Contents

1 Preliminaries **2**
1.1 Introduction . . . 3
1.2 Template . . . 4

2 Authoring tricks **7**
2.1 Hand-Written Annotations . . . 8
2.2 Calculations in LaTeX . . . 11
2.3 Figure Wrapping . . . 14
2.4 The `cleveref` Package . . . 16
2.5 Replace the Missing Reference Symbol (??) . . . 18

3 Tables **19**
3.1 Color in Table Elements . . . 20

4 Miscellaneous **21**
4.1 Custom Labels . . . 22

5 Mathematics Typesetting **25**
5.1 Math environment in list . . . 26
5.2 Summation with Limits on the Side . . . 27

6 tikz tricks **28**
6.1 File System Structure . . . 29
6.2 File System Structure 2 . . . 31
6.3 Adding Noise to Plots . . . 34
6.4 Dual Axes in tikz . . . 36
6.5 Tikz Arrow in Sentence . . . 38

Chapter 1

Preliminaries

Introduction

Description

At some point during my LaTeXwriting life, I decided to put together a portfolio of cool tricks and tips, which I had used in the past or wanted to apply i the future. Something like an index or a portfolio.

I wanted that document to consist of multiple small parts, one for each tip, and each part should contain both example code and the complied example. A great deal of searching was done to find a technical solution which would allow for a document which would allow for:

1. indexable, possible multi-page examples
2. examples to be compiled in isolation, free of package conflicts which would surely emerge in such a document
3. a modular structure, with each example be a short .tex file which would be included to the collection

I did a related question a while back on TeX-StackExchange, but there was no good answer.

Finally, I managed to get a satisfactory result with the `pdfpages` package. I put together a template document to be used for each example, compile it and then include the output .pdf to the collection document.

The code pattern of the top-level document can be seen below.

Enjoy!

NOTE: Some of the tricks use the `minted` package to typeset LaTeXcode. This requires the addition of the `-shell-escape` option when building your LaTeXdocument. For example, my build command in a Linux environment is:

```
"/usr/local/texlive/2016/bin/x86_64-linux/pdflatex" -synctex=1 -shell-escape
      -interaction=nonstopmode %.tex
```

WARNING: Compiling LaTeXdocuments with the `-shell-escape` option enabled is considered a security risk. Only use it to compile documents you trust.

Used Packages

`pdfpages`

Code

```
\documentclass[onepage]{book}

\usepackage{pdfpages}

\begin{document}

\tableofcontents

\chapter{Preliminaries}
\includepdf[pages=-, fitpaper=false, addtotoc={1,section,1,Template,sec:template}]{tricks/
    introduction/introduction.pdf}

<other-includes>

\end{document}
```

Check on the next section for the template code of each section.

Template

Description

A uniform template is useful to keep all the tips and examples of this portfolio consistent and searchable. The related code, both for the preamble and the document, as well as the compiled examples are provided in separate sections. For this section, the template code is presented.
To create a new trick:

1. Create a new folder inside the tricks subfolder
2. Copy the code presented in the *Document* subsection into a new `root.tex` document
3. Build that `root.tex` document
4. Add an entry to the `portfolio.tex` overarching document and compile it

Used Packages

`xcolor, listings, url, hyperref`

Preamble

```
\documentclass{article}

\usepackage[a4paper, total={7in, 9in}]{geometry}
\pagestyle{empty}

\usepackage{xcolor} % Required for listings color definitions
\definecolor{Brown}{cmyk}{0,0.81,1,0.60}
\definecolor{OliveGreen}{cmyk}{0.64,0,0.95,0.40}
\definecolor{CadetBlue}{cmyk}{0.62,0.57,0.23,0}
\definecolor{lightlightgray}{gray}{0.9}

\usepackage{listings} % computer code language formatting

\lstdefinestyle{tex-style} {
language=TeX,                       % Code langugage
basicstyle=\ttfamily,               % Code font, Examples: \footnotesize, \ttfamily
%keywordstyle=\color{OliveGreen},    % Keywords font ('*' = uppercase)
commentstyle=\color{gray},          % Comments font
numbers=none,                       % Line nums position
numberstyle=\tiny,                  % Line-numbers fonts
stepnumber=1,                       % Step between two line-numbers
numbersep=5pt,                      % How far are line-numbers from code
backgroundcolor=\color{lightlightgray}, % Choose background color
frame=single,                         % A frame around the code
tabsize=2,                          % Default tab size
captionpos=b,                       % Caption-position = bottom
breaklines=true,                    % Automatic line breaking?
breakatwhitespace=false,            % Automatic breaks only at whitespace?
showspaces=false,                   % Dont make spaces visible
showtabs=false,                     % Dont make tabls visible
columns=flexible,                   % Column format
morekeywords={__global__, __device__} % CUDA specific keywords
}
```

```
\lstset{style=tex-style}

\usepackage[hyphens]{url}
\usepackage{hyperref}
\hypersetup{
colorlinks=true,
citecolor=black,
filecolor=black,
linkcolor=blue,
urlcolor=blue
}
```

Document

```
\documentclass{article}

\usepackage[a4paper, total={7in, 9in}]{geometry}
\pagestyle{empty} % Prevent relative page numbers

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Default definitions
\input{../default-setup}


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble


\begin{document}

\section*{<example-title>}

\subsection*{Description}
<a-few-words-on-the-example>

\subsection*{Sources}
\textit{<add reference sources here>}
\url{<or paste a url>}

\subsection*{Used Packages}
\verb|<state-the-required-packages-here|

\subsection*{Preamble}
\begin{latex}
<paste-the-example-related-preamble-code-here>
\end{latex}

\subsection*{Document}
\begin{latex}
```

```
<paste-the-document-code-here-for-presentation>
\end{latex}

\subsection*{Result}
<write-the-example-document-code-here-for-compilation>

\end{document}
```

Chapter 2

Authoring tricks

Hand-written annotations

Description

Steven B. Segletes has shown a beautiful way to represent hand-written annotations over a text, with arrows and variable orientation.

However, a severe limitation is that these notes float at specified coordinates over a page and will not follow the text during pagination.

Note: My system didn't have the *emerald* font installed. I used the instructions found in
https://tex.stackexchange.com/questions/79713/install-emerald-font-package-on-ubuntu
to install it. This official link
https://www.tug.org/fonts/fontinstall-personal.html
may provide some additional context.

Sources

https://tex.stackexchange.com/questions/317190/handwritten-comments-and-annotations-in-margin?newsletter=1&nlcode=544695%7c921f

Used Packages

lipsum, stackengine, scalerel, everypage, xcolor, emerald, rotating, fontenc

Preamble

```
\usepackage{lipsum} % Used to create dummy text
\usepackage{stackengine} % Highly customised stacking of objects, insets, baseline changes,
    etc
\usepackage{scalerel} % Constrained scaling and stretching of objects
\usepackage{everypage} % Provide hooks to be run on every page of a document
\usepackage{xcolor} % Driver-independent color extensions for LaTeX and pdfLaTeX
\usepackage{emerald} % The used annotation font
\usepackage{rotating} % Rotation tools, including rotated full-page floats
\usepackage[T1]{fontenc} % Standard package for selecting font encodings


\def\PageTopMargin{1in}
\def\PageLeftMargin{1in}
\newcommand\atxy[3]{%
  \AddThispageHook{\smash{\hspace*{\dimexpr-\PageLeftMargin-\hoffset+#1\relax}%
      \raisebox{\dimexpr\PageTopMargin+\voffset-#2\relax}{#3}}}}
\newcommand\handnote[6][0]{\atxy{#2}{#3}{\rotatebox{#1}{\parbox[t]{#4}{%
        \raggedright\fontfamily{fts}\selectfont\color{#5}#6}}}}
\newcommand\handxform[7]{% SCALING IS EMPLOYED TO EXPAND THE USEFUL RANGE OF #7
  \raisebox{#1}{%
    \scalebox{4}[#6]{% THE 4X SCALING IS LATER COUNTERED BY 0.25
      \raisebox{#2}{%
        \rotatebox{#3}{%
          \stretchto{%
            \rotatebox{#4}{%
              \char#5}%
          }{0.25\dimexpr#7\relax}% THE 0.25 IS COUNTERED BY THE EARLIER 4X SCALING
        }%
      }%
    }%
  }%
}
\newcommand\handline          [2][1]{\handxform{.20ex}{0.30ex}{-90}{ 12}{47}{#1}{#2}}
\newcommand\handrightarrow    [2][1]{\handxform{.55ex}{0.45ex}{-90}{ 90}{62}{#1}{#2}}
\newcommand\handleftarrow     [2][1]{\handxform{.55ex}{-0.5ex}{ 90}{ 90}{62}{#1}{#2}}
\newcommand\handhookrightarrow [2][1]{\handxform{.70ex}{0.60ex}{-90}{-30}{35}{#1}{#2}}
\newcommand\handhookleftarrow [2][1]{\handxform{.50ex}{0.60ex}{-90}{ 30}{36}{#1}{#2}}
\newcommand\handstealthrightarrow[2][1]{\handxform{.50ex}{-0.2ex}{ 90}{ 80}{60}{#1}{#2}}
\newcommand\handstealthleftarrow [2][1]{\handxform{.50ex}{0.40ex}{-90}{ 80}{60}{#1}{#2}}

\newcommand\handyform[7]{\raisebox{#1}{\scalebox{#6}[1]{%
     \rotatebox{#3}{\stretchto{\rotatebox{#4}{\raisebox{#2}{\char#5}}}{#7}}}}}
\newcommand\handuparrow       [2][1]{\handyform{.10ex}{-.08ex}{ 0}{ -0}{91}{#1}{#2}}
\newcommand\handdownarrow
```

Document

```
\begin{minipage}[c]{0.7\linewidth}
  \lipsum[1-2]
\end{minipage}

\handnote[1]{15cm}{3cm}{0.25\linewidth}{red}{\lipsum[4]}

\handnote[4]{1cm}{5cm}{3cm}{blue}{I have a really short comment}
\handnote[4]{3.5cm}{5cm}{2cm}{blue}{\handrightarrow[1.7]{4ex}}

\handnote[0]{5cm}{6.8cm}{9cm}{blue}{%
  \handline[2]{7ex} \handline[2.5]{6ex} \handline[2]{5ex}}%

\handnote[-75]{2cm}{10cm}{3cm}{red}{\handhookleftarrow[2.1]{6ex}}
\handnote[20]{3cm}{11cm}{2cm}{red}{This is a relatively short comment
  to display text rotation}

\handnote[89]{1cm}{10cm}{10cm}{red}{This is a vertical note}

%
\handnote[0]{6.5cm}{12cm}{10cm}{red}{{\Large ACCENTS:}\\%
  handlines: \handline{6ex}\handline[3]{12ex}\\
  handrightarrows: \handrightarrow{4ex}\handrightarrow[2]{5ex}\\
  handleftarrows: \handleftarrow{4ex}\handleftarrow[2]{5ex}\\
  handhookrightarrows: \handhookrightarrow[1]{4ex}\handhookrightarrow[2.1]{7ex}\\
  handhookleftarrows:\handhookleftarrow[1]{5ex}\handhookleftarrow[2.1]{9ex}\\
  handstealthleftarrows: \handstealthleftarrow{5ex}\handstealthleftarrow[2]{5ex}\\
  handstealthrightarrows: \handstealthrightarrow{5ex}\handstealthrightarrow[2]{5ex}\\
  handuparrows \handuparrow{2ex}\handuparrow[1.2]{4ex}\\
  handdownarrows \handdownarrow{2ex}\handdownarrow[1.2]{4ex}}
```

Result

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac ~~turpis~~ egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

I have a really short comment

This is a vertical note

This is a relatively short comment to display text rotation

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

ACCENTS:

handlines: ——— ———

handrightarrows: $\rightarrow$ $\rightarrow$

handleftarrows: $\leftarrow$ $\leftarrow$

handhookrightarrows: $\hookrightarrow$ $\hookrightarrow$

handhookleftarrows: $\hookleftarrow$ $\hookleftarrow$

handstealthleftarrows: $\leftarrow$ $\leftarrow$

handstealthrightarrows: $\rightarrow$ $\rightarrow$

handuparrows $\uparrow$ $\uparrow$

handdownarrows $\downarrow$ $\downarrow$

Calculations in LaTeX

Description

Programmatic execution and self-containment often require the performance of calculations within a LaTeXdocument.
Two options dominate:
The `calc` package is mostly intended for use in the manipulation of lengths. Be careful on the way it treats real numbers and the implicit unit conversion.
For more involved calculations, the `fp` package provides a greater set of functionality.
Read the raw code for examples of broken code, in cases of syntactical omissions.

Sources

https://tex.stackexchange.com/questions/30081/how-can-i-sum-two-values-and-store-the-result-in-other-variable
https://ctan.org/pkg/fp
https://ctan.org/pkg/calc

Used Packages

`calc, fp`

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{calc}
\usepackage{fp}
\usepackage{tikz}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
\newlength{\mylength}
\newlength{\myotherlength}
\newlength{\templength}
%\newlength{\templength2} % Does not work, number is separated
\newcommand\FPuse[1]{\FPeval{\result}{#1}{\result}} % Used for direct calculations with fp.
    Manipulates the variable \result
```

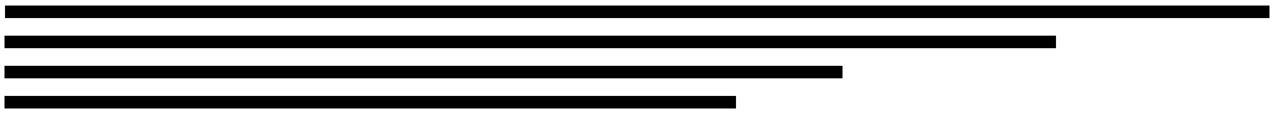
Document

```
  Length manipulation using the \verb|calc| package:

  \setlength{\mylength}{\textwidth}
  \setlength{\myotherlength}{3cm}

  \rule{\mylength}{5pt}

  \setlength{\templength}{\mylength-\myotherlength}
  \rule{\templength}{5pt}

  %\setlength{\templength}{\mylength-2*\myotherlength} % This doesn't work, because the untyped
      2 will lated be assinged a unit type
```

```
\setlength{\templength}{\mylength-\myotherlength*2}
\rule{\templength}{5pt}

\setlength{\templength}{\mylength-\myotherlength*\real{2.5}}
\rule{\templength}{5pt}

~\\
The same commands work inside a tikz environment:\\
\begin{tikzpicture}
\node[draw, anchor=west](n1) at (0,0) {Node 1};
\node[draw, anchor=east](n2) at (\mylength,0) {Node 2};
\draw (\mylength, 0) node[anchor=east] {Node 2};
\setlength{\templength}{\mylength-\myotherlength*\real{2.5}}
\node[draw](n3) at (\templength,0) {Node 3};
\end{tikzpicture}

~\\
Arithmetic and algebra using \verb|fp| package:

\FPset{varA}{10}
varA = \varA

\FPset{varB}{3}
varB = \varB

\FPeval{varC}{clip(varA+varB)}
\varA + \varB = \varC

\FPeval{varD}{round(varA^varB,3)}
\varA \^\ \varB = \varD

\FPmin{\varE}{\varA}{\varB}
min(\varA, \varB) = \varE

\FPlsolve{\varF}{\varA}{\varB}
\varA*x+\varB=0 $\Rightarrow$ x=\FPuse{clip(\varF)}

\FPeval{\varG}{arctan(\varA)}
arctan(\varA) = \FPuse{round(\varG,3)}

~\\
The same functionality can be used within a math environment:
\begin{equation}
10\cdot x + 3 = 0 \Rightarrow x=\FPuse{clip(\varF)}
\end{equation}

~\\
Sadly, the \verb|fp| package requires the full backslash notation (\textbackslash) to define
    variable names for several of its functions. This has the downside that numbers cannot be
    used as part of the variable name, as they cannot be included in a backslashed name.
```

Result

Length manipulation using the `calc` package:



The same commands work inside a tikz environment:

Node 1
Node 3
Node 2

Arithmetic and algebra using `fp` package:
varA = 10
varB = 3
10+ 3= 13
10ˆ 3= 1000.000
min(10, 3) = 3
10*x+3=0 ⇒ x=-0.3
arctan(10) = 1.471

The same functionality can be used within a math environment:

$$10 \cdot x + 3 = 0 \Rightarrow x = -0.3 \tag{1}$$

Sadly, the `fp` package requires the full backslash notation (\) to define variable names for several of its functions. This has the downside that numbers cannot be used as part of the variable name, as they cannot be included in a backslashed name.

Calculations in LaTeX 2: Python Integration

Description

The interfaces of the `calc` and `fp` packages may seem cumbersome to some. Furthermore, their capabilities are definitely limited.
`pythontex` offers Python integration for a LaTeXdocument. A Python shell is invoked and is passed any code contained within the `pythontex` environments. The results can be then accessed from the LaTeXcode.
To use `pythontex`, in this case in a Linux environment:

1. Write your LaTeXdocument, containing any valid `pythontex` code
2. Compile the LaTeXdocument (e.g. `pdflatex root.tex`) to generate `pythontex`-specific code files
3. Run the `pythontex` program on the LaTeXdocument (i.e. `pythontex root.tex`).
   This should read the generated code files and executre all Python commands
4. Compile the LaTeXdocument anew to include the `pythontex` output

Read through the verbatim example code and the result for more information on the usage of the package.
Note 1: Currently `pythontex` does not support a lower build directory (e.g. ./build/output.pdf). All files must be at the same level as the root document.
Note 2: I have not managed to setup an automated build chain of commands within TeXstudio. Calling `pythontex` from within it sometimes produces runtime errors, especially when making `matplotlib` figures, whereas calling `pythontex` manually from the console runs fine. I have not found the cause. I tried calling using an external bash script to run `pythontex` from within TeXstudio but this didn't help either.
Note 3: `pythontex` is not compatible with the listings package. If you need to typeset code listings you can use the `minted` package instead. `minted` calls the `pygments` software, which requires the `-shell-escape` LaTeXcompilation option. This imposes some security risks, so do not use on documents you do not trust.
Note 4: This document uses the `gnuplot` extension of pgfplots, which requires the `-shell-escape` build option.

Sources

https://tex.stackexchange.com/questions/397234/how-to-do-mathematical-programming-in-latex?newsletter=1&nlcode=544695%7c921f
https://www.ctan.org/tex-archive/macros/latex/contrib/pythontex http://tug.ctan.org/tex-archive/macros/latex/contrib/minted/minted.pdf

Used Packages

`graphicx`

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage[gobble=auto]{pythontex}
\usepackage{pgfplots} % Used for plotting within LaTeX
\usepackage{graphicx} % Used to import Python-generated pdfs

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
% Pass LaTeX variables into the Python evironment
\setpythontexcontext{textwidth=\the\textwidth,%
  textheight=\the\textheight%
}
```

Document

```
Python\TeX provides a multitude of environments which allow \LaTeX to interact with Python.

The \mintinline{latex}{\pyb}/\mintinline{latex}{pyblock} environment typesets the passed code into the
↪  \LaTeX document and also executes it. However, it will not print out any results.

\pyb{my_str='hi'}

\begin{pyblock}
  my_str+=' there'
  print('Python says: {0}!'.format(my_str))
\end{pyblock}

You can access the last printout done by Python using the \mintinline{latex}{\printpythontex} command.

\printpythontex

The \mintinline{latex}{\pyc}/\mintinline{latex}{pycode} environment both executes code and typesets
↪  results, but does not typeset the code itself.

\pyc{my_str="calculations"}

\begin{pycode}
  my_str="calculations"
  print("Let's do some {0}".format(my_str))
\end{pycode}

You can use the \mintinline{latex}{\pys}/\mintinline{latex}{pysub} environment to substitute Python
↪  variable values into \LaTeX code, using the \verb|!{...}| format.

In this example, \verb|sympy| expression manipulation is used to generate a function expression, which
↪  is then plotted using \verb|gnuplot|.

\begin{pycode}
  from sympy import *
  x = symbols('x')
  f = integrate(cos(x)*sin(x), x)
\end{pycode}

\begin{pysub}
  \begin{tikzpicture}
  \begin{axis}[xlabel=$x$,ylabel=$y$,samples=200,no markers,title=$!{latex(f)}$]
  \addplot[black] gnuplot {!{f}};
  \end{axis}
  \end{tikzpicture}
\end{pysub}

The \texttt{r} string prefix prevents escaping characters. This can be used to effectively perform
↪  \LaTeX code generation within  a \mintinline{latex}{pycode} block.

\begin{pycode}
  print(r'\begin{center}')
    print(r'\textit{A message from Python!}')
    print(r'\end{center}')
\end{pycode}
```

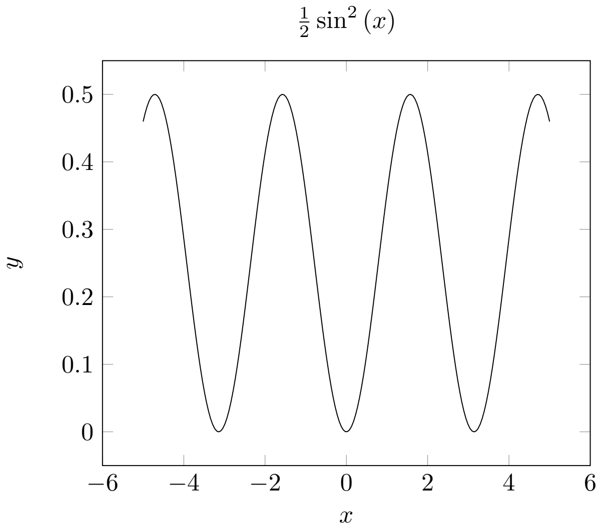
```latex
Now, let us focus on actual calculations done with \verb|pythontex|.

\textbf{Example 1: Typical arithmetic}
\begin{pyblock}
  x1 = 5.1
  x2 = 7.4
  y = 3*x1**2 + 5*x2**0.5
\end{pyblock}
The answer is \py{y}.

~\\
\textbf{Example 2: Math environment code generation}

\pys{$1 + 1 = !{1+1}$}

~\\
\textbf{Example 3: String manipulation}

\begin{pyblock}
  username = "Doctor"
  password = "Zee"
\end{pyblock}
\begin{pysub}
  Your username is: \textit{!{username}}\\
  Your password is: \textit{!{password}}
\end{pysub}

~\\
\textbf{Example 4: Using} \verb|matplotlib| \textbf{to generate figures}

The \mintinline{latex}{pylabcode} block automatically imports the \mintinline{python}{matplotlib}
↪  module.
First of all, notice the usage of the \texttt{textwidth} variable, which was passed to the Python
↪  environment at the preamble, to control the size of the figure.
Following that, the matplotlib commands are juxtaposed.
Finally, a .pdf figure is saved with the \mintinline{python}{savefig} command.

\textbf{Note:} Your initial \LaTeX must have any \mintinline{latex}{\includegraphics} commands
↪  commented out, because these files have not yet been created by \verb|pythontex|. Once the image
↪  files are created, you can un-comment any image include commands.

\begin{pylabcode}
  figwidth = 0.4*pytex.pt_to_in(pytex.context.textwidth)
  figheight = 3/4*figwidth
  rc('text', usetex=True)
  rc('font',**{'family':'serif', 'serif':['Times']})
  rc('font', size=10.0)
  rc('legend', fontsize=10.0)
  x = linspace(0, 3*pi)
  figure(figsize=(figwidth, figheight))
  plot(x, sin(x), label='$\sin(x)$')
  plot(x, sin(x)**2, label='$\sin^2(x)$',  linestyle='dashed')
  xlabel(r'$x$-axis')
  ylabel(r'$y$-axis')
  xticks(arange(0, 4*pi, pi), ('$0$','$\pi$', '$2\pi$', '$3\pi$'))
  axis([0, 3*pi, -1, 1])
  legend(loc='lower right')
  savefig('myplot.pdf', bbox_inches='tight')
\end{pylabcode}


\begin{center}
  \includegraphics{myplot.pdf}
\end{center}
```

Result

PythonTEXprovides a multitude of environments which allow LATEXto interact with Python.
The \pyb/pyblock environment typesets the passed code into the LATEXdocument and also executes it. However, it will not print out any results.

```
my_str='hi'

my_str+=' there'
print('Python says: {0}!'.format(my_str))
```

You can access the last printout done by Python using the \printpythontex command.
Python says: hi there!
The \pyc/pycode environment both executes code and typesets results, but does not typeset the code itself.
Let's do some calculations
You can use the \pys/pysub environment to substitute Python variable values into LATEXcode, using the !{...} format.
In this example, sympy expression manipulation is used to generate a function expression, which is then plotted using gnuplot.



The r string prefix prevents escaping characters. This can be used to effectively perform LATEXcode generation within a pycode block.

A message from Python!

Now, let us focus on actual calculations done with pythontex.
Example 1: Typical arithmetic

```
x1 = 5.1
x2 = 7.4
y = 3*x1**2 + 5*x2**0.5
```

The answer is 91.6314705087.

Example 2: Math environment code generation
$1 + 1 = 2$

Example 3: String manipulation

```
username = "Doctor"
password = "Zee"
```

Your username is: *Doctor*
Your password is: *Zee*

Example 4: Using `matplotlib` to generate figures
The `pylabcode` block automatically imports the `matplotlib` module. First of all, notice the usage of the `textwidth` variable, which was passed to the Python environment at the preamble, to control the size of the figure. Following that, the matplotlib commands are juxtaposed. Finally, a .pdf figure is saved with the `savefig` command.
Note: Your initial LaTeXmust have any `\includegraphics` commands commented out, because these files have not yet been created by `pythontex`. Once the image files are created, you can un-comment any image include commands.

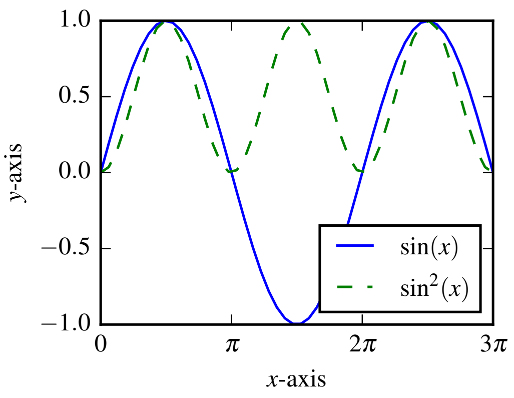


Figure Wrapping

Description

Wrapping a figure around text isn't a stock functionality in LaTeX, but the `wrapfig` package provides a clean interface for that purpose.

Sources

https://tex.stackexchange.com/questions/118602/how-to-text-wrap-an-image-in-latex
Frog image from: http://animal-wildlife.blogspot.gr/2011/08/tree-frog.html

Used Packages

`wrapfig, graphicx`

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{graphicx}
\usepackage{wrapfig}
\usepackage{lipsum} % generates filler text

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
```

Document

```
{\centering
  {\Large Figure Wrapping Example}\\
  \vspace{5pt}
  {\large writeLaTeX}\\
  \vspace{5pt}
  \today\\
  \vspace{5pt}
}

\begin{wrapfigure}{R}{0.3\textwidth}
  \centering
  \includegraphics[width=0.25\textwidth]{frog.jpg}
  \caption{\label{fig:frog1}This is a figure caption.}
\end{wrapfigure}

\lipsum[1]

\begin{wrapfigure}{L}{0.3\textwidth}
  \centering
  \includegraphics[width=0.25\textwidth]{frog.jpg}
  \caption{\label{fig:frog2}This is a figure caption.}
\end{wrapfigure}

\lipsum[2-3]
```

Result

Figure Wrapping Example

writeLaTeX

March 9, 2018

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

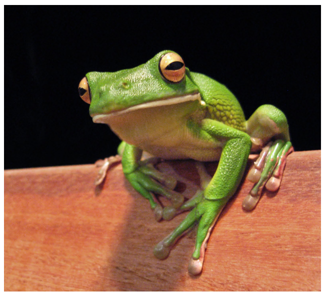


Figure 1: This is a figure caption.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

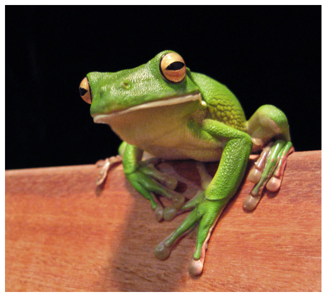


Figure 2: This is a figure caption.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

The cleveref package

Description

cleveref is a very powerful package which provides functionality for cross-referencing within a document. See the example for an overview of the functionality.
Warning 1: cleveref has to be loaded after hyperref
Warning 2: cleveref is not compatible with ieeetrantools (see links below). That means that it cannot refer to IEEEeqnarray environments properly. Some workarounds may exist but they are not guaranteed not to break anything else.

Sources

http://ctan.math.utah.edu/ctan/tex-archive/macros/latex/contrib/cleveref/cleveref.pdf
https://tex.stackexchange.com/questions/1863/which-packages-should-be-loaded-after-hyperref-instead-of-before#1868
https://tex.stackexchange.com/questions/227137/how-to-use-cleveref-with-ieeeeqnarray
https://tex.stackexchange.com/questions/132823/ieeetrantools-clash-with-cleveref
https://tex.stackexchange.com/questions/250733/ieeetran-cls-and-cleveref-the-special-case-of-ieeeeqnarra
https://tex.stackexchange.com/questions/119225/cleveref-cleveref-appears-to-not-be-so-clever-sorry-for-the-pun-but-i-couldnt

Used Packages

graphicx, hyperref, cleveref

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{hyperref}
\usepackage{graphicx}
\usepackage[noabbrev]{cleveref}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
```

Document

```
\begin{figure}[b]
  \centering
  \includegraphics[width=0.25\linewidth]{frog.jpg}
  \caption{This is a figure}
  \label{fig:1}
\end{figure}

\begin{equation}
x^2+y^2=z^2
\label{eq:1}
\end{equation}
\begin{equation}
e^{\pi} = -1
\label{eq:2}
```

```
\end{equation}

The cleveref package allows you to:
\begin{itemize}
  \item prefix the type of the reference automatically: \cref{fig:1}
  \item reference with small and capital initial letters: \Cref{eq:1}, \cref{eq:1}.
  \item reference a range: \crefrange{eq:1}{eq:2}
  \item get the type of the reference: \namecref{fig:1}, \namecref{eq:1}
  \item get the page number of the reference: \cpageref{eq:1}
\end{itemize}
```

Result

Figure 1: This is a figure

$$x^2 + y^2 = z^2 \tag{1}$$

$$e^\pi = -1 \tag{2}$$

The cleveref package allows you to:

- prefix the type of the reference automatically: figure 1
- reference with small and capital initial letters: Equation (1), equation (1).
- reference a range: equations (1) to (2)
- get the type of the reference: figure, equation
- get the page number of the reference: page 2

Replace the Missing Reference Symbol (??)

Description

Typically, when a reference to a missing label occurs, a double question mark appears (??). This is hard to track or search for. Instead, this symbol can be changed to an arbitrary string, for better visibility. The same holds for the missing citation symbol.

Sources

https://tex.stackexchange.com/questions/345438/how-to-replace-not-found-reference-in-an-another-constant-e-g-ref?newsletter=1&nlcode=544695%7c921f
https://tex.stackexchange.com/questions/255533/how-to-replace-default-missing-reference-symbol

Used Packages

etoolbox, xpatch

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{etoolbox} % programming   facilities   geared  primarily   towards  LaTeX class and
    package  authors
\usepackage{xpatch} % generalizes the macro patching commands provided by etoolbox
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
% must be done after loading hyperref
\newcommand{\myunknownrefsymbol}{\color{red}CITE}
\makeatletter
\patchcmd{\@setref}{??}{\color{red}\textless REF \textgreater}{}{}
\def\@citex[#1]#2{\leavevmode
  \let\@citea\@empty
  \@cite{\@for\@citeb:=#2\do
    {\@citea\def\@citea{,\penalty\@m\ }%
      \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
      \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
      \@ifundefined{b@\@citeb}{\hbox{\reset@font\bfseries \myunknownrefsymbol}%
        \G@refundefinedtrue
        \@latex@warning
        {Citation '\@citeb' on page \thepage \space undefined}}%
      {\@cite@ofmt{\csname b@\@citeb\endcsname}}}}{#1}}
\makeatother
```

Document

```
  Figure \ref{missing} is missing.\\
  The same holds for reference \cite{missing-book}.
```

Result

Figure **<REF >** is missing.
The same holds for reference [**CITE**].

Chapter 3

Tables

Colors in table elements

Description

Sometimes tables need some color in them. The `colortbl` package offers some nice functionalities and works with `tabularx` as well.

Sources

https://tex.stackexchange.com/questions/50349/color-only-a-cell-of-a-table

Used Packages

`colortbl`

Preamble

```
\usepackage[table]{xcolor} % table option invokes the colortbl package
\usepackage{tabularx}
```

Document

```
  \centering
  \begin{tabularx}{0.5\linewidth}{%
      >{\columncolor{gray!80}[.5\tabcolsep]}l%
      |%
      >{\columncolor{white}[.5\tabcolsep]}X%
      !{\color{blue}\vline}
      @{\color{blue!50}\vrule width \doublerulesep}
      !{\color{blue}\vline}%
      >{\columncolor{white}[.5\tabcolsep]}r%
    }
    \textbf{Header 1}                      & \textbf{Header 2}        & \textbf{Header 3} \\ \
        hline
    Some                                   & \cellcolor{blue!25}coloured & contents     \\ \hline
    \rowcolor{red!25} And                 & some                      & more            \\
    \rowcolor{red!25} \cellcolor{gray!80}And & some                            & more
```

Result

Header 1	**Header 2**	**Header 3**
Some	coloured	contents
And	some	more
And	some	more

Chapter 4

Miscellaneous

Custom Labels

Description

I wanted an easy-ish way to make and print labels for some storage boxes I use. I had tried tables in MS Word, and while dimensions and images are straightforward to build, it requires a lot of mouse clicking, dragging and typing, each time I want to make a new label.

The *ticket* pakage provides some amenities to do that. The philosophy behind it is that

1. you define some key lengths in the preamble, pertaining to the label geometry
2. you define a default ticket background, placing there any content which is common to all tickets
3. you define a ticket, which places the passed arguments appropriately for each label

Then, you call the `ticket` command for each new label you want to make.

In theory, this allows you to parse `.csv` files and turn their contents into labels (e.g. for conference badges).

Read the annotated code for more insight.

Sources

http://mirror.utexas.edu/ctan/macros/latex/contrib/ticket/doc/manual.pdf

Used Packages

`ticket, graphicx, tikz`

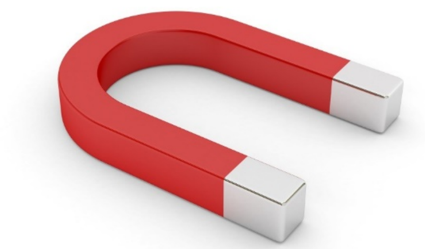
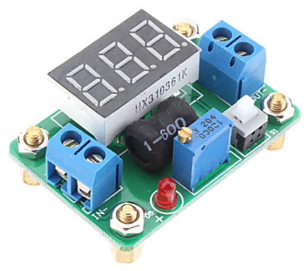
Preamble

```
% Define key quantities
% These are used to manually draw geometry
\newcommand{\labelWidth}{80}
\newcommand{\labelHeight}{60}
\newcommand{\labelSepHeight}{12}
\newcommand{\imageHeight}{45}

% Setup the ticket package lengths
% These are used by the ticket package
\unitlength = 1mm % Used unit
\ticketSize{80}{60} % label dimensions in \unitlength
\ticketNumbers{2}{4} % number of labels in each page
\ticketDistance{0}{0} % separation between each label. I selected 0 to cut both edges at once

% Define the default ticket background
% I used a single tikzpicture to draw the default layout. In theory you could use multiple
    separate \put commands, but I found the coordinate system awkward.
\renewcommand{\ticketdefault}{%
  \put(0,0){%
    \begin{tikzpicture}[x=1mm, y=1mm] % It helps turning tikz units into real mms for easier
        reference
    %    % Add temporary grid for easier positioning of elements
    %    \draw [gray!50, step=10] (0,0) grid + (\labelWidth,\labelHeight);
    %    \node (zeroMark) at (2,2) {0};

    % Draw a frame
    \draw (0,0) -- %
    (\labelWidth, 0) --%
    (\labelWidth, \labelHeight) --%
    (0, \labelHeight) --%
    cycle;
    % Draw the horizontal line
    \draw (0, \labelHeight-\labelSepHeight) -- (\labelWidth, \labelHeight-\labelSepHeight);
    \end{tikzpicture}
  }
}

% Define the ticket content
\newcommand{\myLabel}[2]{\ticket{%
    \put(0, 0){%
      \begin{tikzpicture}[x=1mm, y=1mm]
      % Draw the lable title
      \node[anchor=center] (title) at (0.5*\labelWidth, \labelHeight-0.6*\labelSepHeight) {\
          Huge\textbf{#1}};
      % Insert the image
      \node[anchor=center] (img) at (0.5*\labelWidth, 0.5*\labelHeight-0.5*\labelSepHeight ) {\
          includegraphics[height=\imageHeight mm]{#2}};
      % The two diagnonal points are inserted to create a tikz image as large as the label. In
          this way, it is correctly placed within the image.
      \node at (\labelWidth, \labelHeight) {};
      \node at (0, 0) {};
      \end{tikzpicture}}%
  }
}
```

Document

```
% Just use one \myLabel{title}{imageFile} for each label
\myLabel{Springs}{images/springs.jpg}
\myLabel{Magnets}{images/magnets}
\myLabel{Nails}{images/nails}
\myLabel{Regulators}{images/regulators}
\myLabel{Sandpaper}{images/sandpaper}
\myLabel{Screws}{images/screws2}
```

Result

Springs	**Magnets**
	
Nails	**Regulators**
	
Sandpaper	**Screws**

Chapter 5

Mathematics Typesetting

Set math mode as default for each item in list

Description

Often times, a list containing only math-type content is needed. Typically, for each new list item the $$ math environment definition is required.

Instead, this example shows a workaround in order to pre-load the math environment automatically for each new list item.

Used Packages

`enumitem, xpatch`

Preamble

```
\usepackage{enumitem} % Allows customization of the list environments
\usepackage{xpatch} % Container package extending new command declaration or patching existing
     commands
% Define a new enumeration list type
\newlist{mathlist}{enumerate}{1}
% Set the numerals
\setlist[mathlist,1]{label={\arabic*.}}
% Add math environment opening and closing for each item
\AtBeginEnvironment{mathlist}{%
\xpretocmd{\item}{\ifmmode\)\fi}{}{}
\xapptocmd{\item}{\ifmmode\else\(\fi}{}{}
}
\AtEndEnvironment{mathlist}{%
\ifmmode \)\fi% Close math mode
}
```

Document

```
  \begin{mathlist}
    \item x + y = z
    \item e^{i\pi} + 1 = 0
    \item E=mc^2
    \item \)Non math - mode % Jumping out of math-mode
  \end{mathlist}
  \begin{enumerate}
    \item Foo % It's regular and not in math mode!
  \end{enumerate}
```

Result

1. $x + y = z$
2. $e^{i\pi} + 1 = 0$
3. $E = mc^2$
4. Non math - mode

1. Foo

Summation with Limits on the Side

Description

Some times placing the summation limits to the side of the summation symbol can provide better typesetting or simply save vertical space. This is how to achieve this.

Sources

https://tex.stackexchange.com/questions/283939/having-sub-super-scripts-on-the-side-for-sum

Used Packages

amsmath

Preamble

```
\usepackage{amsmath}
```

Document

```
Standard summation:
\begin{equation}
\sum_{1}^{\infty} 1/x
\end{equation}
Using \verb|\nolimits|:
\begin{equation}
\sum\nolimits_{1}^{\infty}1/x
\end{equation}
Using \verb|\textstyle|:
\begin{equation}
{\textstyle\sum}_{1}^{\infty}1/x
\end{equation}
```

Result

Standard summation:

$$\sum_{1}^{\infty} 1/x \tag{1}$$

Using `\nolimits`:

$$\sum\nolimits_{1}^{\infty}1/x \tag{2}$$

Using `\textstyle`:

$${\textstyle\sum}_{1}^{\infty}1/x \tag{3}$$

Chapter 6

tikz tricks

File System Structure

Description

A tree file system graphical structure is generated, drawn as a tikz image. The trees tikz library is used to build the structure.

Sources

http://tex.stackexchange.com/questions/306415/connecting-tree-nodes-with-double-arrows-a-la-chef?newsletter=1&nlcode=544695%7c921f

Used Packages

tikz

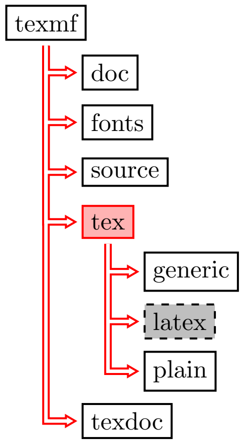
Preamble

```
\documentclass[class=article, crop=false]{standalone}

\usepackage{tikz}
\usetikzlibrary{trees,decorations.markings}
```

Example Code

```
\tikzstyle{every node}=[draw=black,thick,anchor=west]
\tikzstyle{selected}=[draw=red,fill=red!30]
\tikzstyle{optional}=[dashed,fill=gray!50]

\newcommand{\arrowcolor}{red}
\newcommand{\arrowfillcolor}{white}

\pgfdeclarelayer{front}
\pgfsetlayers{main,front}

\makeatletter
\pgfkeys{%
/tikz/path on layer/.code={
\def\tikz@path@do@at@end{\endpgfonlayer\endgroup\tikz@path@do@at@end}%
\pgfonlayer{#1}\begingroup%
}%
}
\makeatother

\begin{tikzpicture}[%
rightarr/.pic={\path[pic actions] (-0.4,0)--(-1,-0.35)--(-1,.35)--cycle;},
grow via three points={one child at (0.5,-0.7) and
two children at (0.5,-0.7) and (0.5,-1.4)},
edge from parent path={(\tikzparentnode.south) |- (\tikzchildnode.west)},
edge from parent/.style={
decoration={
markings,
mark=at position 1 with{\coordinate (0, 0) pic[\arrowcolor,fill=\arrowfillcolor,scale=0.22]{
    rightarr};},
},
draw = \arrowcolor,
line width = 3pt,
shorten >= 5.7pt,
shorten <= 2pt,
postaction = {decorate},
postaction = {draw,line width=1.4pt,white,path on layer=front},
}]
\node {texmf}
child { node {doc}}
child { node {fonts}}
child { node {source}}
child { node [selected] {tex}
child { node {generic}}
child { node [optional] {latex}}
child { node {plain}}
}
child [missing] {}
child [missing] {}
child [missing] {}
child { node {texdoc}};
\end{tikzpicture}
```

Example Result



File System Structure 2

Description

Another way to plot a file system structure is to take advantage of the `forest` package. Note that for this specific example `forest` v2 or greater is required.

Sources

https://tex.stackexchange.com/questions/328886/making-a-directory-tree-of-folders-and-files?newsletter=1&nlcode=544695%7c921f
https://www.ctan.org/pkg/forest

Used Packages

`forest, tikz`

Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage[edges]{forest}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
% Define custom colors
\definecolor{folderbg}{RGB}{124,166,198}
\definecolor{folderborder}{RGB}{110,144,169}
% Define the folder and file drawing commands
\newlength\Size
\setlength\Size{4pt}
\tikzset{%
  folder/.pic={%
    \filldraw [draw=folderborder, top color=folderbg!50, bottom color=folderbg] (-1.05*\Size
        ,0.2\Size+5pt) rectangle ++(.75*\Size,-0.2\Size-5pt);
    \filldraw [draw=folderborder, top color=folderbg!50, bottom color=folderbg] (-1.15*\Size,-\
        Size) rectangle (1.15*\Size,\Size);
  },
  file/.pic={%
    \filldraw [draw=folderborder, top color=folderbg!5, bottom color=folderbg!10] (-\Size,.4*\
        Size+5pt) coordinate (a) |- (\Size,-1.2*\Size) coordinate (b) -- ++(0,1.6*\Size)
        coordinate (c) -- ++(-5pt,5pt) coordinate (d) -- cycle (d) |- (c) ;
  }
}
% Edit forest options
\forestset{%
  declare autowrapped toks={pic me}{}, % This is an option to hold the value desired for each
      node. Eventually, this will take values such as pic {file}. By default, it is empty
  pic dir tree/.style={% We define a style for trees of this kind, pic dir tree
    for tree={%
      folder, % Select the directory-style, provided by the folder/edges library
      font=\ttfamily,
      grow'=0,
    },
    before typesetting nodes={% We'll collect the pic me settings while parsing the tree and add
        them to the edge path by appending the option to the value of edge label
      for tree={%
        edge label+/.option={pic me},
      },
    },
  },
  pic me set/.code n args=2{% pic me set takes 2 arguments. The first is the name of a Forest
      style to-be-created. The second is the name of a pic such as folder or file.
    \forestset{%
      #1/.style={%
        inner xsep=2\Size,
        pic me={pic {#2}},
      }
    }
  },
  pic me set={directory}{folder}, % We just create 2 such styles for now.
  pic me set={file}{file},
}
```

Document

```
\begin{forest}
  pic dir tree, % Import the tree preamble
  where level=0{}{% folder icons by default; override using file for file icons
    directory, % All level 0 nodes will use the directory format
  },
  [system % Now draw the tree
  [config]
  [lib
  [Access]
  [Plugin]
  [file.txt, file] % Override the default folder format and use the file format, as defined
      above
  ]
  [templates]
  [tests]
  ]
\end{forest}
```

Result

```
system
├── config
├── lib
│   ├── Access
│   ├── Plugin
│   └── file.txt
├── templates
└── tests
```

Adding Noise to Plots

Description

Sometimes realistic noise needs to be added to function plots. The following code demonstrates that.

The `\pgfmathstoreseed{arg1}` and `\pgfmathrestoreseed` can be used to manipulate the random number generator seed.

The `rand` function returns uniform noise. It would be nice to find some native command for Gaussian, white, etc noises.

Sources

https://tex.stackexchange.com/questions/144272/noisy-analogue-waveform-in-tikz
https://tex.stackexchange.com/questions/126838/plotting-a-random-function-fx-and-fxsinx-with-tikz

Used Packages

pgfplots

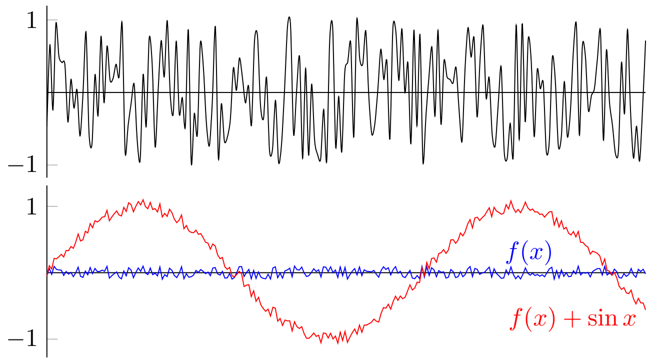
Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{pgfplots}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
\makeatletter
\def\pgfmathstoreseed#1{\let#1\pgfmath@rnd@z}
\let\pgfmathrestoreseed\pgfmathsetseed
\makeatother
```

Document

```
\begin{tikzpicture}[samples=200, domain=0:5*360]
\begin{axis}[
width=10cm, height=4cm,
enlarge x limits=false,
xtick=\empty,
axis lines*=middle,
]
\pgfmathsetseed{126838}
\addplot [no markers, smooth] {rand};
\end{axis}
\end{tikzpicture}

\noindent
\begin{tikzpicture}[samples=250, domain=0:10]
\begin{axis}[
width=10cm, height=4cm,
enlarge x limits=false,
xtick=\empty,
axis lines*=middle,
]
\pgfmathsetseed{126838}
\addplot[color=blue] {rand/10};
\node[right] at (axis cs:7.5,0.3) {\color{blue}$f(x)$};
\pgfmathsetseed{126838}
\addplot[color=red] {rand/10 + sin (deg(x))};
\node[left] at (axis cs:10,-0.7) {\color{red}$f(x) + \sin x$};
\end{axis}
\end{tikzpicture}
```

Result



Dual axes in tikz

Description

Dual scaling in a tikz figure is a common need. This example shows a nice way to have two y axes in a single tikzpicture; one on the left and one on the right.

Sources

https://tex.stackexchange.com/questions/316978/how-to-draw-a-graph-with-several-axes?newsletter=1&nlcode=544695%7c921f

Used Packages

pgfplots

Preamble

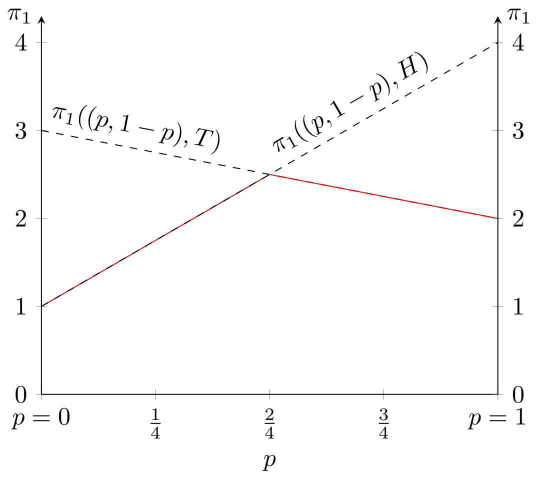
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{pgfplots}
\usetikzlibrary{arrows}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
\pgfplotsset{width=8cm, compat=1.13}
```

Document

```
\begin{tikzpicture}
\pgfplotsset{% <-- common presets for both axes
  ylabel=$\pi_1$,
  domain=0:1,xmin=0,xmax=1,
  ymin=0,ymax=4.3,
  enlargelimits=false,
  clip=false % Controls whether any paths inside of an axis shall be clipped. Used to avoid
      clipping red text which is outside the plot limits.
}
\begin{axis}[
axis y line=left,                % <-- left y axis
ylabel style={at={(0,1)},rotate=-90,anchor=east},% <--
axis x line=bottom,              % <--
x axis line style={-},           % <--
xlabel=$p$,
xtick={0, 0.25, 0.75, 1},        % <--
xticklabels={$p=0$, $\frac{1}{4}$, $\frac{3}{4}$, $p=1$},% <--
]
\addplot[dashed] { -x+3} node[above,sloped,pos=0.2] {$\pi_1((p,1-p),T)$};
\addplot[dashed] {3*x+1} node[above,sloped,pos=0.7] {$\pi_1((p,1-p),H)$};
\addplot[red] {ifthenelse(x>0.5,-x+3,3*x+1)};
\draw[dotted,red] (0,5/2) node[left] {$\underline{v} = \frac{5}{2}$}
-| (0.5,0) node[below] {$\hat{p} = \frac{1}{2}$}; % path elements (i.e. the nodes) are outside
    of the graph area
```

```
\end{axis}
%
\begin{axis}[
axis y line=right,                 % <-- right y axis
ylabel style={at={(1,1)},rotate=-90,anchor=west},% <--
axis x line=none,                  % <-- Disable second x axis to avoid overlap
]
\end{axis}
\end{tikzpicture}
```

Result



Arrows within Text

Description

Tikz can store node positions within the text, which can be later used to rout arrows.

Sources

https://tex.stackexchange.com/questions/408177/drawing-an-arrow-between-two-parts-of-a-sentence?newsletter=1&nlcode=544695%7c921f

Used Packages

tikz

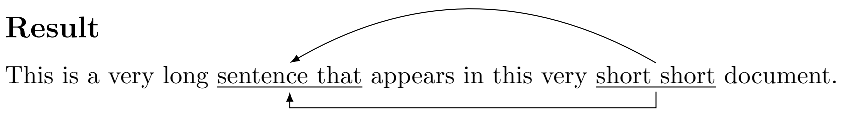
Preamble

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific packages
\usepackage{tikz}
\usetikzlibrary{topaths}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Example-specific preamble
\tikzstyle{every picture}+=[remember picture% Be able to refer to nodes in other pictures
,inner xsep=0% prevent messing up the word spacing in the horizontal direction
,inner ysep=0.25ex] % Adjust vertical spacing
```

Document

```
This is a very long \tikz[baseline=(node1.base)]\node (node1) {\underline{sentence that}};
    appears
in this very \tikz[baseline=(node2.base)]\node (node2) {\underline{short short}}; document.

\begin{tikzpicture}[overlay]
% Bend above text line
\draw[-latex] (node2.north) to[bend right] (node1.north);
% Angled
\draw[-latex] (node2.south) -- ++(0,-1.5ex) -| (node1.south);
```

Result



This is a very long sentence that appears in this very short short document.