

Various T_EX tricks

Helmer Aslaksen
Department of Mathematics
National University of Singapore
Singapore 117543
Singapore

aslaksen@nus.edu.sg
www.math.nus.edu.sg/aslaksen/

Draft: July 22, 2010

1 Graphics

1.1 Programs

1.1.1 L^AT_EX like tools

pict2e Removes the restrictions in the L^AT_EXpicture environment.

epic/eeepic The epic package uses the drawline command instead of the line command, so a line can be specified by its endpoints instead of its length. However, the slope of the line is still restricted. The eeepic package allows for arbitrary slopes, but is not compatible with pdfT_EX.

overpic The overpic package adds T_EX labels.

MetaPost Produces purified EPS files that work with either LaTeX or PDFT_EX.

Asymptote 3D oriented.

PSTricks Does not work directly with PDFT_EX, but must use LaTeX, dvips and ps2pdf. Unless you are using the microtype package there will be no difference between a PDF generated in the PSTricks way with LaTeX and one directly generated with pdflatex.

PGF/TikZ PGF is the base system that provides commands to draw vector images. TikZ is a frontend that provides a user-friendly environment for writing commands to draw diagrams. Can produce either PostScript or PDF output and works with PDFT_EX.

Sketch 3D oriented. Outputs PSTricks or PGF and TikZ.

1.1.2 \LaTeX friendly drawing programs

TpX Windows drawing program that imports EMF/WMF and SVG (and EPS and PDF through PStoedit) and outputs \LaTeX , EPS, PDF, PNG, MetaPost, SVG, PSTricks, EMF/WMF or PGF. I can import EPS files from Mathematica into TpX using PStoedit. I can then replace the Mathematica labels with \TeX labels and save in either PGF or PDF. I must manually edit the caption and reference label.

Ipe Windows/Linux program that imports JPG and PNG and outputs PDF or EPS. IPE comes bundled with a command line tool called pdftoipe that converts arbitrary PDF files to a format that can be edited with IPE.

LaTeXDraw Outputs PSTricks. I can see the PSTricks code generated as I draw.

LaTeXPiX Windows drawing program that imports JPG and PNG and outputs eepic or PGF. Working on eepic, PGF, EMF/WMF and EPS import.

JpgfDraw Outputs PGF, PNG or EPS.

jPicEdt Outputs \LaTeX , eepic or PSTricks.

mfpic Outputs Metapost.

1.1.3 Conversion programs

jPicEdt jPicEdt can import and output \LaTeX , eepic and PSTricks code.

PStoedit Converts EPS and PDF to various formats, including EMF/WMF and SVG. Can be used with TpX.

1.2 Creating graphics

1.2.1 Mathematica

- When creating graphics in Mathematica with labels, use `FormatType -> OutputForm, TextStyle -> FontFamily -> "Times", FontSize -> 12` to avoid Mathematica fonts and for proper text size.

To make sure that Mathematica doesn't use its own fonts for symbols, go to `Edit -> Preferences -> Global options`, and set `Formatting Options -> Font Options -> PrivateFontOptions -> OperatorSubstitution` to false.

- The default for the `Export` function in Mathematica is an image that is four inches wide at 72dpi. When I create an image in Mathematica that I want to use with width x in \LaTeX and also on the web, I use

```
Export["*.pdf", graphics, "PDF", ImageSize -> x*72/2.54];
Export["*.png", graphics, "PNG", ImageResolution -> 600,
      ImageSize -> x*72/2.549]
```

In Mathematica, use `export` instead of `display`. `Display` relies on `psrender`, but PDF export still uses `psrender`.

PDF files from Mathematica use the `MediaBox` (whole page) instead of the `ArtBox` (`BoundingBox`), but I can remove the white margins in Acrobat. The PNG files are fine, but I can use `ThumbsPlus` to autocrop them for an even tighter fit.

- A better alternative is to create an EPS file and convert it to PDF using the `epstopdf` or `epsapdf` macros, the `eps2pdf` GUI or `GSview`.

In Mathematica, `Export["file.eps", %, "EPS"]` creates EPS files without preview that start with `%!` . `Export["file.eps", %, "EPSTIFF"]` creates a binary preview at the beginning of the file.

Mathematica uses `PlotRange` to define the `BoundingBox` for EPS export.

1.3 MetaPost

- MetaPost's PostScript output is a low-featured dialect of the Postscript language, called purified EPS, which can be converted into PDF on-the-fly. For that reason, MetaPost graphics can be handled by both TeX and pdfTeX.

When the `dvips` driver is chosen, the `graphicx` package assumes all files with an unknown file extension to be in the EPS format. MetaPost files with a numeric default file extension are therefore handled correctly — even if only in a fall-back procedure. For the `pdfTEX` driver, the situation is a bit different. Only files with file extension `.mps` are recognized as purified EPS and can be converted to PDF on-the-fly. There are two solution:

- Change MetaPost's output file naming scheme to write files ending with `.mps` via `outputtemplate := "%j-%c.mps"`; and include the graphic files with the `mps` extension. This approach works with the `dvips` driver, too. Even though, again, this time `.mps` is an unknown file extension and triggers EPS file handling in the fall-back procedure.
- Rename MetaPost files to a `.mps` extension. We then have to add the line

```
\usepackage{ifpdf}
\ifpdf
\DeclareGraphicsRule{*}{mps}{*}{}
\fi
```

to the document preamble after loading package `graphicx`. That declaration tells `pdfLaTeX` to load all files with unknown file extensions as `mps` files.

- `MPPreview` is a MetaPost previewer in `WinEdt`. At compile time, an `.mpt` file is created if any output file (`.1`, `.2`, ...) is generated; so, menu items are disabled if this `.mpt` file is not found. I cannot use the `filenametemplate` variable to change extensions to `.mps` or `.eps`.
- Files with `.mpx` extension contain information about `TEX` labels.
- `vardef` returns a value. The call to the function should not include a `;`. `save` makes a variable name local inside a `vardef`.
- Use the `latexMP` package when printing labels and the `emp` package to include the source inside the `TEX` file.

1.4 PSTricks

- The easiest way to get PDF when using PSTricks is to use

```
latex <file>.tex
dvips <file>.dvi
ps2pdf <file>.ps
```

To get all images as eps files create a document that holds only the PSTricks images.

```
\documentclass{article}
\usepackage{pstricks}
\pagestyle{empty}
\begin{document}
... code for an image ...
\clearpage
... code for an image ...
\clearpage
...
\end{document}
```

Finish by running `pdf2eps <picture number> <pdf file without ext>`, where `pdf2eps` is the bat file:

```
pdfcrop "%2.pdf"
gswin32c.exe -sDEVICE=epswrite -dNOPAUSE -dBATCH
-dFirstPage=%1 -dLastPage=%1 -sOutputFile=%2-crop.eps %2-crop.pdf
move %2-crop.eps %2.eps
```

This has to be run for each figure.

- The Perl script `pst2pdf` is the best choice when all graphics are needed as external images. It extracts all `pspicture` and `postscript` environments from the main text body and then runs these code snippets with the same preamble as the main document. The PDF output from each of these single documents is then cropped to get rid of the white space around the figure and also converted into EPS files. After producing all PostScript-related code as a single image, saved in a default subdirectory `images/`, the script `pst2pdf` runs the source one last time with `pdflatex` and replaces all PostScript code with the previously created image.

Download the package and copy `pst2pdf` to `pst2pdf.pl` in `localtexmf\bin`. Then run `pst2pdf.pl <file>.tex`.

- For the `pst-pdf` package, I write (on Linux I use `=` instead of `#`)

```
\documentclass[12pt]{article}
\usepackage{pstricks}
\usepackage{pst-pdf}
\pagestyle{empty}
\begin{document}
```

```

\begin{pspicture}
...
\end{pspicture}
\end{document}

latex <file.tex>
dvips -Ppdf -o <file-pics.ps> <file.dvi>
ps2pdf -dAutoRotatePages#/None <file-pics.ps> <file-pics.pdf>
pdflatex <file.tex>

```

The package uses the preview package to write the pspicture environment into a special DVI file. It is important, that pspicture has the correct coordinates, otherwise you will not get the whole picture. dvips -E writes the contents into a special PS file with one picture per page. With ps2pdf this file is converted into PDF images from which the last pdflatex run reads its images page by page and replaces all with the pspicture environments.

Alternatively I can use the ps4pdf perl script or MiKTeX executable.

- The auto-pst-pdf package uses similar input, but everything is done in a single pdflatex run, and therefore you must allow execution of external programs from within pdflatex. Use the shell-escape option for TEX Live or pdflatex --enable-write18 for MiKTeX. I must first refresh the FNDB in MiKTeX.
- For pdftricks I write

```

\documentclass[a4paper]{article}
\usepackage{ifpdf}
\ifpdf%
  \usepackage[miktex]{pdftricks}
  \begin{psinputs}
    \usepackage{pstricks}
  \end{psinputs}
\else
  \usepackage{pstricks}
  \newenvironment{pdfpic}{}{}
\fi

\begin{document}
\begin{pdfpic}
  \begin{pspicture}(5,2)
  ...
  \end{pspicture}
\end{pdfpic}
\end{document}

```

and run pdflatex --enable-write18. At first I got errors about “(pdftricks) No \write 18 capability”, but then I installed the file modified by Shujun Li and refreshed the MiKTeX database and it worked.

1.5 Asymptote

- Copy asymptote.sty, asycolors.sty and ocg.sty to

```
\localtexmf\tex\latex\asymptote
```

- Add the Asymptote directory to the PATH. Right-click on 'My Computer', choose 'Properties', click the 'Advanced' tab and click the 'Environment Variables' button.
- To run Asymptote from WinEdt I create Asymptote.edt

```
WinExec('', 'C:\Program Files (x86)\Asymptote\asy.exe "%P%\N.asy"', >  
'%P', 'Asymptote', 100, 0, "", "", "%P%\N.alg");  
End;
```

To "Asymptotify", I create Asymptotify.edt

```
Exe('%B\Exec\TeX\PDFLaTeX.edt');  
WinExec('', 'C:\Program Files (x86)\Asymptote\asy.exe "%P%\N.asy"', >  
'%P', 'Asymptote', 100, 0, "", "", "%P%\N.alg");  
Exe('%B\Exec\TeX\PDFTeXify.edt');  
End;
```

I put these in

```
c:\Program Files (x86)\WinEdt Team\WinEdt 6\Exec\Helmer\
```

In Toolbar2.ini I write

```
BUTTON="Asymptote"  
BUTTON="Asymptotify"
```

and in MainMenu.ini I write

```
ITEM="Asymptote"  
CAPTION="Asymptote"  
IMAGE="CapLarge"  
MACRO="Exe('%b\Exec\Helmer\Asymptote.edt');"  
REQ_FILTER="%P%\N.asy"  
ITEM="Asymptotify"  
CAPTION="Asymptotify"  
IMAGE="Complete"  
MACRO="Exe('%b\Exec\Helmer\Asymptotify.edt');"
```

1.6 Scanning

Scan line art and text at grayscale, not black and white.

1.7 Converting graphics

1.7.1 EPS to PDF

- MiKTeX comes with several converters. `eps2eps` and `ps2ps`, which use `gs` to convert (e)ps to simpler, normalized and (usually) faster (e)ps. `ps2pdf` and `pdf2ps` use `gs`, but beware that `pdf2ps` converts fonts to bitmaps. `ps2eps` is a package that uses a Perl script, `gs` and `bbox.exe`.
- There are two versions of `epstopdf.exe`. One is an exe file that is part of MikTeX and one is a PERL script on CTAN. Both use GS. There is also the `epstopdf` package by Oberdiek, which uses shell escapes to automate the process. `eps2pdf.exe` is a stand-alone GUI on CTAN (I install it in `localtexmf`). They seem to give files of comparable size, which are smaller than the PDF files from Acrobat Distiller.
- There are also two WinEdt macros, which must be installed in a subdirectory of `%B\Macros` in the install folder `%B`, which is

```
c:\Program Files (x86)\WinEdt Team\WinEdt 6\
```

`%b` is the local directory

```
c:\Users\Administrator\Application Data\WinEdt Team\WinEdt 6\
```

or

```
c:\Users\Helmer\AppData\Roaming\WinEdt Team\WinEdt 6\
```

but `%b = %B` unless I create profiles (single user mode).

The `epsapdf` macro calls `epstopdf`, while `eps2pdf` macro calls `GSview` directly. `epsapdf` used to have trouble with spaces in the file name, but that seems to have been fixed.

`epsapdf` can be used to correct the bounding box and to reposition the figure in the lower right corner.

`eps2pdf` works on EPS files with binary preview at the beginning, while `epstopdf` requires that I edit the beginning first.

In general, WinEdt no longer requires the MUI plugin, but for `eps2pdf` I must first install the MUI plugin.

`eps2pdf` gives full page PDF files and I must change the paper size to a4 when installing. I can use `Remove White Margins` in Acrobat or the `pdfcrop` tool, which is part of MikTeX.

`eps2pdf` does not read bounding box information correctly.

- To convert multiple files I use

```
for %f in (*.eps) do gswin32c -sDEVICE=png16m  
-sOutputFile=%~nf.png -dEPCrop -r600  
-dSAFER -dPATCH -dNOPAUSE %f
```

or

```
for %%x in (*.eps) do epstopdf %%x
```

- To make the Type 1 Mathematica fonts available to Ghostscript, open GSview, go to Options -> Advanced Configure and under Ghostscript Options add

```
c:\Program Files\Wolfram Research\Mathematica\5.1\SystemFiles\Fonts\Type1
```

```
to -sFONTPATH="c:\psfonts".
```

1.7.2 EPS to PNG

- GSview conversion to PDF, PGN or JPG gives full page files, but I can use autocrop in ThumbsPlus for PGN or JPG or Remove White Margins in Acrobat.

epstopdf.exe (from CTAN) with `epstopdf -sDEVICE=png16m -outfile=p.png p.eps` gives a file with nice resolution, reasonable file size and correct BoundingBox.

epsapdf gives a low resolution file. If I select to scan the HiRes BoundingBox, I get a full page file. If I select advanced options, I get an error message and cannot convert.

Both the eps2pdf macro and GUI give the wrong BoundingBox. In the GUI, I can select `-sDEVICE=png16m` as an additional parameter. I then get a nice file with the correct BoundingBox, but with a PDF extension. In the macro I can unselect the BoundingBox. I then get a full page file that I can crop.

1.7.3 Ghostscript

Ghostview is an X11 user interface for Ghostscript, allowing me to view and navigate PostScript files. GV is a version of Ghostview with an improved user interface and the ability to display PDF files. A similar program called GSview is available for use under Linux/X11, Windows and OS/2.

1.8 Including graphics

- ```
\usepackage{ifpdf}
\ifpdf
 \usepackage[pdftex]{graphicx}
\else
 \usepackage[dvips]{graphicx}
\fi
```

If the graphicx package uses the dvips driver, dvips.def defines the graphic extensions .eps and .ps. If the graphicx package uses the pdfTeX driver, pdftex.def defines the graphic extensions .png,.pdf,.jpg,.mps. So under pdfTeX, includegraphics searches for PNG, PDF, JPG and MPS in that order. Old versions of PDFTeX (prior to version 1.10a) supported TIFF, too.

|                           | 10pt Default | 11pt Option | 12pt Option |
|---------------------------|--------------|-------------|-------------|
| <code>tiny</code>         | 5pt          | 6pt         | 6pt         |
| <code>scriptsize</code>   | 7pt          | 8pt         | 8pt         |
| <code>footnotesize</code> | 8pt          | 9pt         | 10pt        |
| <code>small</code>        | 9pt          | 10pt        | 11pt        |
| <code>normalsize</code>   | 10pt         | 11pt        | 12pt        |
| <code>large</code>        | 12pt         | 12pt        | 14pt        |
| <code>Large</code>        | 14pt         | 14pt        | 17pt        |
| <code>LARGE</code>        | 17pt         | 17pt        | 20pt        |
| <code>huge</code>         | 20pt         | 20pt        | 25pt        |
| <code>Huge</code>         | 25pt         | 25pt        | 25pt        |

Table 1: Default settings of font sizes in  $\LaTeX$ .

- From MiKTeX 2.4 onwards  $\LaTeX$  supports EPS, PNG and JPG. But using PNG or JPG requires some extra work. First, I have to specify the bounding box (which is the image size in this case) manually. If I want to add a 800x600 JPG or PNG image, I have to use the following code:

```
\includegraphics[width=6cm, bb=0 0 800 600]{file.jpg}
```

Notice that I have to give an extension, unless I use the `\DeclareGraphicsExtensions` command.

PDFLaTeX will still work if I use this code, but I get a warning message in the PDFLaTeX output about the `bb` option.

## 2 Text tricks

### 2.1 Font sizes

The default settings of font sizes in  $\LaTeX$  are given in Table 1. The font styles used in various places by the `article` style is shown in Table 2.

### 2.2 Miscellaneous text tricks

- Use

```
\usepackage[latin1]{inputenc}
\usepackage[T1]{fontenc}
```

to input international characters. Use two single quotes instead of double quote.

|               |       |      |
|---------------|-------|------|
| Title         | LARGE |      |
| Author        | large |      |
| Abstract name | small | bold |
| Abstract text | small |      |
| Section       | Large | bold |
| Subsection    | large | bold |

Table 2: Font sizes in article style.

- The `center` environment starts a new paragraph, while the centering declaration doesn't.
- To create a box with width equal to specific text use this construction:

```
\newlength{\mybox}
\settowidth{\mybox}{\framebox{\Large 10}}
```

This can then be used for example as in:

```
\item \framebox[\mybox]{\Large 1}
\item \framebox[\mybox]{\Large 10}
```

- The command `\emph{...}` gives better spacing than the declaration `{\em ...}`. In particular, it automatically handles the italic correction, so I can write `\emph{text}` instead of `{\em text\}`
- An end of file marker (^Z) in a DOS text file that is `\input` prints an æ.
- The form of the footnote mark is determined by `thefootnote`. `latex.tex` sets it to `\fnsymbol{footnote}`, while `maketitle` redefines it to `\arabic{footnote}`. `\footnotetext` leaves no mark in the text, and does not step the counter. But there is no 0 in `fnsymbol`, so if I use `\footnotetext`, `\maketitle` and `fnsymbol` it comes out empty!
- Since there might be several captions in a `figure` or `\table` environment, `\caption` works like a sectioning command within the environment, with the `\label` command going either after the `\caption` command or in its argument.

## 3 Math tricks

### 3.1 Displayed math

|                       |              |           |
|-----------------------|--------------|-----------|
|                       | no alignment | alignment |
| • one equation        | multiline    | split     |
| two or more equations | gather       | align     |

- `split` is actually a twocolumn environment, with the left column right aligned and the right column left aligned. It is meant to split *one* formula inside an `align` or `gather`, whereas `aligned` and `gathered` collect *several* formulas, similar to an array with one column.
- `align` aligns at one point, `alignat` align several `align`'s.
- Do not leave empty lines before math displays. It adds an empty paragraph indent. Compare

$$2 + 2 + 2 + 2 + 2 + 2 + 2 + 2 + 2 = 16$$

with

$$2 + 2 + 2 + 2 + 2 + 2 + 2 + 2 + 2 = 16.$$

- In general, write `& =`, not `= &` in `align`.

Here I write `= &`

$$x =y,$$

and here I write `& =`

$$x = y.$$

However, if I must break the right hand side, I can write

```
\begin{align*}
x = {} & y \\\
 & + z
\end{align*}
```

to get

$$x = y \\ + z.$$

- To number tables within equations, set

```
\renewcommand{\fnum@table}{\tablename~\theequation}
```

and put `\addtocounter{equation}{1}\verb` before each table.

- `\begin{gather}`  
`\begin{gathered}`  
`\end{gathered}`  
`\end{gather}`

will put the tag in the middle.

## 3.2 Fonts and symbols

- In math mode, `:` is a relation symbol while `\colon` is a punctuation symbol.
- `amssymb` redefines certain symbols, including `\Box` for `\square`. I can't use `amssymb` with MathTime Professional, so to avoid using `\square`, which will load a CM font, I use the `amsthm` package. I use `\quedhere` if the proof ends with a displayed equation.
- The `eufrak` gives fraktur, but not `bbb`. `amsfonts` gives fonts, while `amssymb` gives the fonts and defines symbols. I write `\mathbb{R}`.
- `\ldots` is redefined in the `amsmath` package, so I must use `\dots`.
- `*` gives the raised asterisk  $*$ , `$$*` gives  $*$ .
- `\mathbf` uses a bold roman font.

## 3.3 Operator names

- `\DeclareMathOperator{\tr}{tr}` is the `amsmath` version of `\newcommand{\tr}{\mbox{tr}}`.
- There is also a command `\operatorname` such that using `\operatorname{abc}` in a math formula is equivalent to a use of `\abc` defined by `\DeclareMathOperator`. This may be occasionally useful for constructing more complex notation or other purposes. (Use the variant `\operatorname*` to get limits.)

# 4 Tables

## 4.1 Array package

The `array` package lets me insert commands automatically before or after a column's cell.

```
\begin{tabular}{|>{\scriptsize\raggedright\arraybackslash}p{2cm}|}
```

changes the font size, adjusts the spacing in the parboxes and avoids the problem that redefines `\|`, so it can't be recognized as the end of table rows.

## 4.2 Tabs package

To change the height of all rows in a table, use `\renewcommand\arraystretch{2}`. To adjust the row height automatically, use the `tbls` package.

## 4.3 Easytable package

Use

```
\RequirePackage{easy,etex}
\documentclass{beamer}
\usepackage[thinlines,thicklines]{easytable}
\begin{document}
```

to avoid running out of `\dimen`'s when using `easytable`.

## 5 CJK

- Use IME to input in MS Word or EmEditor. Save as text with encoding GB2312, Big5 or UTF8. Open in WinEdt.
- WinEdt does not support Unicode, so CJK text will not display properly. This also includes accented characters with UTF-8 encoding.
- Some Unicode editors may add the “BOM” (Byte Order Mark) at the very beginning of the file, even if the output file encoding is set to UTF-8. The BOM under UTF-8 is the byte sequence 0xEF 0xBB 0xBF. If the output file starts with those three bytes, they should be removed, or I get strange warnings, log entries, or even errors while processing with  $\LaTeX$ .
- To get anything unusual into the bookmarks when using hyperref, like for instance pinyin tone marks, I have to use the `\texorpdfstring` command, for example `\section{\texorpdfstring{\Jie2 \Qi4}{Ji\’e Q\’i}}`. This example was easy, because the tone marks for the second or fourth tones can be obtained by standard accented characters. If I want to get for example the tone marks for the first or third tones, I have to first get the Unicode code from Reading and Writing Chinese Characters and Pinyin on the Web Using Unicode. I then write for example `\section{\texorpdfstring{\Tang1}{T{\001\001}ng}}`. If I want to use Chinese characters, I have to find the codes from the links on Chinese TeX Using the CJK LaTeX Package, Unicode TrueType Fonts and PDFTeX under Windows, and write for example `\section{\texorpdfstring{湯}{\156\157}}`. Please note that the pdfstring uses octal numbers for the hex Unicode strings. I can use <http://www.ascii.cl/conversion.htm> Conversion Table - Decimal, Hexadecimal, Octal, Binary for the conversion.

The Acrobat reader currently does not change the size of the CJK in the bookmarks properly. I have to manually switch the size of all the bookmarks to large.

- Instead of

```
\usepackage[latin1]{inputenc},
\usepackage[T1]{fontenc}
\usepackage{CJK}
```

I just use `\usepackage[T1]{CJKutf8}`.

- Load pinyin.sty after hyperref.sty to avoid conflicts. If that doesn’t work, use

```
\makeatletter
\def\PYding#1{\py@hy d\py@i dn#1ng\py@sp{}}
\makeatother
```

## 6 TeX

- In WinEdt6 the default settings for LaTeX is to compile with option `-synctex=-1` and, in WinEdt5.6, the default settings for LaTeX is without the `-synctex` option. However, MiKTeX does not support the `synctex` option for filenames with spaces. The solution is to open WinEdt and uncheck 'Use `-synctex` switch' in Options -> Execution Modes -> PDF Viewer or to use TeXLive.

This is the problem if you get an error saying "Invalid argument: file.synctex(busy) pdflatex.exe: Data: file.synctex(busy) texify: pdflatex.exe failed for some reason (see log file). PDFTeXify failed to create a pdf file."

## 7 PDFTeX

- The `hyperref` package defines `href` and `url`. `\href{URL}{text}` gives a text with link and `\url{URL}` is the same as `\href{URL}{\noinkurl{URL}}`. The `url` package redefines `url` to allow for line breaks. Load `\usepackage{url} \urlstyle{rm}` after `hyperref`. `href` uses roman, but `url` uses `tt` by default. Use `\urlstyle{rm}` to make `url` use roman. To avoid having the text "mailto:" appear in the document, use `\href{mailto:myfriend@rpi.edu}{myfriend@rpi.edu}`.
- When creating PDFs for printing, colored links are not good as they end up in gray in the final output, making it difficult to read. You can use color frames, which are not printed by setting `colorlinks=false` or by making all links black

```
colorlinks=true,
anchorcolor=black,
citecolor=black,
filecolor=black,
linkcolor=black,
menucolor=black,
pagecolor=black,
urlcolor=black
```

- The `backref` options adds backlinks text to the end of each item in the bibliography, as a list of section numbers. This can only work properly if there is a blank line after each `\bibitem`. Setting `backref=page` gives a list of page numbers. Use Alt + left arrow to move to previously viewed page in acrobat files.
- The `report` and `book` styles have different types of page numbers at the start, and this conflicts with `hyperref`. The solution is to write:

```
\begin{document}
\hypersetup{pageanchor=false}
...
\maketitle
```

```

\begin{abstract}
...
\end{abstract}
\hypersetup{pageanchor=true}

```

- `\phantomsection`  
`\addcontentsline{toc}{section}{Title}`  
adds a bookmark at the right place.
- `{\samepage`  
`\PDFbookmark[0]{Mark}{Mark}`  
`\section*{\centering Mark}`  
`}`

centers a section without a number, but with a PDF bookmark.

- `{\samepage`  
`\phantomsection`  
`\addcontentsline{toc}{section}{Contact Information}`  
`\section*{\centering Contact Information}`  
`}`

centers a section without a number, but with ToC and PDF bookmark.

## 8 Letters

- `\address{\vspace*{-3.65truecm}}\ \` moves the return address up or use `\vspace*{-3truecm}`. By using both, I can put a left aligned letterhead logo and the return address level.

I use the `labels` package to print mailing labels. Print out with no reduction to fit margins. It does not like more than five lines in the labels.

- `\enlargethispage*{100pt}` increases the page.

## 9 Empty space

- Some commands, e.g., `\vspace`, uses `\@bsphack` to put `\ignorespaces` after itself if there is a space in front (L, 153). It is a good idea to use `\ignorespaces` at the end of macros. Spaces are ignored at the beginning of `\halign` constructions like `tabular` and `array`, but if the first entry is a control sequence we might want to use `\ignorespaces`.
- Leaving space before and/or after an “invisible command” can generate unwanted space (L, 153 and `addendum.tex`). When I started writing this document I wrote

```

text \vspace{-.5\topsep}
\begin{verbatim}

```

to decrease the space before and (in a similar way) after `verbatim` environments. But one time `text` happened to fill up the previous line and the end of line marker started a new line. When `\vspace` is used within a paragraph, it inserts space after the current line (using `\vadjust`). In this case that meant after the current empty line, which gave me too much vertical space. One solution is to write `text%`.

Notice the difference between `\vspace` and `\vskip`. If we had written

```
text \vskip -.5\topsep
\begin{verbatim}
```

then the `\vskip` would have finished the paragraph, and the space would have been removed (K, 99). That would have started a new paragraph, however.

- Spaces are ignored in the following situations.
  1. In vertical mode.
  2. At the beginning of `\halign` constructions like `tabular` and `array`.
  3. At the beginning of input lines (K, 47).

I therefore believe that I can state the following rules.

1. Put a `%` when the line ends with a horizontal command that ends with a nonletter.
2. Put a `%` at the end of a line if spaces are not ignored there and the next line starts with a `\vspace`.

## 10 Spacing

- $29.7\text{cm} = 11.69\text{in}$      $21\text{cm} = 8.27\text{in}$   
 $27.94\text{cm} = 11\text{in}$      $21.59\text{cm} = 8.5\text{in}$   
 $\text{textwidth} = 390\text{pt}$  for both letter and a4  
 $390\text{pt} = 13.76\text{cm} = 5.42\text{in}$
- `\newlength{\test}`  
`\setlength{\test}{\linewidth}`  
`\addtolength{\test}{-3.72in}%`  
`\hspace{\test}`  
`\the\test`  
`\showthe\test`
- The `tabular` environment puts half the intercolumn space before the first column and after the last column (L, 183). This can be removed by `@{}` or be adjusted by `\setlength{\tabcolsep}{.5\tabcolsep}`.
- When using a `\parbox` instead of `tabular`, we must include struts and specify the size of the box.
- Interrow spacing in the `tabular` and `array` environments is obtained by putting the default strut in each row. The default strut has height `.7\baselineskip` and depth `.3\baselineskip`. The commands

```

\begin{tabular}{l}table1\
\begin{tabular}{ll}
table2
\end{tabular}
\[\<glue>]table3
\end{tabular}

```

will add `<glue>` to the depth of the default strut in the second row in the outer tabular. But the inner tabular might already have increased the depth to something bigger, in which case `\[\<glue>]` will have no effect. If we write `\{\}\[\<glue>-\baselineskip>]` we will get the desired spacing.

- The construction

```

\makebox[\textwidth]{%
\makebox[0in][l]{\@nameofcourse}%
\hfill Tutorial #1 \hfill
\makebox[0in][r]{\@academicyear}
}

```

puts the number of the tutorial on the middle of the line, and not halfway between the name of the course and the year. We need `\hfill` since `\makebox[\textwidth]` uses `\hfil`.

- `\hss` has both infinite stretchability and shrinkability so it can put things flush left or right in a box without complaining about an overfull box if the contents exceeds the box.
- Article style sets `\floatpagefraction` to be `.5`. If we try to put a bigger float on a page, we must increase `\floatpagefraction`.
- `\newpage` puts `0pt plus 1fil` at the bottom of the page. The `\raggedbottom` declaration defines `\@textbottom` to be `0pt plus .0001fil`. The reason for the number `.0001` is to make sure that I still have (essentially) `1fil` at the bottom. This is important if I use some other `\fil` command earlier on the page and the write `\newpage`.
- In order to print mailing labels, `\end{letter}` issues a `\pagebreak`. Since `\raggedbottom` is in effect, `\@textbottom` is `0pt plus .0001fil`, so `letter.sty` sets `\@texttop` to be `0pt plus .00006fil`.
- `eqnarray` uses `\halign` to `\displaywidth{...}` while `array` uses `\vcenter{\halign{...}}`.  $\TeX$  is in internal vertical mode when it works on a `\noalign`. A `\noalign` inside an array environment uses the width of the array. `\makebox` and `\mbox` use `\leavevmode`. Hence `\noalign{\hbox{and}}` puts it on the left, while `\noalign{\mbox{and}}` enters horizontal mode, includes a `\parindent` and creates a paragraph, i.e., a box of width `\textwidth`. If this happens inside an array, no centering will take place, since the width of the array is already `\textwidth`. `\makebox` includes horizontal glue, so `\makebox[\textwidth]{and}` works, while `\hbox to \textwidth{and}` gives an underfull `\hbox` error.

## 11 Macro expansion

- The `&`'s and `\`'s serve as delimiters for the `\tabular` macro, so if we have an `&` or a `\` in a conditional inside a macro, we get in trouble. If we include the troublesome parts in braces,  $\TeX$  will not see the `&` or `\` (K, 202), so it won't complain about the `\ifx`'s, but it will mess up my `\tabular`. This might also cause a `\` to appear inside a group. The solution is to write `\edef\tabcontents{...the contents of the table, with noexpand in front of all the "nonconditional" commands...}` and then say

```
\begin{tabular}{...}
\tabcontents
\end{tabular}
```

Then all the "iffy" stuff gets expanded in peace, and then put inside the macro.

- If `\test` is defined, `\csname test \endcsname` calls `\test`. If `\test` is undefined, `\csname test \endcsname` defines `\test` to be `\relax`.
- $\TeX$  sets `\catcode'@=12`, so if we write `\def\@test{a}`  $\TeX$  will think that we are trying to redefine `\@` to be a macro with parameter text `test`. If we write `\def\test@at{a}`  $\TeX$  will think that we are trying to define `\test` to be a macro with parameter text `@at`.  $\LaTeX$  allows us to use `@` in style files that are called in the `\documentstyle` command, but not in files that are called by `\input`. We can't use commands with `@` in them in the text (unless we change the category code), but we can use macros that involve commands with `@` in them, because  $\TeX$  uses the category code that was in force when the command was read.
- The command

```
\uppercase{\romannumeral2}
```

prints `ii`. `\romannumeral2` is the argument of the primitive command `uppercase` so it is not expanded before `\uppercase` acts on it, and since it is a control sequence it is not changed by `\uppercase`. The result of

```
\expandafter\uppercase{\romannumeral2}
```

is `ii`. The second token after `\expandafter` is `{`, so `\expandafter` does not have any effect. If we write

```
\uppercase\expandafter{\romannumeral2}
```

we will get `II`. Here the second token after `\expandafter` is `\romannumeral`, so the argument is read and expanded (K, 213) and then transformed by `\uppercase`.

- After

```
\def\up#1{\uppercase{#1}} \def\text{test}
```

the command `\up\text` prints  `test`, while `\expandafter\up\text` prints `Test`. But if we use

```
\def\text{{test}}
\expandafter\up\text
```

we get `TEST`.

- A `\verb` command may not appear in the argument of any other command.

## 12 Modes

Vertical mode generates pages from vertical boxes. Internal vertical mode is used inside a `\vbox`. Pages are not broken in internal vertical mode. Horizontal mode generates paragraphs. Internal horizontal mode is used inside a `\hbox`. Lines are not broken in internal horizontal mode.