Technical Report Writing using LaTeX

[†]Dr. Shubhankar Majumdar, *Member, IEEE* and [*]Dr. Soumen Moulik, *Member, IEEE*
[†]Department of Electronics and Communication Engineering
[*]Department of Computer Science and Engineering
National Institute of Technology Meghalaya, India
Email: [†]shub@nitm.ac.in, [*]smoulik@nitm.ac.in

***Abstract*—This document provides few basics on how to write technical papers in LaTeX. The technical reports will be shown as examples like journal article/conference paper (IEEE template). Then the beamer utility will also be shown which will be utilized for the presentation purpose.**

***Keywords*—LaTeX, NIT Meghalaya.**

I. Introduction

TeX is a low-level markup and programming language which is predominantly used for typesetting professional scientific papers attractively and consistently. LaTeX is a macro package based on TeX. Its purpose is to simplify TeX typesetting, especially for documents containing mathematical formulae. Many authors have contributed extensions, called packages or styles, to LaTeX. Some of these are bundled with most TeX/LaTeX software distributions; more can be found in the Comprehensive TeX Archive Network or CTAN [1].
There are many aspects which can be covered to have a comprehensive lesson on LaTeX. However, in this workshop, we shall learn few basics of these aspects so that a beginner can start with LaTeX to prepare his/her own technical reports or papers. More detailed features can be learned through various resources available online [2] (readers will be redirected to those resources as and when necessary)
In this document, We shall learn how to write mathematical equations, algorithms/pseudo code, proofs of theorems, how to create tables and figures and accordingly resize and place them in your document, how to create bulleted/numbered list, how to use different formats of the document required for review process.

II. Basics: LaTeX syntax

LaTeX uses a markup language in order to describe document structure and presentation. The input for LaTeX is a plain text file. You can create it with any text editor. It contains the text of the document, as well as the commands that tell LaTeX how to typeset the text.

1) **Space** - The LaTeX compiler normalizes whitespace so that whitespace characters, such as [space] or [tab], are treated uniformly as “space”: several consecutive “spaces” are treated as one, “space” opening a line is generally ignored, and a single line break also yields “space”. A double line break (an empty line), however, defines the end of a paragraph; multiple empty lines are also treated as the end of a paragraph.
2) **Reserved Characters** - The following symbols are reserved characters that either have a special meaning under LaTeX or are unavailable in all the fonts. If you enter them directly in your text, they will normally not print, but rather make LaTeX do things you did not intend. # $ % ˆ & _ { } ˜ \ are reserved characters. These characters can be used in your documents all the same by adding a prefix backslash, such as, \# \$ \% \ˆ \& _ \{ \} \˜ \textbackslash.
3) **Groups** - A group is basically defined by a pair of braces. The range of commands put between braces is limited to them. The and commands are equivalent to opening brace and closing brace.
4) **Commands** - LaTeX commands are case sensitive. Some commands need an argument, which has to be given between curly braces after the command name. Some commands support optional parameters, which are added after the command name in square brackets [ ].
5) **Environments** - Environments in LaTeX have a role that is quite similar to commands, but they usually have effect on a wider part of the document. Example: \begin{environmentname}...\end{environmentn ame}. Anything in LaTeX can be expressed in terms of commands and environments.

III. Document Structure

When LaTeX processes an input file, it expects it to follow a certain structure. Thus every input file must contain the commands

```
\documentclass{...}

\begin{document}
...
\end{document}
```

The area between \documentclass{...} and \begin{document} is called the preamble. It normally contains commands that affect the entire document. After the preamble, the text of your document is enclosed between two commands which identify the beginning and end of the actual document.
There are two types of preambles.

Dr. Shubhankar Majumdar and Dr. Soumen Moulik both are Assistant Professors at NIT Meghalaya, India.

1) **Document class** - when processing an input file, LaTeX needs to know the type of document the author wants to create. This is specified with the \documentclass command. It is recommended to put this declaration at the very beginning.

   \documentclass[options]{class}

2) **Packages** - While writing your document, you will probably find that there are some areas where basic LaTeX cannot solve your problem. If you want to include graphics, colored text or source code from a file into your document, you need to enhance the capabilities of LaTeX. Such enhancements are called packages. Some packages come with the LaTeX base distribution. Others are provided separately. Modern TeX distributions come with a large number of packages pre-installed. Packages are activated with the

   \usepackage[options]package

the document body, in general, contains different commands such as, \title{...}, \author{...}, \date{...}, \maketitle followed by the main content in plain text format. To arrange your text into different section and subsections, you can use \section{...}, \subsection{...} etc.
In the next sections, we will specifically look into how mathematical equations, figures, tables, algorithms, proofs etc. which are often required for scientific paper writing, can be typeset using LaTeX. A sample source code to create simple document in LaTeX is as follows:

```
\documentclass[12pt,a4paper,oneside]{article}
\begin{document}
\title{This is a Test Report}
\author{A. Author}
\date{}
\maketitle
\section{Introduction}
\textbf{Hello World!}
\\ Start using LaTeX.
\end{document}
```

IV. How to write mathematical equation

If your document requires only a few simple mathematical formulas, plain LaTeX has most of the tools that you will need. If you are writing a scientific document that contains numerous complicated formulas, the *amsmath* package introduces several new commands that are more powerful and flexible than the ones provided by LaTeX. Here, we enlist some useful features that we often require to implement while writing equations in LaTeX.

- Inline math notation - you should enclose the mathematical notation within dollar signs ($) while writing along the main text like $(x-y)=4$ which results into $(x-y)=4$.
- Separate equation form - a equation should be written within \begin{equation}...\end{equation} environment as shown in the following example.

```
\begin{equation}
\label{eq1}
     (x-y) = 4
\end{equation}
```

The above source code yields:

$$(x-y)=4 \tag{1}$$

- Equation without numbering - use *equation** environment instead of *equation*. So the equation should be enclosed within \begin{equation*}...\end{equation*}.

$$(x-y)=4$$

Another way of doing this is to use \nonumber after the equation line. This is mostly needed when we write multi-line equations, to avoid numbering each and line erroneously.

- Multiline equation - in case of multi-line equations, one option is to enclose the equation within \begin{align}...\end{align}.

```
\begin{align}
\label{eq3}
& x-y = 4 \nonumber \\
\Rightarrow & x = y + 4
\end{align}
```

output:

$$\begin{aligned} & x-y=4 \\ \Rightarrow & x=y+4 \end{aligned} \tag{2}$$

This is also very useful in case of breaking a long equation into more than one line in order to fit it properly. For example, in order to generate the following equation -

$$\begin{aligned} \eta = \big[(a+b)*c\big] &- \bigg[\big[(a+b)*c\big]+\big[(a+b)*c\big]\bigg] \\ &* \big[(a+b)*c\big] \end{aligned} \tag{3}$$

we need to write this code -

```
\begin{align}
\eta = \big[(a + b) * c \big] &-
\bigg[ \big[(a + b) * c \big] +
\big[(a + b) * c \big] \bigg] \nonumber \\
&* \big[(a + b) * c \big]
\end{align}
```

Writer must be very careful regarding the position of '&', before which he/she must put a *nonumber* to avoid the unnecessary numbering.

- Fractional Number - fractional number can be written, for example, like \frac{x}{2}.

$$\left(\frac{x}{2}-y\right)=4 \tag{4}$$

- Superscript/Subscript - mathematical notations having superscript is written, for example, like x^{2} and subscript is written, for example, like x_{1}.

$$x_1^2-y=4 \tag{5}$$

$$\varphi = \frac{\big[(a+b)*c\big]-\bigg[\big[(a+b)*c\big]+\big[(a+b)*c\big]\bigg]*\big[(a+b)*c\big]}{\Bigg[\big[(a+b)*c\big]-\bigg[\big[(a+b)*c\big]+\big[(a+b)*c\big]\bigg]*\big[(a+b)*c\big]\Bigg]*\big[(a+b)*c\big]} \tag{6}$$

- Double-column equation - it is possible to fit big equations using both the columns. However, for this one must put the whole equation environment into another environment - *strip*. It is advisable to put such equations either at the extreme top or at the extreme down of a particular page. However, One may need to adjust the placing of the corresponding code to achieve this. In addition to this, \rule can be used to differentiate the text area and the equation area.

```
\begin{strip}
\rule{\textwidth}{1pt}
\begin{equation}
...
\end{equation}
\end{strip}
```

A full list of syntax for different mathematical symbols and operators can be found in the following URL: **http://www.artofproblemsolving.com/Wiki/index.php/LaTeX:Symbols**

V. How to create tables

Basic tables are not too difficult to create in LaTeX, but anything more advanced can take a fair bit of construction. It makes sense to use a dedicated tool to build tables and then to export these tables into the document. To use this table, you have to use the package *tabularx*. A table should be enclosed within \begin{table}...\end{table}. Following is the simplest example of a table in LaTeX.

```
\begin{table}[ht!]
\centering
\caption{A simple table structure}
\begin{tabular}{|c|c|c|} \hline
Col.1 & Col.2 &  Col.2 \\ \hline
a & 1 & X\\ \hline
b & 2 & Y\\ \hline
c & 3 & Z\\ \hline
\end{tabular}
\label{table_1}
\end{table}
```

The above source code yields:

TABLE I. A simple table structure

Col.1	Col.2	Col.2
a	1	X
b	2	Y
c	3	Z

More complicated table structure can be obtained by first creating it in a spreadsheet like environment (which we are more comfortable with) and then embed the corresponding LaTeX code into the main TeX file and compile. One such tool can be found here: **http://www.tablesgenerator.com/latex_tables#**. A more complicated example is given in the form of the following table.

TABLE II. A nested table structure

Col. 1	Col. 2		Col. 3	
	Col. 2.1	Col. 2.2	Col. 3.1	Col. 3.2
R1	a1	b1	c1	d1
	a2	b2	c2	d2
R2	w1	x1	y1	z1
	w2	x2	y2	z2

If width of the table is too big to be accommodated within the space of one column, then you may want the table to be extended across two columns. This can be easily achieved by using *table** environment in stead of *table*.
You can learn more features from: **http://en.wikibooks.org/wiki/LaTeX/Tables**.

VI. How to import figures

LaTeX supports the most common picture formats around. However, if the imported image is int EPS/PDF format then the quality of the image will be retained once printed. If EPS format is used then the file should be compiled by *latex* compiler (not the *pdflatex* compiler). For importing figures another package is required which is *graphicx*. A figure should be enclosed within \begin{figure}...\end{figure}.

```
\begin{figure}[h!]
\centering
\includegraphics[scale=1.3]{./figs/NIT.eps}
\caption{National Institute of Technology, Meghalaya}
\label{figure_1}
\end{figure}
```

The above source code yields the Fig. 1

ADMINISTRATIVE BLOCK
NATIONAL INSTITUTE OF TECHNOLOGY MEGHALAYA
AN INSTITUTE OF NATIONAL IMPORTANCE

Fig. 1. National Institute of Technology, Meghalaya

If width of the figure is too big to be accommodated within the space of one column, then similar to the table you may want the figure to be extended across two columns. This can be easily achieved by using *figure** environment instead of *figure*.
Apart from using *scale*, one may mention the height and width explicitly such as \includegraphics[height = 5cm, width = 5cm].
The figure can be trimmed/cropped by using additional attribute in option list: \includegraphics[scale=0.3, trim=1cm 2cm 3cm 0cm, clip=true].

Another important aspect of importing figures is to use of sub-figures. It is a very common situation when we need to place more than one sub-figure, each having their own caption, under the main caption, which is the caption of the whole figure. Usually we use sub-figures by using both the columns. However, it can be used in a single column also.

You can learn more features from: **http://en.wikibooks.org/wiki/LaTeX/Importing_Graphics**.

ADMINISTRATIVE BLOCK
NATIONAL INSTITUTE OF TECHNOLOGY MEGHALAYA
AN INSTITUTE OF NATIONAL IMPORTANCE

(a) Caption of Sub-figure 1

(b) Caption of Sub-figure 2

Fig. 2. Caption of the main Figure

VII. Create a List

Creating a list in LaTeX is very easy. All you need to do is to enclose your list items within the *itemized* environment or *enumerate* environment (if you want a numbered list). Every entry of the list should start with the command \item. For example,

```
\begin{enumerate}
\item Item 1
\item Item 2
\item Item 3
\end{enumerate}
```

The above source code yields the following list.

1) Item 1
2) Item 2
3) Item 3

VIII. How to write proofs of theorems

For *Theorem*, *Lemma*, *Proposition*, *Corollary*, we first need to use the package *amsthm* and then we need to define these environments with the use of command \newtheoremnamePrint. Certainly, this command has two parameters, the first one is the name of the environment, and the second one is the word that will be printed, in boldface font, at the beginning of the environment. After the creation of a particular environment we need to enclose the content of a particular theorem / lemma / proposition / corollary inside the corresponding environment. Here, in this document, we have defined the theorem-related environments in the following manner.

```
\usepackage{amsthm}
\newtheorem{theorem}{Theorem}
\newtheorem{lemma}{Lemma}
\newtheorem{corollary}{Corollary}
\newtheorem{definition}{Definition}
```

According to this definition, a theorem should be enclosed within \begin{theorem}...\end{theorem}, a lemma should be enclosed within \begin{lemma}...\end{lemma}, and so on. A proof should be enclosed within \begin{proof}...\end{proof}.

```
\begin{theorem}
This is a simple theorem.
\label{theorem1}
\end{theorem}
\begin{proof}
Here goes the proof of your theorem.
\end{proof}
```

The above source code yields:

Theorem 1. *This is a simple theorem.*

Proof: Here goes the proof of your theorem. ■

```
\begin{lemma}
This is a simple lemma.
\label{lemma1}
\end{lemma}
\end{theorem}
\begin{proof}
Here goes the proof of your lemma.
\end{proof}
```

The above source code yields:

Lemma 1. *This is a simple lemma.*

Proof: Here goes the proof of your lemma. ■

Similarly, corollary, definition, etc. can also be written.

IX. How to write algorithms/pseudo codes

A pseudo code can give the reader a brief insight of the total algorithm. Packages like *algorithmic*, *algorithmicx*, *algorithm2e* can be used for this purpose.The algorithm should be enclosed within \begin{algorithm} ... \end{algorithm}. Following is a sample code -

```
\begin{algorithm}[!h]
\KwData{this text}
\KwResult{how to write algorithm
with \LaTeX}
initialization\;
\While{Some condition is true}{
read   current\;
\eIf(\tcc*[f]{then comment})
{if current is something}{
do some work\;
}{
do some more work\;
}
}
\caption{How to write algorithms}
\end{algorithm}
```

The above source code yields:

An algorithm have some input data and some precise result. The input data and the result can be shown using the commands like \KwData {} and \KwResult {}. The data of the

Algorithm 1: How to write algorithms

```
  Data: this text
  Result: how to write algorithm with LaTeX
1 initialization;
2 while Some condition is true do
3     read current;
4     if if current is something then            /* then comment */
5         do some work;
6     else
7         do some more work;
```

algorithm goes right after that. Various statements like if else, while , for , switch etc can be written using the commands like \uIf {if condition}{text} \uElseIf{else if condition}, \While {condition}{while text}, \For{loops condition}{Fors text}, \Switch{the value} \uCase{a value} \lCase{another value} \Case {last value}.

X. Miscellaneous

Some frequently-required things should be remembered while creating a latex document. They are summarized as follows.

- More than one blank spaces, a single 'enter', tab(s) are treated as a single blank space.
- More than one 'enter' starts a new paragraph.
- Hyphenation of words by - (a single dash).
- Ranges of numbers by – (two dashes).
- To underline, italicize, and bold use *\underline{your text}*, *\textit{your text}*, and *\textbf{your text}* respectively.
- For text in math displays use *\mbox{your text}*.
- *\cdots*, *\ddots*, *\ldots*, *\vdots* represents center height dots, diagonal dots (required in matrices), lower height dots, and vertical dots, respectively.
- A good strategy to find errors is to insert *\end{document}* temporarily in different places. The reason behind this is – LaTeX stops after encountering a *\end{document}* command.

XI. Reference and Citation

For any academic/research writing, incorporating references into a document is an important task. Fortunately, LaTeX has a variety of features that make dealing with references much simpler, including built-in support for citing references. However, a much more powerful and flexible solution is achieved thanks to an auxiliary tool called BibTeX (which comes bundled as standard with LaTeX). BibTeX provides for the storage of all references in an external, flat-file database. This database can be referenced in any LaTeX document, and citations made to any record that is contained within the file. This is often more convenient than embedding them at the end of every document written; a centralized bibliography source can be linked to as many documents as desired. all you need to do are -

- Populate your *.bib* file with necessary references. There are certain formats for different type of articles (e.g., *journal article*, *conference papers*, *thesis*, *online resources*, *technical reports* etc.). You will find different formats in the *.bib* file attached with this particular document. Each of the entries should have a unique *label*.
- Include the *.bib* file into your main TeX file through \bibliography{bib_file_name} syntax.
  ```
  \bibliographystyle{IEEEtran}
  \bibliography{ref}
  ```
- Cite a particular reference by *cite* command and the *label* of the corresponding reference. For example, \cite{Reis2013} yields [3].

You can cite to many references just by separating them by comma within a single *cite* command.

XII. Conclusion

Readers are instructed to look into the TeX file for the syntax used to generate this document for better understanding about how different tables, equations, figures, sections/subsections, citation, proofs have been created through LaTeX.

References

[1] Comprehensive TeX archive network. [Online]. Available: http://www.ctan.org/

[2] Latex. [Online]. Available: http://en.wikibooks.org/wiki/LaTeX

[3] S. Reis and A. Reis, "How to write your first scientific paper," in *Interdisciplinary Engineering Design Education Conference (IEDEC), 2013 3rd*, March 2013, pp. 181–186.