Advanced LaTeX functions and Environments

Text

Previously, we have learned how to write simple paragraphs in multiple different ways. There are however other ways in which text can be presented.

Text Styles

In LaTeX it is possible to apply effects such as bold and italics to your text. You can do so by enclosing the target text with the following commands.

For bolding text:

\textbf{ Your text to be bolded goes here }

For italicizing text:

\textit{ Your text to be italicized goes here }

These can be used within each other for producing text that is both bolded and italicized: ***like so***.

These environments can also be used within math environments. They will produce normal text (without omitting spaces and italicizing everything like the math environment does).

If you would like to write "normal" text within math environments there are multiple ways you can do this.

First is the "mathrm" command. It writes in unatalicized letters, but still ignores spaces.

The following input:

$$\mathrm{Your normal text here}$$

Yields an output of:

$$\mathrm{Yournormaltexthere}$$

This is great for writing units at the end of calculation statements.

You can learn more about other effects that can be applied to text in math environments ***here***.

If you would like to write normal text with normal spaces, you can use the "text" command from the "amsmath" package.

Recall to add this before your begin document statement:

\usepackage{amsmath}

Then the following input:

$$\text{Your normal text here}$$

Yields the following output:

Your normal text here

The last and final text command is a little different from the standard ones we have discussed before. Sometimes it may be useful to label parts of your LaTeX code for yourself, but you would not like these comments to be visible in the pdf. For this you can use the "%" command. Everything on the same line as a percentage sign, will be ignored by the LaTeX compiler and not be added to the pdf.

Lists

With LaTeX it is possible to make unnumbered as well as numbered lists. This can be achieved with the "itemize" (for bulleted lists) and "enumerate" (for ordered lists) environments.

The basic structure is identical for both environments.

The environment is declared with the standard "\begin{environmentname}" command.

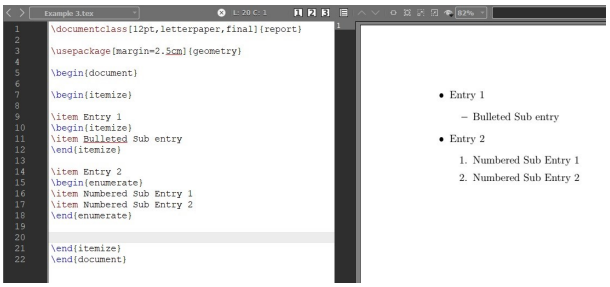
Each entry in a list is preceded by a "\item" command.

If you would like a nested list (sub-points) under a specific point, you can simply add another "itemize" or "enumerate" below said point (item). It is possible to have a numbered list as a subset of a bullet point list (and vice versa).

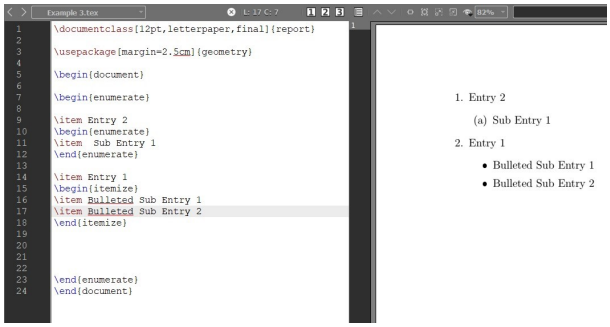
Here is the overall syntax:

\begin{itemize}
\item *Entry*
...
\end{itemize}

Here are examples:



(a) Bulleted list with bulleted and ordered sub-lists



(b) Ordered list with ordered and bulleted sub-lists

Figure 1: Examples of lists with "itemize" and "enumerate" environments

If you do not like the bullet points (black dots) you can change them that to any symbol, by placing it in square brackets after the "\item" command.

Here is an example:

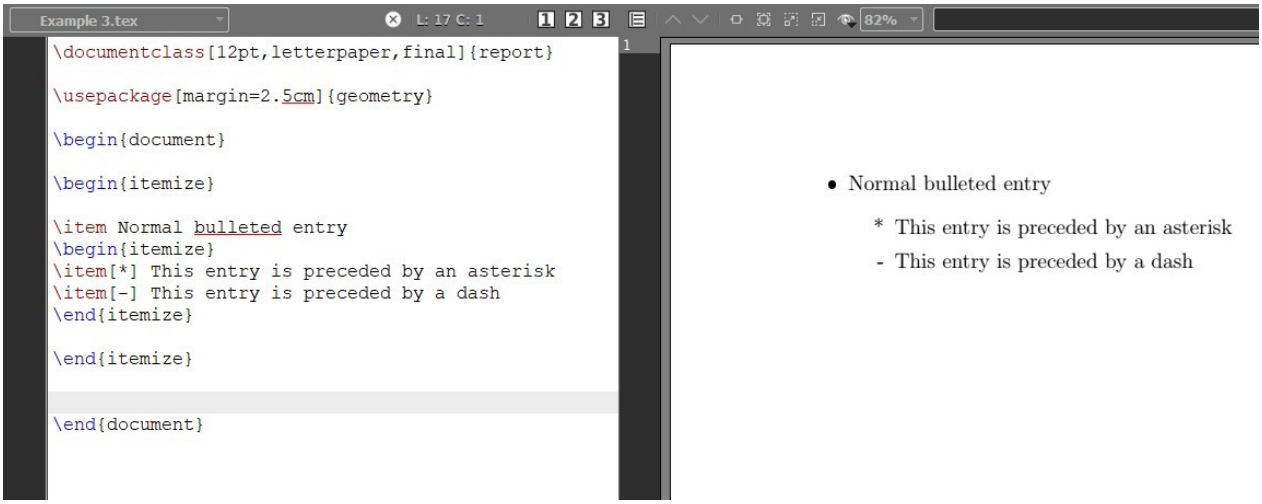


Figure 2: Bulleted entries with different symbols

Math

There are many ways of displaying math in LaTeX. We will now consider a couple more very common features, widely used in almost all works.

Equation Labeling

Referencing equations through out your paper, is an effective way of achieving clarity in a long and potentially confusing rigorous argument. For this purpose,LaTeX has a built in equation

numbering system. It can however be inefficient to manually write (1) or (2), every time you need to reference an equation. Even worse, what if you add a new equation somewhere in the beginning, changing your entire numbering system.

For this issue, LaTeX also has a built in solution. It is possible to label equations with a text name, rather than a number. This label will only be visible within the code, and will just be used to reference this equation.

To assign an equation a text label, we will need to use the "label" command. It should be placed inside of an environment that automatically numbers equations on the same line as the equation you would like to name.

It has the following structure:

\label{ eq:NameofYourEquationHere **}**

The "eq:" is necessary to specify that this is a label for an equation. Note that the space after the "eq:" counts as part of the name.

If you want to reference this specific equation later, you can us the "ref" command. It has the following structure:

\ref{ eq:NameofYourEquationHere **}**

Here is an example:

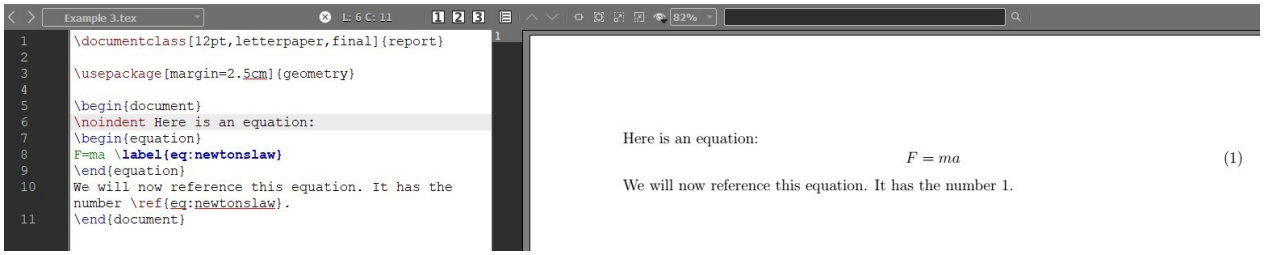


Figure 3: A labeled equation that is later referenced in text.

The "label" and "ref" commands also work with figures that have a caption. You just have to replace "eq:" with "fig:".

Here is an example:

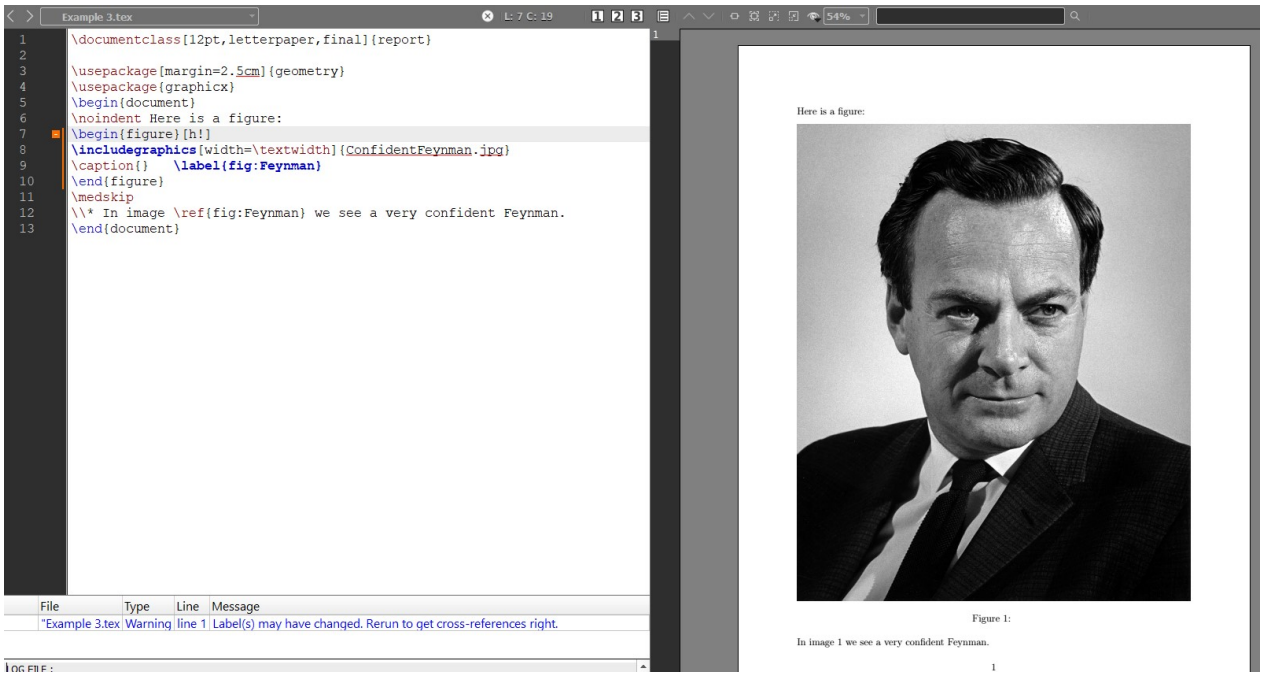


Figure 4: Labeling and referencing figures.

*Note that in order for all of your references to update, you must compile your document twice. If you only compile it once, LaTeX would have only saved your label names, but not actually put them up in your references. Your references would then just be question marks. Here is an example:

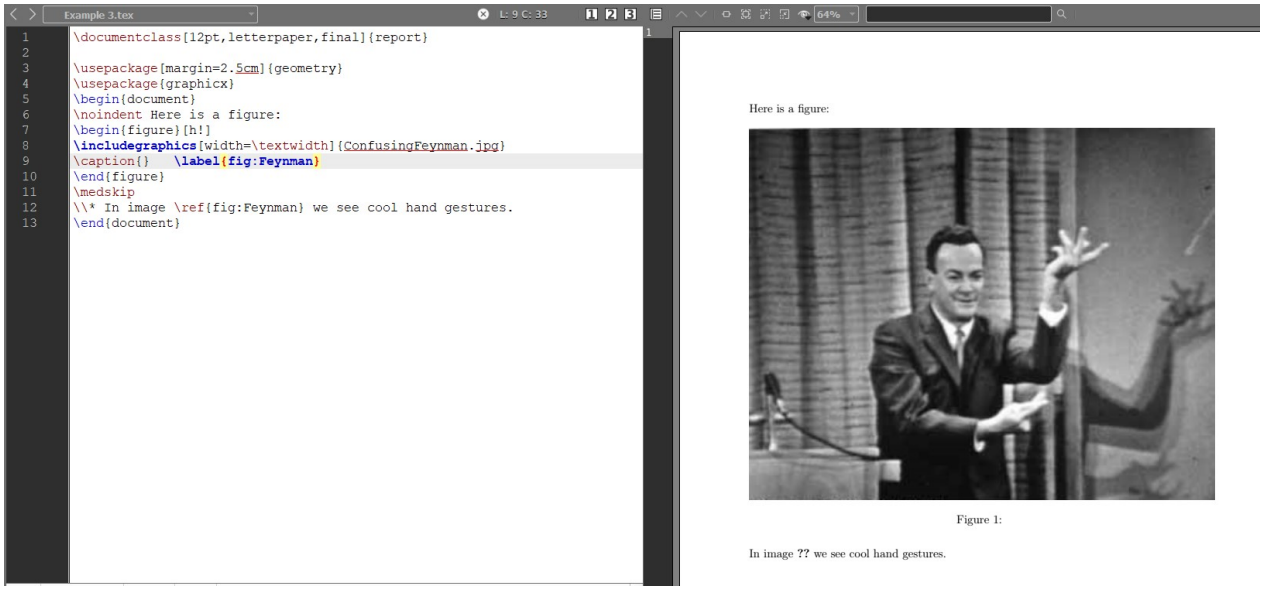


Figure 5: The references are replaced with question marks if only compiled once

Aligning Different Lines

Another useful skill when it comes to writing mathematical equations, is being table to align elements on different lines (for example equal signs). To achieve this we can use the "align" environment. It is part of the "amsmath" package.

The align environment is automatically considered a math environment, therefore no dollar bill signs (or brackets preceded by slashes) are not required. It will also automatically center and number your equations.

The align environment can only process content input as "lines" and not paragraphs. This means that you have to specify where you end each one of your lines. You can do so by placing a double backwards slash "\\" at the end of each line.

On different lines you can place a "&" symbol. LaTeX will then vertically align the lines for these symbols to be right above each other. The symbols themselves will not be printed or take up any space. Here is an example:

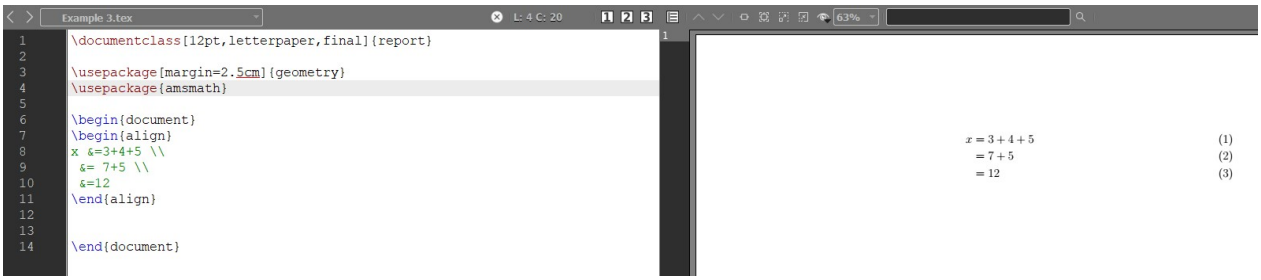


Figure 6: Example where the equal signs are aligned by placing a preceding "&" symbol.

You can label equations on separate lines using the "label" command by placing it before the "\\" end of line statement.

If you would not like any of the equations to be labeled, you can declare the align environment with a star at the end of it. This will get rid of all tags:

\begin{*align**}*
Equations and align symbols here
\end{*align**}*

If you would prefer some equations to be labeled and some not, place a "\notag" command at the end of each line you do not want to be labeled.

If you would prefer equations to be left aligned, you can use the "flalign" environment. To left align your equations in the "flalign" environment, add another "&" at the end of the first line in the environment (If you do not, it will work like a normal align environment.). Here is an example.

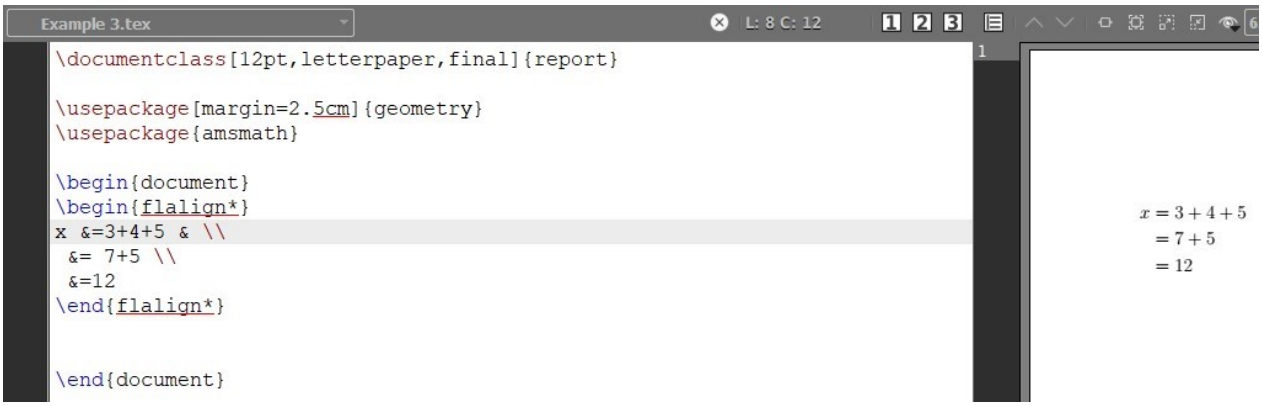


Figure 7: Example of flalign environment

Sub Figures

It is often useful to be able to place multiple sub-figures with "sub labels" (labeled a,b,c) as part of a larger figure (labeled Figure:1).

This can be achieved with the "subfigure" environment. To be able to use it, the "caption" and "subcaption" packages are required.

The subfigure environment is placed inside a figure environment and has the following structure:

\begin{*figure*}
\begin{*subfigure*}[position]{Container width}
Include graphics and caption statement
\end{*subfigure*}%
\hfill
Other subfigures following identical format
\end{*figure*}

Here are the environment's arguments

- Position - used to signify the position of the sub-image within the figure container. This input is optional. Here are two of the arguments it accepts:
  - b - places subfigure at bottom of figure container
  - t - places subfigure at top of figure container
- Container width
  - Determines the space allotted for your subfigure
  - It is easiest to express this in terms of \textwidth.

  - If you use \textwidth for your include graphics inside of your subfigure, it will assume 1 textwidth is the width of a the subfigure environment.

- \hfill
  - This command should be placed after the subfigure if you want all the subfigures to be placed on one line.

- %
  - Percentage sign comment "%" is added at the end of each subfigure to also ensure that the images are on one line.
  - Make sure your \hfill command is on a separate line than the percentage sign (otherwise LaTeX thinks the hfill is a comment and should be ignored)

Here is an example:

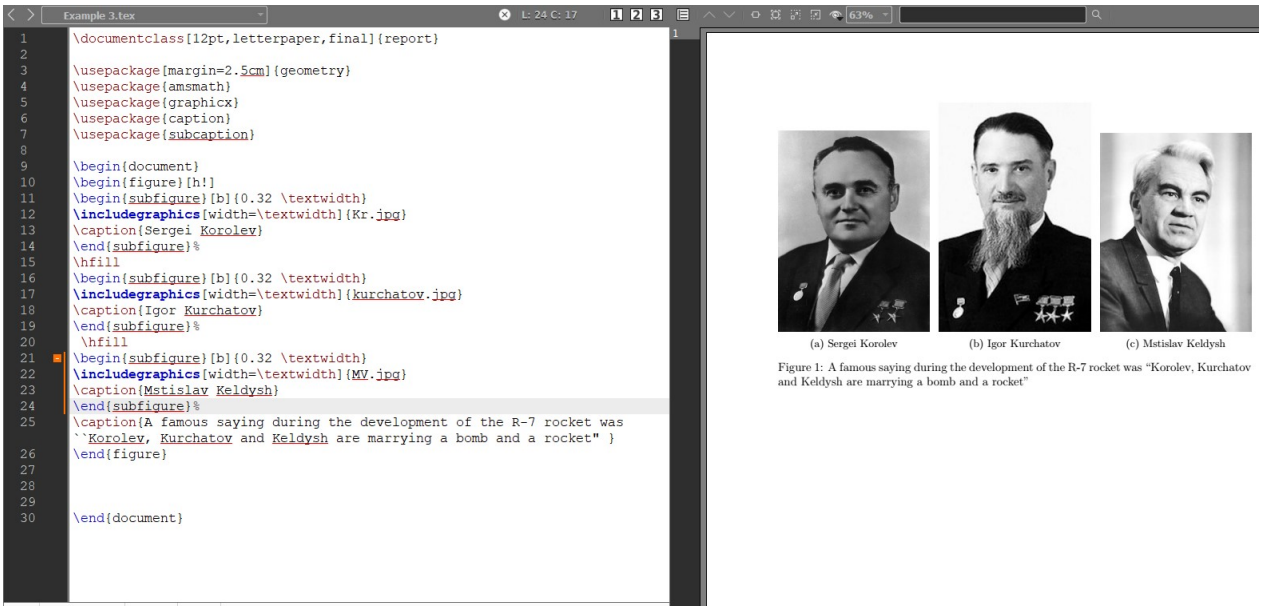


Figure 8: An example of three subfigures in one figure

Note that the width for each of the images was chosen to be less than a third of the page (even though there are 3 images), just to ensure that they will be placed on one line. The image widths in include graphics are therefore set to fill the entire available space.