+b$ `\scriptscriptstyle`

$$a+b \quad \texttt{\textstyle}$$

$$a+b \quad \texttt{\displaystyle}$$

Remark:

- The `textstyle` differs from the `displaystyle` in mainly two ways; in `textstyle`, variable sized symbols are smaller and subscripts and superscripts are usually placed beside the expression instead of below and above, respectively. Otherwise the font size is the same.
- LaTeX switches the style automatically; if we write a simple exponent, it will be typeset in script style, that is, with a smaller font size.
- You may force a desired style using one of the commands in the aforementioned table. This allows you, for instance, to:
  - Type formulas within the text exactly like they would appear in a displayed formula: bigger fraction, bigger sum signs, further subscripts are set below, and superscripts are set above. Note, all of this increases the line spacing.
  - Write exponents or indexes with bigger symbols.

6 Defining new commands

Introduction

LaTeX is not limited to built-in commands and structures but we can also create our own. This practical focuses on the customization features of LaTeX that allows us to extend its functionality and make documents more efficient and consistent.

6.1 Defining Commands

LaTeX allows us to create our own commands for commonly used expressions or symbols.
Syntax: `\newcommand{\cmdName}[num]{definition}`

LaTeX also provides the ability to redefine existing commands. This is useful when customizing document layout or formatting.
Syntax: `\renewcommand{\cmdName}[num]{new definition}`

Example:

```
\newcommand{\area}[2]{The area is $#1 \times #2$ square units.}
```


Remark:

- `\cmdName` is the name of the command. `num` is the number of arguments (optional). The last part is what the command will produce.
- `#1` or `#2` refers to the first or second argument passed to the command.

6.2 Defining Environments

We can create custom environments to format special sections uniformly. Custom environments are used to create formatted text blocks such as notes, examples, definitions, or remarks.
Syntax: `\newenvironment{name}{begin code}{end code}`.
Example:

```
\newenvironment{important}{
  \begin{center}\bfseries Important:\\[5pt]
}{
  \end{center}
}
```


Remark:

- `name` is the name of the environment, `begin code` is what happens at the start, and `end code` is what happens at the end.
- After defining, we can use the environment like:
  `\begin{name}`
  ... content ...
  `\end{name}`

6.3 Theorems

Mathematical documents include elements that require special formatting and numbering such as theorems, definitions, propositions, remarks, corollaries, lemmas and so on. `amsthm` package helps in defining theorem-like environment.

1. Use package `\usepackage{amsthm}`.
2. Define environment`\newtheorem{theorem}{Theorem}[section]`, the first one is the name of the environment that is defined, the second one is the word that will be printed, in boldface font, at the beginning of the environment, third restarts the theorem counter at every new section.
3. Now, this environment can be used like this: `\begin{theorem} and \end{theorem}`.
4. We can also use `proof` environment without defining them: `\begin{proof}...\end{proof}`.

Remark:

- In the output, these environments are numbered and labeled "Theorem" and "Definition", and so on.
- Before using these environment, we must define them in preamble.
- When we want a new environment(say lemma) such that it also use the same counter as other(say theorem) environment, then we can use this: `\newtheorem{lemma}[theorem]{Lemma}`,
- Some famous theorems have their own names, for these cases we can add said name inside brackets in the environment opening command like `\begin{theorem}[Pythagorean theorem]`.
- To create unnumbered theorem-like environment apply asterik like this `\newtheorem*{remark}{Remark}`.
- The package amsthm provide special commands, one of them is—theorem styles. The command `\theoremstyle{ }` sets the styling for the numbered environment defined right below it.
- Available theorem styles are-
  - `definition:` **boldface title**, Roman body. Commonly used in definitions, conditions, problems and examples.
  - `plain:` **boldface title**, *italicized body*. Commonly used in theorems, lemmas, corollaries, propositions and conjectures.
  - `remark:` *italicized title*, Roman body. Commonly used in remarks, notes, annotations, claims, cases, acknowledgments and conclusions.

7 Tables and Figures

Introduction

Scientific documents and many other technical writings do not consist only of plain text; they also organize data using well-structured tables and enhance understanding through diagrams, charts, and drawings. These visual and tabular elements help present complex information more clearly, allow easier comparison of values, and improve the overall readability of the document.

7.1 Tabbing environment

The tabbing environment in LaTeX is used to create aligned text using tab stops, similar to how tabs work in a typewriter or text editor.
Example:

```
\begin{tabbing}
\emph{Info:} \= Software \= : \= \LaTeX \\
             \> Author \> : \>Shiv Shankar Saran \\
             \> Website \> : \> https://shivedit.onrender.com/
\end{tabbing}
```

Output:

Info: Software : LaTeX
Author : Shiv Shankar Saran
Website : https://shivedit.onrender.com/

Remark:

- Set tab stops by \= , and end the line by \\.
- Move to the next tab stop by \>, and again end lines by \\ .

7.2 Tables

LaTeX provides the `tabular` environment for typesetting simple and complex tables which can be nested. Tables in LaTeX allow us to organize information clearly using the `tabular` environment.
Syntax:

```
\begin{tabular}[position]{column specifiers}
row 1 col 1 entry & row 1 col 2 entry ... & row 1 col n entry\\
 ...
\end{tabular}
```

Example:

```
\begin{tabular}{||>{\centering} m{2cm} r| c||}
 \hline
 Col1 & Col2 & Col3 \\
 \hline\hline
 Centre and middle aligned & right aligned& centre aligne \\
 \hline
 2 & shiv & shankar \\[1cm]
 \hline
\end{tabular}
```

Col1	Col2	Col3
Centre and middle aligned	right aligned	centre aligne
2	shiv	shankar

Table 7.1: Tables

7.2.1 Combining rows and columns

We can combine two or more rows or column to create a larger table cell. We have to use `\usepackage{multirow}` for merging rows while no package required for merging columns.

- For multi-column (merging columns):
  Syntax: `\multicolumn{<num>}{<alignment>}{<text>}`.
- For multi-row (merging rows):
  Syntax: `\multirow{<num>}{<width>}{<text>}`.

Example:

```
\begin{tabular}{|c|c|c|c|}
\hline
\multicolumn{4}{|c|}{\textbf{Multi-Table}} \\ \hline
\multirow{2}{*}{Row-merged}
  & \multicolumn{2}{c|}{Column-merged}
```

```
  & \multirow{2}{*}{Row-merged} \\ \cline{2-3}
  & Shiv & Shankar &  \\ \hline
\end{tabular}
```

Multi-Table			
Row-merged	Column-merged		Row-merged
	Shiv	Shankar	

Table 7.2: Merged tables

7.2.2 Multi-page tables

If we want to insert a very long table, which takes up two or more pages, we can use package `\usepackage{longtable}` and can then use the longtable environment.

Basic Syntax:

```
\begin{longtable}{|c|c|c|}
\hline
Header1 & Header2 & Header3 \\ \hline
\endfirsthead

\hline
Header1 & Header2 & Header3 \\ \hline
\endhead

\hline
\endfoot

\hline
\endlastfoot

% Body content goes here
Row1 & A & B \\ \hline
Row2 & C & D \\ \hline
...
\end{longtable}
```

Here are the meaning of each command used here.

`\endfirsthead` Everything above this command will appear at the beginning of the table, in the first page.

`\endhead` Whatever we put before this command and below `\endfirsthead` will be displayed at the top of the table in every page except the first one.

`\endfoot` Similar to `\endhead`, what we put after `\endhead` and before this command will appear at the bottom of the table in every page except the last one.

`\endlastfoot` Similar to `\endfirsthead`. The elements after `\endfoot` and before this command will be displayed at the bottom of the table but only in the last page where the table appears.

7.2.3 Table environment

The table environment is a floating container used to hold tabular material, captions, and labels. It does not itself create a table grid — that job is done by the tabular environment inside it. Some of it sapplications are:

Makes the table float LaTeX moves the table to the best position on the page for spacing and layout.

Allows captions Captions must be placed inside the `table` environment, not inside `tabular` like this `\caption{caption of table}` We can place it above or below the table.

Allows labels for referencing Use `\label` immediately after `\caption`. This allows referencing the table number using `\ref{tab:example}`.

Enables optional position arguments Example options: `[h]`, `[t]`, `[b]`, `[p]`.

- `h` → Will place the table here approximately.
- `t` → top of page
- `b` → bottom of page
- `p` → Put the table in a special page, for tables only.
- `H` → Place the table at this precise location.

Basic Syntax:

```
\begin{table}[position]
  \centering
  \caption{Our caption}
  \label{tab:example}
  \begin{tabular}{|c|c|c|}
    \hline
    A & B & C \\ \hline
    1 & 2 & 3 \\ \hline
  \end{tabular}
\end{table}
```

7.2.4 The Tabbing Environment

The `tabbing` environment allows us to create aligned blocks of text using tab stops. It works similarly to using tabs in a word processor or typewriter and is useful for creating columns or formatted lists without the complexity of tables.

Key features:

- Define tab stops with `\=`.
- Move to the next tab stop with `\>`.
- Use `\kill` at the end of a line to set tab stops without printing the line.

Example:

```
\begin{tabbing}
Name \hspace{3cm} \= Age \hspace{2cm} \= City \kill
Shiv \> 21 \> Delhi \\
Shankar \> 22 \> bgp \\
\end{tabbing}
```

This produces aligned columns for Name, Age, and City:

Shiv	21	Delhi
Shankar	22	bgp

7.2.5 Coloured table

We can apply colours to the rows of table by using the `xcolor` package with the table option. Before starting our table, we can put command like this: `{\rowcolors{3}{green!80!yellow!50}{green!70!yellow!40}`.

The command `\rowcolors` takes three parameters each passed inside braces: the row to start, the colour for odd rows and the colour for even rows.

Remark:

- A column specifier controls:
  - **Alignment** (left, center, right)
  - **Width** of the column
  - **Vertical lines** between columns
  - **Special formatting** using packages (like fixed-width or paragraph columns)
- For example, `\begin{tabular}{|c|c|c|}` declares that three columns, separated by a vertical line, are going to be used in the table. Each c means that the contents of the column will be centred. We can also use r to align the text to the right and l for left alignment.
- To insert a horizontal line on top of the table and at the bottom or between rows, we can use `\hline`.
- Each & is a cell separator and the double-backslash \\ sets the end of this row.
- `[length]` after \\ —specify additional vertical space below that row.
- We can use `\usepackage{array}` which gives us more control over columns, custom column types, and better alignment.
- New column specifiers in `array` package are m{length}, b{length}, and p{length} with improved behavior.
  m = vertically centered cell
  b = cells aligned at bottom
  p = paragraph column (top-aligned)
- If we want to combine both horizontal as well as vertical alignment with fixed size of column, then we can use our commannd like this `>{\raggedright}m{3cm}`, `>{\centering}m{3cm}` or `>{\raggedleft}m{3cm}`.
  The prefix sets the alignment of each column: the first one to right, the second one to center and the third one to left.
- `\setlength{\arrayrulewidth}{length}`: This sets the thickness of the borders of the table.
- `\setlength{\tabcolsep}{length}`: The space between the text and the left/right border of its containing cell is set by this command.

7.3 Graphics

LaTeX provides powerful tools for handling images, mainly through the `graphicx` package. Images can be inserted, resized, rotated, wrapped with text, and arranged in subfigures.
Basic Syntax: `\includegraphics[key=value list]{file name}`.
Steps to include picture in environment:

1. After `\usepackage{graphicx}`, upload the picture in Overleaf by clicking upright arrow below menu option in top-left corner.
2. In our document include this command wherever we need to use picture— `\includegraphics[width=...,height=...]{file}`. There is no need to include extension.
3. Put the file into the same directory as our document; otherwise we need to specify a full or relative path name like this : `\includegraphics{Folder/figure}`. The picture will be embedded with its original size.

7.3.1 Figure environment

The figure environment is used to insert images as floating objects. "Floating" means LaTeX automatically moves the figure to the best position on the page for optimal layout.
Basic Syntax:

```
\begin{figure}[<position>]
    \centering
    \includegraphics[width=<size>]{<filename>}
    \caption{<caption text>}
    \label{<label>}
\end{figure}
```

7.3.2 Sub-figures

We can create multiple figure within one figure like subfigure. For this we have to use package: `\usepackage{subcaption}`. It is also used to insert sub-table as well.
Example:

```
\begin{figure}
    \centering
    \begin{subfigure}{0.45\textwidth}
        \includegraphics[width=\textwidth]{photo}
        \caption{A}
    \end{subfigure}
    \begin{subfigure}{0.45\textwidth}
        \includegraphics[width=\textwidth]{photo}
        \caption{B}
    \end{subfigure}
    \caption{Main caption}
\end{figure}
```

Output:

(a) A (b) B

Figure 7.1: Subfigures

Remark:

- Some keys available in `includegraphics` are:
  - **width**: The graphic would be resized to this width.
  - **height**: The graphic would be resized to this height.
  - **scale**: The graphic would be scaled by this factor. Example: scale=0.5.
  - **angle**: The graphic would be turned by this angle. Example: angle=90.
- Different positions we can try for our pictures are:
  - h stands for here. The float may appear where it's been written in the source code.
  - t stands for top.
  - b stands for bottom.
  - p stands for page. The float is allowed to appear on a separate page, where only floats may reside but no normal text.
  - ! tells LaTeX to try harder! Some constraints may be ignored, easing the placement.
- There is another useful package `\usepackage{wrapfig}`, which can be used to wrap figures around text. After using package, we need to start environment by `\begin{wrapfigure}{placement}{width}` ...`\end{wrapfigure}`. Similarly, we can use `wraptable` for tables.
- `placement` can be one of the characters r, l, i, o for right, left, inner, or outer side or the corresponding uppercase letters R, L, I, O with the same meaning, but allowing the figure to float. The final and mandatory argument gives the width of the figure.

7.4 Marginal Notes

We can add notes in the margin using the command `\marginnote`. For instance, this sentence includes a marginal note.
To use this , first import package by `\usepackage{marginnote}`.
Syntax: `\marginnote{content}`.

This is a note in the margin.

Remark:

- Marginal notes are useful for short comments or reminders that do not fit naturally into the main text.
- We can also move marginal notes up or down by specifying length after content like this : `\marginnote{content}[-2cm]`. Negative length moves up and positive moves down.

8 Referencing

Introduction

Referencing in LaTeX provides a systematic way to cross-refer sections, figures, tables, equations, and other structural elements within or outside a document.

8.1 Table of Contents (TOC)

A Table of Contents (often abbreviated as TOC) is a list that shows the structure of document — usually including sections, subsections, and other major divisions — along with their corresponding page numbers. LaTeX automatically generates table of contents using the `\tableofcontents` command. It includes all sections and subsections marked by `\section`, `\subsection`, etc.

```
Contents
1 Introduction ..................................... 1
2 Methods ......................................... 2
   2.1 Data Collection ........................... 2
3 Results ......................................... 3
```

8.1.1 Index

Extensive documents often contain an index. It is a list of words or phrases and page numbers pointing to where related material can be found in the document. In contrary to a full-text search feature, the index provides selective pointers to relevant information.
To create an index in LaTeX, we need to follow these basic steps:

1. In the preamble, load the **index** package.
2. Use the `\makeindex` command in preamble to declare that an index will be used.
3. Index required point with keyword using `\index{keyword}` command. For example, if we want to index the word LATEX with latex, we can use `\index{latex} LATEX`.
4. Index page is not included in Table of contents automatically, so to include them, use `\usepackage{tocbibind}`.
5. Generate the index at the end of the document using the `\printindex` command.

Remark:

- Always compile LaTeX file twice to update the table of contents and references correctly.
- The first run records all the section titles and page numbers in an auxiliary file (.toc file).
- The second run reads that file and updates the Table of Contents in document.
- To control how many levels (sections, subsections, etc.) appear, use `\setcounter{tocdepth}{n=1/2/3}` in preamble.-1 = part(in book and report)
  0 = chapter(not available in article)
  1 = only sections
  2 = include subsections
  3 = include subsubsections
  4 = include paragraph
- We can use alternative for chapter name to be shown in toc as `\chapter[Chapter short name]{Chapter name}`
- Commands for creating lists of figures and tables are `\listoffigures` and `\listoftables`. Depending on the class, they produce a fine list of all captions together with the figure respectively the table number and the corresponding page numbers.
- Starred commands like `\chapter*` and `\section*` don't produce a TOC entry. We can manually add them using this command: `\addcontentsline{file extension}{sectional unit}{text}`. This command can be used in several contexts: the file extension may be:
  - **toc** for the table of contents file
  - **lof** for the list of figures file
  - **lot** for the list of tables file

  Or, any another extension of such a file type known to LaTeX.

8.2 Cross-Referencing

Cross-referencing is a powerful feature in LaTeX that allows authors to create automatic references to sections, figures, tables, equations, and other numbered elements within a document. This ensures consistency, improves navigation, and eliminates the need for manual renumbering when the document structure changes.

There are three major commands for cross-referencing.

Command	Meaning	Purpose
`\label{unique-name}`	Assigns a unique name to something	Marks a location to refer back to
`\ref{name}`	Displays the number of the labeled item	Creates a reference to section/figure/etc.
`\pageref{name}`	Displays the page number of the labeled item	Refers to which page something is on

Table 8.1: Cross-referencing

Example:

```
\section{A}\label{sec:a}
See Section~\ref{sec:b} on page~\pageref{sec:b}.
\section{B}\label{sec:b}
This refers to Section~\ref{sec:a} on page~\pageref{sec:a}.
```

Explanation of output: Here we can see that the section A is labelled as 'a' by using`\label` and then we are referring to it in last line by using `\ref`.

Remark:

- Compile the document 2-3 times for cross-referencing to work properly.
- In the figure or table environments, `\caption` is responsible for the numbering. That's why as to be placed after `\caption`, not before.
- The name may consist of letters, digits, or punctuation characters. They are case-sensitive. We could prefix them with the type of environment, label figures with fig:name, tables with tab:name, sections with sec:name, item of list with item:name and similar in other cases.
- It is very tiring to always write something like 'see Figure `\ref{fig:figure1}`'. There is special package in LaTeX — **`cleveref`** which automatically determines the type of cross-reference and the context in which it's used.
  Basically, we could just use `\cref` instead of `\ref` or `\Cref`(to capitalize).
- The xr(standing for eXternal References) package is used to refer to an external document. Use `\externaldocument{external_document-name(no extension)}` in preamble.
- For avoiding conflicts when an external document uses the same `\label` like the main document, we can declare a prefix using the optional argument like `\externaldocument[prefix-]{document}`. This way, all references from external document would be prefixed and for referring we can write `\ref{prefix-name}` to refer to name in external document. We may choose any prefix that transforms our labels such that they become unique.
- We can use hyperref package in our document right before cleveref. All references will become hyperlinks. Click any of those numbers to jump to the referred table, list item, section, or page.

8.3 Bibliography and Citation

A **Bibliography** is a list of sources that we used while writing a document. It includes books, research papers, articles, websites, and any other materials that we referred to.
Especially in scientific documents, a list of references or bibliography is very common. We shall work out how to typeset a bibliography and how to refer to its entries. LaTeX's standard environment for bibliographies has the following form:

```
\begin{thebibliography}{widest label}
\bibitem[label]{key} author, title, year etc.
\bibitem…
…
\end{thebibliography}
```

Each item is specified using the command `\bibitem`. This command requires a mandatory argument determining the key. We may simply refer to this key by `\cite{key}` or `\cite{key1,key2}`.

8.3.1 Using bibliography databases with BibTeX

Manually creating the bibliography is laborious. Hence, we will create a separate database file containing the references.

1. Create a new document. We will put our entry here like this:

   ```
   @entrytype{keyword,
   fieldname = {field text},
   fieldname = {field text},
   …
   }
   ```

2. Save the file with same name as our main file but with extension `.bib` . Then, in our main file, instead of writing all reference in the end, we will just add this before `\end{document}`.

   ```
   \bibliographystyle{style}
   \bibliography{tex}
   ```

Remark:

- If no label has been given, LaTeX will number the items consecutively in square brackets.
- The mandatory item of the `thebibliography` environment should contain the widest label for the alignment of the items. It sets the width of the reference labels.
- Standard bibliography styles are:
  plain Arabic numbers as labels, sorted by authors' names. Numbers appear in square brackets, including with `\cite`.
  unsrt Like `plain`, but entries are not sorted— they appear in the order they are cited in the text.
  alpha Sorted by authors' names; labels are alphabetic shortcuts made from the author's name and publication year. Square brackets are used as in the other styles.
  abbrv Like `plain`, but first names and other fields are abbreviated.
  These style should be chosen after `\begin{document}` and before `\bibliography`.
- Some standard entry field in BibTeX are **address, author, book, chapter, edition, journal, pages, pubisher, series, title, volume, year, howpublished**.
  Today we often refer to online sources. To put Internet addresses into BibTeX fields, we can use the `\url` command of the url or hyperref package, for example, howpublished = `{\url{https://shivedit.onrender.com}}`.
- Some standard entry types are as follows:

Type	**Required fields**	**Optional fields**
article	author, title, journal, year	volume, number, pages, month, note
book	author *or* editor, title, publisher, year	volume *or* number, series, address, edition, month, note
misc	none	author, title, howpublished, month, year, note

- We can also list references in LaTeX without citing them in the text. Use `\nocite{*}` to list all references, or `\nocite{key}` for specific reference.

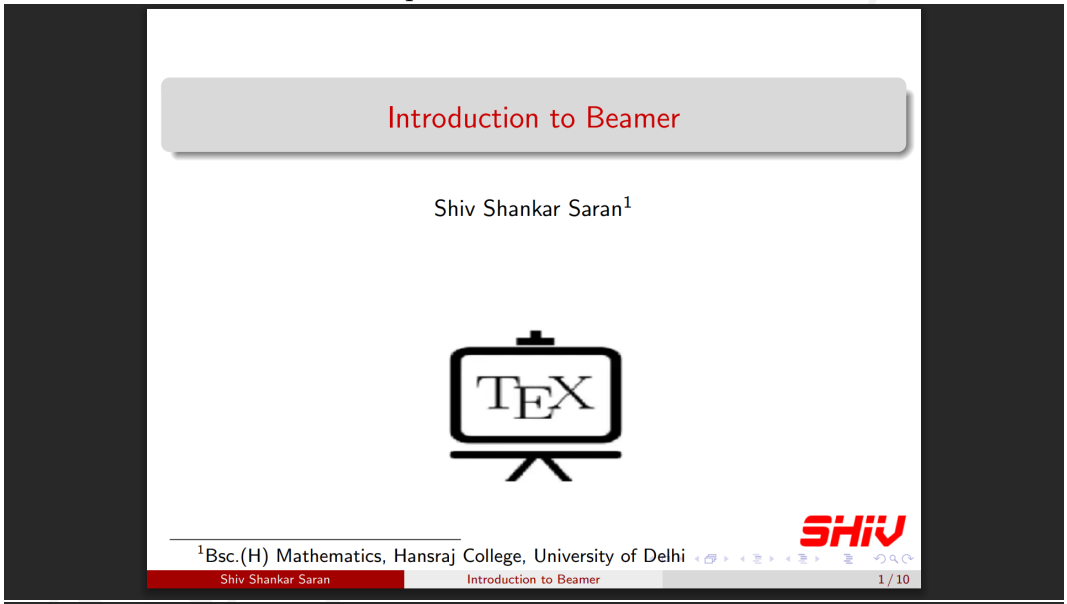
9 Beamer

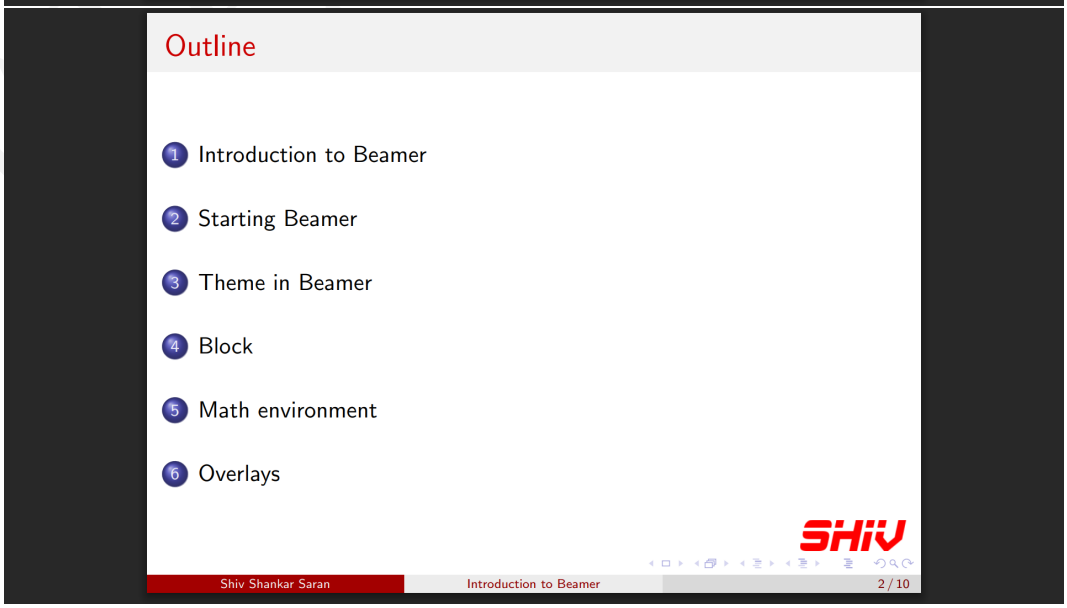
Introduction

Beamer is a LaTeX class used for creating professional presentations. It allows users to design slides using LaTeX syntax, ensuring consistent formatting, high-quality typography, and easy inclusion of mathematical equations, figures, and references. Beamer is widely used in academic, research, and technical presentations because it integrates seamlessly with LaTeX documents and produces PDF slides of excellent quality.

9.1 Presentation

Here are the slides of the presentation made in Beamer.





Introduction to Beamer

Beamer is a LaTeX class used for creating professional presentations. It allows users to design slides using LaTeX syntax, ensuring consistent formatting, high-quality typography, and easy inclusion of mathematical equations, figures, and references. Beamer is widely used in academic, research, and technical presentations because it integrates seamlessly with LaTeX documents and produces PDF slides of excellent quality.

Starting Beamer

Creating a Beamer presentation is similar to LaTeX. The basic syntax is as follows:

Basic syntax

```
\documentclass{beamer}
\usetheme{Madrid}
\begin{document}
\begin{frame}{My first frame}
  Hello world!
\end{frame}.
\end{document}
```

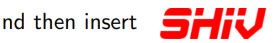
Starting Beamer

Creating a Beamer presentation is similar to LaTeX. The basic syntax is as follows:

Basic syntax

```
\documentclass{beamer}
\usetheme{Madrid}
\begin{document}
\begin{frame}{My first frame}
  Hello world!
\end{frame}.
\end{document}
```

Note: We can also create a title page in similar fashion just like in LaTeX . To insert a graphics in title page, use `\titlegraphic{}` command in preamble and insert link of image here like we do in LaTeX. To use footnote in titlepage, use `\thanks{}`.
To use logo, we can use `\logo{}` in preamble, and then insert link there like in it like in LaTeX .



Themes in Beamer

Beamer has many themes:

- `Madrid`
- `Warsaw`
- `Berkeley`
- `CambridgeUS`
- `AnnArbor`

We can try different ones using `\usetheme{ThemeName}`

Note: There is a option to use color theme in addition to the main theme. It is used mainly to change the color of selected theme. We can use `\usecolortheme{ThemeName}` to change color theme. One such example is Beaver.

Block

There are three basic blocks which contains content in a container which makes it different from others.

Standard Block

Syntax: `\begin{block}{enter Title}...\end{block}`

Block

There are three basic blocks which contains content in a container which makes it different from others.

Standard Block

Syntax: `\begin{block}{enter Title}...\end{block}`

Alert Message

Syntax: `\begin{alertblock}{enter Title}...\end{alertblock}`

Block

There are three basic blocks which contains content in a container which makes it different from others.

Standard Block

Syntax: `\begin{block}{enter Title}...\end{block}`

Alert Message

Syntax: `\begin{alertblock}{enter Title}...\end{alertblock}`

Example

Syntax: `\begin{exampleblock}{Title}...\end{exampleblock}`

Block

There are three basic blocks which contains content in a container which makes it different from others.

Standard Block

Syntax: `\begin{block}{enter Title}...\end{block}`

Alert Message

Syntax: `\begin{alertblock}{enter Title}...\end{alertblock}`

Example

Syntax: `\begin{exampleblock}{Title}...\end{exampleblock}`

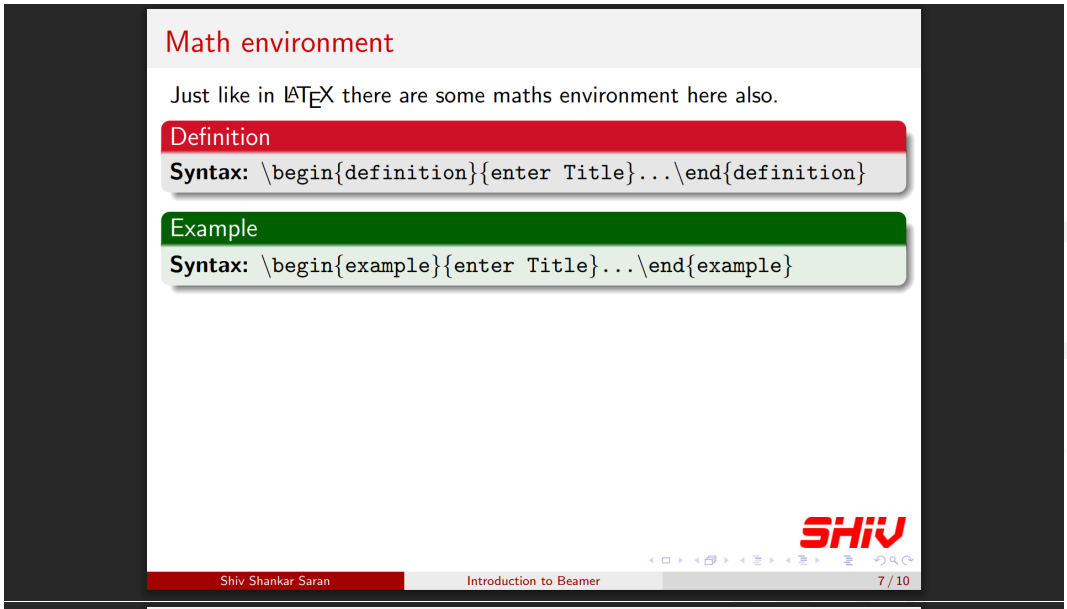
Note: To change title color of block use in preamble:`\setbeamercolor{alertblock title}{bg=red, fg=green}`. For body, use: `\setbeamercolor{block body}{bg=red}`.

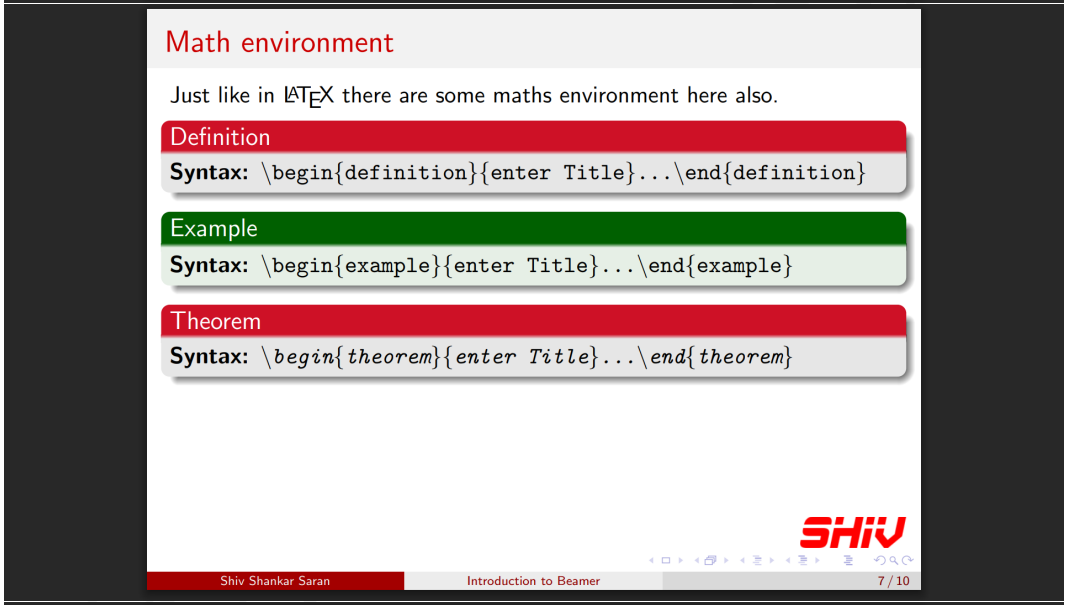
Math environment

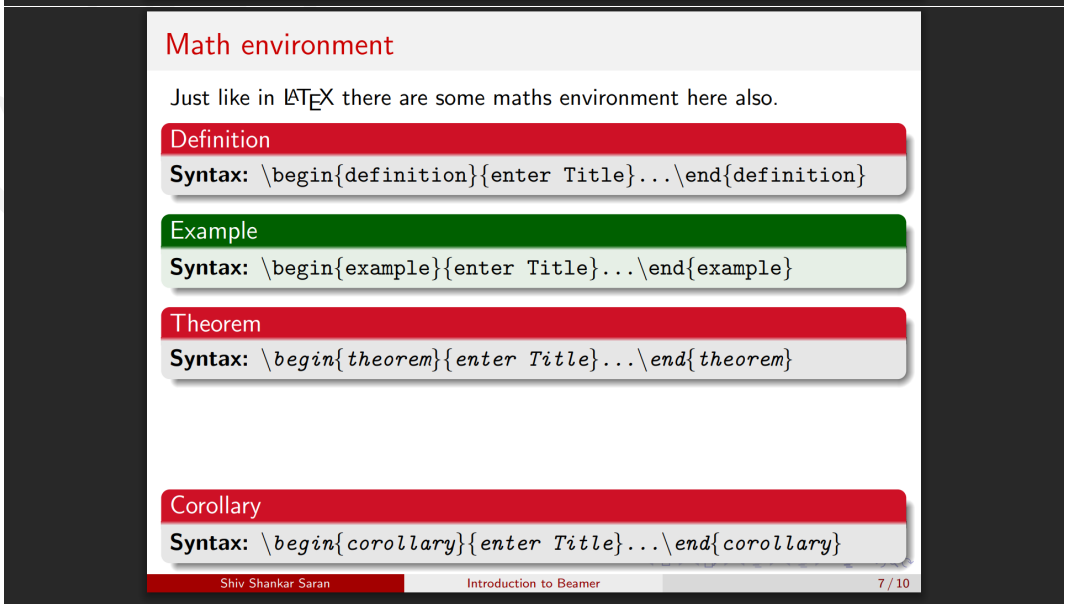
Just like in LaTeX there are some maths environment here also.

Definition

Syntax: `\begin{definition}{enter Title}...\end{definition}`


Math environment
Just like in LaTeX there are some maths environment here also.
Definition
Syntax: \begin{definition}{enter Title}...\end{definition}
Example
Syntax: \begin{example}{enter Title}...\end{example}
SHiV
Shiv Shankar Saran
Introduction to Beamer
7 / 10


Math environment
Just like in LaTeX there are some maths environment here also.
Definition
Syntax: \begin{definition}{enter Title}...\end{definition}
Example
Syntax: \begin{example}{enter Title}...\end{example}
Theorem
Syntax: \begin{theorem}{enter Title}...\end{theorem}
SHiV
Shiv Shankar Saran
Introduction to Beamer
7 / 10


Math environment
Just like in LaTeX there are some maths environment here also.
Definition
Syntax: \begin{definition}{enter Title}...\end{definition}
Example
Syntax: \begin{example}{enter Title}...\end{example}
Theorem
Syntax: \begin{theorem}{enter Title}...\end{theorem}
Corollary
Syntax: \begin{corollary}{enter Title}...\end{corollary}
Shiv Shankar Saran
Introduction to Beamer
7 / 10

Math environment

Just like in LaTeX there are some maths environment here also.

Definition

Syntax: `\begin{definition}{enter Title}...\end{definition}`

Example

Syntax: `\begin{example}{enter Title}...\end{example}`

Theorem

Syntax: `\begin{theorem}{enter Title}...\end{theorem}`

Proof.

Syntax: `\begin{proof}{enter Title}...\end{proof}` □

Corollary

Syntax: `\begin{corollary}{enter Title}...\end{corollary}`

Shiv Shankar Saran | Introduction to Beamer | 7 / 10

Overlays

Before understanding Overlays, first look at the difference between frame and slide.

SHiV

Shiv Shankar Saran | Introduction to Beamer | 8 / 10

Overlays

- **Frame**: The environment that defines one presentation unit.

SHiV

Shiv Shankar Saran | Introduction to Beamer | 8 / 10

Overlays

- **Frame**: The environment that defines one presentation unit.
- **Slide**: The visible output seen by the audience when the frame is displayed.

Overlays

- **Frame**: The environment that defines one presentation unit.
- **Slide**: The visible output seen by the audience when the frame is displayed.

Note: A single frame can produce multiple slides using overlays

Overlays

Examples

We can create overlays with `\pause` command. It offers the easiest way to create the frame with multiple slides.
Basically what happen is that whenever we put `\pause` at a given point in the frame, the first slide contains all the elements until this first appearance. Then the following slide will contain all the elements until the second appearance of `\pause`, and so on until the last appearance, where the remaining elements of the frame are shown in the last slide.

Overlays

Examples

We can create overlays with \pause command. It offers the easiest way to create the frame with multiple slides.
Basically what happen is that whenever we put \pause at a given point in the frame, the first slide contains all the elements until this first appearance. Then the following slide will contain all the elements until the second appearance of \pause, and so on until the last appearance, where the remaining elements of the frame are shown in the last slide.
This is the line after \pause.

Overlays

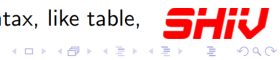
Examples

We can create overlays with \pause command. It offers the easiest way to create the frame with multiple slides.
Basically what happen is that whenever we put \pause at a given point in the frame, the first slide contains all the elements until this first appearance. Then the following slide will contain all the elements until the second appearance of \pause, and so on until the last appearance, where the remaining elements of the frame are shown in the last slide.
This is the line after \pause.
Note: Sometimes we want something to be shown in only one slide, and other on next slide. In this we can insert text in \only<i>{...} command. Here 'i' means it will be shown on only on *i* slide. If we want it to show from i slide to j slide, then use <i-j>. If we want to show it to last slide, then <i->.
If we want to do the same with some environment, then \begin{environmentName}<i> will work. For lists, use \item<i>.

There are some common errors that we usually do in Beamer. So we can keep some points in our mind.

- While using \verb|| enviornment, always use **fragile** in frame like this: \begin{frame}[fragile]{framename}.
  Try to avoid use of **verbatim** here as it creates some error here in beamer!
- While making table of content, remember that table of content do not include frame title. It only includes section name. So ensure that we have used section in it or not, because most of the time we only create frame and do not think about section.
  We can also use **tableofcontents[hideallsubsections]**
- By default text in Beamer are left-aligned but we can make it justified by using package-ragged2e and using command \justifying before starting text.

Note: Every other things which are not discussed here can be treated as same as in LaTeX , meaning that it has same syntax, like table, image, lists, etc.




Thank You!
SHiV
Shiv Shankar Saran
Introduction to Beamer
10 / 10


Thank You!
SHiV
Shiv Shankar Saran
Introduction to Beamer
10 / 10


Thank You!
SHiV
SHiV
Shiv Shankar Saran
Introduction to Beamer
10 / 10

10 PSTricks

Introduction

PSTricks is a powerful graphics package for LaTeX used to create high-quality PostScript-based illustrations directly within LaTeX documents. It allows precise drawing of shapes, graphs, and plots without external image files. PSTricks works by embedding PostScript commands inside LaTeX code. When compiled with a PostScript-compatible workflow, it produces vector graphics with typographic precision.

10.1 Starting PSTricks

Before we start working with PSTricks, we have to do some setup.

1. Change compiler to XeLaTex.
2. Use package 'pstricks' in preamble.
3. All set! We can now simply write commands and get our desired results.

10.1.1 pspicture environment

The pspicture environment in PSTricks is the main area used for creating drawings and graphics in LaTeX documents. It defines a rectangular region in which all graphical objects such as lines, circles, curves, and shapes are drawn. It works on a coordinate system, where every point is represented by an x and y value. The lower-left and upper-right corners of the drawing area are specified when defining the environment. Inside this area, different PSTricks commands can be used to create geometric figures, plots, and labels.
Syntax:

```
\begin{pspicture}(xMin,yMin)(xMax,yMax)
    ....
\end{pspicture}
```

10.1.2 psgrid command

The psgrid command in PSTricks is used to draw a reference grid inside the pspicture environment. It helps visualize coordinates, align objects, and understand positions in the drawing area.
Syntax: `\psgrid`

10.1.3 psaxes command

The `\psaxes` command in PSTricks is used to draw coordinate axes inside the pspicture environment.
Syntax:
`\psaxes[option]{<->}(xOrigin,yOrigin)(xMin,yMin)(xMax,yMax)`

Remark:

- To change the compiler, click on Menu in the top left corner of overleaf window. In settings, there is a option to change compiler.
- Default compiler is pdfLaTeX. We can change it to either XeLaTeX or LaTeX.
- By default, `\psgrid` draw a grid over the entire pspicture area. If we want grid over a specific area, then we can define the area by using `\psgrid(x1,y1)(x2,y2)`.
- There are different options available in `\psaxes` like `linecolor=color` and `labels=none/all/x/y`.

10.2 Plotting points

In PSTricks, plotting of points can be done using command `\psdots`.
Syntax: `\psdots[options](x1,y1)(x2,y2)...(xn,yn)`
Example:

```
\begin{center}
\begin{pspicture}(-3,-3)(3,3)
\psaxes[labels=all,linecolor=royalblue]{<->}(0,0)(-3,-3)(3,3)
    \psdots[dotsize=10pt,linecolor=green, dotstyle=x](0,0)
    \psdots[dotsize=7pt,linecolor=red, dotstyle=+](2,0)
    (1.414,1.414)(0,2)(-1.414,1.414)(-2,0)(-1.414,-1.414)
    (0,-2)(1.414,-1.414)
    \psdots[dotsize=5pt,linecolor=blue, dotstyle=o](1,0)(0,1)
    (-1,0)(0,-1)
\end{pspicture}
\end{center}
```

Output:

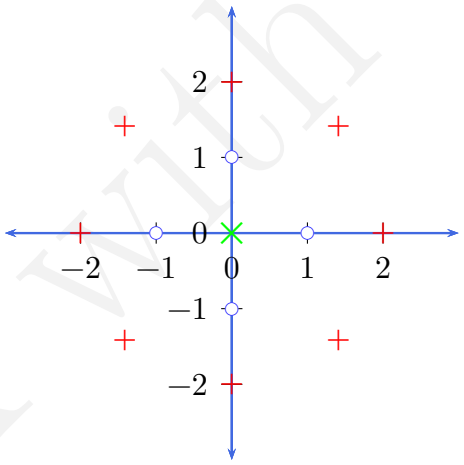


Figure 10.1: Points

Remark:

- There are different options available in `\psdots`. Again by using linecolor we can change tehe color of dots; dotsize is for siize; dotstyle is for styling the dots.
- In dotstyle there are different values available like “x,o,+,*(solid)”. By default it is solid.

10.3 Lines

In PSTricks, lines are drawn using the `\psline` command inside the pspicture environment. It connects two or more coordinate points with straight line segments.

Syntax: `\psline[options](x1,y1)(x2,y2)...(xn,yn)`

Example:

```
\begin{pspicture}(-3,-3)(3,3)
    \psgrid(-3,-3)(3,3)
    \psset{linewidth=1.5pt, arrows=o-o, linecolor=royalblue,
    linestyle=solid}
    \psline[linecolor=royalblue,linestyle=dashed](-3,0)(0,3)
    (3,0)(0,-3)(-3,0)
    \psline[linecolor=brown, linestyle=dotted](-2,0)(0,2)(2,0)
    (0,-2)(-2,0)
    \psline[linecolor=orange, linestyle=dashed](-1,0)(0,1)(1,0)
    (0,-1)(-1,0) \psline(-3,-3)(3,3)
    \psline(-3,3)(3,-3)
    \psset{linecolor=purple}
    \psline(-3,1.5)(3,-1.5)
    \psline(-3,-1.5)(3,1.5)
    \psline(-1.5,3)(1.5,-3)
    \psline(1.5,3)(-1.5,-3)
    \psdots[linecolor=black,dotstyle=o,dotsize=6pt](0,0)
\end{pspicture}
```

Output:

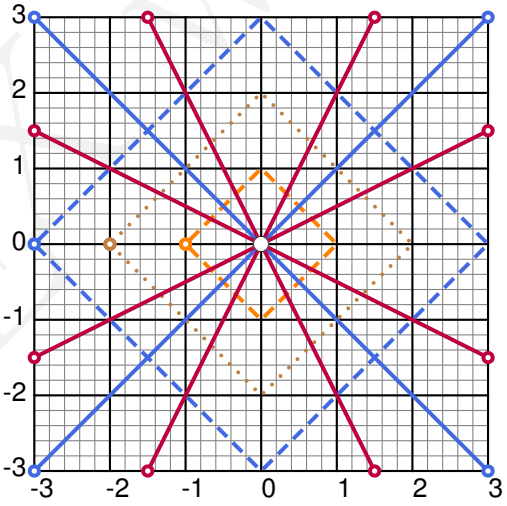


Figure 10.2: Lines

Remark:

- `\psset` is a command in PSTricks used to set or change drawing parameters globally or locally within a picture.
- There are different options available in `\psline` like linewidth, arrows, linecolor, linestyle(dotted, dashed or solid).

10.4 Circle

To draw circles in PSTricks, use the `\pscircle` command inside a pspicture environment.
Syntax: `\pscircle[options](<center_x>,<center_y>){<radius>}`
Example:

```
\begin{center}
\begin{pspicture}(-3,-3)(3,3)
\psset{linecolor=white, linewidth=2pt, linestyle= dashed,
opacity=0.9}
  \pscircle[linecolor=royalblue](0,0){3}
  \pscircle[fillstyle=solid, fillcolor=royalblue, linewidth=2pt]
  (0,0){3}
  \pscircle[fillstyle=solid, fillcolor=red!70](0,0){2.4}
  \pscircle[fillstyle=solid, fillcolor=yellow!70](0,0){1.8}
  \pscircle[fillstyle=solid, fillcolor=lightgray](0,0){1.2}
  \pscircle[fillstyle=crosshatch](0,0){0.6}
  \psdots[linecolor=royalblue, dotsize=8pt](0,0)
\end{pspicture}
\end{center}
```

Output:

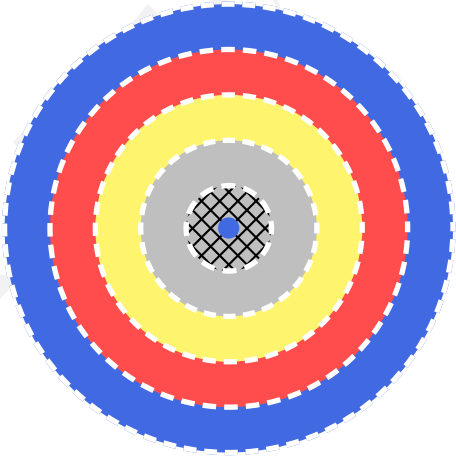


Figure 10.3: Circles

Remark:

- Options available in circle are linecolor, linewidth, fillstyle, fillcolor, linestyle, opacity.
- fillstyle includes none, solid, crosshatch.
- Opacity is for transparency of the shape (0–1).

10.5 Polygon

To draw a polygon in PSTricks, you use the command \pspolygon.
Syntax: \pspolygon[options](<x1>,<y1>)(<x2>,<y2>)...(<xn>,<yn>)
Example:

```
\begin{pspicture}(-4,-4)(4,4)
  \pspolygon[linecolor=royalblue, linestyle=dashed, fillstyle=
  crosshatch](-4,-4)(4,-4)(4,4)(-4,4)
  \pspolygon[linewidth=2pt, linecolor=red, fillstyle=solid,
  fillcolor=yellow!40]
    (0,4)(3.46,2)(3.46,-2)(0,-4)(-3.46,-2)(-3.46,2)
  \pspolygon[linewidth=2pt, linecolor=blue, fillstyle=solid,
  fillcolor=royalblue!40]
    (2,3.46)(4,0)(2,-3.46)(-2,-3.46)(-4,0)(-2,3.46)
  \pspolygon[linewidth=1.5pt, linecolor=green, fillstyle=solid,
  fillcolor=lime!40]
    (0,2)(1.73,1)(1.73,-1)(0,-2)(-1.73,-1)(-1.73,1)
  \psdots[linecolor=black, dotsize=6pt]
    (0,4)(3.46,2)(3.46,-2)(0,-4)(-3.46,-2)(-3.46,2)
\end{pspicture}
```

Output:

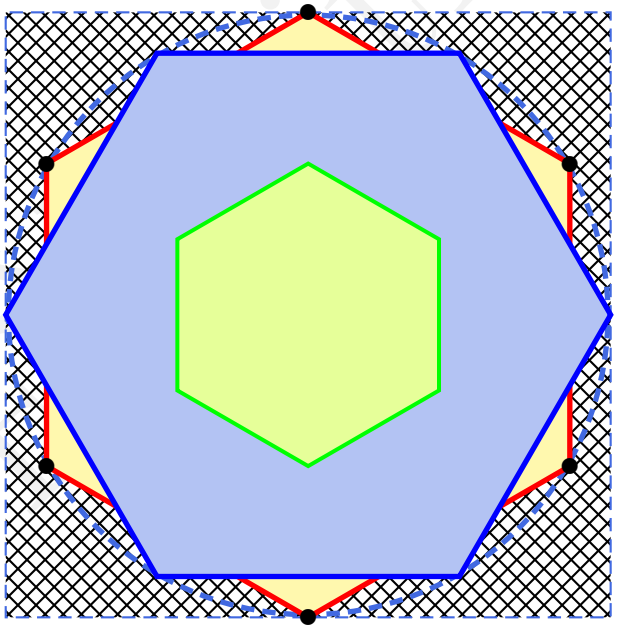


Figure 10.4: Polygons

Remark:

- Options available in polygons are linecolor, linewidth, fillstyle, fillcolor, linestyle, opacity.
- W can also use polar coordinates in place of cartesian.
  Syntax:$(x, y) - > (r; \theta)$

10.6 Arcs and Wedges

In PSTricks, both `\psarc` and `\pswedge` draw circular portions — but they differ in whether the area is filled.In wedge, area is filled.
Syntax: `\psarc[options](center){radius}{startAngle}{endAngle}`
Syntax: `\pswedge[options](center){radius}{startAngle}{endAngle}`
Example:

```
\begin{pspicture}(-6,-3)(6,3)
  \pspolygon[linecolor=lime,linewidth=5pt](-3,-3)(3,-3)(3,3)
  (-3,3)
  \pspolygon[linecolor=lime,linewidth=5pt](-6,-2)(-6,2)(-3,2)
  (-3,-2)
  \rput{90}(-4.5,0){\textcolor{lime}{\scalebox{5}{BEN}}}
  \rput(4.5,0){\textcolor{lime}{\scalebox{7}{10}}}
  \pspolygon[linecolor=lime,linewidth=5pt](6,-2)(6,2)(3,2)(3,-2)
  \pscircle[linewidth=3pt,linecolor=lime](0,0){3}
  \pscircle[linewidth=3pt,fillstyle=solid,fillcolor=black,
  linecolor=black](0,0){2.4}
  \pswedge[fillstyle=solid,fillcolor=lime,linecolor=lime](0,0)
  {2.2}{30}{150}
  \pswedge[fillstyle=solid,fillcolor=lime,linecolor=lime](0,0)
  {2.2}{210}{330}
  \pscircle*[linecolor=white](0,0){0.8}
  \psset{linewidth=3pt, linecolor=lime}
  \pscircle(0,0){0.8}
  \psarc[arrows=o->](0,0){2.6}{30}{150}
  \psarc[arrows=o->](0,0){2.6}{210}{330}
  \pscircle[linestyle=dotted](0,0){2.2}
\end{pspicture}
```

Output:

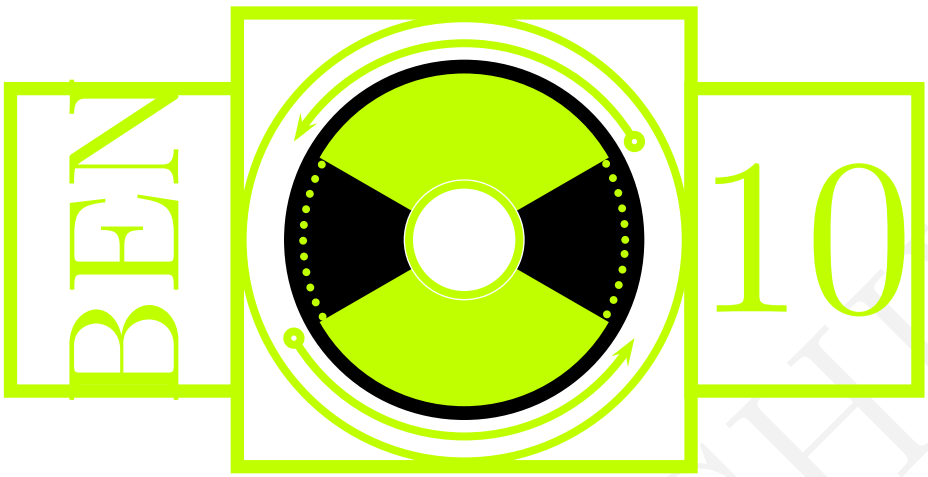


Figure 10.5: Arcs and Wedges

Remark:

- `\rput` stands for "rotated put", and it's used to place (and optionally rotate) any text, symbol, or graphic at a specified coordinate within a PSTricks picture.
- Syntax: `\rput[<refpoint>]{<angle>}(x, y){<content>}`. Common reference point are: t,l,c,r,b,tl,tr,bl,br.
- <angle> — rotation in degrees, counterclockwise.

10.7 Ellipses

In PSTricks,`\psellipse` draws an ellipse or circle. This is the basic command to draw ellipses (and circles, if both radii are equal).
Syntax: `\psellipse[options](xCenter, yCenter)(xRadius, yRadius)`
Example:

```
\begin{pspicture}(-6,-2.7)(6,2.7)
  \pspolygon[fillstyle=solid,fillcolor=blue!50](-6,-2.6)(6,-2.6)
  (6,2.6)(-6,2.6)
  \pscircle[fillstyle=solid,fillcolor=yellow](0,0){1}
  \rput(0,0){\textcolor{orange}{\Huge Sun}}
  \psellipse[linecolor=white,linestyle=dashed](0,0)(2.5,1.5)
  \psellipse[linecolor=white,linestyle=dashed](0,0)(3.5,2)
  \psellipse[linecolor=white,linestyle=dashed](0,0)(4.5,2.5)
  \psset{fillstyle=solid}
  \rput(2.5,0){\pscircle[fillcolor=gray](0,0){0.2}}
  \rput(2.5,-0.4){\textcolor{white}{Mercury}}
  \rput(-3.5,0){\pscircle[fillcolor=orange](0,0){0.25}}
  \rput(-3.5,-0.4){\textcolor{white}{Venus}}
  \rput(4.5,0){\pscircle[fillcolor=blue](0,0){0.3}}
  \rput(4.5,-0.4){\textcolor{white}{Earth}}
\end{pspicture}
```

Output:

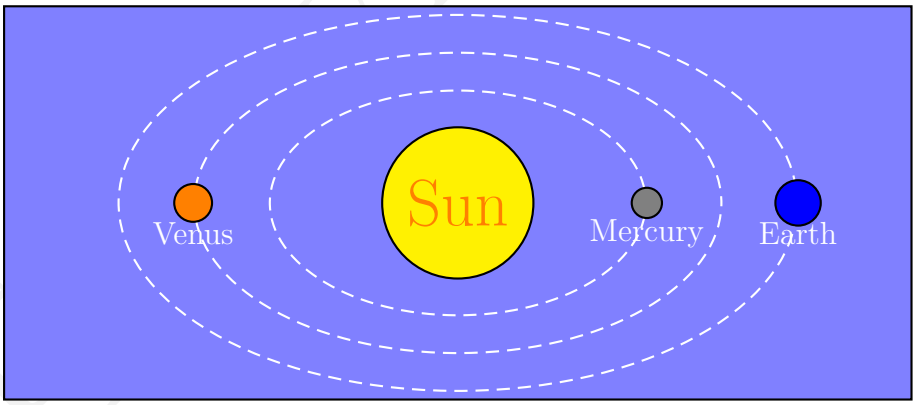


Figure 10.6: Ellipse

Remark:

- The options in ellipse are same as in others, with one addition is of rot. It is the rotation angle through which it is rotated.
- There is a `\psclip` environment which is used to define a clipping region — anything drawn afterward will be visible only inside that region. Syntax: `\psclip{<shape command>} ... \endpsclip`

10.8 Plotting functions

To plot a function using PSTricks, we need the `pst-plot` package. We can include it by using `\usepackage{pst-plot}`. For function graphs, it draws many points and joins them with line segments.
Syntax: `\psplot[options]{xmin}{xmax}{function}`

PSTricks uses PostScript internally. We can't simply write the function as it is. Suppose our function is `y=f(x)`, then we have to write it as `x dup mul`.

To convert a mathematical function (in standard infix notation, such as `a+b×c`) to its PostScript (postfix) expression, we need to rewrite the formula so that each operator follows its operands. Below are some examples.

1. **Infix:** $a + b$
   Postfix (PostScript): `a b add`
2. **Infix:** $a \times (b + c)$
   Postfix (PostScript): `a b c add mul`
3. **Infix:** $(a + b) \times (c - d)$
   Postfix (PostScript): `a b add c d sub mul`

Example: $y = x^2 + 2 - (5 \times x^2) - \frac{x}{5}$

```
\begin{figure}[h]
  \centering
  \begin{pspicture}(-5,-3)(5,3)
  \psset{xunit=01cm, yunit=01cm, plotpoints=2500,
  linewidth=2pt}
  \psgrid(0,0)(-5,-3)(5,3)
  \psaxes[ticks=none, labels=none]{<->}(0,0)(-5,-2.5)(5,2.5)
  \psplot[linestyle=dotted, linecolor=blue, arrows=<->]
  {-1}{1}
  {x x mul 2 add 5 x x mul mul sub x 5 div sub}
  \end{pspicture}
  \caption{Functions}
\end{figure}
```

Output:

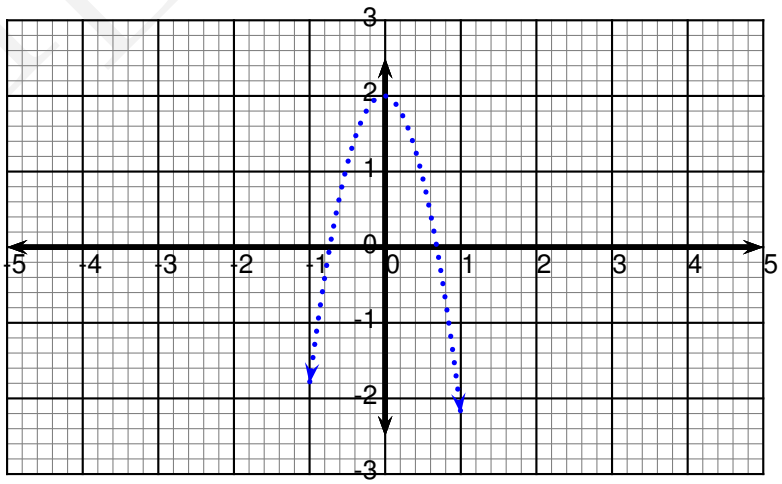


Figure 10.7: Functions

10.8.1 Rational Functions

$$y = \frac{1}{x} \quad ; x \neq 0$$

```
\begin{pspicture}(-3,-3)(3,3)
\psaxes{<->}(0,0)(-3,-3)(3,3)
\psplot[linecolor=blue]{-3}{-0.35}{1 x div}
\psplot[linecolor=blue]{0.35}{3}{1 x div}
\end{pspicture}
```

Output:

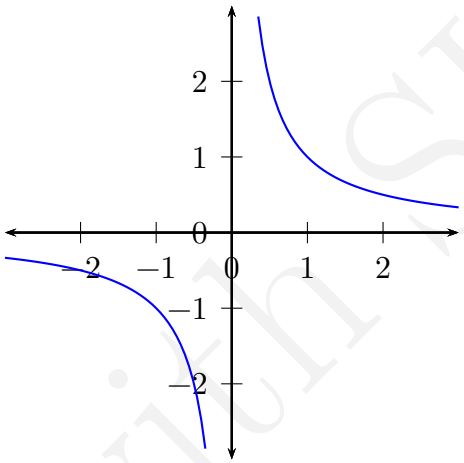


Figure 10.8: Rational functions

10.8.2 Exponential and Logarithmic Functions

```
\begin{pspicture}(-3,-3)(3,3)
    \psaxes{<->}(0,0)(-3,-3)(3,3)
    \rput[dr](3.25,0){$x$}
    \rput[dl](0,3.2){$y$}
    \psplot[linecolor=blue,linewidth=1.2pt]{-3}{1.1}
    {Euler x exp}
    \rput[l](1.6,3){\textcolor{blue}{$y = e^x$}}
    \psplot[linecolor=red,linewidth=1.2pt]{0.05}{3}{x ln}
    \rput[r](3,1.5){\textcolor{red}{$y = \ln x$}}
    \rput(4.5,2.3){
    \psframebox{
        \begin{tabular}{l}
        {\color{blue}\rule{1em}{1pt}} $y = e^x$ \\
        {\color{red}\rule{1em}{1pt}} $y = \ln x$
        \end{tabular}}
    }
\end{pspicture}
```

Output:

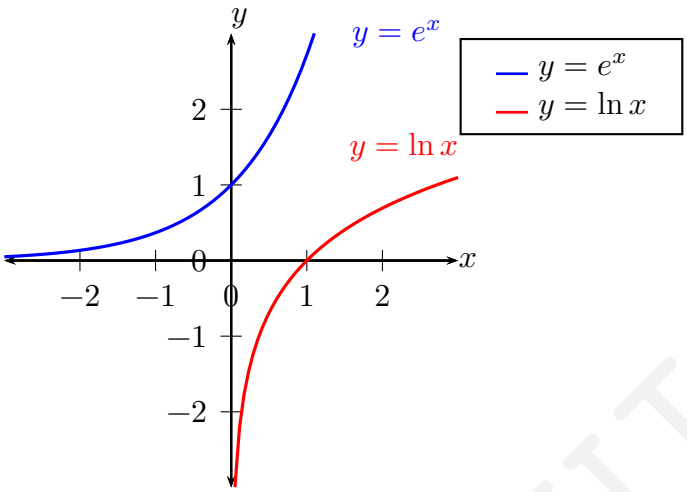


Figure 10.9: Exponential and Logarithmic Functions

10.8.3 Trigonometric and Inverse Trigonometric Functions

```
\begin{pspicture}(-1,-2.5)(7,3)
\psaxes[Dx=1.57,Dy=0.5, labels=none]{->}(0,0)(-0.5,-2.5)
(6.5,2.5)[$x$,0][$y$,90]
\psplot[linecolor=blue,linewidth=1.5pt,plotpoints=500,algebraic]
{0}{6.283}{sin(x) - 1}
\rput[l](6.3,-0.5){\textcolor{blue}{$\sin x - 1$}}
\psplot[linecolor=red,linewidth=1.5pt,plotpoints=500,algebraic]
{0}{6.283}{sin(x)}
\rput[l](6.3,0.5){\textcolor{red}{$\sin x$}}
\psplot[linecolor=orange,linewidth=1.5pt,plotpoints=500,
algebraic]{0}{6.283}{sin(x) + 1}
\rput[l](6.3,1.5){\textcolor{orange}{$\sin x + 1$}}
\rput[t](1.5708,-0.5){$\frac{\pi}{2}$}
\rput[t](3.1416,-0.5){$\pi$}
\rput[t](4.7124,-0.5){$\frac{3\pi}{2}$}
\rput[t](6.2832,-0.5){$2\pi$}
\rput[l](-0.5,1){1}
\rput[l](-0.7,1.5){1.5}
\rput[l](-0.7,0.5){0.5}
\end{pspicture}

\begin{pspicture}(-1.8,-2.5)(2.5,3.5)
  \psaxes[Dx=0.5,Dy=1.5708,labels=none]{<->}(0,0)(-1.5,-2.5)
  (1.5,3.5)[$x$,0][$y$,90]
  \psplot[linecolor=blue,linewidth=1.5pt,plotpoints=2000,
  algebraic]{-1}{1}{asin(x)}
  \psplot[linecolor=red,linewidth=1.5pt,plotpoints=2000,
  algebraic]{-1}{1}{acos(x)}
  \rput[d](-1,0){$-1$}
  \rput[d](-0.5,0){$-\frac{1}{2}$}
  \rput[d](0,0){$0$}
  \rput[d](0.5,0){$\frac{1}{2}$}
  \rput[d](1,0){$1$}
  \rput[r](0,-1.5708){$-\frac{\pi}{2}$}
```

```
  \rput[r](0,0){$0$}
  \rput[r](0,1.5708){$\frac{\pi}{2}$}
  \rput[r](0,3.1416){$\pi$}
  \rput(1.2,1.3){
    \psframe[fillstyle=solid,fillcolor=white,linestyle=dashed]
    (0,0)(2.7,1.3)
    \psline[linewidth=1.5pt,linecolor=blue](0.2,1)(0.8,1)
    \rput[l](1.2,1){$\arcsin x$}
    \psline[linewidth=1.5pt,linecolor=red](0.2,0.5)(0.8,0.5)
    \rput[l](1.2,0.5){$\arccos x$}
  }
\end{pspicture}
```

Output:

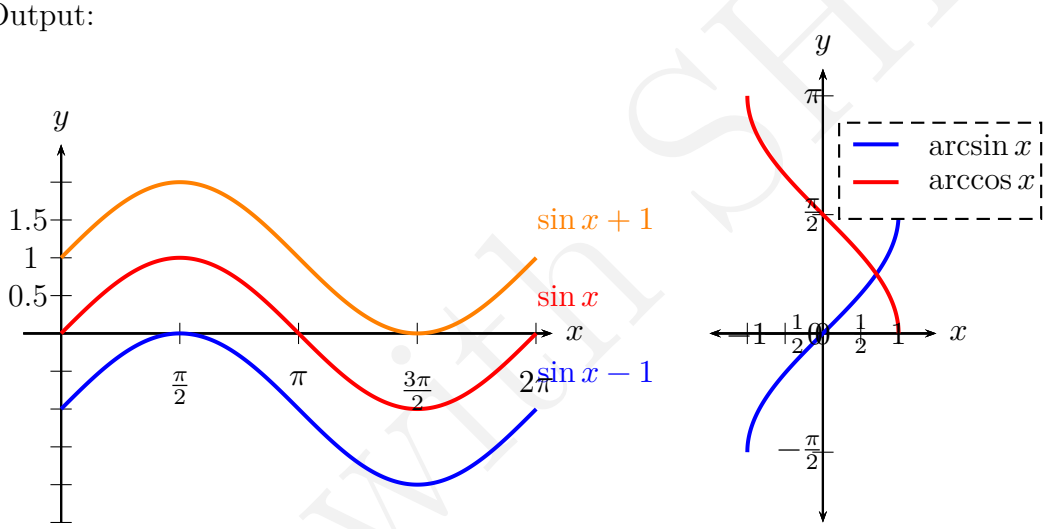


Figure 10.10: Trigonometric and Inverse Trigonometric Functions

10.8.4 Parametric Plot

Syntax: `\parametricplot[options]{t_min}{t_max}{ x(t) | y(t) }`

```
\begin{pspicture}(-2,-5.5)(10,5.5)
\psframe*[linecolor=pink!20](-2,-5.5)(10,5.5)
  \psframe*[linecolor=gray](-0.35,-5)(-0.7,5)
  \psframe*[linecolor=darkgray](0,-5)(-1,-4.5)
  \psframe*[linecolor=orange](-0.35,4.5)(8,3)
  \psframe*[linecolor=white](-0.35,3)(8,1.5)
  \psframe*[linecolor=green](-0.35,1.5)(8,0)
\parametricplot[algebraic,linecolor=blue, linewidth=2pt]{0}
{6.283}{0.75*cos(t)+4 |
      0.75*sin(t)+2.25}
\parametricplot[algebraic,linecolor=blue, linewidth=2pt]{0}
{6.283}{0.15*cos(t)+4 |
      0.15*sin(t)+2.25}
\parametricplot[algebraic,linecolor=blue, linewidth=2pt]{0}
{6.283}{0.25*cos(t)+4 |
      0.25*sin(t)+2.25}
\parametricplot[algebraic,linecolor=blue, linewidth=2pt]{0}
{6.283}{0.45*cos(t)+4 |
      0.45*sin(t)+2.25}
\parametricplot[algebraic,linecolor=brown]{0}{6.283}
{0.25*cos(t)-0.53 | 0.25*sin(t)+4.85}
\parametricplot[linewidth=2pt,linecolor=brown,algebraic]{0}{6.283}
{cos(t)/(1 + sin(t)^2)-0.5 | cos(t)*sin(t)/(1 + sin(t)^2)+4.85}
\end{pspicture}
```

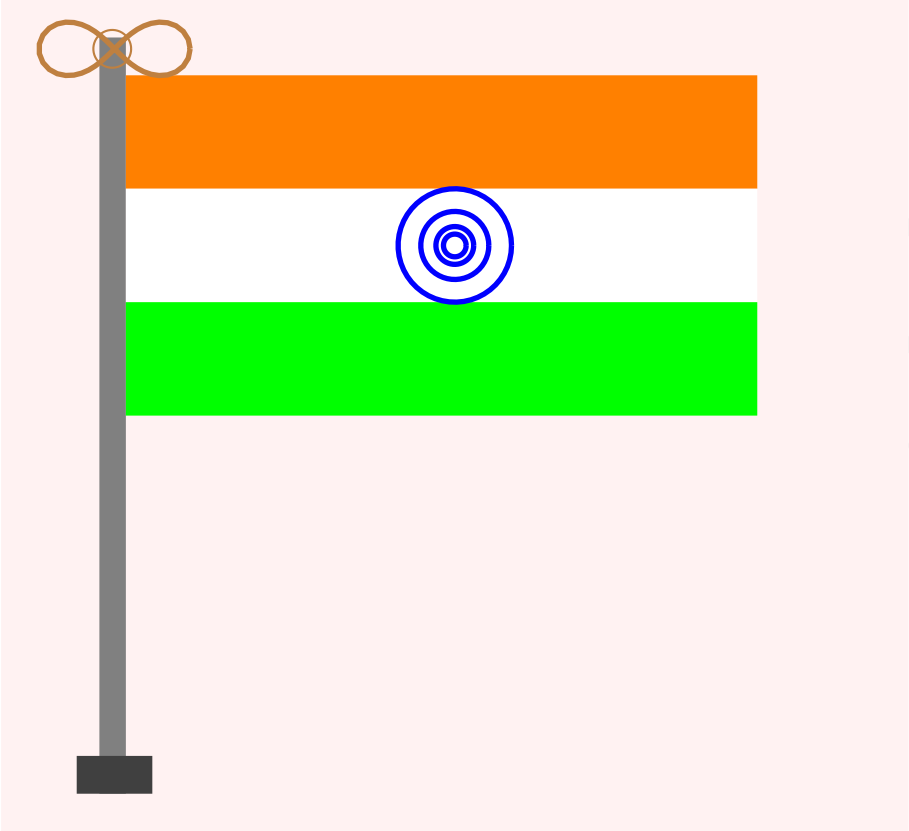


Figure 10.11: Parametric Plot

Remark:

- In psaxes, we can set ticks and labels to none to suppress ticks and numbers on axes.
- We can set length of unit in our plot by using `xunit or yunit` or simply by `unit`.
- `plotpoints` represents to number of points which are used to plot the function. More the plotpoints, more smoother the curve.
- We can create a small box in the picture using `\psframebox`.
- In psaxes, Dx and Dy control the spacing between tick marks on the x-axis and y-axis. We can also set ticks=none and labels=none to hide them.
- **We can also write any function as we write in math mode. We just have to include `algebraic` in square bracket after `\psplot`.**
- `\psframe*` draws a filled rectangle with no border line. We can include option as **linecolor** in it to specify color.

11 Demonstration of Web Resources

Introduction

Mathematical research, in the digital age, is supported by a rich ecosystem of web-based resources that assist in every stage of the research process—from finding literature to publishing work. In this chapter we explores web platforms grouped into key categories relevant to the practice of mathematics research.

11.1 Research Paper Databases

11.1.1 MathSciNet

Website: `https://mathscinet.ams.org`
MathSciNet is a database maintained by the American Mathematical Society (AMS). It provides reviews and bibliographic information for articles in mathematics journals. It allows researchers to search by topic, author, or MSC (Mathematics Subject Classification) codes.

11.1.2 1.2 zbMATH Open

Website: `https://zbmath.org`
zbMATH Open is an openly accessible alternative to MathSciNet, run by the European Mathematical Society. It provides abstracts, keywords, and sometimes full-text links to mathematical publications across journals and conferences.

Remark:

- Researchers often cross-check results by comparing MathSciNet reviews with zbMATH summaries.
- Both platforms track MSC (Mathematics Subject Classification) codes, which help in organizing literature by topic.

11.2 Preprint Archives

11.2.1 arXiv.org (Mathematics Section)

Website: `https://arxiv.org/archive/math`
arXiv is a free distribution service and open-access archive for scholarly articles. In the mathematics section, researchers upload preprints of papers that are yet to be peer-reviewed. It is useful for keeping up with the latest findings in all fields of mathematics.

11.2.2 HAL Archives Ouvertes (France)

Website: `https://hal.science`
HAL is an open archive where authors can deposit scholarly documents from all academic fields. It includes a significant collection of mathematics research papers, especially from French institutions.

Remark:

- Preprints enable early dissemination of research results before journal publication.
- arXiv versions are frequently used when the published version is inaccessible due to paywalls.

11.3 Bibliography and Citation Tools

11.3.1 Google Scholar

Website: `https://scholar.google.com`
Google Scholar allows users to search for academic papers and export BibTeX-formatted citations for use in LaTeX. It provides citation counts and links to related articles.

11.3.2 Mendeley / Zotero

Websites: `https://www.mendeley.com`, `https://www.zotero.org`
These are free reference managers that help in organizing, annotating, and citing research articles. Both can be integrated with BibTeX for LaTeX users.

Remark:

- Reference managers prevent citation errors, especially in lengthy mathematical documents.
- BibTeX compatibility ensures seamless integration with LaTeX-based writing.

11.4 Mathematical Encyclopedias and Databases

11.4.1 Wolfram MathWorld

Website: `https://mathworld.wolfram.com`
MathWorld is a comprehensive mathematics encyclopedia created by Wolfram Research. It includes thousands of entries on topics from algebra to calculus to advanced geometry.

11.4.2 Encyclopedia of Mathematics (EOM)

Website: `https://encyclopediaofmath.org`
EOM is a collaborative online encyclopedia supported by Springer and the European Mathematical Society. It covers detailed, high-level topics with references to original research.

11.4.3 OEIS (Online Encyclopedia of Integer Sequences)

Website: `https://oeis.org`
The OEIS is a searchable database of integer sequences. Each entry includes formulae, references, and connections to other areas of mathematics.

Remark:

- These encyclopedias help quickly recall definitions, theorems, and formulas during research.
- OEIS is widely used in number theory to identify unknown integer sequences.

11.5 Journals and Publishers

11.5.1 Springer Mathematics

Website: `https://www.springer.com/gp/mathematics`
Springer publishes hundreds of journals and books in pure and applied mathematics. The site allows browsing journals by subject and checking submission guidelines.

11.5.2 Elsevier Mathematics

Website: `https://www.elsevier.com/mathematics`
Elsevier's platform includes well-known journals such as *Journal of Mathematical Analysis and Applications* and *Linear Algebra and its Applications*.

11.5.3 Taylor & Francis Math

Website: `https://www.tandfonline.com/math`
This publisher hosts several journals focusing on mathematical modeling, analysis, and computational mathematics.

Remark:

- Different publishers require authors to follow specific formatting and submission guidelines.
- Researchers often choose journals based on scope, peer-review quality, and impact factor.

11.6 Community and Q&A Platforms

11.6.1 6.1 MathOverflow

Website: `https://mathoverflow.net`
MathOverflow is a Q&A site for professional mathematicians. Questions are often research-level and include detailed discussions.

11.6.2 6.2 Mathematics Stack Exchange

Website: `https://math.stackexchange.com`
This site caters to a wider range of mathematical questions, from high school to research level. It is useful for clarification of concepts, reviewing proofs, and peer feedback

Remark:

- MathOverflow is research-focused, while Mathematics Stack Exchange supports all levels of mathematical inquiry.
- These platforms help refine proofs, verify ideas, and learn modern solution techniques.

Bibliography

[1] Stefan Kottwitz. *LaTeX Beginner's Guide: Create Visually Appealing Texts, Articles, and Books for Business and Science*. 2nd ed. Packt Publishing, 2021. ISBN: 978-1-80107-865-8. URL: https://static.latexstudio.net/wp-content/uploads/2015/03/LaTeX_Beginners_Guide.pdf.

LaTeX Practicals

Research Methodology

Unlock the Art of Professional Typesetting with LaTeX

This **Research Methodology Edition** takes you on an immersive journey through the world of LaTeX typesetting. From **beginner-friendly tutorials** to **advanced techniques**, you'll discover:

- Elegant and professional design templates
- Expert tips for creating flawless documents
- Exclusive digital resources to enhance your workflow
- Insights into mastering both academic and creative publishing

Whether you're a student, researcher, or professional, this is your ultimate guide to producing **high-quality, polished documents** with precision and style.