

# L<sup>A</sup>T<sub>E</sub>XiPy Example

Jean Nassar

August 24, 2017

## Contents

<b>1</b>	<b>Online resources</b>	<b>2</b>
<b>2</b>	<b>Assumptions</b>	<b>2</b>
<b>3</b>	<b>Plotting</b>	<b>2</b>
3.1	Without L <sup>A</sup> T <sub>E</sub> Xify . . . . .	2
3.2	With L <sup>A</sup> T <sub>E</sub> Xify . . . . .	3
3.3	Custom parameters . . . . .	4
3.4	Reverting the settings . . . . .	6
<b>4</b>	<b>Programming tips</b>	<b>6</b>
<b>5</b>	<b>Using in L<sup>A</sup>T<sub>E</sub>X</b>	<b>6</b>

## List of Listings

1	List of imports . . . . .	2
2	Figure generation . . . . .	2
3	lp.figure() signature . . . . .	3
4	Example of L <sup>A</sup> T <sub>E</sub> Xify . . . . .	3
5	Default L <sup>A</sup> T <sub>E</sub> Xify settings . . . . .	4
6	Using custom L <sup>A</sup> T <sub>E</sub> Xify settings . . . . .	5
7	Permanently changing L <sup>A</sup> T <sub>E</sub> Xify settings . . . . .	5
8	Using partial function application . . . . .	6
9	L <sup>A</sup> T <sub>E</sub> X Minimum Working Example . . . . .	7
10	Importing PGF with raster . . . . .	8

## List of Figures

1	Default plot . . . . .	3
2	Default L <sup>A</sup> T <sub>E</sub> X plot . . . . .	4
3	Custom L <sup>A</sup> T <sub>E</sub> X plot . . . . .	6

## 1 Online resources

The Github repository is at <https://github.com/masasin/latexipy>, and the full example file is at [examples/examples.py](#). Full documentation is available at <https://latexipy.readthedocs.io/>.

## 2 Assumptions

In order to generate plots with L<sup>A</sup>T<sub>E</sub>XiPy, install the package and import it.

This document assumes that the following imports are made:

```
from functools import partial
import matplotlib.pyplot as plt
import latexipy as lp
```

Listing 1: The imports used in this example.

Also, it assumes that there is a function, `plot_sin_and_cos()` which uses a Matplotlib-based package to generate a plot without calling `plt.savefig()` or `plt.close()`. L<sup>A</sup>T<sub>E</sub>XiPy is known to work well with various libraries, including Matplotlib, Numpy, Pandas and Seaborn, among others.

Finally, `figure()` in the listings (without `lp.`) is the partially applied version of the `lp.figure()` function to save it in the right directory. See [Section 4](#) for more details.

## 3 Plotting

### 3.1 Without L<sup>A</sup>T<sub>E</sub>Xify

If you don't L<sup>A</sup>T<sub>E</sub>Xify, Matplotlib's defaults are used. The typeface is sans-serif, and the font a bit larger. The code in [Listing 2](#) generates [Figure 1](#).

```
with figure('sincos'):
    plot_sin_and_cos()
```

Listing 2: Generate figures with one extra line.

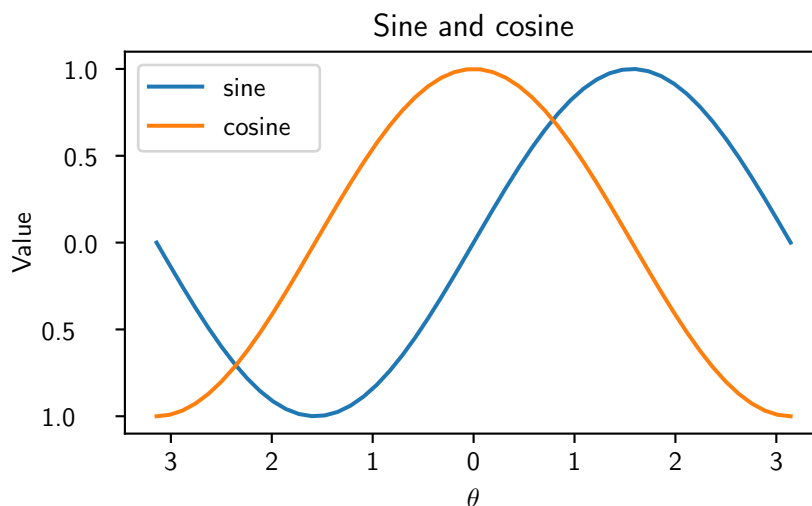


Figure 1: The plot, with default parameters before L<sup>A</sup>T<sub>E</sub>Xifying.

`figure` is a context manager with the following signature:

```
@contextmanager
def figure(filename, *, directory='img', exts=['pgf', 'png'],
           size=None, mkdir=True):
```

Listing 3: The signature for `lp.figure()`.

Note that the default for `directory`, `./img`, is relative to the calling location, and that `size` is a tuple with the  $x$ - and  $y$ -dimensions in inches.

### 3.2 With L<sup>A</sup>T<sub>E</sub>Xify

Figure 2 shows a plot using the default L<sup>A</sup>T<sub>E</sub>XiPy configuration, with the same typeface as the body, but a size of 8 pt. To generate it, we can use the code in Listing 4.

```
lp.latexify()

with figure('sin_cos'):
    plot_sin_and_cos()
```

Listing 4: `lp.latexify()` generates plots that fit well with L<sup>A</sup>T<sub>E</sub>X.

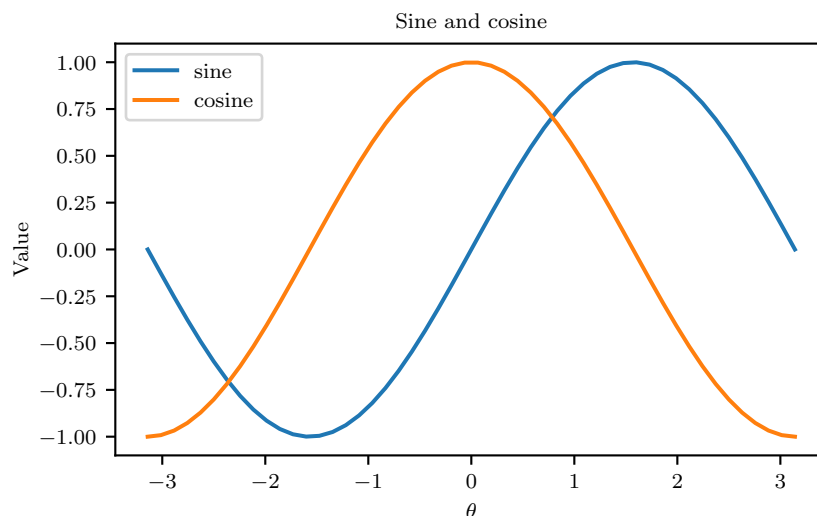


Figure 2: The plot, with default parameters after L<sup>A</sup>T<sub>E</sub>Xifying.

### 3.3 Custom parameters

`lp.latexify()` uses `lp.PARAMS` by default. Its values are shown in Listing 5.

`FONT_SIZE = 8`

```
PARAMS = {
    'pgf.texsystem': 'xelatex', # pdflatex, xelatex, lualatex
    'text.usetex': True,
    'font.family': 'serif',
    'font.serif': [],
    'font.sans-serif': [],
    'font.monospace': [],
    'pgf.preamble': [
        r'\usepackage[utf8x]{inputenc}',
        r'\usepackage[T1]{fontenc}',
    ],
    'font.size': FONT_SIZE,
    'axes.labelsize': FONT_SIZE,
    'axes.titlesize': FONT_SIZE,
    'legend.fontsize': FONT_SIZE,
    'xtick.labelsize': FONT_SIZE,
    'ytick.labelsize': FONT_SIZE,
}
```

Listing 5: The default parameters changed by L<sup>A</sup>T<sub>E</sub>Xify.

To change some parameters temporarily, such as the font size, you can use the `lp.temp_params` context manager. To keep the settings applied, simply pass a different dictionary to `lp.latexify()`. Either can be done at any time. [Listing 6](#) shows an example of increasing the font size temporarily. [Listing 7](#) shows an example of increasing the font size permanently.

```
font_size = 10
with lp.temp_params(font_size=font_size):
    with figure('sincos_big_font_temp'):
        plot_sin_and_cos()
```

Listing 6: Increasing the font size is as simple as setting it once.

```
font_size = 10
params = lp.PARAMS.copy()
params.update({param: font_size
               for param in params
               if 'size' in param})
lp.latexify(params)

with figure('sincos_big_font_permanent'):
    plot_sin_and_cos()
```

Listing 7: Pass a dictionary to `lp.latexify()` to permanently change a setting.

Either way, rerunning [Listing 2](#) after updating the parameters gives [Figure 3](#), with a 10 pt font.

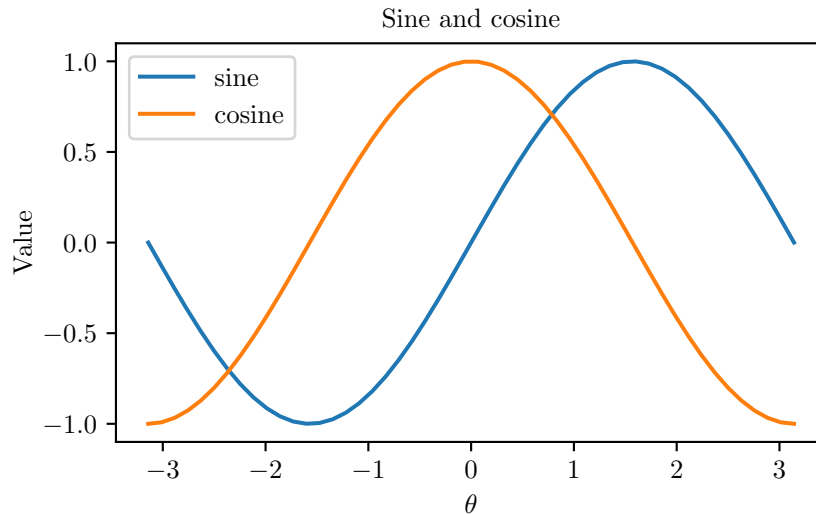


Figure 3: The plot, with the font size increased.

### 3.4 Reverting the settings

To revert the settings, just run `lp.revert()`.

## 4 Programming tips

If you keep passing the same arguments to `lp.figure()` (for example, an output directory, a set of filetypes, or a certain size), you can save it for reuse by using the `partial` function from the `functools` module, as shown in [Listing 8](#). After that, you can use it just like `lp.figure()`. Note that you would not be able to redefine an argument that you had previously applied.

Here, `DIRECTORY` refers to any output directory, either as a string, or as a `pathlib.Path()` object.

```
figure = partial(lp.figure, directory=DIRECTORY)

with figure('sincos_partial'):
    plot_sin_and_cos()
```

Listing 8: The `sincos_partial` plots will be stored in `DIRECTORY`.

## 5 Using in L<sup>A</sup>T<sub>E</sub>X

To include a PGF file in your L<sup>A</sup>T<sub>E</sub>X document, make sure that the `pgf` package is loaded in the preamble:

```
\usepackage{pgf}
```

After that, you can include it in the correct location with:

```
\input{<filename>.pgf}
```

Listing 9 shows a minimum working example of adding an image within a figure.

```
\documentclass{article}
```

```
\usepackage{pgf}
```

```
\begin{document}
```

```
  \begin{figure}[h]
```

```
    \centering
```

```
    \input{img/filename.pgf}
```

```
    \caption[LOF caption]{Regular caption.}
```

```
    \label{fig:pgf_example}
```

```
  \end{figure}
```

```
\end{document}
```

Listing 9: A minimum working example of using PGF with L<sup>A</sup>T<sub>E</sub>X.

Note that figures using additional raster images can only be included by `\input` if they are in the same directory as the main L<sup>A</sup>T<sub>E</sub>X file. To load figures from other directories, you can use the `import` package instead.

```
\usepackage{import}
```

```
\import{<path to file>}{<filename>.pgf}
```

An example is given in Listing 10.

```

\documentclass{article}

\usepackage{import}
\usepackage{pgf}

\begin{document}
  \begin{figure}[h]
    \centering
    \import{/path/to/file/}{filename.pgf} % Note trailing slash.
    \caption[L0F caption]{Regular caption.}
    \label{fig:pgf_example}
  \end{figure}
\end{document}

```

Listing 10: Importing a raster-using PGF from a different directory requires the `\import` package.