Some LaTeX 2ε tips and tricks (V)

Luca Merciadri

1 Introduction

This time, as usual, we shall see some LaTeX hints (numbered according to the following sections):

2. Numbering paragraphs,
3. Incorporating MATLAB graphics,
4. Incorporating MATLAB code,
5. Customizing an index.

2 Numbering paragraphs

2.1 Example

Here is an example of numbering paragraphs:

The Title
1 One line of text that is long enough to wrap as a paragraph that is long enough to wrap
2 Second batch of text that is long enough to wrap as a paragraph that is long enough to wrap as a paragraph.
3 Still more lines of text that are long enough to wrap as a paragraph that is long enough to wrap as a paragraph.

2.2 Code

This can be done with the following code:

```
\newcounter{vcount}
\def\Header#1{\medskip%
  \hbox{\bfseries #1}%
  \setcounter{vcount}{1}%
  \everypar{\arabic{vcount}%
  \stepcounter{vcount}\ }%
}
```

You can then use

```
\Header{The Title}
First paragraph.

Second paragraph. [...]
```

3 Incorporating MATLAB graphics

MATLAB can output graphics, and of course one may want to incorporate them into a LaTeX document. This can be achieved easily, and it produces a better-looking document, because of a more coherent presentation. There are two essentially different approaches: `laprint` and Ti*k*Z/PGF-related ones — `Matfig2PGF` and `matlab2tikz`. Thanks to Marc van Dongen for telling me about the second of these. We will describe each separately.

3.1 LaPrint

For this, you need `laprint.m`. According to [9], once `LaPrint` has been launched into MATLAB, it can perform the following tasks:

- Replace all occurrences of text in the MATLAB figure by tags,
- Save the modified figure in PostScript format (`eps` file),
- Create a `tex` file with commands from the LaTeX `psfrag` package to replace the tags by the original text and to call the PostScript file.

Let's assume you have typed

```
>> set(0,'defaulttextinterpreter','none')
>> figure(1),clf
>> plot([1 2])
>> ylabel('A straight line')
```

where "`>>`" denotes MATLAB's prompt. Let's then type (assuming `laprint.m` is in your current MATLAB working directory)

```
>> laprint
```

`LaPrint` thus asks you the "Number of Figure to be Saved" and "Basename of Files to be Created". You can modify several options, then click on "`Go!`." The `laprint` script will then create two files: an `eps` one and a `tex` one.

The `tex` file can be included into LaTeX documents using the packages `graphicx`, `color` and `psfrag`. Thus, if you let "Basename of Files to be Created" to "`unnamed`", a simple `tex` file showing your graphics will be generated, and will have content like this:

```
\documentclass{article}
\usepackage{graphicx,color,psfrag}
\begin{document}
\input{unnamed}
\end{document}
```

For other pieces of information (such as how to give a predetermined size to your graphics, ...), do not hesitate to read [9].

3.2 Matfig2PGF

`matfig2pgf` converts a MATLAB figure to the Portable Graphics Format (PGF). This PGF file can be included in a LaTeX document. Once `matfig2pgf` has been launched in MATLAB, you just need to generate your plot in MATLAB, and then invoke `matfig2pgf` using

```
>> matfig2pgf('myfile.pgf')
```

where `>>` is MATLAB's prompt and `myfile.pgf` is the output file. You can now write your `.tex` document according to the following minimal structure:

```
\documentclass{article}
\usepackage{pgf}
\usepackage{pgffor}
\usepgflibrary{plothandlers}

% Or, for older PGF versions (<= 1.01)
%\usepackage{pgf}
%\usepackage{pgffor}
%\usepackage{pgflibraryplothandlers}

\begin{document}
  \begin{figure}
    \centering
    \input{myfile.pgf}
    \caption{Figure created by Matfig2PGF}
  \end{figure}
\end{document}
```

This is an easy way to put a MATLAB figure into a LaTeX document. For more information, try typing

```
>> help matfig2pgf
```

in MATLAB.

3.3 Matlab2Tikz

A third way to achieve this is to use `matlab2tikz`. Once `matlab2tikz` has been launched in MATLAB, again you just need to generate your plot in MATLAB, and then invoke `matlab2tikz` using

```
>> matlab2tikz('myfile.tikz');
```

where `>>` is MATLAB's prompt and `myfile.tikz` is the output file. You can now write your `.tex` document according to the following minimal structure:

```
\documentclass{article}
\usepackage{tikz}
\usepackage{pgfplots}
\begin{document}
\input{myfile.tikz}
\end{document}
```

For more information, please have a look at [1].

You may note that you can do all these things by using SageTeX, but it is a little bit less straightforward. It is also possible that you might have to use more than one approach (especially coupling `laprint` with the two other approaches). For example, the plot resultig from a `spectrogram` command in MATLAB can only be included in a LaTeX document with `laprint`.

4 Incorporating MATLAB code

To typeset MATLAB code ([8, 10]), one good approach is to use the `listings` package together with `mcode`.[1] Thus, you may put

```
\usepackage{listings}
\usepackage[bw,numbered,framed,final]{mcode}
```

in the preamble of your document. The following options are available for `mcode`:

- `bw` is useful if you intend to print the document,
- `numbered` is useful if you want line numbers to be written before each line of code,
- `framed` is useful if you want a frame around the source code blocks,
- `final` is useful if you have "globally" set the `draft` option, as `listings` will not, in such a case, output the code at all. That forces it to do so anyway.

You can then include a MATLAB source file using

```
\lstinputlisting{/path/to/yourmfile.m}
```

or placing snippets of source code in a `lstlisting` environment. For example, you would then do

```
\begin{lstlisting}
% Example of Matlab code for calculating
% hypotenuse
% § $c=\sqrt{a^{2}+b^{2}}$ §
a = 3;
b = 4;
c = sqrt(a^2+b^2);
\end{lstlisting}
```

Note that "§" allow you to "escape" from LaTeX mode. As a result, you are not obliged to pass lots of parameters to `listings` using `lstset`.

This will give a better presentation than using `lstlisting` together with a declaration like

```
\lstset{language=MATLAB,basicstyle=\small%
\ttfamily,showstringspaces=false,%
numbers=left,commentstyle=\itshape,%
backgroundcolor=\color{white},%
stepnumber=2,numbersep=5pt,%
escapeinside={(*@}{@*)}}
```

5 Customizing an index

5.1 Standard customizations

When generating an index with `makeindex`, one can create a `perso.ist` file with "customizations". For example:

```
heading_prefix "{\\bfseries\\hfil "
heading_suffix "\\hfil}\\nopagebreak\n"
headings_flag 1
delim_0 "\\dotfill"
delim_1 "\\dotfill"
delim_2 "\\dotfill"
```

[1] Note that this package may be downloaded at `http://files.myopera.com/locksley90/blog/mcode.sty`, at `http://web.mit.edu/~paul_s/www/14.170/matlab/mcode.sty` or even at [8].

This writes the first alphabet symbol in bold font, and uses dots as delimiters. This file is generally used jointly with `makeindex` using

```
makeindex -s perso.ist filename.idx
```

where `filename.idx` has been created by executing `latex` on `filename.tex`.

5.2 French tricks

If your document is in French, you could ask for "Symboles" at the place of "Symbols" and "Nombres" at the place of "Numbers." This is achieved by appending

```
symhead_positive   "Symboles"
symhead_negative   "symboles"
numhead_positive   "Nombres"
numhead_negative   "nombres"
```

to the previous code.

5.3 Insensitive letter sort

If you want, for example, an insensitive letter sort for letter A, you may use, according to [3]:

```
sort_rule "A" "a"
```

You can then repeat this rule for every letter.

5.4 Special letter sort

For TEX-style umlaut-macros, you may use, according to [2]:

```
sort_rule "\\\"A" "ae"
sort_rule "\\\"a" "ae"
sort_rule "\\\"O" "oe"
sort_rule "\\\"o" "oe"
sort_rule "\\\"U" "ue"
sort_rule "\\\"u" "ue"
sort_rule "\\ss({})?" "ss"
```

5.5 Math formulae sort

If you use fancy constructs such as

```
\index{log@\texttt{log}}
```

you may use, according to [5]:

```
% first remove enclosing '$'-characters
*merge_rule "\$(.*)\$" "\1"

% function-name macros will be sorted like
% the function they stand for
merge_rule "\\log" "log"
merge_rule "\\lim" "lim"
% etc.
```

5.6 Greek letter sort

For Greek letters, you may use, according to [5]:

```
% the pronounciation of Greek letters
% decides their sort order
merge_rule "\\alpha" "alpha"
merge_rule "\\beta" "beta"
merge_rule "\\gamma" "gamma"
% etc.
```

5.7 Special characters sort

According to [6], you may use

```
% special characters come first
sort_rule "\." "\b\."
sort_rule "\:" "\b\:"
sort_rule "\," "\b\,"
% etc.
```

to handle special characters correctly.

5.8 Last refinements

If the commands `\LaTeX` and `\TeX` are not correctly handled by your `makeindex`, you may use, according to [4, 7]:

```
merge_rule "\\LaTeX" "LaTeX"
merge_rule "\\TeX" "TeX"
```

References

[1] Universiteit Antwerpen. `matlab2tikz`, 2009. `http://win.ua.ac.be/~nschloe/content/matlab2tikz`.

[2] Gabor Herr. `din.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/din.ist`.

[3] Gabor Herr. `icase.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/icase.ist`.

[4] Gabor Herr. `latex.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/latex.ist`.

[5] Gabor Herr. `math.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/math.ist`.

[6] Gabor Herr. `puncts.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/puncts.ist`.

[7] Gabor Herr. `tex.ist`, 1991. `http://mirror.ctan.org/indexing/makeindex/ist/tex.ist`.

[8] Florian Knorn. M-code LaTeX Package, 2009. `http://www.mathworks.com/matlabcentral/fileexchange/8015-m-code-latex-package`.

[9] Arno Linnemann. LaPrint Users Guide (LaPrint Version 3.16), 2004. `http://www.uni-kassel.de/fb16/rat/matlab/laprint/laprintdoc.ps`.

[10] Locksley. How to include MATLAB source code in a LaTeX document, 2009. `http://my.opera.com/locksley90/blog/2008/02/25/how-to-include-matlab-source-code-in-a-latex-document`.

◇ Luca Merciadri
University of Liège
`Luca.Merciadri (at) student dot ulg dot ac dot be`
`http://www.student.montefiore.ulg.ac.be/~merciadri/`