Mathematical Report Typesetting

Prakhar Mittal

July 16, 2020

Contents

1 Equations and Formulae

1.1 Math Mode

To type maths in LaTeX, one needs to switch to math mode, a special mode in LaTeX. There are 2 tpes of modes:

Inline Math The typesetted math sits in the same line as the running text and is not typesetted in a math environment, generally used for simple statements such as $a = b$ to be embedded inside a line.

Math enviornment The mypesetted math sits in its own space, generally used for large or important equations or a number of equations.

Generally we don't write text in math enviornment as it would be rendered *likethis* without any spacing

1.2 Inline Math

For typesetting mathematical expressions in the midst of your text, place the expression between two $ signs. For example :

If a function f is continuous, then for every $\varepsilon > 0$ there exists a δ such that $||\mathbf{x} - \mathbf{x_0}|| < \delta$ implies $||f(\mathbf{x}) - f(\mathbf{x_0})|| < \varepsilon$.[1]

Other ways to typeset inline math is to wrap the expression in:

- \['expression here without the quotes' \]
- $$ 'expression here without the quotes' $$

For small greek letters like θ we use **\theta**. To capitalize it (Θ) use **\Theta**.
To bold in math we use **\mathbf**. However, this does not work for greek symbols. For greek symbols, we use **\boldsymbol**. For example: $\boldsymbol{\omega}$

For using subscripts/superscripts, we use underscore/carat sign before the text in subscript/superscript. If the text does not include any spaces, we can directly use them, eg. x_0^1, however, if the text does include spaces, then we have to wrap it in curly braces, eg. $p_{ij}^{\alpha+\beta}$.

Now consider $t \in \mathbb{R}$, this will typeset as

$$t \in \mathbb{R}$$

See how many other math stuff can be easily typesetted?

In some cases, we wish to use symbols that otherwise hold a meaning in syntax handling such as $ or {}. To escape their special meaning and typeset them as it is, we use backslah followed by the symbols (in most cases), like \$.

Tilde creates a space between symbols in inline math mode. For eg.

$$S := \{x|\ x \equiv 1\ (\text{mod } 9)\}.\ T := \{x|\ x \equiv 1\ (\text{mod } 3)\}.\ S \subset T\ .$$

(Notice the distance between vertical line and x)

There are several symbols which can be typesetted using LaTeX (including this one). A comprehensive list can be found here.

Some other math expressions which are commonly used are :

Combinations $\binom{n}{k}$ also written as $\frac{n!}{(n-k)!}$

[1] \varepsilon → ε whereas \epsilon → ϵ

Continued Product $n! = \prod_{i=1}^{n} i$

Integration An extension of factorial function is the Gamma function. For positive integers n,

$$(n-1)! = \Gamma(n) = \int_0^\infty x^{n-1} e^{-x} dx$$

Continued Sum It follows from the Binomial Theorem that $\sum_{k=1}^{n} \binom{n}{k} = 2^n$.

Root The Binomial Theorem is incredibly powerful and can be used to approximate $\sqrt{1+x}$, or even $\sqrt[71]{1+x}$.

Sometimes we want regular text while writing in math mode, for that we use **\text**. For eg.

$P \subseteq NP$. However, we know that testing the positivity of the term residue(n) is decidable in coNP^{RP}

1.3 The Equation Environment

Equations are often pivotal to a report, and they need to stand out. This is where the equation environment comes into play. Here's an example:

The Ideal Gas Law states that :

$$pV = nRT \tag{1}$$

Notice the label (in the source code :P). This helps to refer the equation in later sections. Hence, it is a good practice to label the equations.

Sometmes when you are absolutely sure that an equation would not come into use later (like an intermediate step of a derivation), then you would like to do away with the side numbering. This can be done in two ways, 1. using modified equation environment (equation*), or by using **\nonumber**.
Eg:

Vanderwall Proposed the real gas equation as a more accurate model.

$$\left(p + \frac{an^2}{V^2}\right)(V - nb) = nRT^2$$

If you introduce a variable compressibility factor Z, this can be ultimately expressed as

$$pV = ZnRT$$

Many a times, it is required to write several equations one after the other. In such cases, it looks nice of they are properly aligned. It is strongly recommended to use the align environment for this, which is a part of the amsmath package. Do not use eqnarray, which is simply base LaTeX.

Use & right before the symbol you want alignment about. Use double backslash (\\) for changing line. Example:

$$\begin{aligned}
\boldsymbol{\nabla} \cdot \mathbf{E} &= \frac{\rho}{\varepsilon_0} \\
\boldsymbol{\nabla} \cdot \mathbf{B} &= 0 \\
\boldsymbol{\nabla} \times \mathbf{E} &= -\frac{\partial \mathbf{B}}{\partial t} \\
\boldsymbol{\nabla} \times \mathbf{B} &= \mu_0 \mathbf{j} + \frac{1}{c^2} \frac{\partial \mathbf{B}}{\partial t}
\end{aligned}$$

See how the 4 equations are neatly aligned such that all the '=' sign lie on the same line.

[2] \left(\right) ensure that the brackets are appropriately sized

1.4 Matrices

Typesetting matrices is straightforward, very similar to tables. Here's an example:

The companion matrix M is given as:

$$\begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ a_0 & a_1 & a_2 & \dots & a_{\kappa-1} \end{bmatrix}$$

M is invertible **iff** $a_0 \neq 0$.

There are various styles of matrices that the amsmath package allows you to typeset.

1.5 Macros: Custom Commands

Many a times we need a certain typing style used again and again. If it requires multiple commands, then this work becomes quite in-efficient. Luckily, you can define custon commands in LaTeX for your ease. Custom commands are known as macros in LaTeX lingo. Macros can take multiple arguments.

Suppose you are writing a report based on **Linear Algebra**, which heavily uses the notation of inner product. Obviously, an expression like $\langle \mathbf{u}, \mathbf{v} \rangle$ is gonna come up left and right. If you were to hardcode it everytime, it would be:

```
$\langle \mathbf{u,v} \rangle$
$\langle \boldsymbol{Theta} , \boldsymbol{zeta} \rangle$
```

ad nauseum.

What you could do instead is write a macro which takes in 2 arguments and returns them in this format. Macros are defined in the preamble.

```
% In the preamble
\newcommand{\innerproduct}[2]{\langle \boldsymbol{#1} , \boldsymbol{#2} \
    rangle}
```

The new command will be invoked by using **\innerproduct**. The [2] shows that it will take in 2 arguments. Specifying this is optional, if not specified, the custom command takes in no arguments. #1 and #2 are the placeholder for each of the arguments. Our command can be invoked as follows:

```
$\innerproduct{u+w}{v}$
```

Which will typeset into $\langle \boldsymbol{u} + \boldsymbol{w}, \boldsymbol{v} \rangle$, hence saving us from excessive labour.

Suppose if you're writing a repor on an extremely efficient algorithm, and have to use the expression $\mathcal{O}(1)$ again and again. If youwish to be a bit lazy, you can just define a macro and save yourself from typing a few extra characters every time.

```
\newcommand{\Oone}{\mathcal{O}(1)}
```

should do the job. Note that the macro names can only contain alphabetic characters.

If you wish to make a custom command having the same name as an already existing command from the packages you have imported, it is a redefinition. To do so, the same syntax is followed, except the **\newcommand** is replaced by **\renewcommand**.

2 Theorems

Theorems, lemmas, propositions, proof, remarks, and corollaries, any mathematical report feels empty without these. The LaTeX package `amsmath` provides us with seperate environments to easily typeset them and be able to showcase all these easily. Recall, in the preamble, we imported `amsmath` package. The folloing lines made it very convenient to define theorem like environments

```
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
\newtheorem{corollary}[theorem]{Corollary}
\newtheorem{proposition}{Proposition}[section]

\theoremstyle{remark}
\newtheorem*{remark}{Remark}
```

The `\newtheorem` takes in 2 arguments. The first argument states the environment that is defined, and the second one is the word that will be printed, in boldface font, at the beginning of the environment. Once this new environment is defined it can be used normally within the document, delimited it with the marks `\begin{theorem}` and `\end{theorem}`.[3]

Theorem 1 (Optional Clarification)**.** *This is our first theorem.*

Lemma 1. *This is our first lemma. LaTeX maintains a seperate counter for theorems and lemmas.*

Corollary 2. *This is a corollary. Since we provided an optional argument [theorem] when defining the environment, Corollaries and Theorems share the same counter.*

Theorem 3. *This proves the point about the counter.*

Lemma 2. *Hope you're satisfied by now.*

Remark. Because of the asterisk, LaTeX does not use a counter for the remarks. Also, we changed the style of Remarks from plain(default boldface un-italicized) to remark(italicized). `\theoremstyle` changed the way the starting word shows.

Proof. Behold the power of `amsmath` package; it provides a seperate environment for proofs as well. Notice the square at the end of the proof as well, this is another feature of the proof environment.[3] □

Proposition 2.1. *Because of the optional [section] argument we provided in the last, the proposition counter resets after every new section. Each proposition would be numbered as [sectionnumber].[propositionnumber].*

[3] Refer Overleaf tutorial

3 Custom Environments

LaTeX also gives us the ability to define new environments. The definition of the new environment is given in the preamble. Here is an example creating an evironment called "claim":

```
\usepackage{xcolor} % Just for the colors (not necessary for environment
    creation)
\newcounter{claim} % To create a new counter for numbering the environments
     (like theorem)
%If you want the counter to reset after every section, add a [section]
    argument after \newcolor{claim}
%\newcounter{claim}[section]
%\newenvironment{nameofenvironment}[no. of arguments]
%{<code at the beginning>}
%{<ending code>}

\newenvironment{claim}[2][black] %The [black] makes the first argument
    optional with a default value of black
% The second argument remains required

{ % This cell executes when the \begin{env.} is used
    \color{#1!50!black}
    \refstepcounter{claim} %increments the value of counter by 1

    \begin{center}
        \textbf{\large{Claim \theclaim}} \\ % prints value of counter
        \textbf{#2}
    \end{center}
}

{ % This cell executes when the \end{env.} is used
    \vspace{1.5em}
}
% For ordered environments like these, you can use labels for later
    referencing them.
```

The new environment would look like this:

Claim 1
First Custom Environment

Hello There

Claim 2
Optional and Necessary Arguments

The optional argument is wrapped in [ ] and the necessary arguments in {}.

Claim 3
Default Color

The color specifications are to be given in [ ] brackets, if none is given, then the environment assumes the default colour, specified in it's definition.

If you'd like to redesign an existing environment, use `\renewenvironment` in place of `\newenvironment`.

4 Figures and Tables

Often, in atechnical setting, you would find the need to paste pictures, and draw tables. LaTeX has environments to do just that.

4.1 Figures

The `floatrow` package will come into use here.

```
\begin{figure}[H] % [H] s from floatrow package which forces the picture to
    be typeset RIGHT HERE.
    \centering
    % make sure your picture is in the same working directory, or else,
       provide a relative path to it
    \includegraphics[width=0.6\linewidth]{<imagename.extension here>} %note
        the optional argument provided to adjust the size of the image.
    \caption{Anything} % This would display below the picture like ``Figure
        1: <caption>''
    \label{fig:img}
\end{figure}
```

And, voila!

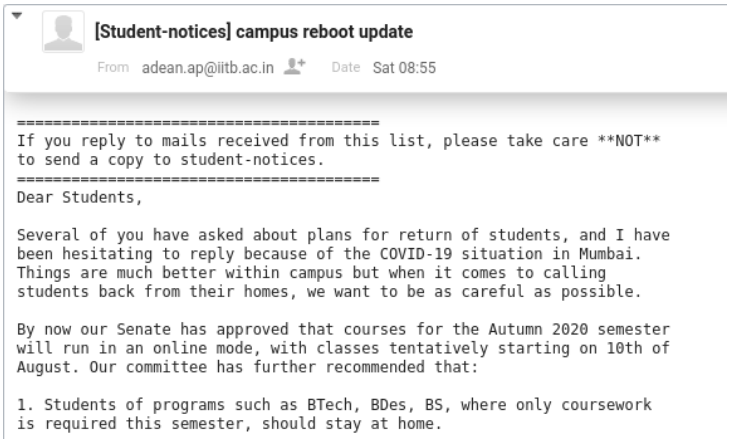

Figure 1: Semester will be online, starting 10th August

You can use `floatrow` for inserting multiple images together.

```
\begin{figure}[H]
    \centering
    \begin{floatrow}
        %Figure boxes inside a floatrow will be placed side by side
        %It's wise to keep a gap whle specifying dimensions
        %\ffigbox[size]{text}{graphics}
        \ffigbox[0.4\textwidth]{\caption{A picture}}{\includegraphics[width
            =0.4\textwidth]{Proof_of_online_semester.png}}
        % Make sure the image width does not exceed that of the box.
        \ffigbox[0.4\textwidth]{\caption{The same picture}}{\
            includegraphics[width=0.4\textwidth]{Proof_of_online_semester.
            png}}
    \end{floatrow}
    \begin{floatrow}
        \ffigbox[0.6\textwidth]{\caption{Same picture, but larger}}{\
            includegraphics[width=0.6\textwidth]{Proof_of_online_semester.
            png}}
    \end{floatrow}
\end{figure}
```

This would show up as follows:

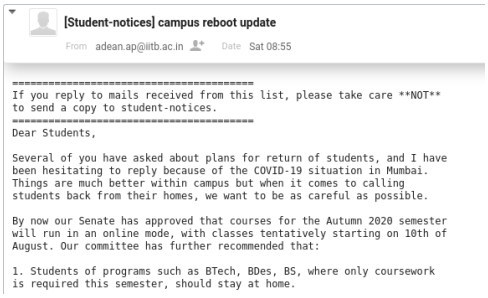

Figure 2: A picture

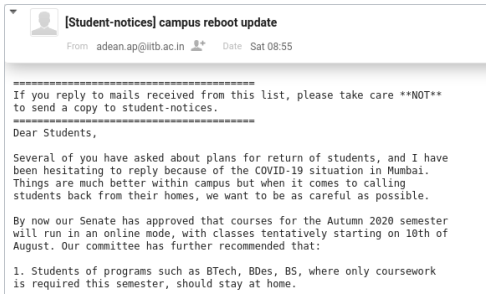

Figure 3: The same picture

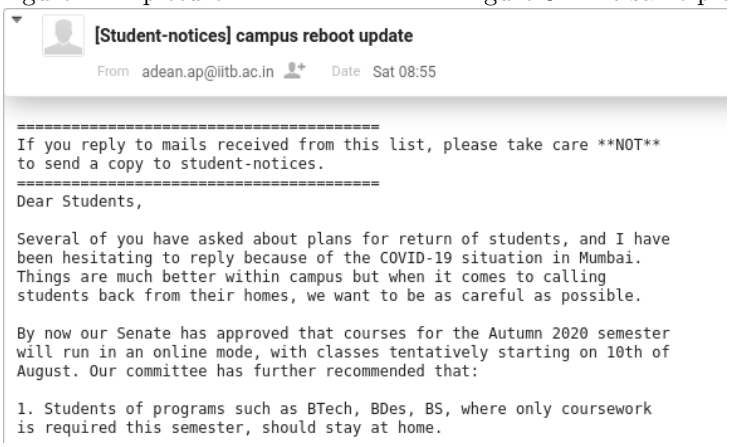

Figure 4: Same picture, but larger

4.2 Tables

The `table` environment encapsulates the LaTeX table. It takes care of the placement of the table in the document, caption, etc. Nested in the `table` environment is the `tabular` environment. The latter is where we enter the contents of out table.

Note that \\ starts a new row, and the ampersand & acts as the column seperator.

```
\begin{table}[H]
    \centering
    \begin{tabular}{l|c|r} % Alignment of text in each column. also the
        vertical lines
        \hline
        \textbf{Left} & \textbf{Center} & \textbf{Right} \\
        \hline
        Liberal & Moderate & Conservative \\
        Blue & vs & Red \\
        Biden & vs & Trump \\
        \hline
    \end{tabular}
    \caption{A Simple Table, USA Votes.}
    \label{tab:votes}
\end{table}
```

Left	Center	Right
Liberal	Moderate	Conservative
Blue	vs	Red
Biden	vs	Trump

Table 1: A Simple Table, USA Votes.

Include packages `multirow` and `multicol` packages in the preamble.

With the `\multirow{NUMBER_OF_ROWS}{WIDTH}{CONTENT}` command, we can have a cell span multiple rows. (Using * as width lets LaTeX automatically determine it):

```
\begin{table}
    \centering
    \begin{tabular}{l|c|r}
        \hline
        \textbf{Left} & \textbf{Center} & \textbf{Right} \\
        \hline
        Liberal & Moderate & Conservative \\
        Blue & \multirow{3}{*}{vs} & Red \\
        Biden &  & Trump \\
        Democrats & & Republicans \\
        \hline
    \end{tabular}
    \caption{Multirow Table. USA votes.}
\end{table}
```

Left	Center	Right
Liberal	Moderate	Conservative
Blue		Red
Biden	vs	Trump
Democrats		Republicans

Table 2: Multirow Table. USA votes.

With the `\multicols{NUMBER_OF_COLS}{ALIGNMENT}{CONTENT}` we can have a cell span several columns.

```
\begin{table}[H]
    \centering
    \begin{tabular}{|c|c|c|c|}
        \hline
        \textbf{Day} & Session 1 & Session 2 & Session 3 \\
        \hline
        1 & \multicolumn{2}{c|}{Australia} & India \\
        \hline
        2 & \multicolumn{3}{c|}{Australia} \\
        \hline
        3 & \multicolumn{2}{c|}{India} & Australia \\
        \hline
        4 & \multicolumn{3}{c|}{India} \\
        \hline
        5 & \multicolumn{2}{c|}{Shared} & India \\
        \hline
```

```
    \end{tabular}
    \caption{Who dominated which session: India v/s Australia, Kolkata
        2001}
    \label{edengardens}
\end{table}
```

Day	Session 1	Session 2	Session 3
1	Australia		India
2	Australia		
3	India		Australia
4	India		
5	Shared		India

Table 3: Who dominated which session: India v/s Australia, Kolkata 2001

We can also use `multirow` and `multicolumn` in tandem. Observe the code carefully to see how it's done.

a^2		ab	.
ab		b^2	.
.	.	.	.

Table 4: Middle School Algebra

```
\begin{table}[H]
    \centering
    \begin{tabular}{|c|c|c|c|}
        \hline
        \multicolumn{2}{|c|}{\multirow{2}{*}{$a^2$}} & \multirow{2}{*}{$ab
            $} & . \\
        \multicolumn{2}{|c|}{} & & . \\ % The empty container is
            aplaceholder, so that the column seperation line does not print
        \hline
        \multicolumn{2}{|c|}{$ab$} & $b^2$ & . \\
        \hline
        . & . & . & . \\
        \hline
    \end{tabular}
    \caption{Middle School Algebra}
    \label{multirowcolumn}
\end{table}
```

4.3 Floats

Figures and Tables in LaTeX are actually examples of Floats, containers for things in a document that cannot be broken over a page. LaTeX Does not put the floats into wherever you write them in your code, but it decides the best place to put it. Sometimes your image or table can even end up in a different page than the content where you wrote the code to insert the table.

You can tell LaTeX where youwould prefer it to put your float, by passing in a location specifier inside []. For example, in the Tables/Figure codes, you'll see something like `\begin{table}[H]` , here the [H]

tells LaTeX that you want to place the float precisely at the position in your LaTeX code. While this may seem desirable, it does not always give desirable results. Her eare the specifiers you can put in

h Place the float here, i.e., *approximately* at the same point it occurs in the source text (however, not *exactly* the same point).

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 LaTeX uses for determining "good" float positions. You would use it in conjunction with another specifier, like `\begin{table}[h]`!. This is kind of like forcing LaTeX to put it wherever you are telling it to.

H As you know, places the float at precisely the location as in the LaTeX code.

5 Highlighting Code

Programming now permeates in almost every field of research, and hence, code is a necessity in most technical reports. To get started, we recommend the easily accessible `listings` package. `minted` does an even more elaborate job, however, it requires some effort to get it running. [1]

Set your preferences in the preamble like this:

```
\lstset{
    numbers=left,
    %language = <set default language>,
    breaklines=true, %automatic line breaks only at whitespace
    keywordstyle=\color{blue}\bfseries,
    numberstyle=\tiny\color{gray},
    commentstyle=\color{green!30!black},
    stringstyle = \color{violet}
}
```

And then when you want to adorn your document with code, use the `lstlisting` environment as follows:

```
\begin{lstlisting}[language=<the programming language>]
<your code goes here!>
\end{lstlisting}
```

And then, it's magic.

5.1 Example: C++ code snippet

```
[language=C++]
```

```
#include <iostream>
//prerocessor directive
using namespace std;

int main(int argc, char* argv[])
{
    string s= "Hello World";
    cout<<s<<endl;
}
```

5.2 Example: Python code snippet

```
[language=python]
```

```
from numpy import *
#insanely powerful library

print('First five odd numbers')
for x in range(10):
    if not x%2==0:
        print(x)
```

5.3 Example: Java code snippet

```
[language=java]
```

```
//yet aother Hello World

public class welcome
{
    public static void main(String[] args)
    {
        System.out.println("Welcome to the world of Java");
    }
}
```

References

[1] Jacob Morzinski. The greek alphabet. URL: https://web.mit.edu/jmorzins/www/greek-alphabet.html.