Python and PGF

Robert Sawko

Department of Engineering Computing, Cranfield University

August 22, 2014

Abstract

This document presents example usage of `matplotlib` together with LaTeX. We will use the PGF in order to create low-level drawing macros which will be compiled by `xelatex`. We will also cover a generation of figures and legend separately which in some situations is beneficial.

1 Introduction

I prepared these files in a response to one of the questions about `matplotlib` capability to generate graphs in LaTeX format. The main advantage of this is a better integration of the document with the graphs as the fonts, font sizes etc. will be obtained during the compilation of the document rather then each figure. This may eliminate invisible labels, ticks and other artifacts haunting people writing large documents.

The integration is achieved via PGF backend. To find out more about backends read [3]. PGF backend is described [2]. PGF stands for Portable Network Graphics - more can be found in [1].

Admittedly, `matplotlib` is a bit more involved than `gnuplot`. You often need to invoke more commands which are not intuitively clear. On the other hand it is more flexible and can be used in any python code with potential loops over directory structure. This cannot be achieved with bare `gnuplot` without some scripting which quickly gets more involved than a readable Python code. Also, `matplotlib` seems to offer more functionalities. Personally, I use `gnuplot` for interactive plotting as I can obtain plots straight from the files. I use `matplotlib` for larger final documents.

Note that I tried to keep the setup simple without a use of fancy build systems or self configuration tools. Graph generation and document compilation could be run with a single command. The use would become very simple but the configuration, the scripts etc would have become more complex. For now the purpose is to demonstrate PGF and

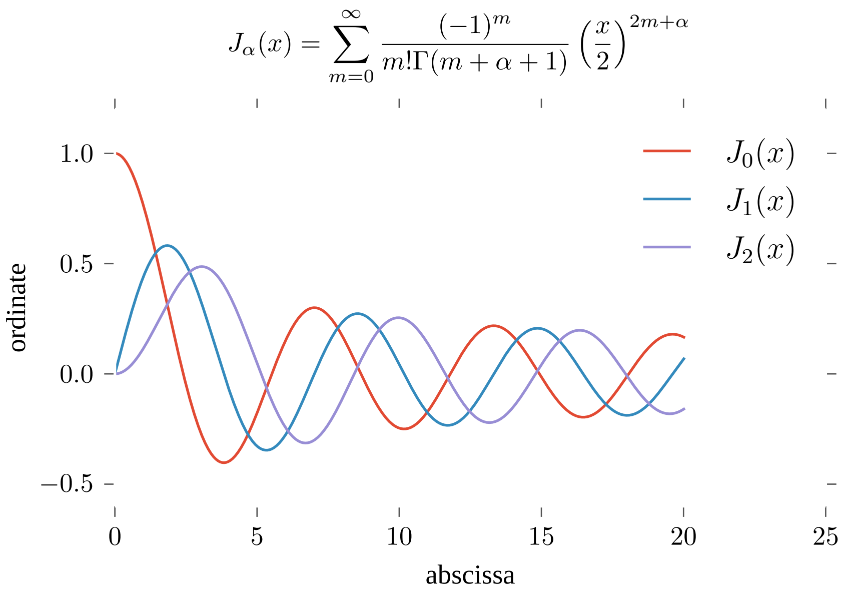

Figure 1: I am printing word ”abscissa” just to prove that the fonts are matched. We’re also admiring a LaTeX equations on the graph.

LaTeXintegration and show some good practices. In the future I may add branches for people who would like to see how to run this with `make` or `cmake`.

2 Example figures

Several examples are presented here. On figure 1 we are simply looking at the capability of plotting via PGF. Figure 2 we are mainly looking at using legend as a separate graph that is only tied to the actual plots at the level of the document. This approach is beneficial if you are plotting many figures of the same type and the legend would have been repeated on each graph. Also, having a legend outside of the plot makes the plot clearer as there is nothing to obscure the results.

3 Future work

1. Include an automated build system like Make or CMake.
2. Include different branches for advanced and novice users.

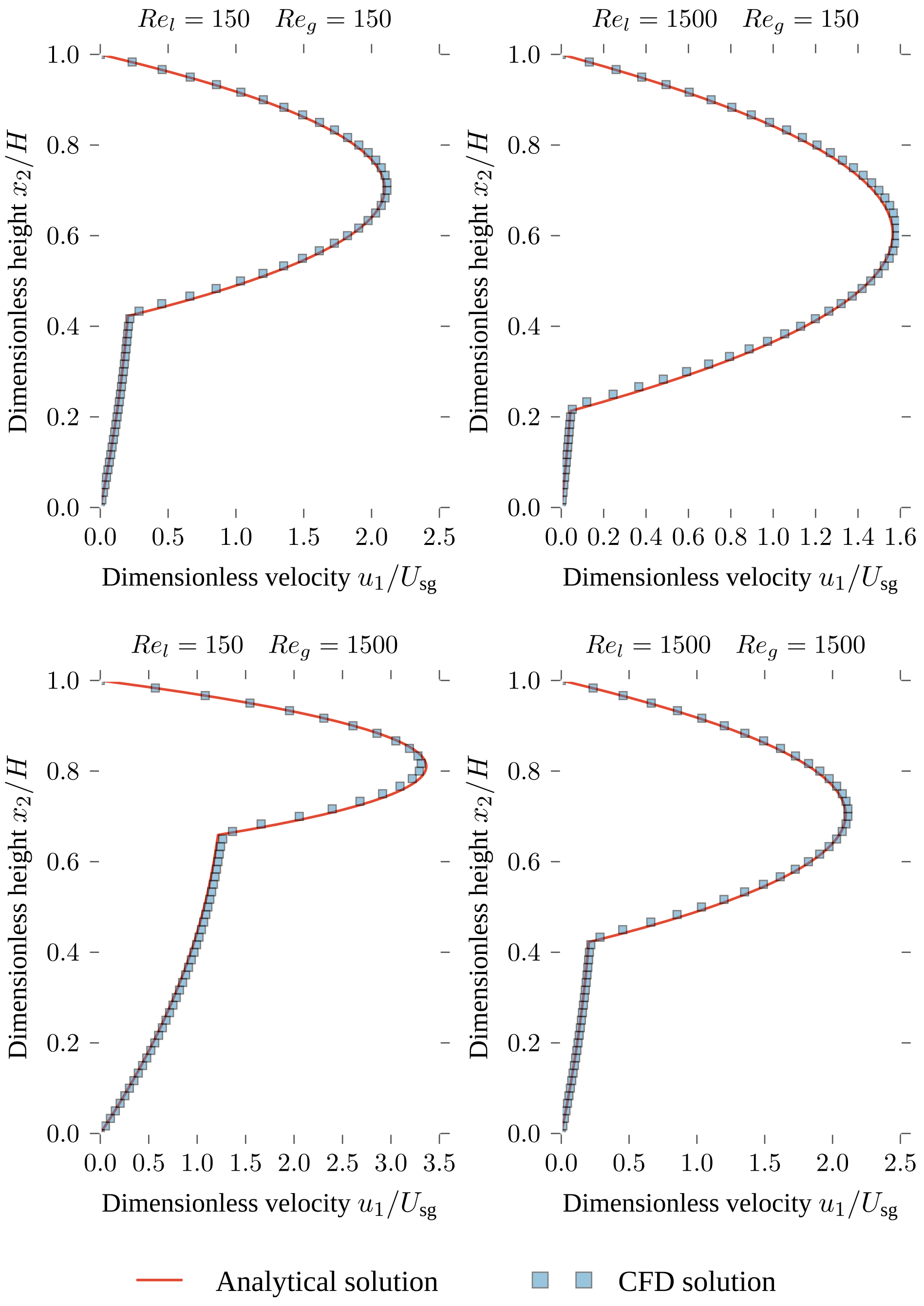

Figure 2: Several figures and a legend. Note the convenience of having the legend in the actual document. It doesn't have to be assigned to any subplot.

References

[1] *PGF/TikZ - Wikipedia article*. Aug. 2014. URL: `http://en.wikipedia.org/wiki/PGF/TikZ`.

[2] *Typesetting With XeLaTeX/LuaLaTeX*. Aug. 2014. URL: `http://matplotlib.org/users/pgf.html`.

[3] *What is a backend?* Aug. 2014. URL: `http://matplotlib.org/faq/usage_faq.html#what-is-a-backend`.