A mixed Mathematica-Python example

This example demonstrates a cooperative effort where Mathematica is used to do the analytic computations while Python is used to plot the data.

The example chosen here is to find and plot the solution to the boundary value problem defined by

$$\frac{d^2y}{dx^2} + 2\frac{dy}{dx} + 10y = 0 \qquad \text{with } y(0) = 3,\ y'(0) = 0$$

This example requires two passes, once for Mathematica and once for Python (and in that order). This example can be run using

```
mmalatex.sh -x -i mixed
pylatex.sh  -x -i mixed
pdflatex         mixed
```

Note that the last pair of commands could also be combined as `pylatex.sh -i mixed`.

The Mathematica code

Here Mathematica is used to first find the general solution of th differential equation. The boundary conitions are then imposed and finally a uniform sampling of the solution is written to a file for later use by Python and Matplotlib.

```
sol = DSolve[y''[x] + 2 y'[x] + 10 y[x] == 0, y, x]
ans = y[x]/.sol[[1]]                    (* mma(ans.101,ans)*)

sol = DSolve[{y''[x] + 2 y'[x] + 10 y[x] == 0, y[0]==3, y'[0]==0}, y, x]
foo = y[x]/.sol[[1]]                    (* mma(ans.102,foo)*)
bah = Simplify[y'[x]/.sol[[1]]]         (* mma(ans.103,bah)*)

(* now sample y and dy at selected points *)
myData = Table[{x, foo, bah}, {x, 0, 2 Pi, 0.02}];
Export["mixed.txt", myData, "Table",  "FieldSeparators" -> " "];
```

The general solution of the differential equation is

$$y(x) = c_1 e^{-x}\sin(3x) + c_2 e^{-x}\cos(3x)$$

while the particular solution satifying the boundary conditions is given by

$$y(x) = e^{-x}(\sin(3x) + 3\cos(3x))$$

The Python code

This is a straighforward use of Matplotlib to plot two functions. The code reads the datafile created previously by Mathematica and then calls Matplotlib to plot that data.

```
import numpy as np
import matplotlib.pyplot as plt

plt.matplotlib.rc('text', usetex = True)
plt.matplotlib.rc('grid', linestyle = 'dotted')
plt.matplotlib.rc('figure', figsize = (5.5,4.1)) # (width,height) inches

x, y, dy = np.loadtxt ('mixed.txt', unpack=True)

plt.plot (x,y)
plt.plot (x,dy)

plt.xlim (0.0,4.0)

plt.legend(('$y(x)$', '$dy(x)/dx$'), loc = 0)
plt.xlabel('$x$')
plt.ylabel('$y(x),\> dy/dx$')
plt.grid(True)
plt.tight_layout(0.5)

plt.savefig('mixed-fig.pdf')
```

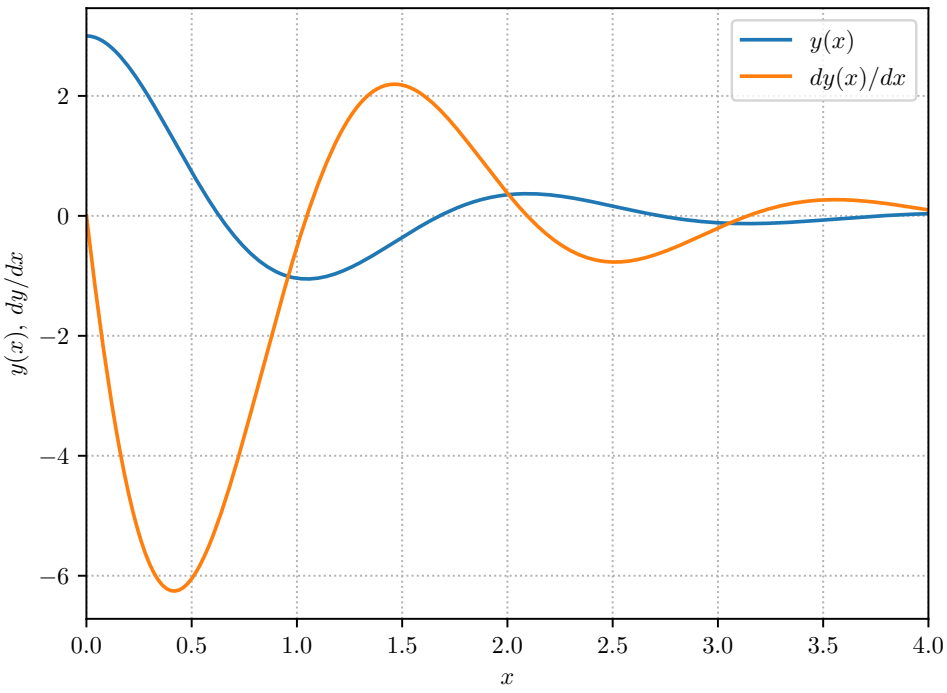

Figure 1: The function and its derivative.