We shall see some tricks:

1. Putting two itemizations on the same line,
2. Using footnotes in boxes,
3. Dealing with PDF cut-and-paste and search functionalities, even when you are using accented characters,
4. Making a distinction with overline for propositional logic formulae,
5. Referring to labels by long names.

We will then discuss some problems people often face when dealing with many figures, and, more generally, floats, in reports. Finally, we will deal with a Beamer scheme which aims at using a single file for both a presentation, and the related report.

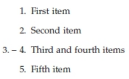
By Luca Merciadri, http://www.student.montefiore.ulg.ac.be/~merciadri/
Download as PDF file

Contents

Putting two itemizations on the same line

Example

Say you want



Code

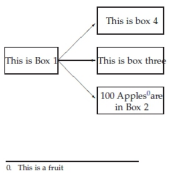
This is achieved thanks to

```
 begin{enumerate}    item First item    item Second item
item[refstepcounter{enumi}labelenumi%      -- refstepcounter{enumi}labelenumi] % Third and
fourth items    item Fifth item  end{enumerate}
```

Thanks to Philipp Stephani (see [7]).

Using footnotes in boxes
Example

Say you want this (note the footnote).



You will notice that a footnote was placed in the bottom of the page, as desired. Generally, generating footnotes from a box is not that easy because of internal footnote limitations. [6]

You might use a different syntax than put and framebox for boxes, but if you want to stick with this syntax, a first solution is to use footnotemark that generates the footnote marking in the text, together with footnotetext{...} that generates the footnote text.

The former needs to be put in the box (so that the number actually appears in the box), when the latter needs to be given outside the box, but on the same page. [6]

Code

This is the approach that I used:

```
 footnotetext{This is a fruit}  addtocounter{footnote}{-1}  {setlength{unitlength}{6mm}
begin{picture}(13,7)(0,0)  put(0,3){framebox(4,2){This is Box 1}}    put(7,0){framebox(5,2){
shortstack{100 Applesfootnotemark are\%  in Box 2}}}  put(4,4){vector(1,1){2.8}}
put(7,3){framebox(5,2){  shortstack{This is box three}}}  put(4,4){vector(1,0){2.8}}
put(7,6){framebox(5,2){  shortstack{ This is box 4}}}  put(4,4){vector(1,-1){2.8}}    end{picture}  }
```

The advantage of this approach is that it is relatively ‘clean’, i.e. you are not obliged to use textsuperscript, be it in the box, or in the footnote, for numbering. Thanks to Heiko Oberdiek at [6]
for this.

Note that many other solutions are available to write diagrams, such as DOT.

PDF compatibility

In our last article, we treated extensively about ‘input encoding’. Here, we will detail two PDF problems: search and copy-and-paste, with their respective solutions. One might generate a PDF document using LaTeX2e, but if this document contains accented characters, searching for words containing accented characters could always fail, depending on the PDF reader.

The first problem is that the default font encoding is OT1. It should generally deal correctly with basic ASCII characters and the PDF operations on it. But once you will use accented characters, it might cause troubles with some PDF readers.

A solution is to use the T1 font encoding, conjointly with the cmap package. As a result, the

```
 usepackage[T1]{fontenc}  usepackage{cmap}
```

set of directives should fix search and copy-and-paste in traditional PDF readers, even for Greek symbols in formulae. [5]

Propositional Logic

Let us express the dual of A as A. The major problem using overline is that if you write things like AB (overline{A}overline{B}), they look exactly the same as

AB
(overline{AB}).

Or the latter is equal to A + B by de Morgan's rules, and thus differs from the former on a truth table. So, a distinction needs to be made. Enrico Gregorio gave me the solution at
[1]
. Implement

```
 newcommand{closure}[2][3]{{}%  mkern#1muoverline{mkern-#1mu#2}}
```

and then write

```
 closure{A}closure{B}
```

and compare it to

```
 closure{AB}
```

The respective results are

$\overline{A}\,\overline{B}$ and $\overline{AB}$

One can now see the difference. This command accepts one optional argument and one mandatory argument. This command works this way (thanks to Claudio Beccari for a detailed explanation):

1. it advances by #1mu (#1 is the optional number); mu is the special math length unit that is equivalent to 1/18th of 1em of the current font, so that it obeys the different sizes used in math for the main formula, and for the first- and second-order sub- and super-scripts,
2. it overlines an argument made up of the following items,
3. a negative math kern of #1mu, in order to get back by the same amount it advanced in step 1,
4. the second argument represented by #2.

The 'step forward – step back' procedure is used because the overline is a little shorter than it would be if the step back step was absent; therefore closure{A}closure{B} do not have touching overlines and produce a different result than closure{AB}.

Of course, the optional amount of kerning may depend from the real obligatory argument. For example, arguments that fill their bounding box such as H, M, N, Z, T, I, would maybe require 3mu, while, for arguments that do not fill their bounding box, such as A, O, S, C, and the like, 2mu or 1.5mu might be better values. The optional argument to closure, such as in closure[2]{A}, is thus meant to correct possible ‘errors’ due to letter form.

Referring to labels by long names

Code

Say you want to label{} e.g. an enumerate’s environment’s item. You then have, for example,

```
 begin{enumerate}    item First item,    item Second item.  end{enumerate}
```

Now, one wants to say ‘see Item 1’. This can be achieved using

```
 begin{enumerate}    item First item, label{item:firstitem}    item Second item.  end{enumerate}
```

and then, ‘see Item ref{item:firstitem}’ somewhere in the text. This is the traditional approach to the ref and label commands. But now, how does one manage if he wants to write ‘see Item ref{item:firstitem}’ and wants ‘see Item [name]’ to appear, where [name] is an attribute of label, for example defined like

```
 begin{enumerate}    item First item, mylabel{item:firstitem}{%      Fruits and Co.}    item Second item.  end{enumerate}
```

where here, name is‘Fruits and Co.’? He could then have, in the output, ‘see Item Fruits and Co.’

For this, I did not try using zref, but used Mr. Oberdiek’s trick ([4]):

```
 makeatletter  newcommand*{mylabel}[2]{%   @bsphack   begingroup   def@currentlabel{#2}%      label{#1}%   endgroup   @esphack  }  makeatother
```

Example

This gives

1. First item,
2. Second item

and a reference to the first item is ‘Fruits and Co.’

Note also that the nameref package exists, which solves the closely-related problem of e.g. including the name of a section when cross-referencing.

Float management

Many LaTeX-beginners are often frustrated. This happens especially when, after compilation, their figures, tables, and, more generally, floats, appear at incongruous places. For example, one might already have a ‘sufficient’ number of floats on one page, the following floats being moved to the following page(s).

On a content-oriented approach, it is, effectively, sometimes disturbing, be it to the reader, or to the writer, to look at such behaviors without any clue on how to deal with this placement.

Before trying anything, an important thing is to understand, briefly, why and how LaTeX manages to put floats this way. It is typographically disadviced to put too much images or tables on one page, or, more generally, in the ‘main matter’ of a document. If you need to put so much images in your document, ask yourself some questions:

- Is it a good way to present things?
- Shouldn’t I select the most important images and tables?
- Couldn’t I put some images and tables on specific pages only, or in the appendices?
- ...

Once you have clear and responsable answers to these questions, you might still need to fine tune the placement of your floats.

If you need to put floats on specific float pages, in the ‘main matter’ of your document, you might use the [p] option after the beginning of your environment: figure, table, etc.

However, typography rules sometimes go against your rules, i.e. the way you (want to) present things. If you know what you are doing, you might simply, sometimes, feel the need to include a float at a place where LaTeX does not want it to be at all.

If you tried even with the [h!] option, meaning ‘put this float here!’, to no avail, you might consider modifying some internal parameters, because putting an exclamation mark in the list of placement options makes LaTeX ignore all the constraints but topfraction, bottomfraction, and the ones with floatpage in their names. [8]

Once again, ask you questions:

- Are there too much floats on my page?
- Is my float too big?
- Where am I including my float?

If your float is too big, either you resize it until it comes on the desired page, or you let it where LaTeX puts it. But if there are too many floats, you need to know that LaTeX defines a limit on the number of floats by page.

Acceptable parameter modifications might be found at [8] :

% See p.105 of "TeX Unbound" for suggested values. % See pp. 199-200 of Lamport’s "LaTeX" book for details. % General parameters, for ALL pages: renewcommand{topfraction}{0.9} % max frac of fl. at top renewcommand{bottomfraction}{0.8} % max frac of fl. at bot. % Parameters for TEXT pages: setcounter{topnumber}{2} setcounter{bottomnumber}{2} setcounter{totalnumber}{4} setcounter{dbltopnumber}{2} % for 2-column pages renewcommand{dbltopfraction}{0.9} % fit big float above 2-col. text renewcommand{textfraction}{0.07} % allow minimal text w. figs % Parameters for FLOAT pages: renewcommand{floatpagefraction}{0.7} % require fuller float pages % N.B.: floatpagefraction MUST be less than topfraction! renewcommand{dblfloatpagefraction}{0.7} % require fuller float pages % remember to use [htp] or [htpb] for placement

If you find that liberal values of the float parameters still are causing trouble, you can try forcing a float page with clearpage to disgorge the accumulated blockage. If you do not want to force a pagebreak, use the afterpage package and tell LaTeX afterpage{clearpage}, which should force a float page when the current page comes to an end. If floats continue to pile up at the end, you probably have one too big to fit on a page; try reducing its size. [8] In extreme cases, you might consider using the float package.
[8]

LaTeX mechanisms are sufficiently well programmed so that, under normal use, they do not cause you too much trouble with floats, i.e. floats are not rejected too far in the document. In my reports, where I often need to include graphics, I generally encounter no problem, without any specific tuning.

Most problematic cases are linked to the question

Do I need to let LaTeX put my floats e.g. two pages after where they are included, between some paragraphs which have absolutely no link with these figures, or should I manage to put my figures differently?

The answer to this question is that you should normally consider using pages for floats, or at least manage to put figures in pages whose context is linked to the appearing figures. If a float is rejected on the following page, it is generally not too much shocking for the reader, and it might even be a good thing if you prefer this way to present things. But if your floats are rejected further, it generally becomes disturbing for the reader, who needs to look for figures through your document.

Beamer and articles

When you use the beamer class, you can evidently produce slides (we will call this the presentation), e.g. for a conference, but you can also produce a handout.

Generally, a handout is considered to be a scaled version of the slides, so that more than one slide fits on the page. This can generally be achieved using PDF tools, and this is not our concern here.

The case we want to treat here is the one where you write a presentation, and want to write a report too, both being related to the same content. It might be interesting if the presentation you are writing mainly consists of a summary of your research.

If you want to reuse your beamer presentation to write the article, or the contrary, you can do this in a very simple way, thanks to Mr. Knudsen explanations at [3] , and to [2] . Consider that you have three files:

- content.tex
- presentation.tex
- report.tex

Then, the idea is to put the whole content (i.e. text that will either be shown in the presentation, or in the report, the ‘or’ being inclusive) in content.tex. That means that all your content will be put in this file. This file does not need to have any preamble, because it will be included by both report.tex and presentation.tex.

Now, we need to specify how your content will be directed to the presentation, the report, or both. There is a quick way for this: mode<presentation> for presentation mode, and mode<article> for article mode. That means that you will have a file content.tex like

```
 section{First Section}  mode<article>  { % Appears in the article }  frame {    frametitle{My Frame}    % Appears in both versions  }  % Appears in both versions  % (except if specified differently)
```

You can then also use the mode command in the frame content e.g. to scale differently images. Say you are in a frame. You can then use

```
 begin{figure}[h]  centering  mode<presentation> {     includegraphics[scale=0.2]{img.eps} % small  }  mode<article> {    includegraphics[scale=1.5]{img.eps} % bigger  }  caption{Appears in both versions.}  end{figure}
```

The report.tex file will be written using e.g.

```
 documentclass{article}  usepackage{beamerarticle}  begin{document}  input{content.tex}  end{document}
```

The presentation.tex file will follow habitual rules:

```
 documentclass{beamer}  begin{document}  input{content.tex}  end{document}
```

By using this scheme, your content will only reside in content.tex and it could be an advantage, for two main reasons:

- You have only one file for the content, for both versions,
- If you use many code snippets, in your report, from your presentation, content modification (e.g. modifying numbers, a caption, a title, . . .) is centralized, and, as a result, you only modify things once. This allows extra consistency between your report and your slides.

References

[1] Enrico Gregorio. *Appunti di programmazione in LaTeX e TeX*, 2009. Second edition. http://profs.sci.univr.it/~gregorio/introtex.pdf
.

[2] HAPPYMUTANT.COM. *A Quick & Dirty Guide to LaTeX – Making LaTeX Beamer Presentations* , 2010. http://happymutant.com/beamer/ .

[3] Torben Knudsen and Luca Merciadri. *Including only frames in the beamer* (comp.text.tex discussion), 2010.

[4] Lars Madsen, Heiko Oberdiek, and Luca Merciadri. *Using the ref and label commands so that ref points to the label but with a user-given name* (comp.text.tex discussion), 2010.

[5] Günter Milde and Luca Merciadri. *Font encodings and PDFs*, 2010.

[6] Heiko Oberdiek and Luca Merciadri. Footnotes in boxes (comp.text.tex discussion), 2010.

[7] Philipp Stephani and Luca Merciadri. *Enumerate package: how to put two itemizations on the same line?* (comp.text.tex discussion), 2010.

[8] Andrew T. Young. *Controlling LaTeX floats*, 2010. http://mintaka.sdsu.edu/GF/bibliog/latex/floats.html .