

# Introduction to L<sup>A</sup>T<sub>E</sub>X

Liu Yihao

SJTU-UMJI Technology Department

August 17, 2020

Liu Yihao

- Lecture I: Hello,  $\text{\LaTeX}$
- Lecture II: Text
- Lecture III: Maths
- Lecture IV: Graphs, Tables and Code
- Lecture V: Beamer

### Getting Started

What is  $\text{\LaTeX}$

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

# Lecture I

# Hello, $\text{\LaTeX}$

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

- 1 Getting Started
  - What is L<sup>A</sup>T<sub>E</sub>X
    - Distributions and IDEs
    - Documentation
- 2 Layout of a Document
- 3 Learn More

# What is L<sup>A</sup>T<sub>E</sub>X

## Getting Started

### What is L<sup>A</sup>T<sub>E</sub>X

#### Distributions and IDEs

#### Documentation

## Layout of a Document

### A Simple Document

#### Document Classes

#### The Preamble

#### Document Body

#### Document Sections

## Learn More

### Commands

### Environments

## From Wikipedia, the free encyclopedia<sup>1</sup>

L<sup>A</sup>T<sub>E</sub>X (lah-tekh, lah-tek or lay-tek, a shortening of Lamport T<sub>E</sub>X) is a document preparation system. When writing, the writer uses plain text in markup tagging conventions to define the general structure of a document (such as [article](#), [book](#), and [letter](#)), to stylize text throughout a document (such as **bold** and *italic*), and to add citations<sup>1</sup> and cross-references.

A T<sub>E</sub>X distribution such as T<sub>E</sub>XLive or MikT<sub>E</sub>X is used to produce an output file (such as PDF or DVI) suitable for printing or digital distribution.

Within the typesetting system, its name is stylized as L<sup>A</sup>T<sub>E</sub>X.

---

<sup>1</sup>L<sup>A</sup>T<sub>E</sub>X - <https://en.wikipedia.org/wiki/Latex>

# A brief History of T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X

## Getting Started

### What is L<sup>A</sup>T<sub>E</sub>X

#### Distributions and IDEs

#### Documentation

## Layout of a Document

### A Simple Document

#### Document Classes

#### The Preamble

#### Document Body

#### Document Sections

## Learn More

### Commands

### Environments

Donald Kunuth from Stanford University is the specialist in programming art. In year 1977, he had just received his first samples from the new typesetting system of the publisher's, and its quality was so far below that of the first edition of Volume 2 that he couldn't stand it. Kunuth decided to implement a mathematical composition system by himself (since he is a computer scientist). He figured that this would take about 6 months (Ultimately, it took nearly 10 years). The system is named as T<sub>E</sub>X, of both the meaning of Greek letters  $\tau\epsilon\chi$ , and "technical".

L<sup>A</sup>T<sub>E</sub>X was created in 1983 by Leslie Lamport, when he was working at SRI. He needed to write T<sub>E</sub>X macros for his own use, and thought with a little extra effort he could make a general package usable by others. Then L<sup>A</sup>T<sub>E</sub>X developed rapidly and now there are thousands of packages written in T<sub>E</sub>X macros available for direct usage.

Liu Yihao

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

- 1 Getting Started
  - What is L<sup>A</sup>T<sub>E</sub>X
  - Distributions and IDEs
  - Documentation
- 2 Layout of a Document
- 3 Learn More

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Installation of L<sup>A</sup>T<sub>E</sub>X

Though there are some other distributions of L<sup>A</sup>T<sub>E</sub>X (like MikT<sub>E</sub>X), T<sub>E</sub>XLive is recommended in this lecture.

## Windows & Linux

Download T<sub>E</sub>XLive on the [tuna mirrors](#)

<https://mirrors.tuna.tsinghua.edu.cn/CTAN/systems/texlive/Images/>

## MacOS

Download MacT<sub>E</sub>X on the [tuna mirrors](#)

<https://mirrors.tuna.tsinghua.edu.cn/CTAN/systems/mac/mactex/>

## Linux (Debian/Ubuntu)

Enter the command (fast with apt source mirror)

```
sudo apt-get install texlive-full
```



## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Selection of IDEs

There are various IDEs recommended that support L<sup>A</sup>T<sub>E</sub>X, for example

## Texmaker

<http://www.xmlmath.net/texmaker/>

## Sublime Text

<http://www.sublimetext.com/>

Follow the instructions on <https://www.zhihu.com/question/36038602>

## Visual Studio Code

<https://code.visualstudio.com/>

Follow the instructions on <https://zhuanlan.zhihu.com/p/38178015>

They all have cross-platform support for Windows, Linux and MacOS.

Liu Yihao

## Getting Started

What is  $\text{\LaTeX}$

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Write $\text{\LaTeX}$ on Overleaf (Online)

Another alternative choice is to write  $\text{\LaTeX}$  online with the technology of [Overleaf](#). It's free for personal usage and supports share editing which is very useful in group work.

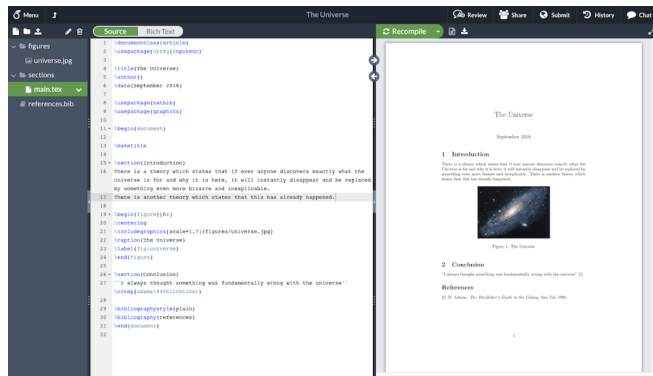


Figure 1: Layout of the Overleaf Online  $\text{\LaTeX}$  Editor.

Liu Yihao

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

- 1 Getting Started
  - What is L<sup>A</sup>T<sub>E</sub>X
  - Distributions and IDEs
  - Documentation
- 2 Layout of a Document
- 3 Learn More

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Documentation of L<sup>A</sup>T<sub>E</sub>X

If you've installed a full version of T<sub>E</sub>XLive (as strongly recommended), the full L<sup>A</sup>T<sub>E</sub>X documentation is already on your computer.

Open the command line and input the command

## Command

```
1 texdoc <docname>
```

You can also use the online version on [Link](#)

For example, you can use the following types for the **docname**

**tex** about T<sub>E</sub>X

**article** about documentclass **article**

**beamer** about documentclass **beamer** (used to create slides)

**pgf** about packages **tikz** and **pgf** (used to draw graphs)

Try to **texdoc** about all new things and then you'll be an expert in L<sup>A</sup>T<sub>E</sub>X.

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

## 1 Getting Started

## 2 Layout of a Document

- A Simple Document
  - Document Classes
  - The Preamble
  - Document Body
  - Document Sections

## 3 Learn More

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# A Simple Document

A typical (simplest) L<sup>A</sup>T<sub>E</sub>X example is presented here.

## Example

```
1 \documentclass[a4paper]{article}
2 \usepackage{amsmath} % Define various maths environments
3 \usepackage{amssymb} % Define various maths symbols
4 \usepackage{geometry} % Adjust the margin, paper size, and etc.
5 \usepackage[shortlabels]{enumerate} % Provide different style of lists
6 \usepackage{graphicx} % Insert image of all types
7 % Use other packages and setup them here
8 \title{A simple \LaTeX\ document}
9 \author{XX XXX}
10 \date{\today}
11
12 \begin{document}
13     \maketitle
14     Hello, \LaTeX !
15 \end{document}
```

Code started with `\` is called a **command**, and a pair of `\begin{}` and `\end{}` is called an **environment**.

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

## 1 Getting Started

## 2 Layout of a Document

- A Simple Document
- **Document Classes**
- The Preamble
- Document Body
- Document Sections

## 3 Learn More

# All Begins with `documentclass`

## Definition

In a L<sup>A</sup>T<sub>E</sub>X file, the **first** line must be

```
\documentclass[options]{class}
```

For example, you can use the following types for the **class**

**article** Write a report or an science article

**report** Write a report

**beamer** Produce a lecture silde like this!

Some options can be added, for example, a typical case can be

```
\documentclass[11pt,twoside,a4paper]{article}
```

Some details about the **article** class will be introduced on the next page. More features about other classes and options can be found in the L<sup>A</sup>T<sub>E</sub>X Document on your own.



# The `article` Class

The `article` class is one of the most basic class in L<sup>A</sup>T<sub>E</sub>X, it provides you with some normalized structure and format for report writing. So usually you will use the following command as the first line of your tex document:

```
\documentclass[options]{article}
```

Some of the options values are listed below (the default values are **alerted**)

- `10pt`, `11pt`, `12pt` or other sizes - the font size of the document
- `a4paper`, `a5paper`, `letterpaper` - the size of paper
- `fleqn` - make the math equations left aligned (default middle aligned)
- `leqno` - display the serial numbers of math equations on the left (default on the right)
- `titlepage`, `notitlepage` - whether to make the title an entire page
- `onecolumn`, `twocolumn` - the number of columns of the document
- `twoside`, `oneside` - influence the position of something on the page

# Other classes

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

This project is open sourced and you can read the source code [▶ Link](#) to learn much (I promise) about the `beamer` class and some very interesting features of L<sup>A</sup>T<sub>E</sub>X itself. There may also be a lecture about the `beamer` class in the future.

When writing a `long` report, `report` class can be used to provide some more layers of document (such as `chapter`) and different type settings. It's very similar to the `article` class, so it won't be specified.

There are some other document classes such as `minimal`, `book`, `letter` and etc., but I think you may never use them.

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

**The Preamble**

Document Body

Document Sections

### Learn More

Commands

Environments

## 1 Getting Started

## 2 Layout of a Document

- A Simple Document
- Document Classes
- **The Preamble**
- Document Body
- Document Sections

## 3 Learn More

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# The Preamble of a Document

As in the simple example of a document, you should notice that there is a pair of

## Command

```
1 \begin{document}
2   % some contents
3 \end{document}
```

This is called the **body** of the document, and everything before the **body**, including the `\documentclass` line, is called the **preamble** of the document.

In the preamble, you define the type of document you are writing and the language, load extra packages you will need, and set several parameters. For instance, a simplified document of the example above preamble would look like this:

## Example

```
1 \documentclass[a4paper]{article}
2 \title{A simple LATEX document}
3 \author{XX XXX}
4 \date{\today}
```

# Title, Author and Date

It's very useful to generate a title on the first page of a document, in order to achieve it, these commands should first be added in the **preamble**.

## Example

```
1 \title{title}
2 \author{author name}
3 \date{\today}
```

You can simply use `\date{\today}` to display your system date now.

Then in the **body** (will be introduced in the next section), use the command `\maketitle` to generate the title, or title page if you added the option `titlepage` in the `\documentclass`.

# Magic of Packages

L<sup>A</sup>T<sub>E</sub>X is a macro-based language, where most of useful commands are not built-in commands. These commands are defined in various packages, which should be included in the **preamble**.

## Command

```
\usepackage[options]{package}
```

There are some very useful packages that you may **ALWAYS** include:

**amsmath** Define various maths environments

**amssymb** Define various maths symbols

**geometry** Adjust the margin, paper size, and etc.

**enumitem** Generate a list like this!

**graphicx** Insert images of all types

The usages of these and more packages will be introduced further.

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

Here I provide a list of commonly used packages, you can start from using them after the lecture.

```
1 \usepackage{hyperref} % Extensive support for hypertext
2 \usepackage{float} % Improved interface for floating objects
3 \usepackage[margin=2.5cm]{geometry} % Flexible document dimensions
4 \usepackage[shortlabels]{enumerate} % Enumerate with redefinable labels
5 \usepackage{multirow} % Tabular cells spanning multiple rows
6 \usepackage{multicol} % Intermix single and multiple columns
7 \usepackage{ulem} % Package for underlining
8 \usepackage{graphicx} % Enhanced support for graphics
9 \usepackage{subfig} % Figures broken into subfigures
10 \usepackage{amsmath} % AMS mathematical facilities
11 \usepackage{amssymb} % AMS symbols
12 \usepackage{amsfonts} % AMS fonts
13 \usepackage{mathrsfs} % Support for using RSFS fonts in maths
14 \usepackage{latexsym} % LaTeX symbols
15 \usepackage{verbatim} % Reimplementation of LaTeX verbatim
```

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

**Document Body**

Document Sections

### Learn More

Commands

Environments

## 1 Getting Started

## 2 Layout of a Document

- A Simple Document
- Document Classes
- The Preamble
- **Document Body**
- Document Sections

## 3 Learn More



# Main Body of Document

The main body of your document which starts with `\begin{document}` and ends with `\end{document}` can be also called the `document` environment. All of the contents you'd like to display should be in it, and it **MUST** be **unique** in the whole file.

## Example

```
1 \begin{document}
2   \maketitle
3   \newpage    %start the following contents in a new page
4   \tableofcontents %automatically generate a table of content
5   \newpage
6   Hello, \LaTeX !
7   % TODO: Add more contents
8   ...
9 \end{document}
```

The position and order of title page and table of contents can be arbitrary, and there can be multiple table of contents in one document.

# The `abstract` Environment

When you are writing a paper, an `abstract` is often necessary in the beginning of the document.

## Example

```
1 \begin{abstract}
2   This is a lecture about how to getting start in \LaTeX!
3 \end{abstract}
```

### Abstract

This is a lecture about how to getting start in L<sup>A</sup>T<sub>E</sub>X!

The styling of the `abstract` will be based on the `documentclass` you are using. The example shows an `abstract` in the `beamer` class, which will be slightly different from that in the `article` class.

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

As in other programming languages, comments are useful when you want to make your code readable. Adding a `%` can make the whole line after it into a comment.

## Example

```
% This is a comment
```

If you need multiline comments, use the `comment` environment provided by the `comment` package. (Add `\usepackage{comment}` to your preamble.)

## Example

```
1 \begin{comment}
2   some comments
3   some other comments
4 \end{comment}
```

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

Note that in the compiling, anything after a % is omitted, including the newline character, so there is no space between “comment” and “no” in the second line.

### Example

```
1  A line
2  with space between ``line'' and ``with''
3
4  A line ended with comment% comments
5  no space between ``comment'' and ``no''
```

A line with space between “line” and “with”

A line ended with commentno space between “comment” and “no”

PS: One newline, or any number of space and tab characters are usually considered as a single “spacing” in L<sup>A</sup>T<sub>E</sub>X compilers. Two or more continuous newlines will cause a line break. We’ll discuss it later in the lecture.

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

## 1 Getting Started

## 2 Layout of a Document

- A Simple Document
- Document Classes
- The Preamble
- Document Body
- Document Sections

## 3 Learn More

# Sections

Commands to organize a document vary depending on the document type, the simplest form of organization is the sectioning, available in all formats.

## Command

```
1 \section{name}
```

```
2 \subsection{name}
```

```
3 \subsection{name}
```

```
1 \section*{name}
```

```
2 \subsection*{name}
```

```
3 \subsection*{name}
```

The default style (can be changed with `\renewcommand`) of sections is like

1 Example Section Name

1.1 Example Subsection Name

1.1.1 Example Subsubsection Name

If an asterisk (\*) is added, the sequence number will be hidden, and it won't be added to the table of contents.

Note: (Sub)sections are **commands**, and the whole contents between two (sub)sections is belonged to the former (sub)section.

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

## Other Structures - Chapter, Part and Paragraph

## Command

```
1 \chapter{name}  
2 \part{name}  
3 \paragraph{name}  
4 \subparagraph{name}
```

```
1 \chapter*{name}  
2 \part*{name}  
3 \paragraph*{name}  
4 \subparagraph*{name}
```

In document classes such as `report` and `book`, some outer structures of `section` (`\chapter` and `\part`) can be used.

`\paragraph` and `\subparagraph` are used for the title of small paragraphs in a (sub)section.

If an asterisk (\*) is added, the effect will be the same as in the sections (sequence numbers will be hidden).

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Levels of Document Structures

There are up to 7 levels of depth for defining sections depending on the document class:

Level	Command
-1	<code>\part{part}</code>
0	<code>\chapter{chapter}</code>
1	<code>\section{section}</code>
2	<code>\subsection{subsection}</code>
3	<code>\subsubsection{subsubsection}</code>
4	<code>\paragraph{paragraph}</code>
5	<code>\subparagraph{subparagraph}</code>

`\part` and `\chapter` are not available in some document classes.



# Introduction to L<sup>A</sup>T<sub>E</sub>X

## Lecture I: Hello, L<sup>A</sup>T<sub>E</sub>X

Liu Yihao

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

- 1 Getting Started
- 2 Layout of a Document
- 3 Learn More**
  - **Commands**
  - Environments

# Common Syntax of L<sup>A</sup>T<sub>E</sub>X Commands

All L<sup>A</sup>T<sub>E</sub>X commands have the following syntax

## Command

```
\commandName<specialArgs>[optionalArgs]{requiredArgs}
```

**specialArgs** Seldom used in basic usage, for certain special usages in some packages

**optionalArgs** Used to define mode of the command, if not specified, L<sup>A</sup>T<sub>E</sub>X will use the default mode

**requiredArgs** Must be filled

If you want to connect a letter after a command, a space must be appended after the command or L<sup>A</sup>T<sub>E</sub>X won't be able to compile it correctly. But two commands can be directly connected since there is a `\` before each command.

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Define New Commands

In L<sup>A</sup>T<sub>E</sub>X, you can define a new command (must not already exist) with

## Command

```
\newcommand{\commandName}[args]{definition}
```

The definitions of new commands are usually put in the preamble. If there are no arguments, you can omit the optional [args]; or use #num to fill in the arguments.

## Example

```
1 \newcommand{\examplelatexcommand}[1]{%  
2 This lecture is #1!%  
3 }%  
4  
5 \examplelatexcommand{interesting}  
6 \examplelatexcommand{great}
```

This lecture is interesting! This lecture is great!

Here I use the comment character % in the end of each line of the definition to prevent adding newlines in the new command.

# Renew Commands

You can also redefine a command (must already exist) with

## Command

```
\renewcommand{\commandName}[args]{definition}
```

## Example

```
1 \newcommand{\examplelatexcommand}[1]{...}%  
2 \renewcommand{\examplelatexcommand}[1]{%  
3   This lecture is not #1!%  
4 }%  
5  
6 \examplelatexcommand{interesting}  
7 \examplelatexcommand{great}
```

This lecture is not interesting! This lecture is not great!

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

`\renewcommand` is often used to change the style of `section`, `subsection` and etc., for example

## Example

```
1 \renewcommand{\thesection}{\Roman{section}.}
2 \renewcommand{\thesubsection}{\Roman{section}.\arabic{subsection}}
```

This example changes the section number to capital roman numbers and subsection number to arabic numbers. Here's a list of available styles:

`\arabic` prints the value as an Arabic number, e.g. 2.

`\alph` prints the value as an alphabetic character (minuscule), e.g. b.

`\Alph` prints the value as an alphabetic character (capital letter), e.g. B.

`\roman` prints the value as a Roman number (minuscules), e.g. ii.

`\Roman` prints the value as a Roman number (capital letters), e.g. II.

`\fnsymbol` prints the value as a symbol in a sequence, this is meant to be used for symbolic footnotes, e.g. †.

# Introduction to L<sup>A</sup>T<sub>E</sub>X

## Lecture I: Hello, L<sup>A</sup>T<sub>E</sub>X

Liu Yihao

### Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

### Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

### Learn More

Commands

Environments

- 1 Getting Started
- 2 Layout of a Document
- 3 Learn More**
  - Commands
  - Environments**

## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

# Common Syntax of L<sup>A</sup>T<sub>E</sub>X Environments

All L<sup>A</sup>T<sub>E</sub>X environments have the following syntax

## Command

```
1 \begin{environmentName}<specialArgs>[optionalArgs]{requiredArgs}
2 % ...
3 \end{environmentName}
```

`specialArgs`, `optionalArgs`, `requiredArgs` are similar to those in a `command`

It is recommended to have a indent in each environment or your tex codes will be difficult to read by others or even **yourself**.

# Define New Environments and Renew Environments

You can define a new environment (must not already exist) with

## Command

```
\newenvironment{environmentName}[args]{before begin}{after end}
```

The difference of defining an `environment` from defining a `command` is that you should specify two code blocks, one is inserted before the `\begin` clause and the other is inserted after the `\end` clause.

Another issue is that arguments can only be used in the first of them (before `\begin`). If you need to save some arguments, use `\newcommand` to define a macro, but it may cause problems in nested usages.

Redefine an environment (must already exist) with

## Command

```
\renewenvironment{environmentName}[args]{before begin}{after end}
```



## Getting Started

What is L<sup>A</sup>T<sub>E</sub>X

Distributions and IDEs

Documentation

## Layout of a Document

A Simple Document

Document Classes

The Preamble

Document Body

Document Sections

## Learn More

Commands

Environments

For example, the examples in this lecture are provided by a self-defined `latexexample` environment:

## Example

```
1  \newenvironment{latexexample}
2  {\VerbatimOut{\jobname.tmp}}
3  {\endVerbatimOut
4  \begin{example}
5  ~
6  \inputminted{latex}{\jobname.tmp}
7
8  \input{\jobname.tmp}
9  \end{example}
10 }
11 \begin{latexexample}
12   some code here
13 \end{latexexample}
```

It is a `verbatim` environment, which accepts L<sup>A</sup>T<sub>E</sub>X code as plain text and deals with them later.

# Lecture II

## Text

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

- 4 Polishing the plain text
  - Special Characters and Accents
  - Fonts
  - Underline

### 5 Typesetting

### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

# Special Characters

Some special symbols can't be directly used since they are reserved by L<sup>A</sup>T<sub>E</sub>X:

<code>\#</code>	<code>#</code>	<code>\\$</code>	<code>\$</code>	<code>\%</code>	<code>%</code>	<code>\&amp;</code>	<code>&amp;</code>	<code>\~</code>	<code>~</code>	<code>\`</code>	<code>`</code>
<code>\{</code>	<code>{</code>	<code>\}</code>	<code>}</code>	<code>\-</code>	<code>-</code>	<code>\textbackslash</code>	<code>\</code>				

Many L<sup>A</sup>T<sub>E</sub>X starters are confused with how to correctly print quotes, hyphens and dots.

``` prints a left single quote, `'` prints a right single quote.

```` prints a left double quote, `''` prints a right double quote.

one hyphen (-) print like -

two hyphens (--) print like –

three hyphens (---) print like —

`\dots` prints the dots with a correct format (...) instead of directly use three dots (...)

# Accent on letters

Sometimes you may need an accent form of a letter, here is an example of letter o

<code>\`{o}</code>	ò	<code>\'{o}</code>	ó	<code>\^{o}</code>	ô	<code>\"{o}</code>	ö	<code>\~{o}</code>	õ
<code>\={o}</code>	ō	<code>\.{o}</code>	ó	<code>\u{o}</code>	ů	<code>\v{o}</code>	ǒ	<code>\H{o}</code>	Ǫ
<code>\t{oo}</code>	ôo	<code>\r{o}</code>	õ	<code>\c{o}</code>	ç	<code>\d{o}</code>	ð	<code>\b{o}</code>	ƚ

## Something interesting

You may be curious about how to print words like L<sup>A</sup>T<sub>E</sub>X, actually it's defined as a command.

- `\TeX` - T<sub>E</sub>X
- `\LaTeX` - L<sup>A</sup>T<sub>E</sub>X
- `\LaTeXe` - L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>

### Polishing the plain text

#### Special Characters and Accents

#### Fonts

#### Underline

### Typesetting

#### Enumeration

#### Alignment

#### Spaces, lines and pages

#### Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

#### Multiple Languages

#### Scope

# Deal with unfamiliar symbols

## Polishing the plain text

### Special Characters and Accents

#### Fonts

#### Underline

## Typesetting

#### Enumeration

#### Alignment

#### Spaces, lines and pages

#### Minipage and Multicolumn

## Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

#### Multiple Languages

#### Scope

Sometimes you may want to deal with symbols you have never seen. In this case, you may refer to <http://detexify.kirelabs.org/classify.html> to find out how to output the character.

### Polishing the plain text

Special Characters and  
Accents

**Fonts**

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

## 4 Polishing the plain text

- Special Characters and Accents
- **Fonts**
- Underline

## 5 Typesetting

## 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

## Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
MulticolumnLearn more - multi  
languages and scope in  
L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

## Basic commands about fonts

First, let's start with some commands that transform font types

- `\bf` - **Sample Text**
- `\it` - *Sample Text*
- `\rm` - Sample Text
- `\sc` - SAMPLE TEXT
- `\sf` - Sample Text
- `\sl` - *Sample Text*
- `\tt` - Sample Text

Note that the commands that transform font types influence the text in the whole scope (`{...}`) until another font type is specified. For example, how to use the first command `\bf` is shown below

`{\bf Sample Text}`



## Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

Learn more - multi  
languages and scope in  
L<sup>A</sup>T<sub>E</sub>X

Multiple Languages  
Scope

Sometimes we don't want to transform all the font types, instead, we can only change the font type of some specified text.

## Example

```
1 \textbf{Sample text}
```

There are more options for fonts.

- `\textit` - *Sample Text*
- `\textsc` - SAMPLE TEXT

However, in a math environment (will be introduced later), some other commands should be used

- `\mathbf` - **Sample Text**
- `\mathit` - *Sample Text*
- `\mathsf` - Sample Text

Note that the math environment doesn't include all of the font types on the previous page. More information about font types can be found [here](#).

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages  
Scope

Font size can also be easily modified

- `\tiny` - Sample Text
- `\scriptsize` - Sample Text
- `\footnotesize` - Sample Text
- `\small` - Sample Text
- `\normalsize` - Sample Text
- `\large` - Sample Text
- `\Large` - Sample Text
- `\LARGE` - Sample Text
- `\huge` - Sample Text
- `\Huge` - Sample Text

# Build a colorful document

## Polishing the plain text

Special Characters and Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

Changing the color is similar to changing font types.

If you want to transform to a color (like transforming to bold with `\bf`), you can use `\color{name}`.

Similarly, you can use `\textcolor{name}` like `\textbf`.

The background color of the whole page can be set using `\pagecolor{name}`.

There are some defined color `name` in the `xcolor` package.

	black		gray		olive		teal		blue
	green		orange		violet		brown		lightgray
	pink		white		cyan		lime		purple
	yellow		darkgray		magenta		red		

You can find more information in the documentation of `xcolor` (`texdoc xcolor`)

### Polishing the plain text

Special Characters and  
Accents

Fonts

**Underline**

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

- 4 Polishing the plain text
  - Special Characters and Accents
  - Fonts
  - **Underline**

### 5 Typesetting

### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

# Ulem package

If you want to add some lines on the text, use the [ulem](#) package.

## Command

```
1 \usepackage{ulem}
2 \uline{Sample Text}
```

There are different kinds of lines supported:

- `\uline` - Sample Text
- `\uuline` - Sample Text
- `\uwave` - Sample Text
- `\sout` - ~~Sample Text~~
- `\xout` - ~~Sample Text~~
- `\dashuline` - Sample Text
- `\dotuline` - Sample Text

## Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

Learn more - multi  
languages and scope in  
L<sup>A</sup>T<sub>E</sub>X

Multiple Languages  
Scope

### Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

#### 4 Polishing the plain text

#### 5 Typesetting

- Enumeration

- Alignment

- Spaces, lines and pages

- Minipage and Multicolumn

#### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

# Enumerate

When you need to enumerate some items as a list, you may use the [enumerate](#) package.

## Command

```
1 \usepackage{enumerate}
2 \begin{enumerate}[style]
3 \item % ...
4 \item % ...
5 \item % ...
6 \end{enumerate}
```

This will generate a normal list with the serial numbers in the specified [style](#), which could be the following (as example)

- **1** - 1, 2, 3, 4, ...
- **(i)** - (i), (ii), (iii), (iv), ...
- **[1.]** - [1.], [2.], [3.], [4.], ...

# Itemize

If you want to generate an unordered list, use `itemize` instead of `enumerate`.

## Command

```
1 \usepackage{enumerate}
2 \begin{itemize}
3 \item[style] % ...
4 \item[style] % ...
5 \item[style] % ...
6 \end{itemize}
```

In this case, `style` must be added after each item, which is different from that in `enumerate`, and the symbol displayed in the beginning of each item will be exactly same as the `style`. If `style` is not added, a default style will be used.

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope



### Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

### Typesetting

Enumeration

#### Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

#### 4 Polishing the plain text

#### 5 Typesetting

- Enumeration
- Alignment
- Spaces, lines and pages
- Minipage and Multicolumn

#### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

# Alignment

If you want to align a paragraph of text, use these three environments for left/center/right align.

## Command

```
1 \begin{flushleft/center/flushright}  
2 % ...  
3 \end{flushleft/center/flushright}
```

However, if only a single line needs to be aligned, use these three commands.

## Command

```
1 \leftline{text}  
2 \centerline{text}  
3 \rightline{text}
```

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

**Spaces, lines and pages**

Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

#### 4 Polishing the plain text



#### 5 Typesetting

- Enumeration
- Alignment
- **Spaces, lines and pages**
- Minipage and Multicolumn

#### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

# Spaces may be confusing

There are defined command of spaces in different width and usages.

-  - the basic space in L<sup>A</sup>T<sub>E</sub>X (printed in yellow since it's transparent). Note that any number of spaces or tabs is equal to one space, and the space after a command is ignored. If you want to add an extra space, use `\` which makes a 1/3 em space (1 em is approximately the width of an M in the current font)
- `~` - If two words can't be separated on two lines, you can tell L<sup>A</sup>T<sub>E</sub>X about it using a tie (`~`), such as Prof.`~`Hamade (Prof. Hamade).
- `\,` - makes a 1/6 em space, commonly used before units (notice the space before em on this page)
- `\;` - makes a 2/7 em space
- `\quad` - makes a 1 em space
- `\qquad` - makes a 2 em space
- `\phantom{text}` - makes actually the space of `text`, but `text` will be invisible.

## Polishing the plain text

Special Characters and Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

# Separate contents into lines and pages

Here are some basic commands about lines and pages in L<sup>A</sup>T<sub>E</sub>X, you will use them everywhere.

- `\newline` - begin a new line
- `\\` - begin a new line (not recommended<sup>1</sup>)
- `\par` - begin a new paragraph (a new line with indent)
- `\\[offset]` - begin a new line with an vertical offset, `offset` is the size of needed space (not recommended, using `\vspace` instead.)
- `\newpage` - begin a new page
- `%` - begin a line comment

---

<sup>1</sup>According to Manuel Charlemagne, `\\` should only be used for a force break (where `\newline` doesn't work).

## Polishing the plain text

- Special Characters and Accents

- Fonts

- Underline

## Typesetting

- Enumeration

- Alignment

- Spaces, lines and pages

- Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

- Multiple Languages

- Scope

# Spacing

When trying to separate two paragraphs by a certain space, many new learners of L<sup>A</sup>T<sub>E</sub>X may use multiple empty lines and linebreaks, which is a very dirty fix and is not so accurate. Actually, L<sup>A</sup>T<sub>E</sub>X provides a precise spacing mechanism.

## Command

```
\vspace{space}
```

```
\vspace*{space}
```

When trying to show the next paragraph or sentence precisely at the bottom of the current page, we can use

## Command

```
\vfill
```

between the contents of two paragraphs to separate them.

# Predefined skipping

More often<sup>1</sup>, we don't need to think about the skipping space, we can use the predefined skipping commands to achieve a small, medium or big skip. They are actually particular cases of `\vspace`

## Command

`\smallskip``\medskip``\bigskip`

You may note that the effects are these skipping commands have been already shown above.

---

<sup>1</sup>According to Manuel Charlemagne, you should always use these skipping commands if possible instead of using `\\` (as in many online tutorials).

# Spacing units

The `space` can be anything representing a size, such as `1cm`, `2em` and `10pt`. In L<sup>A</sup>T<sub>E</sub>X, spacing units can be

- `cm`
- `mm`
- `in` - inch, 1 inch = 2.54 cm
- `pt` - 72 pt = 1 inch, the smallest unit in L<sup>A</sup>T<sub>E</sub>X
- `em` - 1em equals to the width of letter M
- `ex` - 1ex equals to the width of letter x
- `\linewidth` - the width of current line in the container
- `\pagewidth` - the width of the page
- `\pageheight` - the height of the page
- `\textwidth` - the normal width of text on the page
- `\textheight` - the normal height of text on the page



### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

#### 4 Polishing the plain text

#### 5 Typesetting

- Enumeration
- Alignment
- Spaces, lines and pages
- Minipage and Multicolumn

#### 6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

## Minipage

`minipage` is a very useful environment for dividing pages into a grid.

## Example

```

1  \begin{minipage}{0.32\linewidth}
2      % ...
3  \end{minipage}
4  \hfill % Fill horizontal space
5  \begin{minipage}{0.32\linewidth}
6      % ...
7  \end{minipage}
8  \hfill % Fill horizontal space
9  \begin{minipage}{0.32\linewidth}
10     % ...
11 \end{minipage}
12 \vfill % Fill vertical space

13 \begin{minipage}{0.32\linewidth}
14     % ...
15 \end{minipage}
16 \hfill % Fill horizontal space
17 \begin{minipage}{0.32\linewidth}
18     % ...
19 \end{minipage}
20 \hfill % Fill horizontal space
21 \begin{minipage}{0.32\linewidth}
22     % ...
23 \end{minipage}

```

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

The code above generate six minipages in a grid of 3 columns  $\times$  2 rows. Don't try to add up the width of minipages in a line for more than about `0.98\linewidth` (since a minipage have a small margin on each side), or the last minipage may be on a new line.

For each minipage, it can be seem as an independent L<sup>A</sup>T<sub>E</sub>X document, where text, formulas, graphics, tables and etc. can be inserted, and most importantly, they won't affect each other. What's more, you can even use minipages in a minipage to form a multi-level nesting.

# The multicol package

## Polishing the plain text

Special Characters and Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

When typesetting contents with small line width and many lines (for example, source code), the **multicol** package is recommended.

### Command

```
1 \usepackage{multicol}
2 \begin{multicols}{cols}
3     % contents on column one
4     \breakcolumn % break the current column here
5     % contents on column two
6 \end{multicols}
```

Here **cols** is the number of columns, it must be specified. If **\breakcolumn** is not used, the **multicol** package will automatically balance the length of each column.

### Polishing the plain text

Special Characters and Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

4 Polishing the plain text

5 Typesetting

6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

- Multiple Languages
- Scope

# Spelling languages

If you want to use a spelling language with characters similar to English, package `babel` can be used (exactly the same name as `babel`).

## Command

```
\usepackage[languages]{babel}
```

- `languages` - a list of languages, the last one to be the default language

## Example

```
\usepackage[greek,english]{babel}  
\textgreek{abcdefgABCDEFGF}
```

Then L<sup>A</sup>T<sub>E</sub>X will print αβζδεϕγΑΒ<sup>Δ</sup>ΕΦΓ

Of course, you can use some simple commands to print these greek letters directly, such as `\alpha`, `\beta` and etc, which is more convenient only when few of them are needed.

## Polishing the plain text

Special Characters and Accents

Fonts

Underline

## Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and Multicolumn

Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

The Chinese TeX Community maintains a package called `ctex` for inputting Chinese in L<sup>A</sup>T<sub>E</sub>X. Note that it is only a package, which is shipped with most modern T<sub>E</sub>X Suites, not the CTEX Suite. I don't think it's a good choice to use the CTEX Suite directly.

## Command

```
\usepackage{ctex}
```

The default L<sup>A</sup>T<sub>E</sub>X compiler `pdflatex` doesn't have support on Chinese input with `ctex` package, `xelatex` is a recommended modern L<sup>A</sup>T<sub>E</sub>X compiler as a replacement.

However, the `ctex` package is too heavy and it can slow down the total compilation speed seriously.

### Polishing the plain text

Special Characters and  
Accents

Fonts

Underline

### Typesetting

Enumeration

Alignment

Spaces, lines and pages

Minipage and  
Multicolumn

### Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

Multiple Languages

Scope

4 Polishing the plain text

5 Typesetting

6 Learn more - multi languages and scope in L<sup>A</sup>T<sub>E</sub>X

- Multiple Languages
- Scope



# Usage of scope in L<sup>A</sup>T<sub>E</sub>X

First, you should realize the meaning of “scope” in programming. Let's start with a simple example in C/C++ (assuming you know that):

```
1  int main()
2  { // The scope "main" of function main
3    int a = 1; // int a is defined in scope "main"
4    for (int i = 0; i < 10; i++)
5    { // The scope "for" of the for loop
6      int b = i; // int b and i are both defined in scope "for"
7      a += b; // int a can be visited here!
8    }
9    { // The scope "other", we can directly define a scope like this
10     int c; // int c is defined in scope "other"
11     c = a; // int a can be visited here!
12   }
13   a -= c // error: c is not in scope "main", can't be visited!
14 }
```

In the example of C/C++, we use brackets `{}` to define a scope, which is just the same in L<sup>A</sup>T<sub>E</sub>X. In addition, notice that an environment or a command also defines a scope.

### Example

1	<code>black (default) text \\</code>	black (default) text
2	<code>\color{blue}</code>	blue text
3	<code>blue text \\</code>	brown text
4	<code>{ \color{brown} brown text }</code>	
5	<code>\begin{center}</code>	
6	<code>\color{red}</code>	centered red text
7	<code>centered red text</code>	
8	<code>\end{center}</code>	
9	<code>\textbf{ \color{brown}</code>	bold brown text
10	<code>bold brown text } \\</code>	blue text
11	<code>blue text</code>	

With the usage of scopes, you can flexibly change the color, font or anything else you wish in a self-defined range of the document.

# Lecture III

## Maths

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

- Math Expressions
- Math Environments
- Spacing in Math Mode
- Basic Math Commands
- Matrices and Arrays

## 8 Useful Maths Packages

# Introduction

Basic equations in  $\text{\LaTeX}$  can be easily “programmed”, for example: <sup>1</sup>

## Example

```
1 The well known Pythagorean theorem \(x^2 + y^2 = z^2\) was
2 proved to be invalid for other exponents.
3 Meaning the next equation has no integer solutions:
4
5 \[ x^n + y^n = z^n \]
```

The well known Pythagorean theorem  $x^2 + y^2 = z^2$  was proved to be invalid for other exponents. Meaning the next equation has no integer solutions:

$$x^n + y^n = z^n$$

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

# Subscripts and Superscripts

The use of superscripts and subscripts is very common in mathematical expressions involving exponents, indexes, and in some special operators.<sup>1</sup>

## Example

```
1 \[ a_1^2 + a_2^2 = a_3^2 \]
```

$$a_1^2 + a_2^2 = a_3^2$$

Note that here we use `\[` and `\]` to typeset a mathematical expression. You may see many people (including myself in the past) using a pair of `$$` instead. It is a plain-T<sub>E</sub>X command, and is nowadays heavily deprecated. See this discussion [▶ Link](#) on Stack Exchange for more information.

---

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

If the expression contains long superscripts or subscripts, these need to be collected in braces, as L<sup>A</sup>T<sub>E</sub>X normally applies the mathematical commands `^` and `_` only to the following character:

Example

```
1 \[ x^{2 \alpha} - 1 = y_{ij} + y_{ij} \]  
2 \[ (a^n)^{r+s} = a^{nr+ns} \]  
3 \[ x^{abc}, \quad \text{\quad} x_{abc}, \quad \text{\quad} x^{abc}_{abc} \]  
4 \[ x^{abc}, \quad \text{\quad} x_{abc}, \quad \text{\quad} x^{abc}_{abc} \]
```

$$x^{2\alpha} - 1 = y_{ij} + y_{ij}$$

$$(a^n)^{r+s} = a^{nr+ns}$$

$$x^{abc}, \quad x_{abc}, \quad x^{abc}_{abc}$$

$$x^{abc}, \quad x_{abc}, \quad x^{abc}_{abc}$$

# Brackets and Parentheses

Parentheses and brackets are very common in mathematical formulas. You can easily control the size and style of brackets in  $\text{\LaTeX}$ .<sup>1</sup>

Here's how to type some common math braces and parentheses in  $\text{\LaTeX}$ :

Type	$\text{\LaTeX}$	Code
Parentheses; round brackets	$(x + y)$	<code>(x+y)</code>
Brackets; square brackets	$[x + y]$	<code>[x+y]</code>
Braces; curly brackets	$\{x + y\}$	<code>\{x+y\}</code>
Angle brackets	$\langle x + y \rangle$	<code>\langle x+y \rangle</code> <code>\rangle</code>
Pipes; vertical bars	$ x + y $	<code>x+y</code> <code>---</code>
Double pipes	$  x + y  $	<code>\ x+y\ </code>
Floor brackets	$\lfloor x + y \rfloor$	<code>\lfloor x+y \rfloor</code> <code>\rfloor</code>
Ceil brackets	$\lceil x + y \rceil$	<code>\lceil x+y \rceil</code> <code>\rceil</code>

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:



The size of brackets and parentheses can be manually set, or they can be resized dynamically in your document, as shown in the next example:

### Example

```
1 \[ F = G \left( \frac{m_1 m_2}{r^2} \right) \]
```

$$F = G \left( \frac{m_1 m_2}{r^2} \right)$$

Notice that to insert the parentheses or brackets, the `\left` and `\right` commands are used. Even if you are using only one bracket, both commands are mandatory, you can use invisible brackets `\left.` or `\right.` for this.

### Example

```
1 \[ \int_a^b x^2 {\rm d} x = \left. \frac{1}{3} x^3 \right|_a^b \]
```

$$\int_a^b x^2 dx = \left. \frac{1}{3} x^3 \right|_a^b$$

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

Sometimes you may want to control the sizes of the brackets yourselves, which is called manually sized brackets. The commands listed are designed for thus purpose.

Size	L <sup>A</sup> T <sub>E</sub> X	Code
big	$()$	<code>\big ( \big )</code>
Big	$\bigl[ \bigr]$	<code>\Big [ \Big ]</code>
bigg	$\biggl\{ \biggr\}$	<code>\bigg \{ \bigg \}</code>
Bigg	$\Biggl  \Biggr $	<code>\Bigg  </code>

# Mathematical Modes

L<sup>A</sup>T<sub>E</sub>X allows two writing modes for mathematical expressions: the `inline` mode and the `display` mode. The first one is used to write formulas that are part of a text. The second one is used to write expressions that are not part of a text or paragraph, and are therefore put on separate lines.

To put your equations in `inline` mode use `\(` and `\)`, `$` and `$` or `\begin{math}` and `\end{math}`. They all work and the choice is a matter of taste.

## Example

- 1 In physics, the mass-energy equivalence is stated
- 2 by the equation  $E=mc^2$ , discovered in 1905 by Albert Einstein.

In physics, the mass-energy equivalence is stated by the equation  $E = mc^2$ , discovered in 1905 by Albert Einstein.

The `display` mode is usually used with mathematical environments together, which will be discussed in the next subsection.

# Numbering of Equations

The `display` mode has two versions: `numbered` and `unnumbered`.

## Example

```

1 The mass-energy equivalence is described by the famous equation
2 \[E=mc^2\]
3 discovered in 1905 by Albert Einstein.
4 In natural units ( $c = 1$ ), the formula expresses the identity
5 \begin{equation}
6 E=m
7 \end{equation}

```

The mass-energy equivalence is described by the famous equation

$$E = mc^2$$

discovered in 1905 by Albert Einstein. In natural units ( $c = 1$ ), the formula expresses the identity

$$E = m \tag{1}$$

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

- Math Expressions
- **Math Environments**
- Spacing in Math Mode
- Basic Math Commands
- Matrices and Arrays

## 8 Useful Maths Packages

# The `equation` Environment

An `equation` environment contains a set of maths equations

## Command

```
1 \begin{equation(*)}  
2 % ...  
3 \end{equation(*)}
```

## Example

$$\operatorname{rot} F = \left( \frac{\partial F_z}{\partial y} - \frac{\partial F_y}{\partial z} \right) \hat{n}_x + \left( \frac{\partial F_x}{\partial z} - \frac{\partial F_z}{\partial x} \right) \hat{n}_y + \left( \frac{\partial F_y}{\partial x} - \frac{\partial F_x}{\partial y} \right) \hat{n}_z \quad (2)$$

If a star(\*) is added, the sequence number of the equation won't be displayed (this feature is from the `amsmath` package, and should behave very similar as directly using `\[` and `\]`). Note that the environment name in the `\begin` and `\end` statements must be the same (both or neither have a \* here).

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

The L<sup>A</sup>T<sub>E</sub>X script of the equation above is quite long, but not so difficult as you think so. All of the useless spaces are omitted, so please pay attention to the necessary spaces (marked in `\`).

```

1 \begin{equation}
2   \mathop{\rm rot}F=\left(\frac{\partial F_z}{\partial y}
3     -\frac{\partial F_y}{\partial z}\right)\hat{n}_x
4     +\left(\frac{\partial F_x}{\partial z}
5     -\frac{\partial F_z}{\partial x}\right)\hat{n}_y
6     +\left(\frac{\partial F_y}{\partial x}
7     -\frac{\partial F_x}{\partial y}\right)\hat{n}_z
8 \end{equation}
```

In math environments, unlike in plain text, normal spaces will not lead to visible spaces in output. Only `\` or `\quad`, `\qquad` etc. will create spaces between words.

`\partial` prints the symbol  $\partial$ , `\frac{...}{...}` makes a fraction.

`\left(` and `\right(` make braces that fit the equation's height.

It is written in plain-L<sup>A</sup>T<sub>E</sub>X, and things can even be easier with packages like `physics`, which will be demonstrated later.

# The `split` Environment (inline)

In order to deal with extremely long equations or equation with multiple lines, we can use the `split` environment. It is an `inline` environment being used in other maths environments.

## Example

```

1 \begin{equation}
2   \begin{split}
3     F &= 1+2+3+4+5 \\
4     &= 15
5   \end{split}
6 \end{equation}

```

$$F = 1 + 2 + 3 + 4 + 5 = 15 \quad (3)$$

`&` is used to align the equal marks, and `\\` is used to split the equation into two lines. Only one equation number will be generated in an `equation` environment.

The `split` environment is designed to serve as the entire body of an equation, or an entire line of an `align` or `gather` environment. There cannot be any printed material before or after it within the same enclosing structure.



# The `aligned` Environment (inline)

For linear equation systems, the `aligned` environment can be used, which is similar to the `split` environment above. It is also an `inline` environment, which can be used in `inline` mode such as `$$`! Here `split` doesn't work because `\left` and `\right` is an enclosing structure. See this discussion [▶ Link](#) for more information.

## Example

```

1 Equations:
2 $
3 \left\lbrace\begin{aligned}
4   x+y &= 1 \ \&\ x-y &= 1
5 \end{aligned}\right.
6 \Longrightarrow
7 \left\lbrace\begin{aligned}
8   x &= 1 \ \&\ y &= 0
9 \end{aligned}\right.
10 $
```

$$\text{Equations: } \begin{cases} x + y = 1 \\ x - y = 1 \end{cases} \implies \begin{cases} x = 1 \\ y = 0 \end{cases}$$

Actually things can also be easier with packages like `systeme`, which will be demonstrated later.

# The `align` Environment

An `align` environment can be used to simply the `split` or `aligned` in the `equation` environment. But it numbers the equation on each line.

## Example

```
1 \begin{align}
2   F &= 1+2+3+4+5 \\
3   &= 15 \\
4 \end{align}
```

$$F = 1 + 2 + 3 + 4 + 5 \quad (4)$$

$$= 15 \quad (5)$$

Use `align*` so that there will be no number(s).

## Example

```
1 \begin{align*}
2   a+b &\Leftrightarrow b+a \\
3   (a+b)+c &\Leftrightarrow a+(b+c) \\
4 \end{align*}
```

$$a + b \Leftrightarrow b + a$$

$$(a + b) + c \Leftrightarrow a + (b + c)$$

## The `gauss` Package

The ampersand character `&` determines where the equations align. The odd columns are right-aligned, and the even ones are left-aligned, so you can use `&&` if you want to make two neighbor column aligned to the same direction.

```

1 \begin{align*}
2 \text{\texttt{\textbackslash text{(right)}}} & \& \text{\texttt{\textbackslash text{(left)}}} & \& \text{\texttt{\textbackslash text{(left)}}} & \& \text{\texttt{\textbackslash text{(right)}}} \\
3 \& \& \text{\texttt{\textbackslash text{(right)}}} & \& \text{\texttt{\textbackslash text{(left)}}} & \& \\
4 x & \& =y & \& \& w & \& =z & \& \& a\&=b+c \& \& \\
5 2x & \& =-y & \& \& 3w & \& =z/2 & \& \& a\&=b \& \& \\
6 -4+5x & \& =2+y & \& \& w+2 & \& =-1+w & \& \& ab\&=cb \\
7 \end{align*}

```

(right)(left)	(left)	(right)	(right)(left)
$x = y$	$w$	$= z$	$a = b + c$
$2x = -y$	$3w$	$= z/2$	$a = b$
$-4 + 5x = 2 + y$	$w + 2$	$= -1 + w$	$ab = cb$

# The `cases` Environment (inline)

The linear system of equations can also be typeset simply with the `cases` environment. It is less flexible than an `aligned` environment, eg., there can only be one `&` on each row. Another minor difference is that the horizontal space before `&` is larger than other similar environments.

## Example

```

1  \begin{equation}
2  \left\{\begin{aligned}
3  x+y &= 1 \\
4  x-y &= 1
5  \end{aligned}\right.
6  \end{equation}
7
8  \begin{equation}
9  \begin{cases}
10 x+y &= 1 \\
11 x-y &= 1
12 \end{cases}
13 \end{equation}
```

$$\begin{cases} x + y = 1 \\ x - y = 1 \end{cases} \quad (6)$$

$$\begin{cases} x + y = 1 \\ x - y = 1 \end{cases} \quad (7)$$

# The `gather` Environment

## Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

If you just need to display a set of consecutive equations, centered and with no alignment whatsoever, use the `gather` environment. The asterisk trick to `set/unset` the numbering of equations also works here.

### Example

```
1 \begin{gather}
2   2x - 5y = 8 \\
3   3x^2 + 9y = 3a + c
4 \end{gather}
```

$$2x - 5y = 8 \quad (8)$$

$$3x^2 + 9y = 3a + c \quad (9)$$

# The `gathered` Environment (inline)

There is also an `inline` version of `gather`, called `gathered`. The relationship of them is similar to `align` and `aligned`.

## Example

```
1 \begin{equation}
2   \begin{gathered}
3     2x - 5y = 8 \\
4     3x^2 + 9y = 3a + c
5   \end{gathered}
6 \end{equation}
```

$$\begin{aligned} 2x - 5y &= 8 \\ 3x^2 + 9y &= 3a + c \end{aligned} \tag{10}$$

# The `multline` Environment

For equations longer than a line use the `multline` environment. Insert a double backslash to set a point for the equation to be broken. The first part will be aligned to the left and the second part will be displayed in the next line and aligned to the right.

## Example

```

1 \begin{multline}
2   p(x) = 3x^6 + 14x^5y + 590x^4y^2 + 19x^3y^3 \\
3         - 12x^2y^4 - 12xy^5 + 2y^6 - a^3b^3
4 \end{multline}

```

$$\begin{aligned}
 p(x) = & 3x^6 + 14x^5y + 590x^4y^2 + 19x^3y^3 \\
 & - 12x^2y^4 - 12xy^5 + 2y^6 - a^3b^3 \quad (11)
 \end{aligned}$$

The equation number will be in the last line, use `multline*` for no numbering.

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

For equations equal or longer than three lines,

## Example

```
1 \begin{multline*}
2   a+b+c=1 \\
3   b+c=2 \\
4   c+d=1 \\
5   d=3
6 \end{multline*}
```

$$a + b + c = 1$$

$$b + c = 2$$

$$c + d = 1$$

$$d = 3$$

Here, the first column is left-aligned, the last column is right-aligned and the others ones are center-aligned.



# The `flalign` Environment

For equations aligned left, use the `flalign` environment. It is similar to the `align` environment.

## Example

```

1 \begin{flalign}
2   a+b &=1=& b+a \\
3   b   &=2=& c
4 \end{flalign}

```

$$a + b = 1 =$$

$$b + a \quad (12)$$

$$b = 2 =$$

$$c \quad (13)$$

You may notice that the columns are flushed left (start from the left most position) and the right most column is flushed right, different from the `align` environment.

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

- 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X
  - Math Expressions
  - Math Environments
  - Spacing in Math Mode
  - Basic Math Commands
  - Matrices and Arrays

### 8 Useful Maths Packages

# Horizontal Spacing

Horizontal spacing in maths mode is useful in several situations, let's see an example: <sup>1</sup>

## Example

```

1  Assume we have the next sets
2  \[
3  S = \{ z \in \mathbb{C} \setminus, | \setminus, |z| < 1 \} \quad \quad
4  \textrm{and} \quad \quad S_2 = \partial S
5  \]
```

Assume we have the next sets

$$S = \{z \in \mathbb{C} \mid |z| < 1\} \quad \text{and} \quad S_2 = \partial S$$

As you see in this example, a mathematical text can be explicitly spaced by means of some special commands.

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

The spacing depends on the command you insert, the example below contains a complete list of spaces and how they look like.

Example

```
1 \begin{align*}
2 f(x) =& x^2\! + 3x\! + 2 \quad \!
3 f(x) =& x^2+3x+2 \quad
4 f(x) =& x^2\,, +3x\,, +2 \quad
5 f(x) =& x^2\: +3x\: +2 \quad
6 f(x) =& x^2\; +3x\; +2 \quad
7 f(x) =& x^2\! +3x\! +2 \quad
8 f(x) =& x^2\!\quad +3x\!\quad +2 \quad
9 f(x) =& x^2\!\quad\quad +3x\!\quad\quad +2
10 \end{align*}
```

$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$
$$f(x) = x^2 + 3x + 2$$

# Vertical Spacing

When the space between `display` maths and the main body paragraph is considered larger than expectation, is there any way to modify the line spacing?

In default style of `display` mode is like

## Example

```

1  your body paragraph is supposed to be typed here
2  \begin{equation}
3    a \times b = c
4  \end{equation}
5  your body paragraph is supposed to be typed here

```

your body paragraph is supposed to be typed here

$$a \times b = c \tag{14}$$

your body paragraph is supposed to be typed here

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

You can use `\setlength` to set the `displayskip`.

## Command

```
1 \setlength\abovedisplayskip{<length>}
2 \setlength\belowdisplayskip{<length>}
```

## Example

```
1 \setlength\abovedisplayskip{0em}
2 \setlength\belowdisplayskip{0em}
3 your body paragraph is supposed to be typed here
4 \begin{equation}
5   a \times b = c
6 \end{equation}
7 your body paragraph is supposed to be typed here
```

your body paragraph is supposed to be typed here

$$a \times b = c$$

(15)

your body paragraph is supposed to be typed here

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

- 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X
  - Math Expressions
  - Math Environments
  - Spacing in Math Mode
  - **Basic Math Commands**
  - Matrices and Arrays

### 8 Useful Maths Packages

# Fractions and Binomials

Fractions and binomial coefficients are common mathematical elements with similar characteristics - one number goes on top of another. <sup>1</sup>

## Command

```
1 \frac{top}{bottom} % fraction
2 \binom{top}{bottom} % binomial coefficients
```

Using fractions and binomial coefficients in an expression is straightforward.

## Example

```
1 The binomial coefficient is defined by the next expression:
2 \[ \binom{n}{k} = \frac{n!}{k!(n-k)!} \]
```

The binomial coefficient is defined by the next expression:

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

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:



Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

In `inline` and `display` mode, the appearance of the fractions and binomials may differ. You can use `\displaystyle` or `\textstyle` to adjust the size of the fractions and binomials, or use `\dfrac` if not all fractions in an equation need to be resized.

## Example

```

1 When displaying fractions in-line, for example  $\frac{3x}{2}$ 
2 you can set a different display style:  $ \displaystyle \frac{3x}{2} $ .
3 Or you can use  $\dfrac{3x}{2}$ . This is also true the other way around
4  \[ f(x)=\binom{n}{x}=\frac{n!}{x!(n-x)!} \quad \text{and} \quad 
5  f(x)=\textstyle\binom{n}{x}=\frac{n!}{x!(n-x)!} \]

```

When displaying fractions in-line, for example  $\frac{3x}{2}$  you can set a different display style:  $\frac{3x}{2}$ . Or you can use  $\frac{3x}{2}$ . This is also true the other way around

$$f(x) = \binom{n}{x} = \frac{n!}{x!(n-x)!} \quad \text{and} \quad f(x) = \binom{n}{x} = \frac{n!}{x!(n-x)!}$$

The command `\displaystyle` will format the fractions and binomials as if they were in mathematical display mode. On the other side, `\textstyle` will change the style of them as if they were part of the text.

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

The usage of fractions is quite flexible, they can be nested to obtain more complex expressions. And `\cfrac` can be used to make continued fractions.

## Example

- 1 The fractions can be nested
- 2 `\[ \frac{1+\frac{a}{b}}{1+\frac{1}{1+\frac{1}{a}}} \]`
- 3 Now a wild example
- 4 `\[ a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cdots}}} \]`

The fractions can be nested

$$\frac{1 + \frac{a}{b}}{1 + \frac{1}{1 + \frac{1}{a}}}$$

Now a wild example

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3 + \cdots}}}$$

# Operators

Characters in mathematical mode are usually shown in italics, but sometimes especial function names require different formatting (font and skip), this is accomplished by using operators defined in L<sup>A</sup>T<sub>E</sub>X.<sup>1</sup>

Trigonometrical functions, logarithms, and some others can be written in a document by means of some special commands.

## Example

```
1 \[ \sin(a + b) = \sin(a)\cos(b) + \cos(a)\sin(b) \]
2 \[ \log_a b = \frac{\log_c b}{\log_c a} = \frac{\ln b}{\ln a} \]
3 \[ \tan a, \quad \arccos a, \quad \arcsin a, \quad \arctan a \]
```

$$\sin(a + b) = \sin(a) \cos(b) + \cos(a) \sin(b)$$

$$\log_a b = \frac{\log_c b}{\log_c a} = \frac{\ln b}{\ln a}$$

$$\tan a, \quad \arccos a, \quad \arcsin a, \quad \arctan a$$

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

## Integrals

Integral expression can be added using the command

## Command

```
1 \int_{lower}^{upper}
```

Note, that integral expression may seems a little different in `inline` and `display` math mode - in `inline` mode the integral symbol and the limits are compressed.

## Example

```
1 Integral $\int_{a}^{b} x^2 dx$
   ↳ inside text
2 \[ \int_{a}^{b} x^2 dx \]
```

Integral  $\int_a^b x^2 dx$  inside text

$$\int_a^b x^2 dx$$

There is always an argue about whether *italic* or roman style of “d” should be used in integrals and derivatives. There’s no right or wrong. If you prefer to use roman style, try `commath` or `physics` package. Either of them provides some macros to insert the “d” you want simply.

# Multiple Integrals

To obtain double/triple/multiple integrals you must use `amsmath` package.

## Example

```

1 \begin{gather*}
2 \iint_V \mu(u,v) \, du \, dv \\
3 \iiint_V \mu(u,v,w) \, du \, dv \, dw \\
4 \iiint_V \mu(t,u,v,w) \\
5 \quad \rightarrow \, dt \, du \, dv \, dw \\
6 \idotsint_V \mu(u_1, \dots, u_k) \, du_1 \\
7 \quad \rightarrow \, \dots \, du_k \\
8 \end{gather*}

```

$$\iint_V \mu(u, v) \, du \, dv$$

$$\iiint_V \mu(u, v, w) \, du \, dv \, dw$$

$$\iiint_V \mu(t, u, v, w) \, dt \, du \, dv \, dw$$

$$\int \cdots \int_V \mu(u_1, \dots, u_k) \, du_1 \dots du_k$$

# Cyclic Integrals

## Use Maths in L<sup>A</sup>T<sub>E</sub>X

[Math Expressions](#)[Math Environments](#)[Spacing in Math Mode](#)[Basic Math Commands](#)[Matrices and Arrays](#)

## Useful Maths Packages

[Common Packages](#)[The `physics` Package](#)[The `systeme` Package](#)[The `gauss` Package](#)

To obtain cyclic integrals you must use `esint` package.

### Example

```
1 \begin{gather*}
2 \oint_V f(s) \, ds \\
3 \oiint_V f(s,t) \, ds \, dt \\
4 \end{gather*}
```

$$\oint_V f(s) \, ds$$
$$\oiint_V f(s,t) \, ds \, dt$$

# Limits, Sums and Products

Like integrals, limits, sums and products expression are compressed in `inline` mode.

## Command

```
1 \limits_{lower}
2 \sum_{lower}^{\upper}
3 \prod_{lower}^{\upper}
```

## Example

```
1 Limit $\lim_{x\to\infty} f(x)$ inside text
2 \[ \lim_{x\to\infty} f(x) \]
```

Limit  $\lim_{x \rightarrow \infty} f(x)$  inside text

$$\lim_{x \rightarrow \infty} f(x)$$

## Example

- 1 Sum `$\sum_{n=1}^{\infty} 2^{-n} = 1$` inside text
- 2 `\[ \sum_{n=1}^{\infty} 2^{-n} = 1 \]`

Sum  $\sum_{n=1}^{\infty} 2^{-n} = 1$  inside text

$$\sum_{n=1}^{\infty} 2^{-n} = 1$$

## Example

- 1 Product `$\prod_{i=a}^b f(i)$` inside text
- 2 `\[ \prod_{i=a}^b f(i) \]`

Product  $\prod_{i=a}^b f(i)$  inside text

$$\prod_{i=a}^b f(i)$$



# Improvement of Integrals, Limits, Sums and Products

In `inline` math mode the integral/sum/product lower and upper limits are placed right of integral symbol. Similar is for limit expressions. If you want the limits of an integral/sum/product to be specified above and below the symbol in `inline` math mode (or in `display` mode), use the `\limits` command before limits specification.

## Example

- 1 Integral `\int_{a}^{b} x^2 dx` inside text `\par`
- 2 Improved integral `\int\limits_{a}^{b} x^2 dx` inside text `\par`
- 3 Use limits in display mode `\[ \int\limits_{a}^{b} x^2 dx \]`

Integral  $\int_a^b x^2 dx$  inside text

Improved integral  $\int\limits_a^b x^2 dx$  inside text

Use limits in display mode

$$\int\limits_a^b x^2 dx$$

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

Moreover, adding `\displaystyle` beforehand will make the symbol in inline mode large and easier to read, as in display mode.

## Example

- 1 Limit `\lim_{x\to\infty} f(x)` inside text `\par`
- 2 Display style limit `\displaystyle\lim_{x\to\infty} f(x)` inside text

Limit  $\lim_{x \rightarrow \infty} f(x)$  inside text

Display style limit  $\lim_{x \rightarrow \infty} f(x)$  inside text

On the other hand, `\mathlarger` command (provided by `relsize` package) is used to get bigger integral symbol in display.

## Example

- 1 `\int \frac{1}{2} dx - \mathlarger{\int \frac{1}{2} dx}`

$$\int \frac{1}{2} dx - \int \frac{1}{2} dx$$

# Other Math Symbols

Some examples of other common used math symbols are shown.

Name	$\LaTeX$	Code
Square Root	$\sqrt{a} \sqrt[b]{a}$	<code>\sqrt {a}\ \ \sqrt [b]{a}</code>
Over/Under Line	$\overline{a+b} \quad \underline{a+b}$	<code>\overline {a+b}\ \ \underline {a+b}</code>
Over Brace	$\overbrace{1+2+\cdots+n}^n$	<code>\overbrace {1+2+\cdots +n}^n</code>
Under Brace	$\underbrace{1+2+\cdots+n}_n$	<code>\underbrace {1+2+\cdots +n}_n</code>
Over Arrow	$\overrightarrow{a+b} \quad \overleftarrow{a+b}$	<code>\overrightarrow {a+b}\ \ \overleftarrow {a+b}</code>
Under Arrow	$\underrightarrow{a+b} \quad \underleftarrow{a+b}$	<code>\underrightarrow {a+b}\ \ \underleftarrow {a+b}</code>
Dots	$\dots \quad \cdots \quad \vdots \quad \ddots$	<code>\dots \ \ \cdot \ \ \cdots \ \ \vdots \ \ \ddots</code>
Arrows	$\rightarrow \quad \leftarrow \quad \leftrightarrow$ $\Rightarrow \quad \Leftarrow \quad \Leftrightarrow$ $\longleftarrow \quad \Longrightarrow$	<code>\rightarrow \ \ \leftarrow \ \ \leftrightarrow</code> <code>\Rightarrow \ \ \Leftarrow \ \ \Leftrightarrow</code> <code>\longleftarrow \ \ \Longrightarrow</code>

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

## Mathematical Fonts

In mathematical mode as well as in text mode, you can change the typeface as needed. For instance, it's customary to represent real numbers with a blackboard bold font, or topological spaces with calligraphic font.<sup>1</sup>

For some elements is convenient to have the possibility of changing the font typeface.

## Example

```
1 Let \(\mathcal{T}\) be a topological space, a basis is defined as
2 \[ \mathcal{B} = \{B_{\alpha} \mid B_{\alpha} \in \mathcal{T}, \bigcup_{\alpha} B_{\alpha} = U, U \in \mathcal{T}\}
3 U = \bigcup_{\alpha} B_{\alpha} \text{ for all } U \in \mathcal{T} \]
```

Let  $\mathcal{T}$  be a topological space, a basis is defined as

$$\mathcal{B} = \{B_{\alpha} \in \mathcal{T} \mid U = \bigcup_{\alpha} B_{\alpha} \forall U \in \mathcal{T}\}$$

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

# Mathematical Fonts for Capital Letters

There are some font typefaces that support only a limited number of characters; these fonts usually denote some special sets.

## Example

1	<code>\begin{gather*}</code>	
2	<code>RQSZ \\</code>	<i>RQSZ</i>
3	<code>\mathcal{RQSZ} \\</code>	<i>RQSZ</i>
4	<code>\mathfrak{RQSZ} \\</code>	<i>RQSZ</i>
5	<code>\mathbb{RQSZ}</code>	<b>RQSZ</b>
6	<code>\end{gather*}</code>	<i>RQSZ</i>

This example shows Calligraphic, Fraktur and Blackboard bold typefaces. For instance, to display the R in blackboard bold typeface `\mathbb{R}` will do the trick.

# Other Mathematical Fonts

It's possible to set a different font family for a complete mathematical expression.

## Example

```

1 \begin{gather*}
2   3x^2 \in R \subset Q \\
3   \mathnormal{3x^2 \in R \subset Q} \\
4   \mathrm{3x^2 \in R \subset Q} \\
5   \mathit{3x^2 \in R \subset Q} \\
6   \mathbf{3x^2 \in R \subset Q} \\
7   \mathsf{3x^2 \in R \subset Q} \\
8   \mathtt{3x^2 \in R \subset Q} \\
9 \end{gather*}

```

$$3x^2 \in R \subset Q$$

$$3x^2 \in R \subset Q$$

$$3x^2 \in R \subset Q$$

$$3x^2 \in R \subset Q$$

$$\mathbf{3x^2 \in R \subset Q}$$

$$3x^2 \in R \subset Q$$

$$3x^2 \in R \subset Q$$

In this case, not only letters but all characters change its appearance, for example `\mathit{3x^2}` italicises the entire expression.

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

## Define Own Symbols

If you need to add a personalized operator to be displayed in Roman font instead of italics use `\DeclareMathOperator`, provided by the the package `amsmath`.

## Example

```
1 \DeclareMathOperator{\Mr}{M_{\mathbb{R}}}
2 User-defined operator for matrices with Real entries $ x \in \Mr $
```

User-defined operator for matrices with Real entries  $x \in M_{\mathbb{R}}$

The command can be slightly modified if you need that your defined operator uses subscripts, as the `\lim` operator, in such case use `\DeclareMathOperator*`.

You can also use `\mathop` to define a italics math operator supporting subscripts, and change it to Roman font by hand.

## Example

```
1 \[ \mathop{\mathrm{limsup}}_{n \rightarrow \infty} \mathop{\mathrm{rot}} F_n \]
```

$$\limsup_{n \rightarrow \infty} \operatorname{rot} F_n$$

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

**Matrices and Arrays**

### Useful Maths Packages

Common Packages

The **physics** Package

The **systeme** Package

The **gauss** Package

- 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X
  - Math Expressions
  - Math Environments
  - Spacing in Math Mode
  - Basic Math Commands
  - **Matrices and Arrays**

### 8 Useful Maths Packages



# The `matrix` Environment (inline)

There are various kinds of matrix environments defined in `amsmath` package, they are `matrix`, `pmatrix`, `bmatrix`, `Bmatrix`, `vmatrix`, `Vmatrix`.

## Command

```

1 \begin{[p/b/B/v/V]matrix}
2   a_{11} & a_{12} & \dots & a_{1n} \\
3   a_{21} & a_{22} & \dots & a_{2n} \\
4   \dots & \dots & \dots & \dots \\
5   a_{n1} & a_{n2} & \dots & a_{nn} \\
6 \end{[p/b/B/v/V]matrix}
```

## Example

```

1 \begin{equation}
2   \begin{pmatrix}
3     a_{11} & a_{12} & a_{13} \\
4     a_{21} & a_{22} & a_{23} \\
5     a_{31} & a_{32} & a_{33}
6   \end{pmatrix}
7 \end{equation}
```

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad (16)$$

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The **physics** PackageThe **systeme** PackageThe **gauss** Package

Here is some examples of the style of these matrix.

## Example

**matrix**

$$\begin{matrix} a & b \\ c & d \end{matrix}$$

**bmatrix**

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

**vmatrix**

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

**pmatrix**

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

**Bmatrix**

$$\begin{Bmatrix} a & b \\ c & d \end{Bmatrix}$$

**Vmatrix**

$$\begin{Vmatrix} a & b \\ c & d \end{Vmatrix}$$

Some packages may also help simplify the typesetting of matrix, for example, there is some macros defined in the **physics** package to make identity matrix, or generate the examples above more simply.

Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

## Useful Maths Packages

Common Packages

The `physics` PackageThe `systeme` PackageThe `gauss` Package

If you need to create matrices with different delimiters, you can add them manually to a plain matrix. For example:

## Example

```
1 \begin{equation}
2 \left\lceil
3 \begin{matrix}
4 1 & 2 & 3 \\
5 a & b & c
6 \end{matrix}
7 \right\rceil
8 \end{equation}
9
10 \begin{equation}
11 \left\langle
12 \begin{matrix}
13 1 & 2 & 3 \\
14 a & b & c
15 \end{matrix}
16 \right\rangle
17 \end{equation}
```

$$\left[ \begin{array}{ccc} 1 & 2 & 3 \\ a & b & c \end{array} \right] \quad (17)$$

$$\left\langle \begin{array}{ccc} 1 & 2 & 3 \\ a & b & c \end{array} \right\rangle \quad (18)$$

# The `smallmatrix` Environment

When typesetting inline math, the usual `matrix` environments above may look too big. It may be better to use `smallmatrix` in such situations, although you will need to provide your own delimiters.

## Example

```

1  Trying to typeset an inline matrix here
2  $$\begin{pmatrix}
3      a & b \\\
4      c & d
5  \end{pmatrix}
6  but it looks too big, so let's try
7  $$\big(\begin{smallmatrix}
8      a & b \\\
9      c & d
10 \end{smallmatrix}\big) instead.
```

Trying to typeset an inline matrix here  $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$  but it looks too big, so let's try  $\big(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\big)$  instead.

# The `array` Environment

An `array` environment is actually a math mode `tabular` environment, and the usage of them are almost the same. You can refer to the lecture about tables for this part.

A simple example is given here:

## Example

```

1 \begin{equation}
2   \chi(\lambda) =
3   \left| \begin{array}{ccc}
4     \lambda - a & -b & -c \\
5     -d & \lambda - e & -f \\
6     -g & -h & \lambda - i
7   \end{array} \right|
8 \end{equation}

```

$$\chi(\lambda) = \begin{vmatrix} \lambda - a & -b & -c \\ -d & \lambda - e & -f \\ -g & -h & \lambda - i \end{vmatrix} \quad (19)$$

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

## 8 Useful Maths Packages

- Common Packages

- The `physics` Package

- The `systeme` Package

- The `gauss` Package

# The $\text{AMS-}\text{\LaTeX}$ Packages

$\text{AMS-}\text{\LaTeX}$  is a collection of  $\text{\LaTeX}$  document classes and packages developed for the American Mathematical Society (AMS).

It is an extension of plain- $\text{\LaTeX}$  maths, with many new maths environments (most of them were introduced in the previous section), maths symbols and maths fonts.

Usually you can insert all of the commands in the preamble of your document.

## Command

```
1 \usepackage{amsmath} % loads maths environments
2 \usepackage{amssymb} % loads maths symbols
3 \usepackage{amsfonts} % loads maths fonts
```

# Some Other Packages

Recall that we also use some other packages in this lecture:

## Command

```
1 \usepackage{esint}    % for cyclic integrals
2 \usepackage{relsize}  % for \mathlarger
```

For a better `array` environment, though it's not mandatory (you can use it without the package), you're recommended to add the `array` package.

## Command

```
1 \usepackage{array}
```



### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The **physics** Package

The **systeme** Package

The **gauss** Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

## 8 Useful Maths Packages

- Common Packages
- The **physics** Package
- The **systeme** Package
- The **gauss** Package

# The `physics` Package

To use the `physics` package, simply insert the command in the preamble of your document.

## Command

```
1 \usepackage{physics}
```

The goal of this package is to make typesetting equations for `physics` simpler, faster, and more human-readable. But it can also be used in various maths circumstances.

To that end, the commands included in this package have names that make the purpose of each command immediately obvious and remove any ambiguity while reading and editing `physics` code.

The documentation of the `physics` package can be found in <http://mirrors.ctan.org/macros/latex/contrib/physics/physics.pdf>.

Recall the equation:

```

1 \begin{equation}
2   \mathop{\rm rot} F = \left( \frac{\partial F_z}{\partial y} - \frac{\partial F_y}{\partial z} \right) \hat{n}_x
3   + \left( \frac{\partial F_x}{\partial z} - \frac{\partial F_z}{\partial x} \right) \hat{n}_y
4   + \left( \frac{\partial F_y}{\partial x} - \frac{\partial F_x}{\partial y} \right) \hat{n}_z
5   \end{equation}

```

Now we can rewrite it with the commands defined in the `physics` package.

## Example

```

1 \begin{equation}
2   \mathop{\rm rot} F = \qty(\pdv{F_z}{y} - \pdv{F_y}{z}) \hat{n}_x +
3   \qty(\pdv{F_x}{z} - \pdv{F_z}{x}) \hat{n}_y +
4   \qty(\pdv{F_y}{x} - \pdv{F_x}{y}) \hat{n}_z
5   \end{equation}


```


$$\operatorname{rot} F = \left( \frac{\partial F_z}{\partial y} - \frac{\partial F_y}{\partial z} \right) \hat{n}_x + \left( \frac{\partial F_x}{\partial z} - \frac{\partial F_z}{\partial x} \right) \hat{n}_y + \left( \frac{\partial F_y}{\partial x} - \frac{\partial F_x}{\partial y} \right) \hat{n}_z \quad (20)$$


# Automatic Bracing

When typesetting maths equations, you may use something like `\left(` and `\right)` to make the braces taller than the typical ones. The `physics` package provides some macros to simplify and replace them.


`\quantity`      `\qty(\typical)` →       automatic ( ) braces

`\qty(\tall)` → 

`\qty(\grande)` → 

`\qty[\typical]` → 

automatic [ ] braces

`\qty|\typical|` → 


automatic | | braces

`\qty{\typical}` → 


automatic { } braces

`\qty\big{}` → 

manual sizing (works with any of the above bracket types)

`\qty\Big{}` → 

`\qty\bigg{}` → 

`\qty\Bigg{}` → 

`\pqty{}` ↔ `\qty()`

alternative syntax; robust and more  $\text{\LaTeX}$ -friendly

`\bqty{}` ↔ `\qty|`

`\vqty{}` ↔ `\qty|`

`\Bqty{}` ↔ `\qty{}`

`\absolutevalue`

$$\backslash\mathrm{abs}\{a\} \rightarrow |a|$$

$$\backslash\mathrm{abs}\Big{a\Big} \rightarrow \Big|a\Big|$$

$$\backslash\mathrm{abs*}\{\mathrm{grande}\} \rightarrow \big| \big|$$

`\norm`

$$\backslash\mathrm{norm}\{a\} \rightarrow \|a\|$$

$$\backslash\mathrm{norm}\Big{a\Big} \rightarrow \Big\|a\Big\|$$

$$\backslash\mathrm{norm*}\{\mathrm{grande}\} \rightarrow \big\| \big\|$$

`\evaluated`

$$\backslash\mathrm{eval}\{x\}_0^{\infty} \rightarrow x \Big|_0^{\infty}$$

$$\backslash\mathrm{eval}(x|_0^{\infty} \rightarrow \left(x\right) \Big|_0^{\infty}$$

$$\backslash\mathrm{eval}[x|_0^{\infty} \rightarrow \left[x\right] \Big|_0^{\infty}$$

$$\backslash\mathrm{eval}[\mathrm{venti}|_0^{\infty} \rightarrow \left[\mathrm{venti}\right] \Big|_0^{\infty}$$

$$\backslash\mathrm{eval}[\mathrm{venti}|_0^{\infty} \rightarrow \left[\mathrm{venti}\right] \Big|_0^{\infty}$$

$$\backslash\mathrm{eval}[\mathrm{venti}|_0^{\infty} \rightarrow \left[\mathrm{venti}\right] \Big|_0^{\infty}$$

$$\backslash\mathrm{eval*}[\mathrm{venti}|_0^{\infty} \rightarrow \big[\mathrm{venti}\big] \Big|_0^{\infty}$$

`\order`

$$\backslash\mathrm{order}\{x^2\} \rightarrow \mathcal{O}(x^2)$$

$$\backslash\mathrm{order}\Big{x^2\Big} \rightarrow \Big\mathcal{O}(x^2)\Big$$

$$\backslash\mathrm{order*}\{\mathrm{grande}\} \rightarrow \big\mathcal{O}(\big)$$

automatic sizing; equivalent to `\qty`  
`|a|`

inherits manual sizing syntax from `\qty`

star for no resize

automatic sizing

manual sizing

star for no resize

vertical bar for evaluation limits

alternate form

alternate form

automatic sizing

star for no resize

order symbol; automatic sizing and space handling

manual sizing

star for no resize

# Vector Notation

You may use `\mathbf` to make bold maths symbols, However, it won't always work. For example, with `\mathbf{\alpha}` you may have  $\alpha$ , which is actually not bold. These commands will help provide the correct  $\alpha$ .

<code>\vectorbold</code>	<code>\vb{a}</code> → $\mathbf{a}$	upright/no Greek
	<code>\vb*{a}</code> , <code>\vb*{\theta}</code> → $\boldsymbol{a}$ , $\boldsymbol{\theta}$	italic/Greek
<code>\vectorarrow</code>	<code>\va{a}</code> → $\vec{a}$	upright/no Greek
	<code>\va*{a}</code> , <code>\va*{\theta}</code> → $\vec{a}$ , $\vec{\theta}$	italic/Greek
<code>\vectorunit</code>	<code>\vu{a}</code> → $\hat{a}$	upright/no Greek
	<code>\vu*{a}</code> , <code>\vu*{\theta}</code> → $\hat{a}$ , $\hat{\theta}$	italic/Greek

There are also some shorthand for vector operations.

<code>\dotproduct</code>	<code>\vdot</code> → $\cdot$ as in $\mathbf{a} \cdot \mathbf{b}$	note: <code>\dp</code> is a protected T <sub>E</sub> X primitive
<code>\crossproduct</code>	<code>\cross</code> → $\times$ as in $\mathbf{a} \times \mathbf{b}$	alternate name
	<code>\cp</code> → $\times$ as in $\mathbf{a} \times \mathbf{b}$	shorthand name

The default del (nabla) symbol  $\nabla$  used in `physics` vector notation can be switched to appear with an arrow  $\vec{\nabla}$  by including the option `arrowdel` in the document preamble  $\rightarrow$  `\usepackage[arrowdel]{physics}`.

<code>\divergence</code>	<code>\div</code> $\rightarrow \nabla \cdot$	note: <code>amsmath</code> symbol $\div$ renamed <code>\divisionsymbol</code>
	<code>\div{\vb{a}}</code> $\rightarrow \nabla \cdot \mathbf{a}$	default mode
	<code>\div(\vb{a}+\tall)</code> $\rightarrow \nabla \cdot (\mathbf{a} + \blacksquare)$	long-form
	<code>\div[\vb{a}+\tall]</code> $\rightarrow \nabla \cdot \left[ \mathbf{a} + \blacksquare \right]$	
<code>\curl</code>	<code>\curl</code> $\rightarrow \nabla \times$	
	<code>\curl{\vb{a}}</code> $\rightarrow \nabla \times \mathbf{a}$	default mode
	<code>\curl(\vb{a}+\tall)</code> $\rightarrow \nabla \times (\mathbf{a} + \blacksquare)$	long-form
	<code>\curl[\vb{a}+\tall]</code> $\rightarrow \nabla \times \left[ \mathbf{a} + \blacksquare \right]$	
<code>\laplacian</code>	<code>\laplacian</code> $\rightarrow \nabla^2$	
	<code>\laplacian{\Psi}</code> $\rightarrow \nabla^2 \Psi$	default mode
	<code>\laplacian(\Psi+\tall)</code> $\rightarrow \nabla^2 (\Psi + \blacksquare)$	long-form
	<code>\laplacian[\Psi+\tall]</code> $\rightarrow \nabla^2 \left[ \Psi + \blacksquare \right]$	

# Operators

The standard set of trig functions is redefined in `physics` to provide automatic braces that behave like `\qty()`. In addition, an optional power argument is provided. This behavior can be switched off by including the option `notrig` in the preamble  $\rightarrow$  `\usepackage[notrig]{physics}`.

For example,

<code>\sin</code>	<code>\sin(\grande)</code> $\rightarrow$ $\sin(\blacksquare)$	automatic braces; old <code>\sin</code> renamed <code>\sine</code>
	<code>\sin[2](x)</code> $\rightarrow$ $\sin^2(x)$	optional power
	<code>\sin x</code> $\rightarrow$ $\sin x$	can still use without an argument

Similar behavior has also been extended to the following functions:

<code>\exp(\tall)</code>	<code>exp(\tall)</code>	<code>\exponential</code>
<code>\log(\tall)</code>	<code>log(\tall)</code>	<code>\logarithm</code>
<code>\ln(\tall)</code>	<code>ln(\tall)</code>	<code>\naturallogarithm</code>
<code>\det(\tall)</code>	<code>det(\tall)</code>	<code>\determinant</code>
<code>\Pr(\tall)</code>	<code>Pr(\tall)</code>	<code>\Probability</code>

old definitions  $\Rightarrow$



Use Maths in L<sup>A</sup>T<sub>E</sub>X

- Math Expressions
- Math Environments
- Spacing in Math Mode
- Basic Math Commands
- Matrices and Arrays

Useful Maths Packages

- Common Packages
- The `physics` Package
- The `systeme` Package
- The `gauss` Package

There are also some new operators:

<code>\trace</code> or <code>\tr</code>	<code>\tr\rho</code> $\rightarrow \operatorname{tr} \rho$ also <code>\tr(\tall)</code> $\rightarrow \operatorname{tr}(\blacksquare)$	trace; same bracing as trig functions
<code>\Trace</code> or <code>\Tr</code>	<code>\Tr\rho</code> $\rightarrow \operatorname{Tr} \rho$	alternate
<code>\rank</code>	<code>\rank M</code> $\rightarrow \operatorname{rank} M$	matrix rank
<code>\erf</code>	<code>\erf(x)</code> $\rightarrow \operatorname{erf}(x)$	Gauss error function
<code>\Res</code>	<code>\Res[f(z)]</code> $\rightarrow \operatorname{Res}[f(z)]$	residue; same bracing as trig functions
<code>\principalvalue</code>	<code>\pv{\int f(z) \dd{z}}</code> $\rightarrow \mathcal{P} \int f(z) \operatorname{d} z$ <code>\PV{\int f(z) \dd{z}}</code> $\rightarrow \operatorname{P.V.} \int f(z) \operatorname{d} z$	Cauchy principal value
<code>\Re</code>	<code>\Re{z}</code> $\rightarrow \operatorname{Re}\{z\}$	old <code>\Re</code> renamed to <code>\real</code> $\rightarrow \Re$
<code>\Im</code>	<code>\Im{z}</code> $\rightarrow \operatorname{Im}\{z\}$	old <code>\Im</code> renamed to <code>\imaginary</code> $\rightarrow \Im$

# Quick Quad Text

This set of commands produces text in math-mode padded by `\quad` spacing on either side. This is meant to provide a quick way to insert simple words or phrases in a sequence of equations. Each of the following commands includes a starred version which pads the text only on the right side with `\quad` for use in aligned environments such as `cases`.

`\qqtext`

`\qq{}`

`\qq{word or phrase}` →  
 \_\_\_\_ word or phrase \_\_\_\_

general quick quad text with argument  
 normal mode; left and right `\quad`

`\qq*{word or phrase}` →  
 word or phrase \_\_\_\_

starred mode; right `\quad` only

Some special macros:

`\qcomma` or `\qc` → , \_\_\_\_

right `\quad` only

`\qcc` → \_\_\_\_ c.c. \_\_\_\_

complex conjugate; left and right `\quad` unless starred `\qcc*`  
 → c.c. \_\_\_\_

`\qif` → \_\_\_\_ if \_\_\_\_

left and right `\quad` unless starred `\qif*` → if \_\_\_\_

`\qthen`, `\qelse`, `\qotherwise`, `\qunless`, `\qgiven`, `\qusing`, `\qassume`, `\qsince`,  
`\qlet`, `\qfor`, `\qall`, `\qeven`, `\qodd`, `\qinteger`, `\qand`, `\qor`, `\qas`, `\qin`

The default differential symbol `d` which is used in `\differential` and `\derivative` can be switched to an italic form *d* by including the option `italicdiff` in the preamble  $\rightarrow$  `\usepackage[italicdiff]{physics}`.

`\differential`

`\dd`  $\rightarrow$   $d$

`\dd x`  $\rightarrow$   $dx$

`\dd{x}`  $\rightarrow$   $\mathrm{d}x$

`\dd[3]{x}`  $\rightarrow$   $d^3x$

`\dd(\cos\theta)`  $\rightarrow$   $d(\cos\theta)$

`\derivative`

`\dv{x}`  $\rightarrow$   $\frac{d}{dx}$

`\dv{f}{x}`  $\rightarrow$   $\frac{df}{dx}$

`\dv[n]{f}{x}`  $\rightarrow$   $\frac{d^n f}{dx^n}$

`\dv{x}(\grande)`  $\rightarrow$   $\frac{d}{dx} \left( \text{blue square} \right)$

`\dv*{f}{x}`  $\rightarrow$   $df/dx$

- no spacing (not recommended)
- automatic spacing based on neighbors
- optional power
- long-form; automatic braces
- one argument
- two arguments
- optional power
- long-form; automatic braces, spacing
- inline form using `\flatfrac`

Use Maths in L<sup>A</sup>T<sub>E</sub>X

- Math Expressions
- Math Environments
- Spacing in Math Mode
- Basic Math Commands
- Matrices and Arrays

Useful Maths Packages

- Common Packages
- The `physics` Package
- The `systeme` Package
- The `gauss` Package

<code>\partial</code>	<code>\partial</code>	$\frac{\partial}{\partial x}$	alternate name
<code>\partial</code>	<code>\partial</code>	$\frac{\partial}{\partial x}$	shorthand name
<code>\partial</code>	<code>\partial</code>	$\frac{\partial f}{\partial x}$	two arguments
<code>\partial</code>	<code>\partial</code>	$\frac{\partial^n f}{\partial x^n}$	optional power
<code>\partial</code>	<code>\partial</code>	$\frac{\partial}{\partial x} \left( \text{blue square} \right)$	long-form
<code>\partial</code>	<code>\partial</code>	$\frac{\partial^2 f}{\partial x \partial y}$	mixed partial
<code>\partial</code>	<code>\partial</code>	$\frac{\partial f}{\partial x}$	inline form using <code>\flatfrac</code>
<code>\partial</code>	<code>\partial</code>	$\delta F[g(x)]$	functional variation (works like <code>\dd</code> )
<code>\partial</code>	<code>\partial</code>	$\delta(E - TS)$	long-form
<code>\partial</code>	<code>\partial</code>	$\frac{\delta}{\delta g}$	functional derivative (works like <code>\dv</code> )
<code>\partial</code>	<code>\partial</code>	$\frac{\delta F}{\delta g}$	
<code>\partial</code>	<code>\partial</code>	$\frac{\delta}{\delta V} (E - TS)$	long-form
<code>\partial</code>	<code>\partial</code>	$\delta F / \delta x$	inline form using <code>\flatfrac</code>

The following matrix macros produce unformatted rows and columns of matrix elements for use as separate matrices as well as blocks within larger matrices. For example, the command `\identitymatrix{2}` which also has the shortcut `\imat{2}` produces the elements of a  $2 \times 2$  identity matrix  $\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}$  without braces or grouping. This allows the command to also be used within another matrix, as in:

### Example

```
1 \begin{equation}
2   \begin{pmatrix}
3     \imat{2} \\
4     a & b
5   \end{pmatrix}
6 \end{equation}
```

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ a & b \end{pmatrix} \quad (21)$$

To specify elements on the right of left sides of our `\imat{2}` sub-matrix we use the grouping command `\matrixquantity` or `\mqty` to effectively convert `\imat{2}` into a single matrix element of a larger matrix:

### Example

```
1 \begin{equation}
2   \begin{pmatrix}
3     \mqty{\imat{2}} & \mqty{a\\b} \\
4     \mqty{c & d}    & e
5   \end{pmatrix}
6 \end{equation}
```

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ c & d & e \end{pmatrix} \quad (22)$$

The extra `\mqty` groups were required in this case in order to get the  $a$  and  $b$  elements to behave as a single element, since `\mqty{\imat{2}}` also acts like a single matrix element (the same can be said of the grouped  $c$  and  $d$  elements). Finally, the outermost `pmatrix` environment could have also been replaced with the `physics` macro `\mqty()`, allowing the above example to be written on one line:

### Example

```

1 \begin{equation}
2   \mqty(
3     \mqty{\imat{2}} & \mqty{a\\b} \\
4     \mqty{c & d}    & e
5   )
6 \end{equation}

```

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ c & d & e \end{pmatrix} \quad (23)$$

The matrix commands are listed below:

<code>\matrixquantity</code>	<code>\mqty{a &amp; b \\ c &amp; d}</code>	$\rightarrow \begin{matrix} a & b \\ c & d \end{matrix}$
	<code>\mqty(a &amp; b \\ c &amp; d)</code>	$\rightarrow \begin{pmatrix} a & b \\ c & d \end{pmatrix}$
	<code>\mqty*(a &amp; b \\ c &amp; d)</code>	$\rightarrow \begin{bmatrix} a & b \\ c & d \end{bmatrix}$
	<code>\mqty[a &amp; b \\ c &amp; d]</code>	$\rightarrow \left[ \begin{matrix} a & b \\ c & d \end{matrix} \right]$
	<code>\mqty a &amp; b \\ c &amp; d </code>	$\rightarrow \left  \begin{matrix} a & b \\ c & d \end{matrix} \right $
	<code>\pmmqty{}</code>	$\leftrightarrow \mqty{}$
	<code>\Pmqty{}</code>	$\leftrightarrow \mqty*{}$
	<code>\bmqty{}</code>	$\leftrightarrow \mqty\blacksquare$
	<code>\vmqty{}</code>	$\leftrightarrow \mqty  $
<code>\smallmatrixquantity</code>	<code>\smqty{a &amp; b \\ c &amp; d}</code>	$\rightarrow \begin{smallmatrix} a & b \\ c & d \end{smallmatrix}$
	<code>\smqty{}</code> or <code>\sPmqty{}</code>	
	<code>\smqty*{}</code> or <code>\sPmqty*{}</code>	
	<code>\smqty\blacksquare</code> or <code>\sbmqty{}</code>	
	<code>\smqty  </code> or <code>\svmqty{}</code>	
<code>\matrixdeterminant</code>	<code>\mdet{a &amp; b \\ c &amp; d}</code>	$\rightarrow \begin{vmatrix} a & b \\ c & d \end{vmatrix}$
	<code>\smdet{a &amp; b \\ c &amp; d}</code>	$\rightarrow \begin{vmatrix} a & b \\ c & d \end{vmatrix}$
<code>\identitymatrix</code>	<code>\imat{n}</code>	
	<code>\smqty(\imat{3})</code>	$\rightarrow \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

groups a set of matrix elements into a single object

parentheses

alternate parentheses

square brackets

vertical bars

alternative syntax; robust and more L<sup>A</sup>T<sub>E</sub>X-friendly

the `smallmatrix` form of `\mqty`  
small version of `\mqty{}`  
small version of `\mqty*{}`  
small version of `\mqty\blacksquare`  
small version of `\mqty||`

matrix determinant

small matrix determinant  
elements of  $n \times n$  identity matrix

formatted with `\mqty` or `\smqty`



Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

`\xmatrix`

`\xmat{x}{n}{m}`

`\smqty(\xmat{1}{2}{3})`  $\rightarrow \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$   
`\smqty(\xmat*{a}{3}{3})`  $\rightarrow \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$   
`\smqty(\xmat*{a}{3}{1})`  $\rightarrow \begin{pmatrix} a_2 \\ a_3 \end{pmatrix}$   
`\smqty(\xmat*{a}{1}{3})`  $\rightarrow \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}$

`\zeromatrix`

`\zmat{n}{m}`

`\smqty(\zmat{2}{2})`  $\rightarrow \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

`\paulimatrix`

`\pmat{n}`

`\smqty(\pmat{0})`  $\rightarrow \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$   
`\smqty(\pmat{1})`  $\rightarrow \begin{pmatrix} 1 & 0 \\ 0 & -i \end{pmatrix}$   
`\smqty(\pmat{2})`  $\rightarrow \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$   
`\smqty(\pmat{3})`  $\rightarrow \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

`\diagonalmatrix`

`\dmat{a,b,c,...}`

`\mqty(\dmat{1,2,3})`  $\rightarrow \begin{pmatrix} 1 & & \\ & 2 & \\ & & 3 \end{pmatrix}$

`\mqty(\dmat[0]{1,2})`  $\rightarrow \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix}$

`\mqty(\dmat{1,2&3\\4&5})`  $\rightarrow \begin{pmatrix} 1 & & \\ & 2 & 3 \\ & 4 & 5 \end{pmatrix}$

`\antidiagonalmatrix` `\admat{a,b,c,...}`

`\mqty(\admat{1,2,3})`  $\rightarrow \begin{pmatrix} & & 1 \\ & 2 & \\ 3 & & \end{pmatrix}$

elements of  $n \times m$  matrix filled with  $x$

formatted with `\mqty` or `\smqty`

star for element indices

as a vector with indices

$n \times m$  matrix filled with zeros  
equivalent to `\xmat{0}{n}{m}`

$n^{\text{th}}$  Pauli matrix

$n \in \{0, 1, 2, 3 \text{ or } x, y, z\}$

specify up to eight diagonal or  
block diagonal elements

optional argument to fill spaces

enter matrix elements for each  
block as a single diagonal ele-  
ment

same as syntax as `\dmat`

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The **physics** Package

The **systeme** Package

The **gauss** Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

## 8 Useful Maths Packages

- Common Packages
- The **physics** Package
- The **systeme** Package
- The **gauss** Package

# The `systeme` Package

To use the `systeme` package, simply insert the command in the preamble of your document.

## Command

```
1 \usepackage{systeme}
```

This package can make it really easy when typesetting linear systems by the command `\systeme`.

## Example

```
1 \begin{equation}
2   \systeme{
3     2a-3b+4c=2,
4     a+8b+5c=8,
5     -a+2b+c=-5
6   }
7 \end{equation}
```

$$\begin{cases} 2a - 3b + 4c = 2 \\ a + 8b + 5c = 8 \\ -a + 2b + c = -5 \end{cases} \quad (24)$$

It also works for subscripts.

### Example

```
1 \begin{equation}
2   \systeme{
3     4x_1-x_2=3,
4     -x_1+5x_2=-1
5   }
6 \end{equation}
```

$$\begin{cases} 4x_1 - x_2 = 3 \\ -x_1 + 5x_2 = -1 \end{cases} \quad (25)$$

It can also reorder the variables and numbers in the equations.

### Example

```
1 \begin{equation}
2   \systeme{
3     3y+2x=0,
4     x-z+9=0,
5     2+3x+5-y-7+z=0
6   }
7 \end{equation}
```

$$\begin{cases} 2x + 3y & & = 0 \\ x & - z + & 9 = 0 \\ 3x - & y + z + 2 + 5 - 7 = 0 \end{cases} \quad (26)$$

Complicated coefficients can be handle correctly. Note that + and - should be replaced with `\+` and `\-` in the coefficients.

### Example

```

1 \begin{equation}
2   \systeme{
3     (2\+\sqrt{2})x-
4     (1\-\sqrt{2})y=1,
5     x+(1\+\sqrt{2})y=-1
6   }
7 \end{equation}

```

$$\begin{cases} (2 + \sqrt{2})x - (1 - \sqrt{2})y = 1 \\ x + (1 + \sqrt{2})y = -1 \end{cases} \quad (27)$$

The documentation of the `systeme` package can be found in [http://mirrors.ctan.org/macros/generic/systeme/systeme\\_fr.pdf](http://mirrors.ctan.org/macros/generic/systeme/systeme_fr.pdf), however it's in French, and the author is Manuel de l'utilisateur.

### Use Maths in L<sup>A</sup>T<sub>E</sub>X

Math Expressions

Math Environments

Spacing in Math Mode

Basic Math Commands

Matrices and Arrays

### Useful Maths Packages

Common Packages

The `physics` Package

The `systeme` Package

The `gauss` Package

## 7 Use Maths in L<sup>A</sup>T<sub>E</sub>X

## 8 Useful Maths Packages

- Common Packages
- The `physics` Package
- The `systeme` Package
- The `gauss` Package

# The `gauss` Package

To use the `gauss` package, simply insert the command in the preamble of your document.

## Command

```
1 \usepackage{gauss}
```

This package provides L<sup>A</sup>T<sub>E</sub>X-macros for typesetting operations on a matrix. By an “operation on a matrix” we understand a row operation or a column operation. It is named `gauss` because **Gauss Elimination** is a widely used application of matrix operations.

The documentation of the `systeme` package can be found in <http://mirrors.ctan.org/macros/latex/contrib/gauss/gauss-doc.pdf>.

For example, if you are taking VV285 or working with other linear algebra stuffs in L<sup>A</sup>T<sub>E</sub>X, you may use the `gmatrix` environment provided by the `gauss` package.

### Example

```
1 \begin{equation}
2   \begin{array}{ccc|}
3     4 & 2 & -2 & \\
4     -3 & 1 & 0 & \\
5     1 & 4 & 2 & 
6   \end{array}
7   \begin{gmatrix}
8     -2 & 6 & -9
9   \end{gmatrix}
10  \begin{rowops}
11    \swap{0}{2}
12    \add[*]{3}{0}{1}
13  \end{rowops}
\end{equation}
```

$$\begin{array}{ccc|c} 4 & 2 & -2 & -2 \\ -3 & 1 & 0 & 6 \\ 1 & 4 & 2 & -9 \end{array} \begin{array}{l} \leftarrow \\ \leftarrow \\ \leftarrow \end{array} \begin{array}{l} \leftarrow \\ \leftarrow \\ \leftarrow \end{array} \begin{array}{l} * (3) \\ + \\ \end{array} \quad (28)$$



## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

# Lecture IV

## Graphs, Tables and Code

### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

## 9 Graphs

- Include Graphs
  - Figures
  - Draw Graphs

## 10 Tables

## 11 Code

## Graphs

### Include Graphs

#### Figures

#### Draw Graphs

## Tables

### Tabulars

### Tables

## Code

### Pseudo Code

### Code Listing

# Include Graphs

Before all, you need the `graphics` or `graphicx` package, where `graphicx` is an extended and enhanced one. So you are recommended to insert the command in the preamble of your document.

## Command

```
\usepackage{graphicx}
```

Then you can use the command `\includegraphics` to insert images of many formats, including `jpg`, `png` images and even other `pdf` files. `eps` images should be supported by most modern L<sup>A</sup>T<sub>E</sub>X distributions as well.

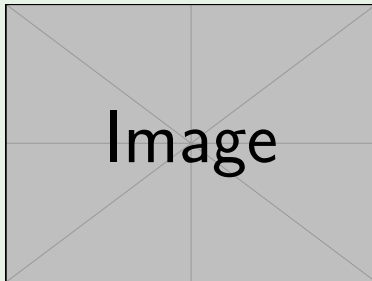
## Command

```
\includegraphics[options]{filename}
```

There are some example images defined, you can insert them if the figure is not yet ready when writing L<sup>A</sup>T<sub>E</sub>X code. They are `example-image`, `example-image-golden`, `example-image-a`, `example-image-b` and etc.

## Example

```
1 \includegraphics[width=0.4\textwidth]{example-image}
```



We usually use the `width` option to adjust the size of the image, according to a ratio of `\textwidth`, which means the maximum width of text here.

# Options of Include Graphs

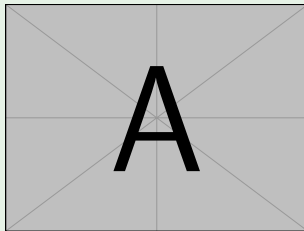
Here some useful **options** are listed:

- **height** - use any L<sup>A</sup>T<sub>E</sub>X measuring unit.
- **width** - use any L<sup>A</sup>T<sub>E</sub>X measuring unit.
- **scale** - scale the graph to this proportion
- **angle** - rotate the graph in anti-clockwise by this angle

L<sup>A</sup>T<sub>E</sub>X measuring unit can be `\textwidth`, `\linewidth`, `\textheight`, `\lineheight`, cm, pt, em, and etc..

## Example

```
1 \includegraphics[width=4cm] %  
2 {example-image-a}
```



### Graphs

Include Graphs

**Figures**

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

- 9 Graphs
  - Include Graphs
  - **Figures**
  - Draw Graphs

### 10 Tables

### 11 Code

# The `figure` Environment

The `figure` environment provides a wrapper of image inserted by `\includegraphics`, which add caption and label (reference) to an image. They are especially useful in report and paper writing, here is a template of how to use the environment.

## Command

```
1 \begin{figure}[position]
2   \centering
3   \includegraphics[options]{filename}
4   \caption{caption}
5   \label{fig:label}
6 \end{figure}
```

- `filename` - the filename or relative path of the graph you want to insert, usually placed in the same or child directory as the tex file
- `position` - we usually use `!htbp` or `!H` here, which will be introduced later in this chapter
- `caption` - the caption displayed above/under the graph
- `label` - used for references in a document (will be introduced later)

# Labels and References

You can use `\ref` to have a reference of a figure by its label. The figures will be automatically numbered (like equations), and the reference is also a hyperlink.

## Example

```
1 \begin{figure}[!htbp]
2   \centering
3   \includegraphics[
4     width=0.8\textwidth,
5     angle=90
6   ]{example-image-b}
7   \caption{Example Image B rotated by 90
8     ↪ degree.}
9   \label{fig:img-b}
10 \end{figure}
11 B was shown in Figure
12 \ref{fig:img-b}.
```

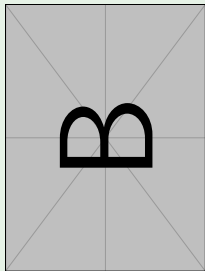


Figure 2: Example Image B rotated by 90 degree.

B was shown in Figure 2.



# Floats and Positions

Floats are containers for things in a document that cannot be broken over a page. L<sup>A</sup>T<sub>E</sub>X by default recognizes `figure` and `table` (will be introduced later) floats.

If you don't provide the `position` option, L<sup>A</sup>T<sub>E</sub>X will try to help you find a place to set the figure. However, the position is often not ideal, so you need to add some specifiers yourselves.

- `h` - Place the float `here`, i.e., approximately at the same point it occurs in the source text (however, not exactly at the spot)
- `t` - Position at the `top` of the page.
- `b` - Position at the `bottom` of the page.
- `p` - Put on a special `page` for floats only.
- `!` - Override internal parameters L<sup>A</sup>T<sub>E</sub>X uses for determining “good” float positions.
- `H` - Places the float at precisely the location in the L<sup>A</sup>T<sub>E</sub>X code. Requires the float package, i.e., `\usepackage{float}`.

# Include Multiple Graphs

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

A useful extension is the `subcaption` package, which provides a `subfigure` environment to add multiple subfigures in a figure.

Note that there is also a package called `subfigure`, but it has been deprecated (not maintained), please do not use it. Another package called `subfig` provides the same commands as that of `subfigure` package. However, they can't be used together.

In simplicity, if there is some compatibility problem with your template after you tried the `subcaption` package, choose the `subfig` package.

Here is an example with the `subcaption` package.

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

## Example

```
1 \begin{figure}
2   \centering
3   \begin{subfigure}{0.3\textwidth}
4     \includegraphics[width=\textwidth]{example-image-a}
5     \caption{Example Image A.}
6     \label{fig:subcaption-a}
7   \end{subfigure}
8   ~
9   \begin{subfigure}{0.3\textwidth}
10    \includegraphics[width=\textwidth]{example-image-b}
11    \caption{Example Image B.}
12    \label{fig:subcaption-b}
13  \end{subfigure}
14
15  \begin{subfigure}{0.3\textwidth}
16    \includegraphics[width=\textwidth]{example-image-c}
17    \caption{Example Image C.}
18    \label{fig:subcaption-c}
19  \end{subfigure}
20  \caption{Example Images}\label{fig:subcaption}
21 \end{figure}
```

### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

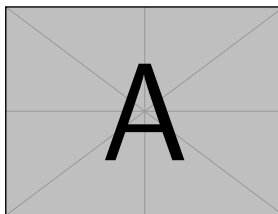
Tabulars

Tables

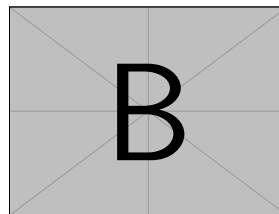
### Code

Pseudo Code

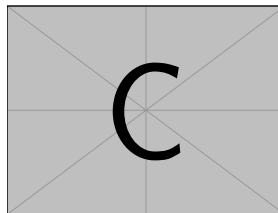
Code Listing



(a) Example Image A.



(b) Example Image B.



(c) Example Image C.

Figure 3: Example Images

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

As shown in Figure 3, the figures can be arranged in columns and rows.

Between Figure 3a and Figure 3b, a `~` was added. You can add desired spacing between images, e. g. `~`, `\quad`, `\qquad`, `\hfill` (fill all rest horizontal spaces) and etc..

Between Figure 3b and Figure 3c, a newline was added. It will force the subfigure onto a new line.

The references of subfigures can be used by their `\label` as well. For example, above references are generated by these commands:

## Example

```
1 \ref{fig:subcaption}  
2 \ref{fig:subcaption-a}  
3 \ref{fig:subcaption-b}  
4 \ref{fig:subcaption-c}
```

### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

- 9 Graphs
  - Include Graphs
  - Figures
  - Draw Graphs

### 10 Tables

### 11 Code

# The tikz and pgf packages

The `tikz` and `pgf` packages can help you draw graphs in L<sup>A</sup>T<sub>E</sub>X for example:

## Example

```

1 \begin{tikzpicture}[scale=2, bend angle=22.5]
2 \tikzstyle{every node}=[draw,shape=circle];
3 \foreach \i in {1,...,8}
4 {
5 \path (45*\i-45:1cm) node (v\i) {$v\_i$};
6 }
7 \draw
8 (v1) -- (v2) (v3) -- (v4) (v5) -- (v6) (v7) -- (v8)
9 (v1) -- (v3) (v3) -- (v5) (v5) -- (v7) (v7) -- (v1)
10 (v2) -- (v5) (v4) -- (v7) (v6) -- (v1) (v8) -- (v3)
11 (v1) -- (v5) (v3) -- (v7);
12 \end{tikzpicture}

```

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

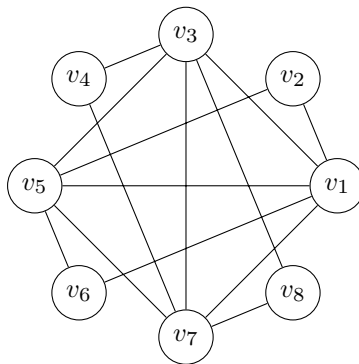
Tables

## Code

Pseudo Code

Code Listing

This will generate a simple graph which consists of eight nodes:



There may be a lecture about [tikz](#) and [pgf](#) in the future. If you are now interested in it, please refer to the [pgf manual](#) by `texdoc tikz` or `texdoc pgf`.



# Another example:

## Example

```

1 \begin{tikzpicture}[scale=0.8]
2 \tikzstyle{every node}=[draw,shape=circle,minimum size=0.8cm];
3 \node {17}[sibling distance=4cm]
4   child { node {17}[sibling distance=2cm]
5     child {
6       node {17}[sibling distance=1cm]
7       child { node {17} }
8       child { node {4} }
9     }
10    child {
11      node {5}[sibling distance=1cm]
12      child { node {1} }
13      child { node {5} }
14    }
15  }
16  child { node {14}[sibling distance=2cm]
17    child {
18      node {13}[sibling distance=1cm]
19      child { node {13} }
20      child { node {10} }

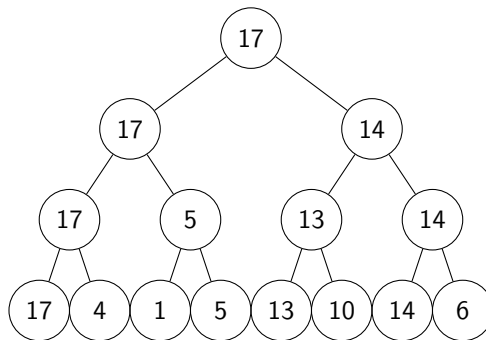
```

```

21     }
22     child {
23         node {14}[sibling distance=1cm]
24         child { node {14} }
25         child { node {6} }
26     }
27 };
28 \end{tikzpicture}

```

This will generate a binary tree:



### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

## 9 Graphs

## 10 Tables

### • Tabulars

### • Tables

## 11 Code

# The `tabular` Environment

Table is another common element in L<sup>A</sup>T<sub>E</sub>X, usually you will need the `array` package for enhanced functions of tables. You can insert the command in the preamble of your document.

## Command

```
\usepackage{array}
```

## Example

```
1 \begin{tabular}{|l|c|r|}
2 \hline
3 Title 1 & Title 2 & Title 3 \\
4 \hline
5 1 & 2 & 3 \\
6 \hline
7 \end{tabular}
```

Title 1	Title 2	Title 3
1	2	3

The syntax is similar to the `align` environment in maths. `&` is used to split the columns are `\\` is used to split the rows.

# Column Format

## Command

```
1 \begin{tabular}{format}  
2 ...  
3 \end{tabular}
```

`format` can be set as follow:

- `|` - represents a vertical separate line between two columns
- `l` - align left in this column
- `c` - align center in this column
- `r` - align right in this column

## Example

`|l|l|l|`

Title 1	Title 2	Title 3
1	2	3

`||c|cc||`

Title 1	Title 2	Title 3
1	2	3

## Graphs

Include Graphs  
Figures  
Draw Graphs

## Tables

Tabulars  
Tables

## Code

Pseudo Code  
Code Listing

With the help of the `array` package, more formats are available:

- `p{width}` - Equivalent to `\parbox[t]{width}`, vertically aligned **bottom**
- `b{width}` - Equivalent to `\parbox[b]{width}`, vertically aligned **top**
- `m{width}` - Equivalent to `\parbox{width}`, vertically aligned middle
- `>{decl.}` - Can be used before a letter option, inserts `decl` before the column.
- `<{decl.}` - Can be used after a letter option, inserts `decl` after the column.

`t` and `b` may be very confusing, but that's how they work in `\parbox`. With these new formats, the columns can be defined more flexibly.

## Example

```
1 \begin{tabular}  
2 { |p{1.2cm}|b{1.2cm}|m{1.2cm}| }  
3 \hline  
4 Aligned Bottom & Aligned Top &  
5 Aligned Middle \\  
6 \hline  
7 1 & 2 & 3 \\  
8 \hline  
9 \end{tabular}
```

Aligned Bottom	Aligned Top	Aligned Middle
1	2	3

`t`, `b` and `m` only affect the vertical alignment. If you want to control the width and make the text horizontally centered as well, you can use `>\centering` to insert a `\centering` before the text in that column. You can also insert `>{ $ }` and `<{ $ }` to generate a column in math mode.

## Example

```
1 \begin{tabular}{|>\centering m{2cm}|>{ $ } b{2cm}<{ $ }|}  
2 \hline  
3 Row of Text &  
4 \text{Row of Maths} \\\br/>5 \hline  
6 First & x \\\br/>7 Second & x^2 \\\br/>8 \hline  
9 \end{tabular}
```

Row of Text	Row of Maths
First	$x$
Second	$x^2$

If a column type will be used many times, and also very long, you can define a new column type by yourselves. You can use

## Command

```
\newcolumntype{new type}{>{some declarations}{old type}<{some more declarations}}
```

If you want to repeat a format for multiple times, you can use `*{num}{format}`. Here's an example of the usage of `\newcolumnntype` with multiple columns form.

Example

```
1 \newcolumnntype{C}{>{$}c<{$}}
2 \newcolumnntype{L}{>{$}l<{$}}
3 \newcolumnntype{R}{>{$}r<{$}}
4
5 \begin{tabular}{|L| *{2}{C|} R|}
6 \hline
7 \text{First} & \text{Second} &
8 \text{Second} & \text{Third} \\
9 \hline
10 x & x^2 & x^2 & x^3 \\
11 \hline
12 y & y^2 & y^2 & y^3 \\
13 \hline
14 \end{tabular}
```

First	Second	Second	Third
$x$	$x^2$	$x^2$	$x^3$
$y$	$y^2$	$y^2$	$y^3$



# Horizontal Lines

We usually need horizontal lines in tables. As shown in the examples above, you can add a `\hline` at the beginning of a row.

If you only want to draw a partial line, use `\cline[start-end]`.

## Example

```
1 \begin{tabular}{c|l|c|r}
2 \hline\hline
3 & Title 1 & Title 2 & Title 3 \\\
4 \cline{2-4}
5 Table & 1 & 2 & 3 \\\
6 \cline{2-4}
7 & 4 & 5 & 6 \\\
8 \hline\hline
9 \end{tabular}
```

Table	Title 1	Title 2	Title 3
	1	2	3
	4	5	6

Here we draw a table with a multirow, but it only works with multirows of odd row number. A more convenient method of drawing multirows will be introduced.

# Combining Rows and Columns

There are two commands being used to combine rows and columns

## Command

```
\multicolumn{ncols}{format}{text}
```

- `ncols` - the number of columns to be merged
- `format` - the format of the merged column, excluding the left | (eg. `c|`)
- `text` - the text in the merged column

```
\multirow{nrows}{width}[fixup]{text}
```

- `nrows` - the number of rows to be merged
- `width` - the width of the merged rows (use `*` for auto)
- `fixup` - the vertical position of the text (optional, default in the center)
- `text` - the text in the merged row

To use the `\multirow` command, you need to insert the package `multirow` in the preamble of your document.

## Example

```
1 \centering
2 \begin{tabular}{|c|c|c|c|c|}
3 \hline
4 \multirow{4}{*}{Table} & Title 1 & Title 2 & Title 3 & Title 4 \\
5 \cline{2-5}
6 & \multicolumn{2}{c|}{Text 1} & & 
7 \multicolumn{2}{c|}{\multirow{3}{*}{Text 3}} \\
8 \cline{2-3}
9 & \multicolumn{2}{c|}{Text 2} & \multicolumn{2}{c|}{} \\
10 \cline{2-3}
11 & Text 4 & Text 5 & \multicolumn{2}{c|}{} \\
12 \hline
13 \end{tabular}
```

Table	Title 1	Title 2	Title 3	Title 4
	Text 1		Text 3	
	Text 2			
	Text 4	Text 5		

Just leave blank in the rest rows of `\multirow`.

# Coloring Rows and Columns

The rows and columns can also be colored, with the `colortbl` package. You may also need the `xcolor` package to define new colors.

## Command

```
1 \usepackage{xcolor}
2 \usepackage{colortbl}
```

Some commands are provided by these packages

## Command

```
1 \definecolor{name}{system}{definition}
2 \rowcolor{color}
3 \columncolor{color}
```

Here `system` can be `rgb/hsb/cmyk/gray`. Please refer the lecture about defining colors.

# Example

```
1 \definecolor{mygray}{gray}{.9}
2 \definecolor{barblue}{RGB}{153,204,254}
3 \centering
4 \begin{tabular}{|*2{>{\columncolor{mygray}}c|c|}{}}
5 \hline\rowcolor{barblue}
6 Title 1 & Title 2 & Title 3 & Title 4 \\
7 \hline
8 Text 1 & Text 2 & Text 3 & Text 4 \\
9 \hline
10 Text 5 & Text 6 & Text 7 & Text 8 \\
11 \hline
12 \end{tabular}
```

Title 1	Title 2	Title 3	Title 4
Text 1	Text 2	Text 3	Text 4
Text 5	Text 6	Text 7	Text 8

Note that the `\rowcolor` overwrites the `\columncolor`.

# Styling Rows and Columns

Styling columns can be easily achieved by prepending styles in the `>\decl.` introduced before.

However, styling rows is much more complicated. You should only consider using this method when your table is really too large to style them one by one.

First, you may define these in the preamble of your document:

## Command

```
1 \newcolumntype{+}{>\global\let\currentrowstyle\relax}}
2 \newcolumntype{^}{>\currentrowstyle}}
3 \newcommand{\rowstyle}[1]{\gdef\currentrowstyle{#1}%
4   #1\ignorespaces
5 }
```

Then you should add a `+` before the first column definition and a `^` before any other column definitions. (You can change the symbols `+` and `^` in the definition above.)

Graphs

Include Graphs  
Figures  
Draw Graphs

Tables

Tabulars  
Tables

Code

Pseudo Code  
Code Listing

Example

```
1 \newcolumntype{+}{>{\global\let\currentrowstyle\relax}}
2 \newcolumntype{^}{>{\currentrowstyle}}
3 \newcommand{\rowstyle}[1]{\gdef\currentrowstyle{#1}%
4   #1\ignorespaces
5 }
6 \centering
7 \begin{tabular}{|+>{\ttfamily}c|^c|^>{\ttfamily}c|^c|}
8   \hline\rowstyle{\bfseries\sffamily}
9   Title 1 & Title 2 & Title 3 & Title 4 \\
10  \hline
11  Text 1 & Text 2 & Text 3 & Text 4 \\
12  \hline
13  Text 5 & Text 6 & Text 7 & Text 8 \\
14  \hline
15 \end{tabular}
```

Title 1	Title 2	Title 3	Title 4
Text 1	Text 2	Text 3	Text 4
Text 5	Text 6	Text 7	Text 8

Note that the `\rowstyle` also overwrites the column style set in the formats.

# Table Generators

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

With `\multirow` and `\multicolumn`, we can almost draw tables of any style, but this coding process can never be as easy as the graphic one, like making tables in Word or Excel. Is there any ways to convert graphic tables into L<sup>A</sup>T<sub>E</sub>X codes directly?

- Use L<sup>A</sup>T<sub>E</sub>X Table Generator: <http://www.tablesgenerator.com/>
- L<sup>A</sup>T<sub>E</sub>X Complex Table Editor: <https://www.latex-tables.com/>
- Excel2latex: <https://ctan.org/tex-archive/support/excel2latex/>



### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

## 9 Graphs

## 10 Tables

- Tabulars

- **Tables**

## 11 Code

# The `table` Environment

The `table` environment is used to arrange the place of a tabular, similar to the `figure` environment. Here is a template of how to use the environment.

## Command

```
1 \begin{table}[position]
2   \centering
3   \begin{tabular}{format}
4     ...
5   \end{tabular}
6   \caption{caption}
7   \label{table:label}
8 \end{table}
```

The `position`, `caption`, `label` are same as those in the `figure` environment.

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

# Recall the Positions

We usually want to place the graphs or tables just below or above the content where we mention them, but even when we type `[h]` in position, you can not ensure that it will appear at the ideal position, and there are several methods to make up for this. You can try them one by one:

- 1 Change `[h]` to `[!h]`
- 2 Change `[!h]` to `[!H]`
- 3 Use `\newpage` to move the following content to the next page

Usually you don't need to pay too much attention about where the figures and tables are exactly are because you can use `\ref` to reference them. And the numbering of figures and tables will strictly follow the order of their code.

## figure and table in Two-column Documents

If you are writing a document using two columns (i.e. you started your document with something like `\documentclass[twocolumn]{article}`), you might have noticed that you can't use floating elements that are wider than the width of a column (using a L<sup>A</sup>T<sub>E</sub>X notation, wider than `0.5\textwidth`), otherwise you will see the figure or table overlapping with text.

If you really have to use such wide elements, the only solution is to use the “starred” variants of the floating environments:

### Command

```
1 \begin{figure*}[position]
2   ...
3 \end{figure*}

1 \begin{table*}[position]
2   ...
3 \end{table*}
```

Those “starred” versions work like the standard ones, but they will be as wide as the page, so you will get no overlapping.

# The `array` Environment

When you use `tabular` in maths environment, the text format in the `tabular` won't be italic. However, there is a replacement of `tabular`, which is the `array` environment.

## Command

```
1 \begin{array}{format}  
2 ...  
3 \end{array}
```

The options and usages of these two environment are exactly the same.

Though the environment is not provided by the `array` package (it's built-in one), you are also recommended to use this package for enhancements.

# List of Floats

The figures and tables are all called floats. Captions can be listed at the beginning of a paper or report in a “List of Figures” or a “List of Tables” section with the commands:

## Command

- 1 `\listoffigures`
- 2 `\listoftables`

The caption used for each figure will appear in these lists, along with the figure numbers, and page numbers that they appear on.

The `\caption` command also has an optional parameter, which is used for the List of Tables or List of Figures.

## Command

```
\caption[short]{long}
```

Typically the `short` description is for the caption listing, and the `long` description will be placed beside the figure or table.

### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

9 Graphs

10 Tables

11 Code

- Pseudo Code
- Code Listing

# The `algorithm` Environment

L<sup>A</sup>T<sub>E</sub>X has several packages for typesetting algorithms in form of “pseudocode”. They provide stylistic enhancements over a uniform style (i.e., all in typewriter font) so that constructs such as loops or conditionals are visually separated from other text. The pseudocode is usually put in an `algorithm` environment. Include it by adding the command to your document’s preamble.

## Command

```
\usepackage{algorithm}
```

Then you can use the `algorithm` environment, which acts similar as the `figure` and `table` environments.

## Command

```
1 \begin{algorithm}[position]
2   \caption{caption}
3   \label{algorithm:label}
4   <the actual pseudocode environment>
5 \end{algorithm}
```



# The `algorithmic` Package

One of the packages, the `algorithmic`, defines the `algorithmic` environment. Include it by adding the command to your document's preamble.

## Command

```
\usepackage{algorithmic}
```

The basic commands are:

## Command

```
1  \STATE <text>
2  \IF{<condition>} \STATE {<text>} \ELSE \STATE{<text>} \ENDIF
3  \IF{<condition>} \STATE {<text>} \ELSIF{<condition>} \STATE{<text>} \ENDIF
4  \FOR{<condition>} \STATE {<text>} \ENDFOR
5  \FOR{<condition>} \TO <condition> } \STATE {<text>} \ENDFOR
6  \FORALL{<condition>} \STATE{<text>} \ENDFOR
7  \WHILE{<condition>} \STATE{<text>} \ENDWHILE
8  \REPEAT \STATE{<text>} \UNTIL{<condition>}
9  \LOOP \STATE{<text>} \ENDLOOP
10 \REQUIRE <text>, \ENSURE <text>, \RETURN <text>, \PRINT <text>
11 \AND, \OR, \XOR, \NOT, \TO, \TRUE, \FALSE, \COMMENT{<text>}
```

## Example

```

1 \begin{algorithm}[H]
2   \caption{Calculate  $y = x^n$ }
3   \label{algorithm:n-square}
4   \begin{algorithmic}
5     \REQUIRE  $n \geq 0 \wedge x \neq 0$ 
6     \ENSURE  $y = x^n$ 
7     \STATE  $y \leftarrow 1$ 
8     \IF{ $n < 0$ }
9       \STATE  $X \leftarrow 1 / x$ 
10      \STATE  $N \leftarrow -n$ 
11    \ELSE
12      \STATE  $X \leftarrow x$ 
13      \STATE  $N \leftarrow n$ 
14    \ENDIF
15    \WHILE{ $N \neq 0$ }
16      \IF{ $N$  is even}
17        \STATE  $X \leftarrow X \times X$ 
18        \STATE  $N \leftarrow N / 2$ 
19      \ELSE[ $N$  is odd]
20        \STATE  $y \leftarrow y \times X$ 
21        \STATE  $N \leftarrow N - 1$ 
22      \ENDIF
23    \ENDWHILE
24  \end{algorithmic}
25 \end{algorithm}

```

---

**Algorithm 1** Calculate  $y = x^n$ 

---

**Require:**  $n \geq 0 \vee x \neq 0$ **Ensure:**  $y = x^n$  $y \leftarrow 1$ **if**  $n < 0$  **then** $X \leftarrow 1/x$  $N \leftarrow -n$ **else** $X \leftarrow x$  $N \leftarrow n$ **end if****while**  $N \neq 0$  **do****if**  $N$  is even **then** $X \leftarrow X \times X$  $N \leftarrow N/2$ **else**[ $N$  is odd] $y \leftarrow y \times X$  $N \leftarrow N - 1$ **end if****end while**

---

# The `algorithmicx` Package

Another package `algorithmicx` provides more functionalities, but it is not compatible with the `algorithmic` package. Include it by adding the command to your document's preamble.

## Command

```
\usepackage{algpseudocode}
```

Note that `\usepackage{algorithmicx}` only defines some common macros and it is not enough. Don't insert `\usepackage{algorithmic}` in this situation.

The main difference of these two packages is that all of the command name are changed, so that only the first letter in a word is capital. For example, `\STATE` is changed to `\State` and `\ENDFOR` is changed to `\EndFor`.

The command `\begin{algorithmic}` can be given the optional argument of a positive integer, which if given will cause line numbering to occur at multiples of that integer. E.g. `\begin{algorithmic}[5]` will enter the `algorithmic` environment and number every fifth line.

## Example

```
1 \begin{algorithm}[H]
2   \caption{Euclids algorithm}
3   \label{algorithm:euclid}
4   \begin{algorithmic}[1]
5     \Procedure{Euclid}{$a,b$}\Comment{The g.c.d. of a and b}
6       \State $r\gets a\bmod b$
7       \While{$r\neq 0$}\Comment{We have the answer if r is 0}
8         \State $a\gets b$
9         \State $b\gets r$
10        \State $r\gets a\bmod b$
11      \EndWhile\label{euclidendwhile}
12      \State \textbf{return} $b$\Comment{The gcd is b}
13    \EndProcedure
14  \end{algorithmic}
15 \end{algorithm}
```

## Algorithm 2 Euclids algorithm

```

1: procedure EUCLID( $a, b$ )                                ▷ The g.c.d. of  $a$  and  $b$ 
2:    $r \leftarrow a \bmod b$ 
3:   while  $r \neq 0$  do                                     ▷ We have the answer if  $r$  is 0
4:      $a \leftarrow b$ 
5:      $b \leftarrow r$ 
6:      $r \leftarrow a \bmod b$ 
7:   end while
8:   return  $b$                                              ▷ The gcd is  $b$ 
9: end procedure

```

Algorithms can also be listed like figures and tables, by the command:

Command

`\listofalgorithms`

### Graphs

Include Graphs

Figures

Draw Graphs

### Tables

Tabulars

Tables

### Code

Pseudo Code

Code Listing

9 Graphs

10 Tables

11 Code

- Pseudo Code

- Code Listing

# The `verbatim` Environment

The default tool to display code in L<sup>A</sup>T<sub>E</sub>X is `verbatim`, which generates an output in monospaced font.

## Example

```
1 \begin{verbatim}
2 Text enclosed inside \texttt{verbatim} environment
3 is printed directly
4 and all \LaTeX{} commands are ignored.
5 \end{verbatim}
```

Text enclosed inside `\texttt{verbatim}` environment  
is printed directly  
and all `\LaTeX{}` commands are ignored.

There's a starred version (`verbatim*`) whose output is slightly different.

Text enclosed inside `\texttt{verbatim}` environment  
is printed directly  
and all `\LaTeX{}` commands are ignored.



# The `\verb` Command

Verbatim-like text can also be used inline with the command `\verb`

## Example

```
1 In the directory \verb|C:\Windows\system32| you can find a lot of Windows
2 system applications.
3
4 The \verb+\ldots+ command produces \ldots
```

In the directory C:\Windows\system32 you can find a lot of Windows system applications.

The `\ldots` command produces ...

The command `\verb|C:\Windows\system32|` prints the text inside the delimiters `|` in verbatim format. Any character, except letters and `*`, can be used as delimiter. For instance `\verb+\ldots+` uses `+` as delimiter.

# The listings Package

A better form of code listing can be done by the `listings` package. To use it, simply insert the command in the preamble of your document.

## Command

```
\usepackage{listings}
```

It provides a `lstlisting` environment.

## Command

```
1 \begin{lstlisting}[language=name]
2   ...
3 \end{lstlisting}
```

You can also input source code from file.

## Command

```
\lstinputlisting[language=name]{filename}
```

## Example

```
1 \begin{lstlisting}[language=Python]
2 import numpy as np
3
4 def incmatrix(genl1,genl2):
5     m = len(genl1)
6     n = len(genl2)
7 \end{lstlisting}
```

```
import numpy as np
```

```
def incmatrix(genl1 , genl2 ):
    m = len(genl1)
    n = len(genl2)
```

You can add code coloring and styling by some complicated configurations, see the Overleaf tutorial [▶ Link](#).

The documentation of the `listings` package can be found in [▶ Link](#).

# The `minted` Package

All of the code in this lecture are highlighted by the `minted` package. To use it, simply insert the command in the preamble of your document.

## Command

```
\usepackage{minted}
```

This is a very special package, it depends a program out of L<sup>A</sup>T<sub>E</sub>X called `pygmentize`, which is a code highlighting package written in `Python`.

You can install the package through `pip` (assuming you have `Python` 2 or 3 and `pip` installed) in your terminal:

## Command

```
pip install Pygments
```

And then you can examine in your terminal whether `pygmentize` is your `PATH` by directly running it. You also need to add an option `-shell-escape` to your L<sup>A</sup>T<sub>E</sub>X compiler because L<sup>A</sup>T<sub>E</sub>X need this permission to run other programs on shell.

# The `minted` Environment

You can use the `minted` environment to insert a block of code in the specific language.

## Command

```
1 \begin{minted}[options]{language}
2   ...
3 \end{minted}
```

You can use the command in the terminal to find the supported languages.

## Command

```
pygmentize -L lexers
```

There is also a list of languages on the online document [▶ Link](#). Note that if you want to insert plain text, use the `text` language which doesn't have any highlight.

## Graphs

Include Graphs  
Figures  
Draw Graphs

## Tables

Tabulars  
Tables

## Code

Pseudo Code  
Code Listing

# The Inline `minted`

For a single line of source code, you can alternatively use a shorthand notation:

## Command

```
\mint[options]{language}|...|
```

Here we use a pair of `|`, same as the usage of the `\verb` command, which is also an inline verbatim command.

Or you can also use

## Command

```
\mintinline[options]{language}|...|
```

Here `|` can also be replaced with `{}`, a pair of `+`, etc., the key is there should not exist the same delimiter inside the code.

# Input File with `minted`

When you have a source code file alone, you can use the command to input the file.

## Command

```
\inputminted[options]{language}{filename}
```

There are some commonly used options (not only for this command):

- `linenos` - Turn on line numbers
- `breaklines` - Automatically break long lines in `minted` environment and `\mint`, and wrap longer lines in `\mintinline`.
- `fontsize` - The size of the font to use, as a size command, e.g.  
`fontsize=\footnotesize`.
- `tabsize` - The number of spaces a tab is equivalent to. (default is 8, but often set to 4)
- `firstline` - The first line to show. (default is 1, useful when showing part of a file)
- `lastline` - The last line to show. (default is the last line of the input)

# Using Different Styles

You can use various styles of highlighting scheme provided by [pygmentize](#).

## Command

```
\usemintedstyle{name}
```

You can use the command in the terminal to find the supported styles.

## Command

```
pygmentize -L styles
```

There is also a demo of languages and styles on the online demo [▶ Link](#). The [autumn](#) style is used in this lecture.

In the end, [XeLaTeX](#) might be the best L<sup>A</sup>T<sub>E</sub>X compiler working with the [minted](#) package. It also supports typesetting with Chinese, if you meet problems when using the default [pdflatex](#) compiler, switch into [XeLaTeX](#) may solve your issues.

The documentation of the [minted](#) package can be found in [▶ Link](#).



## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

# Custom Floats

If tables and figures are not adequate for your needs, then you always have the option to create your own! Examples of such instances could be source code examples, or maps. For a program float example, one might therefore wish to create a `float` named program. The package `float` is your friend for this task. All commands to set up the new float must be placed in the preamble, and not within the document.

## Command

```
1 \usepackage{float}
2 \floatstyle{style}
3 \newfloat{type}{placement}{ext}[outer counter]
4 \floatname{type}{floatname}
```

The default name that appears at the start of the caption is the type. If you wish to alter this, use `\floatname{type}{floatname}`.

For the `\floatstyle` command, `style` can be:

- `plain` - the normal style for L<sup>A</sup>T<sub>E</sub>X floats, but the caption is always below the content.
- `plaintop` - the normal style for L<sup>A</sup>T<sub>E</sub>X floats, but the caption is always above the content.
- `boxed` - a box is drawn that surrounds the float, and the caption is printed below.
- `ruled` - the caption appears above the float, with rules immediately above and below. Then the float contents, followed by a final horizontal rule.

For the `\newfloat` command,

- `type` - the new name you wish to call your float, in the example, “program”.
- `placement` - t, b, p, or h (as previously described in Placement), where letters enumerate permitted placements.
- `ext` - the file name extension of an auxiliary file for the list of figures (or whatever). L<sup>A</sup>T<sub>E</sub>X writes the captions to this file.
- `outer counter` - the presence of this parameter indicates that the counter associated with this new float should depend on outer counter, for example “chapter”.

## Graphs

Include Graphs

Figures

Draw Graphs

## Tables

Tabulars

Tables

## Code

Pseudo Code

Code Listing

## Example

```
1 \documentclass{article}
2 \usepackage{float}
3 \floatstyle{ruled}
4 \newfloat{program}{thp}{lop}
5 \floatname{program}{Program}
6
7 \begin{document}
8
9 \begin{program}[H]
10   \begin{minted}{java}
11   class HelloWorldApp {
12     public static void main(String[] args) {
13       //Display the string
14       System.out.println("Hello World!");
15     }
16   }
17   \end{minted}
18   \caption{The Hello World! program in Java.}
19   \label{program:hello-world}
20 \end{program}
21
22 \end{document}
```

---

**Program 1** The Hello World! program in Java.

---

```
1  class HelloWorldApp {  
2      public static void main(String[] args) {  
3          //Display the string  
4          System.out.println("Hello World!");  
5      }  
6  }
```

---

You can also reference the custom floats.

### Example

```
1  Program \ref{program:hello-world} is the Hello World! program in Java.
```

Program 1 is the Hello World! program in Java.

# Lecture V

## Beamer Slides

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

- Beamer Document
- Beamer Structure

## 13 Overlay and Animation

## 14 Special Structures

# Why beamer?

For L<sup>A</sup>T<sub>E</sub>X users, **beamer** has a number of advantages over PowerPoint or other presentation software: <sup>1</sup>

- If you are creating slides from a larger document, you can simply re-use your L<sup>A</sup>T<sub>E</sub>X source material from that document.
- If you need mathematical content in your slides, you have the wealth of mathematical constructs in L<sup>A</sup>T<sub>E</sub>X to draw upon.
- The slides you create are multi-platform.

**beamer** allows you to create slides featuring overlays, animation and so on in L<sup>A</sup>T<sub>E</sub>X. You simply insert some calls to **beamer** macros in your L<sup>A</sup>T<sub>E</sub>X source file, and compile it into a pdf file. You can then use a pdf viewer to present your slides.

---

<sup>1</sup><http://heather.cs.ucdavis.edu/~matloff/beamer.html>

# The beamer Class

In order to use `beamer`, you should use the following command as the first line of your tex document:

## Command

```
\documentclass[options]{beamer}
```

Then you can create frames with the `frame` environment or the `\frame` command in the `document` body.

## Example

```
1 \documentclass[options]{beamer}
2 \begin{document}
3 \begin{frame}
4   some content
5 \end{frame}
6 \frame{some content}
7 \end{document}
```



# The Title Page

You can add title, author, date and some other information in the preamble of the document, similar to the document class `article`.


## Example

```
1 \title{Introduction to \LaTeX}
2 \author{Liu Yihao}
3 \date{\today}
4 \institute{SJTU-UMJI Technology Department}
```

Then you can use the `\titlepage` command to generate a title page.

## Command

```
1 \begin{frame}
2   \titlepage
3 \end{frame}
```

This is how the  of this document is generated.

There are some more options for the title page than the ones presented. The next example is a complete one, most of the commands are optional. <sup>1</sup>

## Example

```
1 \documentclass{beamer}
2 \usetheme{Boadilla}
3 \usecolortheme{seahorse}
4
5 \title[About Beamer] %optional
6 {About the Beamer class in presentation making}
7 \subtitle{A short story}
8 \author[Arthur, Doe] % (optional, for multiple authors)
9 {A.~B.~Arthur\inst{1} \and J.~Doe\inst{2}}
10 \institute[VFU] % (optional)
11 {
12   \inst{1} Faculty of Physics \\\ Very Famous University
13   \and
14   \inst{2} Faculty of Chemistry\\ Very Famous University
15 }
16 \date[VLC 2013] % (optional)
17 {Very Large Conference, April 2013}
18 \logo{\includegraphics[height=1.5cm]{example-image}}
```

<sup>1</sup>Some of this part is ported from the tutorial of Overleaf:

The distribution of each element in the title page depends on the theme, which will be introduced later. Here is a description of each command:

- `\title[short title]{title}` - The title of your presentation must be inside braces. You can set an optional shorter title in the square brackets.
- `\subtitle{subtitle}` - Subtitle can be omitted if unnecessary.
- `\author[short author]{author}` and `\institute[short institute]{institute}` - The usages can be referred to the example code. Use the `\inst` command to state the institute of each author if needed.
- `\date[short date]{date}` - You can also use `\today` as a date.
- `\logo{logo}` - You can use text or image, it will be shown on every slide.

The short versions of title, author, institute and date are often used in the headline or footline in the presentation. If omitted, the long versions will be used there.

The complete example of title page is demonstrated on the next page.

`\frame{\titlepage}`

# Outline

- 1 Section 1
  - Subsection 1.1
    - Subsubsection 1.1.1
  - Subsection 1.2
    - Subsubsection 1.2.1
    - Subsubsection 1.2.2
- 2 Section 2
  - Subsection 2.1
    - Subsubsection 2.1.1
  - Subsection 2.2
    - Subsubsection 2.2.1
    - Subsubsection 2.2.2



# Frame Title

You may notice that some of the slides have a title (eg., “Frame Title” in this slide). You can use this command to add one:

## Command

```
1 \begin{frame}  
2   \frametitle{frame title}  
3 \end{frame}
```

Alternatively, you can add the title as an argument of the `frame` environment:

## Command

```
1 \begin{frame}{frame title}  
2  
3 \end{frame}
```

It is worth noting that in `beamer` the basic container is a `frame`. A `frame` is not exactly equivalent to a slide, one `frame` may contain more than one slides.

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

- Beamer Document
- Beamer Structure

## 13 Overlay and Animation

## 14 Special Structures

# Sections and Parts

You can also structure a `beamer` document into sections, subsections and subsubsections. Usually subsubsections are not very useful in small presentations.

## Command

```
1 \section{section}
2 \subsection{subsection}
3 \subsubsection{subsubsection}
```

For large presentations or lectures (such as this one), another structure called `\part` can be used.

## Command

```
1 \part{part}
```

The contents of different parts are often split from each other completely, eg., the counter of figures and tables, the table of contents, etc.

# Table of Contents

After dividing your presentation into sections and subsections, you can add a table of contents at the beginning of the document, or before each section, or anywhere.

## Command

```
1 \begin{frame}{Outline}
2   \tableofcontents[options]
3 \end{frame}
```

For example, if the document structure is

## Example

```
1 \section{Section 1}
2 \subsection{Subsection 1.1}
3 \subsubsection{Subsubsection 1.1.1}
4 \subsection{Subsection 1.2}
5 \subsubsection{Subsubsection 1.2.1}
6 \subsubsection{Subsubsection 1.2.2}
7 \section{Section 2}
8 \subsection{Subsection 2.1}
9 \subsubsection{Subsubsection 2.1.1}
10 \subsection{Subsection 2.2}
11 \subsubsection{Subsubsection 2.2.1}
12 \subsubsection{Subsubsection 2.2.2}
```

An example of the default table of contents is shown on the next page.



# Outline

- 1 Section 1
  - Subsection 1.1
    - Subsubsection 1.1.1
  - Subsection 1.2
    - Subsubsection 1.2.1
    - Subsubsection 1.2.2
- 2 Section 2
  - Subsection 2.1
    - Subsubsection 2.1.1
  - Subsection 2.2
    - Subsubsection 2.2.1
    - Subsubsection 2.2.2



By default, all sections, subsections and subsubsections in the current part will be shown in the table and contents. You can also use options to set whether some of the sections or subsections should be shaded, or be hided. The allowed styles are `show`, `shaded` and `hide`. The available options are

- `sectionstyle=<style for current section>/<style for other sections>`
- `subsectionstyle=<style for current subsection>/<style for other subsections in current section>/<style for subsections in other sections>`
- `subsubsectionstyle=<style for current subsubsection>/<style for other subsubsections in current subsection>/<style for subsubsections in other subsections in current section>/<style for subsubsections in other subsections in other sections>`

The later styles can be omitted in each options, in this case, the omitted styles will be set to the last explicit style. For example, these two lines are equivalent:

- `subsectionstyle=show/shaded`
- `subsectionstyle=show/shaded/shaded`

They both cause all subsections except the current subsection in the current section to be shown in a semi-transparent way.

There are also some shorthands of the options above, you can use them alone, or mix them with any other options.

- `currentsection` - `sectionstyle=show/shaded,subsectionstyle=show/show/shaded`
- `currentsubsection` - `subsectionstyle=show/shaded`
- `hideallsubsections` - `subsectionstyle=hide`
- `hideothersubsections` - `subsectionstyle=show/show/hide`

Some other options include

- `part=<part number>` - shows the table of contents of a specific part.
- `pausesections` - causes a `\pause` command to be issued before each section. This is useful if you wish to show the table of contents in an incremental way.
- `pausesubsections` - causes a `\pause` command to be issued before each subsection.

The `\pause` command can split a frame into two slides, which will be introduced in the next section.

Some examples are shown on the next pages.

```
\frame{\tableofcontents[pausesection]}
```

# Outline (pausesections)

- 1 Section 1
  - Subsection 1.1
    - Subsubsection 1.1.1
  - Subsection 1.2
    - Subsubsection 1.2.1
    - Subsubsection 1.2.2



```
\frame{\tableofcontents[pausesection]}
```

Outline (pausesections)

- 1 Section 1
  - Subsection 1.1
    - Subsubsection 1.1.1
  - Subsection 1.2
    - Subsubsection 1.2.1
    - Subsubsection 1.2.2
- 2 Section 2
  - Subsection 2.1
    - Subsubsection 2.1.1
  - Subsection 2.2
    - Subsubsection 2.2.1
    - Subsubsection 2.2.2



```
\frame{\tableofcontents[currentsection]}
```

## Outline (Now we are at subsection 1.2.1)

- ① Section 1
  - Subsection 1.1
    - Subsubsection 1.1.1
  - Subsection 1.2
    - Subsubsection 1.2.1
    - Subsubsection 1.2.2
- ② Section 2
  - Subsection 2.1
    - Subsubsection 2.1.1
  - Subsection 2.2
    - Subsubsection 2.2.1
    - Subsubsection 2.2.2



```
\frame{\tableofcontents[currentsubsection]}
```

## Outline (Now we are at subsection 2.2.1)

### 1 Section 1

- Subsection 1.1
  - Subsubsection 1.1.1
- Subsection 1.2
  - Subsubsection 1.2.1
  - Subsubsection 1.2.2

### 2 Section 2

- Subsection 2.1
  - Subsubsection 2.1.1
- Subsection 2.2
  - Subsubsection 2.2.1
  - Subsubsection 2.2.2



```
\frame{\tableofcontents[sectionstyle=show/shaded,  
subsectionstyle=show/shaded/hide,subsubsectionstyle=show/shaded/hide]}
```

## Outline (Now we are at subsubsection 2.2.2)

### 1 Section 1

### 2 Section 2

- Subsection 2.1

- Subsection 2.2

  - Subsubsection 2.2.1

  - Subsubsection 2.2.2





# Bibliography

Like the `article` class, you can use `\cite` to create citations and add a bibliography at the end of the file.

Unfortunately, `bibtex` is not perfectly supported in `beamer`, so usually you need to typeset them by hand.

## Example

```
1 \begin{frame}
2   \frametitle{For Further Reading}
3   \begin{thebibliography}{Dijkstra, 1982}
4     \bibitem[Salomaa, 1973]{Salomaa1973} A.~Salomaa.
5     \newblock {\em Formal Languages}. \newblock Academic Press, 1973.
6     \bibitem[Dijkstra, 1982]{Dijkstra1982} E.~Dijkstra.
7     \newblock Smoothsort, an alternative for sorting in situ.
8     \newblock {\em Science of Computer Programming}, 1(3):223--233, 1982.
9   \end{thebibliography}
10  \end{frame}
```

# Appendix

You can add an appendix by using the `\appendix` command. The command essentially just starts a new part named `\appendixname`. However, it also sets up certain hyperlinks.

All frames, all `\subsection` commands, and all `\section` commands used after this command will not be shown as part of the normal navigation bars.

## Example

```
1 \begin{document}
2 \frame{\titlepage}
3 \section*{Outline}
4 \frame{\tableofcontents}
5 \section{Main Text}
6 \frame{Some text}
7 \section*{Summary}
8 \frame{Summary text}
9
10
11 \appendix
12 \section{\appendixname}
13 \frame{\tableofcontents}
14 \subsection{Additional material}
15 \frame{Details}
16 \frame{Text omitted in main talk.}
17 \subsection{Even more additional
    ↳ material}
18 \frame{More details}
19 \end{document}
```

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

## 13 Overlay and Animation

- Overlay
- Animation

## 14 Special Structures

# Simple Overlay

In the introduction, it was mentioned that a frame is not equivalent to a slide.

## Introduction

Beamer Document

Beamer Structure

## Overlay and Animation

Overlay

Animation

## Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

# Simple Overlay

In the introduction, it was mentioned that a frame is not equivalent to a slide.

The simplest way to verify this is to add a simple overlay with the command

## Command

```
1 \pause
```

# Simple Overlay

In the introduction, it was mentioned that a frame is not equivalent to a slide.

The simplest way to verify this is to add a simple overlay with the command

## Command

```
1 \pause
```

A direct example is this frame itself:

## Example

```
1 \begin{frame}{Simple Overlay}  
2   some contents  
3   \pause  
4   some contents  
5   \pause  
6   some contents  
7 \end{frame}
```

Note that the page numbers in the bottom right corner of these three slides are the same, the page counter only counts the number of frames.

# Overlay Specifications

However, the `\pause` command only provide a basic support of splitting slides. Using commands with an overlay specification will be more flexible, which means you can have different effects with the same command on different slides.

Overlay specifications can only be written behind certain commands, not every command. `\textbf` is one of these commands.

## Example

```
1 \textbf{This line is bold on all slides.}
2 \textbf<2>{This line is bold only on the second slide.}
```

**This line is bold on all slides.** This line is bold only on the second slide.

The syntax of (basic) overlay specifications is the following: They are comma-separated lists of slides and ranges. Ranges are specified like this: `2-5`, which means slide two through to five.

The start or the end of a range can be omitted. For example, `3-` means “slides three, four, five, and so on”.

# Overlay Specifications

However, the `\pause` command only provide a basic support of splitting slides. Using commands with an overlay specification will be more flexible, which means you can have different effects with the same command on different slides.

Overlay specifications can only be written behind certain commands, not every command. `\textbf` is one of these commands.

## Example

```
1 \textbf{This line is bold on all slides.}
2 \textbf<2>{This line is bold only on the second slide.}
```

**This line is bold on all slides. This line is bold only on the second slide.**

The syntax of (basic) overlay specifications is the following: They are comma-separated lists of slides and ranges. Ranges are specified like this: `2-5`, which means slide two through to five.

The start or the end of a range can be omitted. For example, `3-` means “slides three, four, five, and so on”.



For the following commands, adding an overlay specification causes the command to be simply ignored on slides that are not included in the specification: `\textbf`, `\textit`, `\textmd`, `\textnormal`, `\textrm`, `\textsc`, `\textsf`, `\textsl`, `\texttt`, `\textup`, `\emph`, `\color`, `\textcolor`, `\alert`, `\structure`. If a command takes several arguments, like `\color`, the specification should directly follow the command.

## Example

```
1 \color<2>[rgb]{1,0,0} This text is red on slides 2, otherwise black.
```

This text is red on slides 2, otherwise black.

For the following commands, adding an overlay specification causes the command to be simply ignored on slides that are not included in the specification: `\textbf`, `\textit`, `\textmd`, `\textnormal`, `\textrm`, `\textsc`, `\textsf`, `\textsl`, `\texttt`, `\textup`, `\emph`, `\color`, `\textcolor`, `\alert`, `\structure`. If a command takes several arguments, like `\color`, the specification should directly follow the command.

### Example

```
1 \color<2>[rgb]{1,0,0} This text is red on slides 2, otherwise black.
```

This text is red on slides 2, otherwise black.

# The `onslide` Command

When you want to display or hide some contents on some slides, you can use

## Command

```
\onslide(modifier)<overlay specification>{text}
```

## Example

```
1 \onslide<1>{(1) onslide}
2 \uncover<1>{(1) uncover}
3
4 \onslide+<2>{(2) onslide+}
5 \visible<2>{(2) visible}
6
7 \onslide*<3>{(3) onslide*}
8 \only<3>{(3) only}
```

(1) onslide (1) uncover

# The `onslide` Command

When you want to display or hide some contents on some slides, you can use

## Command

```
\onslide(modifier)<overlay specification>{text}
```

## Example

```
1 \onslide<1>{(1) onslide}  
2 \uncover<1>{(1) uncover}  
3  
4 \onslide+<2>{(2) onslide+}  
5 \visible<2>{(2) visible}  
6  
7 \onslide*<3>{(3) onslide*}  
8 \only<3>{(3) only}
```

(2) onslide+ (2) visible

# The `onslide` Command

When you want to display or hide some contents on some slides, you can use

## Command

```
\onslide(modifier)<overlay specification>{text}
```

## Example

```
1 \onslide<1>{(1) onslide}  
2 \uncover<1>{(1) uncover}  
3  
4 \onslide+<2>{(2) onslide+}  
5 \visible<2>{(2) visible}  
6  
7 \onslide*<3>{(3) onslide*}  
8 \only<3>{(3) only}
```

(3) onslide\* (3) only

## Some explanations:

- `\onslide` is equivalent to `\uncover`, the text is only shown (“uncovered”) on the specified slides. On other slides, **the text still occupies space** and it is still typeset, but it is not shown or only shown as if transparent (can be set with the command `\setbeamercovered`).
- `\onslide+` is equivalent to `\visible`, it does almost the same as `\uncover`, but it is never transparent, but rather it is not shown at all.
- `\onslide*` is equivalent to `\only`, the text is inserted only into the specified slides. For other slides, the text is simply thrown away. In particular, **it occupies no space**.

## There are also some similar commands:

- `\invisible<overlay specification>{text}` is opposite to `\visible`.
- `\alt<overlay specification>{text}{alternate text}` will show the text on the specified slides and the alternate text on other slides.
- `\temporal<overlay specification>{before text}{text}{after text}` will show the text on the specified slides, the before text on the slides before the interval, and the after text on the slides after the interval.

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

**Animation**

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

## 13 Overlay and Animation

- Overlay
- **Animation**

## 14 Special Structures

# Zooming

Zooming is necessary when you want to explain a part of a frame (or a very complicated graphic).

## Command

```
\framezoom<button overlay specification><zoomed overlay  
specification>[options](x,y)(w,d)
```

This command should be given somewhere at the beginning of a frame. The button overlay specification is your main slide with the whole graph, and there will be a clickable area which will navigate you to the zoomed slide, specified by the zoomed overlay.

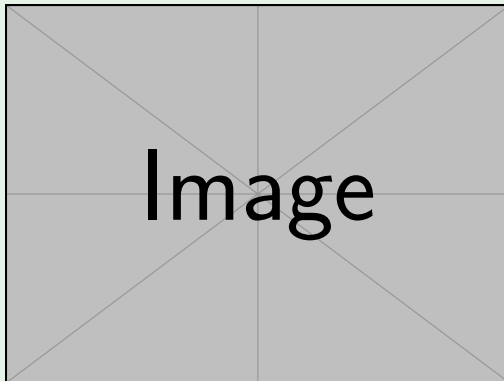
$(x,y)$  is the upper left corner of the clickable area. Thus, the location  $(0pt,0pt)$  is at the beginning of the normal text (which excludes the headline and also the frame title).  $(w,d)$  is the width and depth (height) of the clickable area.

You can also add `border=width` as `options` so that there will be a border around the clickable area.



## Example

```
1 \framezoom<1><2>(0cm,0cm)(4cm,3cm)
2 \framezoom<1><3>(1.5cm,3.5cm)(4cm,3cm)
3 \pgfimage[height=5cm]{example-image}
```



Try to click on the code and the image.

# Example

```
1 \framezoom<1><2>(0cm,0cm)(4cm,4cm)
2 \framezoom<1><3>(1.5cm,3.5cm)(4cm,4cm)
3 \pgfimage[height=5cm]{example-image-10x10}
```



# Image

# Limitations of Zooming

Though `\framezoom` is very powerful, it still has some limitations.

- You can click on the zoomed slide to jump back to the origin slide, but it only works on Adobe Reader or Acrobat.
- The backend of X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X will ignore all hyperlinks without text in it, so when compiling with `xelatex`, the area is no longer clickable. You can use `pdflatex` or `lualatex` when you need the zooming feature.

You can also redefine some macros in the `beamer` package to use `xelatex` with expected behavior, but it will require some knowledge of the inner concept of L<sup>A</sup>T<sub>E</sub>X. You can check `lecture.sty` for more details.

# The `\againframe` Command

You can use the `\againframe` command to “continue” frames that you previously started somewhere, but where certain details have been suppressed. You need to add a label for the frame to be repeated.

## Command

```
\againframe<overlay specification>[options]{label}
```

You can use this command together with the `\framezoom` command to put the zoomed slides at the end of the presentation.

## Example

```
1 \begin{frame}<1>[label=zooms]
2 \frametitle<1>{A Complicated Picture}
3 \framezoom<1><2>[border](0cm,0cm)(2cm,1.5cm)
4 \framezoom<1><3>[border](1cm,2cm)(2cm,1.5cm)
5 \pgfimage[height=6cm]{example-image}
6 \end{frame}
7 % other slides
8 \againframe<2->[noframenumbering]{zooms}
```

Liu Yihao

Introduction

Beamer Document

Beamer Structure

Overlay and Animation

Overlay

Animation

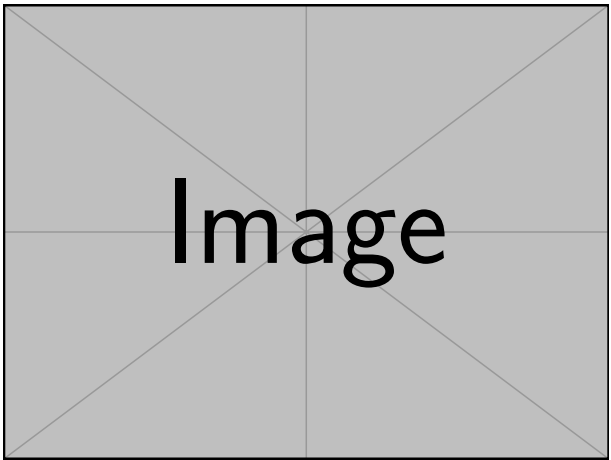
Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

# A Complicated Picture



Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

## 13 Overlay and Animation

## 14 Special Structures

- Blocks and Columns
- Hyperlinks and Buttons
- Fragile Frame

# The `block` Environment

Blocks in `beamer` are based on the `tcolorbox`, with all styles configured.

## Command

```
1 \begin{block}<action specification>{title}  
2   % block contents  
3 \end{block}
```

If the `action specification` is present, the given actions are taken on the specified slides.

There are three types of blocks: `block`, `alertblock` and `exampleblock`. The only difference is their color and style.



## Example

```
1 \begin{block}{Normal Block}
2   This is a normal block.
3 \end{block}
4 \begin{alertblock}{Alert Block}
5   This is an alert block.
6 \end{alertblock}
7 \begin{exampleblock}{Example Block}
8   This is an example block.
9 \end{exampleblock}
```

### Normal Block

This is a normal block.

### Alert Block

This is an alert block.

### Example Block

This is an example block.

# Predefined block environments

Some other block environment are predefined for ease to use. They are `theorem`, `corollary`, `definition`, `definitions`, `example`, `examples`. Here only a few of them are shown below.

## Example

```
1 \begin{theorem}[additional text]
2   The additional text will be in brackets if the option is provided.
3 \end{theorem}
4 \begin{proof}[Proof Name]
5   The default title is ``Proof.'', which can be replaced by the option.
6 \end{proof}
```

## Theorem (additional text)

*The additional text will be in brackets if the option is provided.*

## Proof Name.

The default title is “Proof.”, which can be replaced by the option. ☐

# The `columns` Environment

In an `article` class, `minipage` is often used to split contents into multiple columns; In `beamer`, you can use the `columns` environment as an alternative.

## Command

```
1 \begin{columns}[options]
2   \begin{column}[placement]{width}
3     % contents
4   \end{column}
5   \column[placement]{width}{...}
6 \end{columns}
```

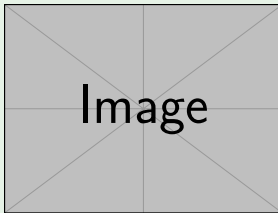
For the `options` in the `columns` environment,

- `b`, `c` and `t` - will cause the columns to be vertically aligned `bottom`, `center` and `top`.
- `T` - similar to `t`, if strange things happen in `t`, try this option.
- `totalwidth=width` - will cause the columns to occupy not the whole page width, but only `width`.

## Example

```
1 \begin{columns}[c]
2   \begin{column}{0.5\textwidth}
3     \begin{center}
4       The first line \\
5       The second line
6     \end{center}
7   \end{column}
8   \column{0.5\textwidth}{
9     \includegraphics[width=0.6\textwidth]{example-image}
10  }
11 \end{columns}
```

The first line  
The second line



You should place only `column` environments or `\column` commands in the `columns` environment.

For the `placement` in the `column` environment or the `\column` command, you can overwrite the `b`, `c`, `t` and `T` in the outer `columns` environment. The default of `placement` is `t` if not specified.

The `width` is the same as other `width` in L<sup>A</sup>T<sub>E</sub>X. For example, you can use `5cm`, or `\textwidth`.

Generally speaking, there are few differences between `minipage` and `columns`, but the `columns` environment has a more user-friendly structure, and overlap is supported better in it. Despite which do you prefer, choosing one of them and sticking to it throughout a presentation is suggested.

# Footnote in column

Using footnotes is usually not a good idea. They disrupt the flow of reading.

When you really need it, you can use the `\footnote` command, which is slightly different from common L<sup>A</sup>T<sub>E</sub>X.

## Command

```
\footnote<overlay specification>[options]{text}
```

As usual, you can give a number as `options`, which will cause the footnote to use that number.

You can also add a `frame` as `options` so that the footnote will be shown at the bottom of the frame. This is normally the default behavior anyway, but in `minipage`, `columns` and certain blocks it makes a difference.

In a `minipage` or `column`, the footnote is usually shown as part of the minipage rather than as part of the frame.

## Example

```
1 \begin{columns}[c,totalwidth=0.9\textwidth]
2   \begin{column}{0.3\textwidth}
3     The first line \footnote[frame,1]{footnote 1}
4   \end{column}
5   \begin{column}{0.3\textwidth}
6     The second line \footnote[frame,2]{footnote 2}
7   \end{column}
8   \begin{column}{0.3\textwidth}
9     The third line \footnote[3]{footnote 3}
10  \end{column}
11 \end{columns}
```

The first line <sup>1</sup>	The second line <sup>2</sup>	The third line <sup>c</sup>
		<sup>c</sup> footnote 3

As you can see, placing footnote at the bottom of a `column` is not a good idea, so using the `frame` option is preferred in most situations.

---

<sup>1</sup>footnote 1  
<sup>2</sup>footnote 2

## 12 Introduction

## 13 Overlay and Animation

## 14 Special Structures

- Blocks and Columns
- Hyperlinks and Buttons
- Fragile Frame



# Hyperlinks and Buttons

Here is a simple three-step workflow of how to create hyperlinks and buttons in your slides:

- ❶ You specify a target using the command `\hypertarget` or (easier) the command `\label`. In some cases, see below, this step may be skipped.
- ❷ You render the button using `\beamerbutton` or a similar command. This will render the button, but clicking it will not yet have any effect.
- ❸ You put the button inside a `\hyperlink` command. Now clicking it will jump to the target of the link.

# The `beamerbutton` command

## Command

```
\beamerbutton{text}
```

There are four types of buttons, the suggested usages of them are:

- `\beamerbutton` - a normal button
- `\beamergotobutton` - jump to another area of the presentation
- `\beamerskipbutton` - skip a well-defined part
- `\beamerreturnbutton` - return back to a previous part

## Example

```
1 \beamerbutton{normal} \beamergotobutton{goto}
2 \beamerskipbutton{skip} \beamerreturnbutton{return}
```

normal

▶ goto

▶▶ skip

◀ return

The color of the buttons will follow the current color of the block.

normal

▶ goto

▶▶ skip

◀ return

# The `hyperlink` command

## Command

```
\hyperlink<overlay specification>{target}{text}
```

If the overlay specification is present, the hyperlink (including the text) is completely suppressed on the non-specified slides.

You will jump to the `target` you defined before with a `\hypertarget` command.

You can also use a L<sup>A</sup>T<sub>E</sub>X command from the `hyperref` package as a target, e.g., `\href`.

## Example

- 1 You can find the source of the slides on
- 2 `\href{https://github.com/SJTU-UMJI-Tech/LaTeX}{\beamerbutton{GitHub}}`.

You can find the source of the slides on [GitHub](https://github.com/SJTU-UMJI-Tech/LaTeX).

# The `\hypertarget` command

## Command

```
\hypertarget<overlay specification>{target}{text}
```

If the overlay specification is present, the text is the target for hyper jumps only on the specified slide. On all other slides, the text is shown normally.

## Example

```
1 \begin{itemize}
2   \item<1-> First item.
3   \item<2-> Second item.
4 \end{itemize}
5 \hyperlink{jumptosecond}{\beamergotobutton{Jump to second slide}}
6 \hypertarget<2>{jumptosecond}{}

```

- First item.

▶ Jump to second slide

# The `\hypertarget` command

## Command

```
\hypertarget<overlay specification>{target}{text}
```

If the overlay specification is present, the text is the target for hyper jumps only on the specified slide. On all other slides, the text is shown normally.

## Example

```
1 \begin{itemize}
2   \item<1-> First item.
3   \item<2-> Second item.
4 \end{itemize}
5 \hyperlink{jumptosecond}{\beamergotobutton{Jump to second slide}}
6 \hypertarget<2>{jumptosecond}{}

```

- First item.
- Second item.

▶ Jump to second slide

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

## 12 Introduction

## 13 Overlay and Animation

## 14 Special Structures

- Blocks and Columns
- Hyperlinks and Buttons
- Fragile Frame

# Fragile Frame

When you need to insert any verbatim (such as `minted`) in a frame, you have to add the option `[fragile]`, and the `\end{frame}` must be alone on a single line (except for any leading whitespace).

## Example

```
1 \begin{frame}[fragile]
2   \begin{minted}{latex}
3   \documentclass{beamer}
4   \end{minted}
5   \end{frame}
```

Using this option will cause the frame contents to be written to an external file and then read back.

# Frame in Fragile Frame

A more tricky situation is that you want to put `\begin{frame}` and `\end{frame}` inside a verbatim environment (like this lecture).

One possible solution is to define a new environment so that the new environment name won't conflict with the content in the verbatim environment.

## Example

```
1 \newenvironment{fragileframe}%  
2 {\begin{frame}[fragile,environment=fragileframe]}%  
3 {\end{frame}}
```

Then you can use `\begin{fragileframe}` to substitute `\begin{frame}[fragile]` for all fragile frames.



# Introduction to L<sup>A</sup>T<sub>E</sub>X

## Lecture V: Beamer Slides

Liu Yihao

### Introduction

Beamer Document

Beamer Structure

### Overlay and Animation

Overlay

Animation

### Special Structures

Blocks and Columns

Hyperlinks and Buttons

**Fragile Frame**

Liu Yihao

Introduction

Beamer Document

Beamer Structure

Overlay and Animation

Overlay

Animation

Special Structures

Blocks and Columns

Hyperlinks and Buttons

Fragile Frame

lm