More Advanced LATEX

How to do ‘difficult’ things

Tuesday 21st July 2009 10:00–12:30

1 Initial Setup

1.1. Make sure that you can run MikTEX on the computer that you are logged in to:

(a) for TEXnicCenter users: you should make sure that you can run both pdfLATEX and XƎLATEX. If you don't have a XƎLATEX profile, copy the pdfLATEX one and modify the executable specified from `pdflatex.exe` to `xelatex.exe`, leaving everything else alone.

(b) for command-line users: Cygwin should simply allow you to run the various executables (such as `tex` and `latex`) directly from the command-line. If you get stuck at an input prompt, such as ‘`**`’, typing `\endinput` or `\bye` might help you get out of it; a ‘`?`’ prompt should exit given `x` or `q`.

(c) for GNU Emacs users: you're on your own; unfortunately, the AUCTEX mode[1] for editing and processing LATEX is not installed. It's possible that a shell within Emacs has the right paths, or not. Good luck!

1.2. One thing that will probably cause annoyance at some point is that viewing an output document in the Adobe Acrobat PDF viewer prevents that document from being overwritten by LATEX. If you feel confident, try producing a PDF document from LATEX in your `public_html` subdirectory of your home directory on `igor`, which should be mounted as the `G` drive, and viewing the output document through a web browser.

2 Graphics

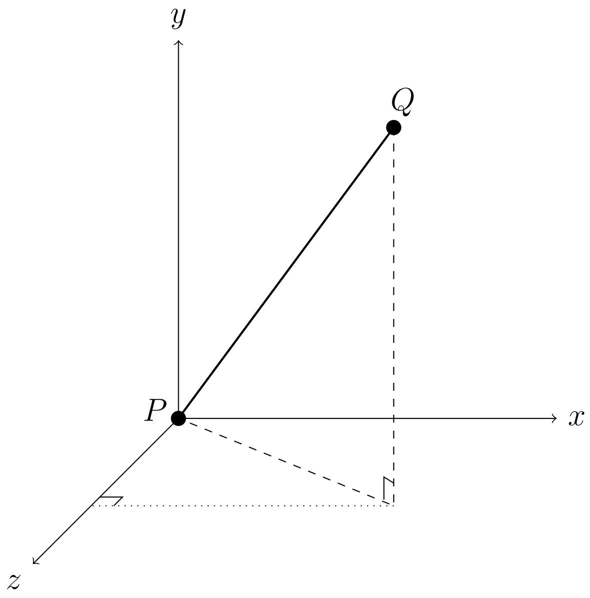
2.1. This part explores the capabilities of Ti*k*Z for the construction of graphics.

(a) Generate a document containing a plot of the function $f(x) = x^2$ for x from -10 to +10. Label your axes appropriately.

(b) A set of data files tabulating the Sound Pressure Levels for equal loudness at different audio frequencies is provided. Generate a graph illustrating these data.

(c) Ti*k*Z can also do ‘three-dimensional’ diagrams, simply by giving points that need it a third coördinate . Below is a diagram illustrating the Euclidean distance between P and Q in three-dimensional space; have a go at reproducing it.

[1] Highly recommended!



3 Fonts

3.1. This part explores the capabilities of pdfLaTeX:

(a) Using pdfLaTeX create a document with a number of paragraphs, setting each in a different font shape or weight (*e.g.* italic, roman, sans-serif, bold, small caps, oblique, typewriter). You may wish to generate or otherwise acquire some filler text; an Internet search for "lorem ipsum" may prove helpful.

(b) Examine the effects of using the utopia, charter and times LaTeX packages in your document.

3.2. This part explores the capabilities of XeLaTeXand the fontspec package:

(a) First, take your document from part 3.1, and try running it through XeLaTeX with and without the various font packages. Do any of them work?

(b) The fontspec package allows XeLaTeX to work with any installed font. Usually, using such a font is as straightforward as using the fontspec package in the document preamble, and then defining the main font for a document with, for example, `\setmainfont{Georgia}`[2]. Try setting the main font of your document to a different family.

(c) Selecting a font within a document is done with the `\fontspec` command: for example `\fontspec{Arial}`. Play a bit with setting portions of your document to different fonts.

[2] The Georgia font is one of the 'core fonts for the web' originally distributed by Microsoft. It also happens to be one of the fonts in Goldsmiths' house style guidelines.

(d) The fontspec package allows accessing advanced features of OpenType fonts, such as alternate forms, contextual swashes, and rare ligatures. There are at least two OpenType fonts with interesting features installed on the lab machines: Linux Libertine O, which is a freely-available font, and Warnock Pro from Adobe. Try setting your document in those fonts, giving the `\fontspec` command options such as:

- `Numbers=Oldstyle` and/or `Numbers=SlashedZero`
- `Ligatures=Rare` and/or `Ligatures=NoCommon`
- `UppercaseSmallCaps`
- `Contextuals=Swash`
- `Variant=1`

4 Packages and Classes

4.1. A minimal 'package' or style file consists of a `\NeedsTeXFormat` declaration and a `\ProvidesPackage` line:

(a) Create such a minimal package file;

(b) Write a document that uses your package file. Verify that using your package has no effect, other than writing out a diagnostic (and logging it in the `.log` file).

4.2. A minimal class file, as well as a `\NeedsTeXFormat` declaration and a `\ProvidesClass` line, needs to define four things: values for `\textheight` and `\textwidth`, a value for `\normalsize`, and a style for pagenumbering:

(a) Create such a minimal class file[3];

(b) Write a document using your minimal class file. What normal LaTeX features do not work?

4.3. One convention for LaTeX software is not to develop 'bare' class or style files as above, but to write "documented LaTeX": interspersing the LaTeX macro definitions with user documentation and implementation notes, and using the doc and docstrip packages to produce the printable documentation and the LaTeX files from a single source file. The format of these files is slightly tricky; to start with, look at the `esexamreport.dtx` and `.ins` files provided:

(a) the `esexamreport.ins` file is already almost in the correct form for use: however, all instances of '`esexamreport`' need to be changed to match the filename you have saved your minimal class file under, and of course the copyright declarations need to be modified. Go ahead and do this.

[3] If you get stuck, look for the `minimal.cls` file, distributed with LaTeX.

(b) the `esexamreport.dtx` (for "documented TEX") file illustrates the usual form: the header of the file contains some gymnastics motivated by the fact that multiple different programs will be reading this file. There is then a short ordinary-looking LATEX document using the `ltxdoc` class, which inputs the file itself; following this, there is what looks like the body of a LATEX document, commented out except for the pieces of code that make up the class or style file (and these pieces of code are within `macrocode` environments). Borrowing liberally from this example, convert your minimal class file to documented TEX format.

(c) if all has gone well, you should be able to run LATEX on your `.dtx` file to generate documentation, and LATEX on the `.ins` file to generate the class file. Check that this works, and if not, fix any problems that are revealed.

Further Reading

- The LATEX3 Project, *LATEX2ε for class and package writers*
- Frank Mittelbach, *The* doc *and* shortvrb *Packages*
- Scott Pakin, *How to Package your LATEX Package*
- Will Robertson, *The* fontspec *Package*
- Till Tantau, *The TikZ and PGF Packages: Manual for version 2.00*