Github Repository

My LaTeX 2_ε template documentation

J. Langedijk

Summary. *The features of my custom LaTeX 2_ε template are documented here for reference.*

Contents

---

1 Introduction and motivation

After writing for 3+ years in LaTeX 2_ε, I still found myself often jumping inbetween projects to copy certain "complex" pieces of code that I needed, like fancy looking code blocks that I used once in an older project or some obscure `usepackage` I need. Thus, building a clearly documented template with custom environments and commands would solve this issue; everything I need while writing is now instantly at my fingertips!

It is highly recommended to compile documents made with this template with LuaLaTeX to avoid the malfunctioning of certain `usepackages`. pdfLaTeX is quite primitive on its own anyways. Luckily, TeXlive-full comes with LuaLaTeX out of the box. Furthermore, *Overleaf* works on TeXlive, so with a simple setting switch you can use LuaLaTeX there, too!

2 Commands

The template comes with a set of new useful commands.

2.1 Document property variables

A few *document variables* are available, embedded in the output PDF as metadata, as well as used to fill certain fields in the template.

- `\Title` prints the title of the document.
- `\Author` prints the author(s) of the document.
- `\Institute` prints the institute associated with the document, like a university or company.
- `\Date` prints the date related to the document, like a deadline for a report.
- `\Docsummary` prints the summary of the document.

All these variables are declared at the top of the template file, and are supposed to be altered as desired.

2.2 The `now` command

The `\now` command prints the current date and time using `\datetime` and `\today`. This document is compiled at `\now`: 4th September, 2022, 23:38.

3 Environments

The template defines some environments, that safe some precious time of setting up styles and nesting environments.

3.1 The `code` environment

The `code` environment renders a nice box with title and label to throw code in. For now, it does **not** support to import code from other files; it expects code to be nested in the environment. The environment has three arguments:

```
\begin{code}{<title>}{<language>}{<label>}
<code>
\end{code}
```

Example 3.1. *Imagine you are writing a tutorial on GNU Octave, the free and open source alternative to MATLAB by the GNU Foundation. You want to embed a Octave script to show how to plot two functions, $f(x) = x^2$ and $g(x) = \sin(\cos x)$, in a figure for a fixed domain. You now only need to write:*

```
\begin{code}{Plotting two functions in GNU Octave}{octave}{lst:2func}
clear vars;close all; clc;
%% VARS
x1 = -1:1e-3:1;
x2 = -2:1e-3:2;
%% FUNCS
y1 = x1.^2;
y2 = sin(cos(x2))
%% PLOT
figure(1)
hold on; grid on;
plot(x1,y1);
plot(x2,y2);
xlabel(x);
ylabel(y);
\end{code}
```

When compiling this LaTeX 2_ε code from the template, the output PDF renders

Listing 3.1: Plotting two functions in GNU Octave

```
1 clear vars;close all; clc;
2 %% VARS
3 x1 = -1:1e-3:1;
4 x2 = -2:1e-3:2;
5 %% FUNCS
6 y1 = x1.^2;
7 y2 = sin(cos(x2))
```

```
%% PLOT
figure(1)
hold on; grid on;
plot(x1,y1);
plot(x2,y2);
xlabel(x);
ylabel(y);
```

which looks quite nice! Using `\ref` *we can refer to the code block using the label: 3.1.*

Parameters other than the provided arguments can ofcourse be tuned manually in the template preamble itself.

3.2 The `plot` environment

The `plot` environment is simply an alias to a *PGFplot* `axis` enviroment, which needs to be initiated inside a `tikzpicture` evironment. This on its own does not allow as much flexibility as a `figure` environment, so `plot` also nests this whole structure inside a figure with `H` float. The syntax now becomes very simple:

```
\begin{plot}{<caption>}{<label>}
[<pgfplotssset settings to add/override defaults>]
<contents of the PGFplot axis object>
\end{plot}
```

It is assumed one knows how to use PGFplots when reading this; if not, go to the *Comprehensive TEX Achive Network* for the full documentation.

Example 3.2. *Plots generated by GNU Octave look fine, but not stellar. The output of Listing 3.1 can look much nicer if we do the whole thing in PGFplots in LaTeX 2ε natively. Let's use the* `plot` *environment for that. Listing 3.2 shows the code that plots the two functions. The plot is visible in Figure 1.*

Listing 3.2: PGFplot code to draw the two function in a subset of $\mathbb{R}^2$

```
\begin{plot}{The two functions of Listing \ref{lst:2func}}{fig:2func}
  [clip=false,
  xmin=-2,
  xmax=+2,
  axis lines=middle,
  xlabel=$X$,
  ylabel=$Y$,
  grid style=dashed]
  \addplot[samples=50,domain=-1:1,blue]{x^2}
  node[right,pos=0.95]{ $f(x)=x^{2}$};
  \addplot[samples=50,domain=-2:2,red]{sin(deg(cos(deg(x))))}
  node[right,pos=0.95]{ $g(x)=\sin{(\cos{x})}$};
\end{plot}
```

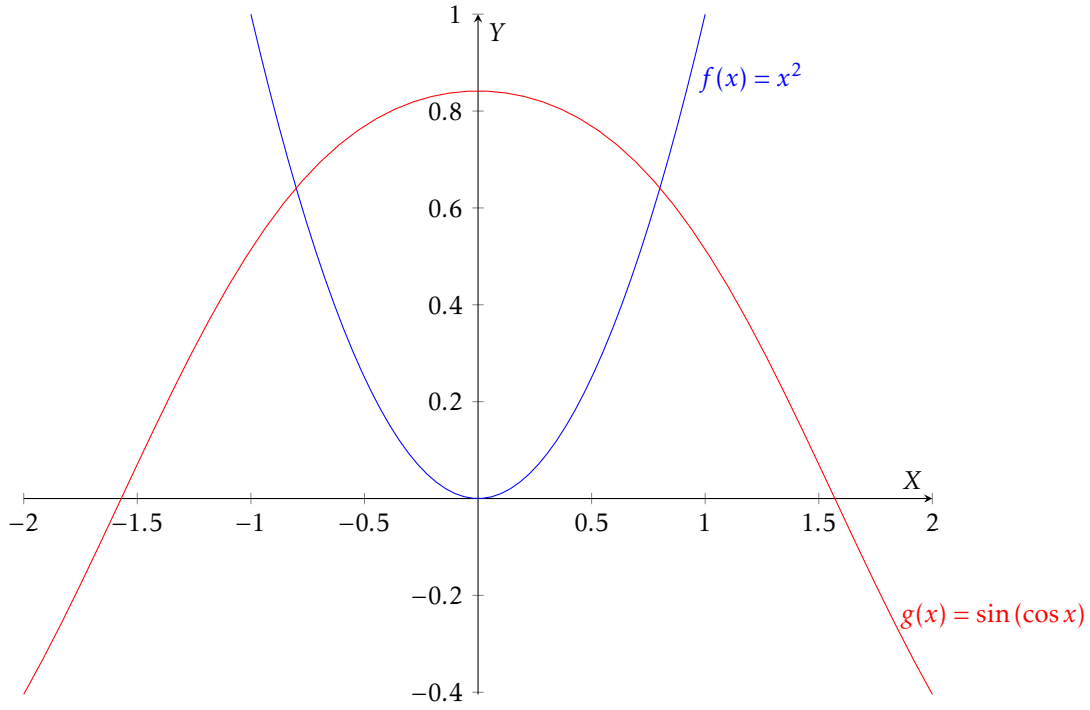

Figure 1: The two functions of Listing 3.1

Appendices

A The template file

The template file can be compiled as a LaTeX 2_ε file as standalone and thus be used as a startpoint for any new projects. Below code functions as a offline reference to the template to see how things work, or just to grab some ideas from it. The full source can be found on the corresponding Github repository, found on this documents title page.

Listing A.1: Template

```
%% PREAMBLE
    ------------------------------------------------------------------------

% --> Document Variables
\newcommand{\Title}{TITLE}      %Document title
\newcommand{\Author}{AUTHOR}    %Document author
\newcommand{\Date}{\today}      %Document date
\newcommand{\Institute}{INSTITUTE}  %Document institute
\newcommand{\Docsummary}{       %Document summary
This is the summary of the contents of this document
}
% --> Setting Document Class, Input, Language and Time
\documentclass[leqno]{article}  %`leqno` places equation tags to the
    left
\usepackage[utf8]{inputenc}     %Unicode Support
\usepackage[english]{babel}     %Language
\usepackage[
    long,                       %Long format date
    nodayofweek,                %Don't show day of week
    level,                      %No clue
    24hr                        %24h format
    ]{datetime}                 %Set date settings
\newcommand{\now}{\today, \currenttime} %Print date+time
% --> Fonts
\usepackage{kpfonts}            %Font, supported by TeXlive
\usepackage[T1]{fontenc}        %Magic thing to get the font working
% --> Header and Footer Styles
\pagestyle{headings}            %Headings page number style
\usepackage{lastpage}           %Last page
% --> Multicolumns
\usepackage{multicol}           %Multi-Columns
% --> TiKz Things
\usepackage{tikz}               %TiKz
\usepackage{pdfpages}           %PDF importer, at the wrong spot
\usetikzlibrary{positioning}    %Positioning TiKz Library
% --> Table Things
```

```
\usepackage{tabularx}        %TabularX, to make more sexy tables
\usepackage{booktabs}        %Booktabs in tables
% --> AMSTeX Things
\usepackage{amsmath}         %All the AMSMath things
\usepackage{amssymb}         %Cute symbols
\usepackage{amsthm}          %Theorems
% --> Graphics and Figures
\usepackage{xcolor}          %Fancy colors
\usepackage{geometry}        %Page geometry settings
\usepackage{graphicx}        %Graphics support
\usepackage{float}           %Float support
\usepackage{caption}         %For subcaptions in subfigures
\usepackage{hologo}          %Fancy Logos
% --> Graphs and Plots
\usepackage{pgfplots}        %Plotting
\pgfplotsset{                %Defaults
  compat=1.18,               %Compatibility
  clip=false,                %Do not clip plots
  width=0.9\textwidth,       %Plot width
  height=0.7\textwidth,      %Plot height
  axis lines=middle,         %Axis style
  xlabel=$x$,                %xlabel
  ylabel=$y$,                %ylabel
  grid style=dashed          %Grid style
  }
\newenvironment{plot}[2]
  {
  \newcommand{\plotcap}{#1}
  \newcommand{\plotlab}{#2}
  \begin{figure}[H]
    \centering
    \begin{tikzpicture}
    \begin{axis}
  }
  {
    \end{axis}
    \end{tikzpicture}
  \caption{\plotcap}
  \label{\plotlab}
  \end{figure}
  }
% --> Referencing and `hyperref` Settings
\usepackage{hyperref}        %For cross-referencing, links, forms and
    more
\hypersetup{
    colorlinks=true,         %Links have colors
    linkcolor=cyan,          %Link color
    filecolor=magenta,       %File Color
    urlcolor=cyan,           %URL Color
```

```
  pdftitle={\Title, \Author},    %Embedded PDF Title
  pdfauthor={\Author},           %Embedded PDF Author
    pdfsubject={\Title},         %Embedded PDF Subject
    }
\renewcommand\LayoutCheckField[2]{#1\hfill #2}
% --> Listings and Code
\usepackage{listings}          %Code Listings
\lstdefinestyle{mystyle}{
  backgroundcolor= {},         %Transparent background
  commentstyle=\color{green},  %Comment color
  keywordstyle=\color{magenta}, %Keyword color
  numberstyle=\tiny\color{black}, %Line number color
  stringstyle=\color{purple},  %String color
  basicstyle=\ttfamily\small   %->
  \color{black},               %Font style
  breakatwhitespace=false,     %Do not break at whitespace
  breaklines=true,             %Break lines
  captionpos=b,                %Add caption below
  keepspaces=true,             %Keep spaces in code
  numbers=left,                %Line number position
  numbersep=5pt,               %Line number separation
  showspaces=false,            %Show space indicator
  showstringspaces=false,      %Show space indicator in strings
  showtabs=true,               %Show tab indicator
  tabsize=2,                   %Tab size
}
\lstset{style=mystyle}         %Set `mystyle` as default listing style
% --> Tcolorbox and Code Environment
\usepackage{tcolorbox}         %Fancy Boxes around text, e.g. for code
    blocks
\tcbuselibrary{                %Set libraries for `tcolorbox`
  skins,                       %Skins
  breakable,                   %Breakable boxes
  listings}                    %Listing boxes
\tcbset{listing engine=listings}  %Set listing engine
\newtcblisting[
  auto counter,
  number within=section]{code}[3]{  %Code environment
  listing only,                %Only show listing
  fonttitle=\bfseries,         %Title font
  colback=black!5!white,       %Background color
  colframe=blue!20!black,      %Frame color
  title=Listing \thetcbcounter: #1, %Box title
  size=normal,                 %Box size
  bottom=0.1 mm,               %Bottom sep.
  sharp corners,               %Use sharp corners
  breakable,                   %Allow box to break
  label=#3,                    %Label for referencing
  listing options={            %Options for the `listing` package
```

```
    language=#2,          %Programming Language
  }}
% --> Emoji Support (Needs Compilation via LuaLaTex!)
\usepackage{emoji}        %Emoji Support
\setemojifont{TwemojiMozilla}   %Use Mozilla Emojis
% --> Appendices
\usepackage[toc,page]{appendix}   %Appendix usepackage
% --> Miscallaneous
\usepackage{lipsum}       %Generates Lorem Ipsums on demand
\usepackage{paralist}
\usepackage{soul}
\usepackage{parskip}      %Stop auto-indenting
\usepackage{import}       %Import other .tex files for structure
\usepackage{enumitem}     %Custom itemize behavior
\setitemize{noitemsep,
topsep=0pt,parsep=0pt,partopsep=0pt}


%% BEGIN DOCUMENT
    ------------------------------------------------------------------------
\begin{document}

%% TITLE
    ------------------------------------------------------------------------------


\begin{titlepage}
  \vspace*{1cm}
  \Large
    \textit{\Institute}
    \vspace{0.25cm}

  \Huge
    \textbf{\Title}
    \vspace{0.25cm}

  \Large
  \Author
  \vspace{1.0cm}

  \normalsize
  \rule{\textwidth}{0.4pt}
  \newtheorem*{summary}{Summary}
  \begin{summary}
    \Docsummary
  \end{summary}
  \setcounter{tocdepth}{1}
  \tableofcontents
  \vfill
```

```
\rule{\textwidth}{0.4pt}
\end{titlepage}

%% MAIN
    ---------------------------------------------------------------------------


%<Main code for sections, chapters, imports, appendices etc.>

%% END DOCUMENT
    ------------------------------------------------------------------
\end{document}
%%
    ---------------------------------------------------------------------------
```