Analysing and Plotting data using python inside LaTeX

README file from developers:

```
1- download the package from  http://www.imada.sdu.dk/~ehmsen/python.sty
2-\usepackage{python}
3- put your python code inside \begin{python} and \end{python}
4- run (pdf)latex with -shell-escape option
```

Experimental data: Potentiostatic pulses of different amplitudes.
The oscilloscope output (filename: F000CH1PotentialPulse.CSV and F000CH2CurrentTransient.CSV):

```
Record Length,2.500000e+03,,   0.000000000000,   0.00000,
Sample Interval,1.000000e-02,,   0.010000000000,   0.00000,
Trigger Point,0.000000000000e+00,,   0.020000000000,   0.00000,
,,,   0.030000000000,   0.00000,
,,,   0.040000000000,   0.00000,
,,,   0.050000000000,   0.00000,
Source,CH1,,   0.060000000000,   0.00000,
Vertical Units,V,,   0.070000000000,   0.00000,
Vertical Scale,1.000000e+00,,   0.080000000000,   0.00000,
Vertical Offset,-2.040000e+00,,   0.090000000000,   0.00000,
Horizontal Units,s,,   0.100000000000,   0.00000,
Horizontal Scale,2.500000e+00,,   0.110000000000,   0.00000,
Pt Fmt,Y,,   0.120000000000,   0.04000,
Yzero,0.000000e+00,,   0.130000000000,   0.00000,
Probe Atten,1.000000e+00,,   0.140000000000,   0.04000,
Model Number,TDS1001B,,   0.150000000000,   0.00000,
Serial Number,C020357,,   0.160000000000,   0.04000,
Firmware Version,FV:v21.20,,   0.170000000000,   0.00000,
,,,0.180000000000,   0.04000,
,,,0.190000000000,   0.04000,
,,,0.200000000000,   0.00000,
,,,0.210000000000,   0.04000,
,,,0.220000000000,   0.00000,
,,,0.230000000000,   0.00000,
,,,0.240000000000,   0.00000,
....
....
```

The "pulsefile.py"program:

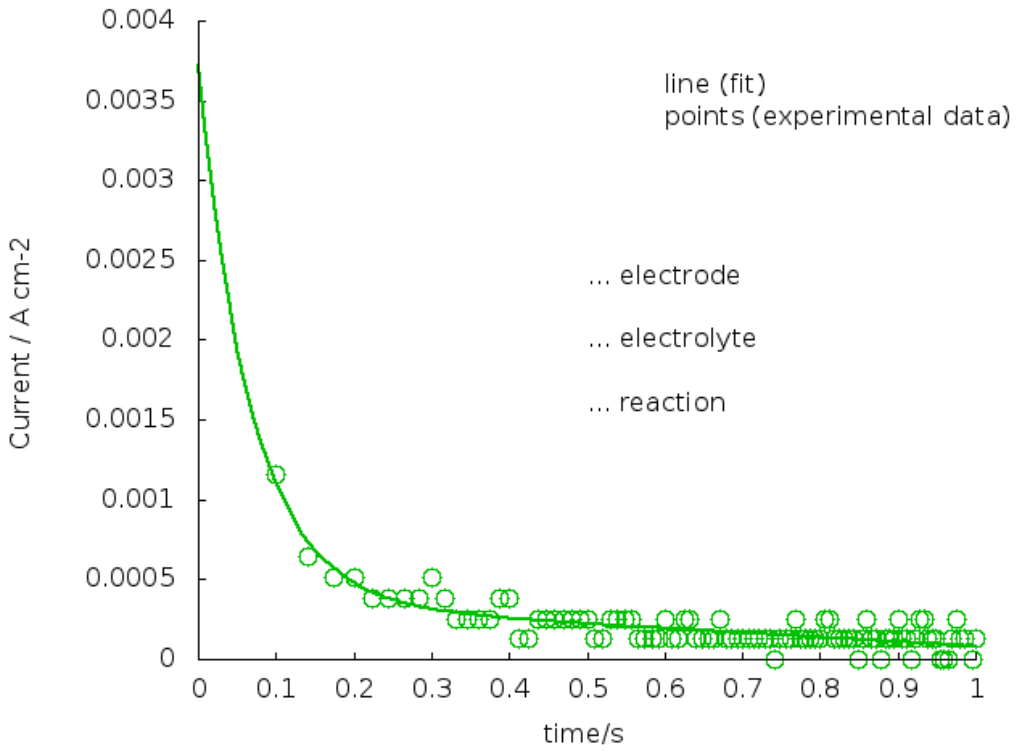
```
#!/usr/bin/env python
# Copyright 2011--2013
# Luc\'ia B. Avalle
# Grupo de Electroqu\'imica Experimental y Te\'orica, FaMAF, UNC
# C\'ordoba, Argentina

# This program is free software: you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation, either version 3 of the License, or
# (at your option) any later version.

# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
# GNU General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program.  If not, see <http://www.gnu.org/licenses/>.

# Authors:
#       Leoncio J.E. L\'opez (anicholo at gmail dot com)


    """The potentiostatic pulses applied to the solid/liquid interface consist of different amplitudes,

    starting at a potential where no hydrogen evolves.

    This program was developed to load transient data from file at location filename.

    The in files are formatted according to the output of two-channel oscilloscope.

    The out files have only time and CH* (*= 1, 2) columns space-separated.

    The structure and program code meet educational goals"""


import sys

try:
    infilename = sys.argv[1];  outfilename = sys.argv[2]
except:
    print "Usage:",sys.argv[0], "infile outfile"; sys.exit(1)


def load_data(infilename):
    f = open(infilename, 'r'); lines = f.readlines(); f.close()

    i = 0; values = [] #empty list
    for line in lines[24:]: #eliminates the headers
        arglist = line.split(",")
        value = float(arglist[3]), float(arglist[4])
        #start from zero: space[0],space[1],space[2],number[3],number[4],
values.append(value) #tuple list
i = i+1
    return values

def dump_data(filename, values):
    # write out 2-column files:
    of = open(filename, 'w')
    for i in range(len(values)):
        of.write('%12.5e %12.5e\n' % values[i])
    of.close()

values = load_data(infilename) #function call for execution
dump_data(outfilename, values)
print 'We are done!'
```

Data fitting according to a given physical model:
Run python inside latex. Use gnuplot to do calculations inside python and plot data.

```
\begin{python}
#!/usr/bin/python
import sys
from collections import defaultdict

#This piece of code was written by Leoncio J.E. L\'opez (anicholo at gmail dot com)
def fit_data(archivo):
    f = os.popen('gnuplot' ,'w')

    FIG_NAME = archivo + '.png'
    LABEL_1 = "... electrode"
    LABEL_2 = "... electrolyte"
    LABEL_3 = "... reaction"
    LABEL_4 = "line (fit)"
    LABEL_5 = "points (experimental data)"

    print >>f, "f(x)= 0.012345 + a*exp(-c*x) + b*exp(-d*x)"
    print >>f, "a=-0.0015"
    print >>f, "b=-0.10"
    print >>f, "c=1"
    print >>f, "d=1"

    print >>f, "fit [0.0:1][] f(x) '%s' via a, b, c, d" % archivo

    print >>f, "set terminal png"
    print >>f, "set output 'plot-include1.png'"
    print >>f, "set label 1 \"%s\" at graph 0.5,0.60" % LABEL_1
    print >>f, "set label 2 \"%s\" at graph 0.5,0.50" % LABEL_2
    print >>f, "set label 3 \"%s\" at graph 0.5,0.40" % LABEL_3
    print >>f, "set label 4 \"%s\" at graph 0.6,0.90" % LABEL_4
    print >>f, "set label 5 \"%s\" at graph 0.6,0.85" % LABEL_5
    print >>f, "set xlabel \"time/s\""
    print >>f, "set ylabel \"Current / A cm-2\""
    print >>f, "set fit"
    print >>f, "set fit logfile"
    print >>f, "set fit errorvariables"
    print >>f, "set border 3"
    print >>f, "set xtics 0.10"
    print >>f, "set xtics nomirror"
    print >>f, "set ytics nomirror"
    print >>f, "plot [0.0:1][0:] f(x) t '' w l lt 2 lw 2, '%s' t '' with points pt 6 ps 2" % archivo

    f.flush()


if __name__ == '__main__':
    SUFFIX = 'Data'
    import os

    all = filter(lambda f: f.endswith(SUFFIX), os.listdir('.'))

    for archivo in all:
        fit_data(archivo)

    print r'\begin{center}\includegraphics[width=0.8\linewidth]{plot-include1.png}\end{center}'
```

\end{python}



Different physical models can be tested, and the resulting parameters can be stored in a file:

1- Perform the experimental measurement at different temperatures, potentials and electrodes

2- Give the appropriate name to the files (filename:*date_temperature_potential_electrode*)

3- Fit experimental data to a given model.

4- The following example generates a png file and dumps the *"potential"*, *"a"* and *"b"* parameters to a txt file.

```
\begin{python}
import sys
from collections import defaultdict

#This piece of code was written by Leoncio J.E. L\'opez (anicholo at gmail dot com)
def fit_data(archivo):
    f = os.popen('gnuplot' ,'w')

    FIG_NAME = archivo + '.png'
    LABEL_1 = "Data"
    LABEL_2 = "electrolyte"
    LABEL_3 = "linear regression"

    print >>f, "set terminal png #font \"c059013l.pfb\" 9 enhanced"
    print >>f, "set output '%s'" % FIG_NAME
    print >>f, "set label 1 \"%s\" at graph 0.6,0.89" % LABEL_1
    print >>f, "set label 2 \"%s\" at graph 0.6,0.85" % LABEL_2
```

```
    print >>f, "set label 3 \"%s\" at graph 0.6,0.80" % LABEL_3
    print >>f, "set xlabel \"t^0.5 [time^0.5/s^0.5]\""
    print >>f, "set ylabel \"Current / A cm-2\""
    print >>f, "set fit"
    print >>f, "set fit logfile"
    print >>f, "set fit errorvariables"
    print >>f, "f(x)= a + b*(x)"
    print >>f, "a=-0.0015"
    print >>f, "b=-0.10"
    print >>f, "fit [0.0:0.18][] f(x) '%s' via a, b" % archivo
    print >>f, "plot [0.0:1][0:] f(x) w l, '%s' with points" % archivo
    f.flush()

#This piece of code was written by Natalia Bidart (nataliabidart at gmail dot com)
def parse_fit_log():
    data = defaultdict(list)
    f = open('fit.log')
    line = f.readline()
    # parse data
    while line != '':
        in_filename = out_a_filename = out_b_filename = None
        a = b = None
        a_error = b_error = None
        date = temp = potential = electrode = None

        if line.startswith('FIT: '): # new entry
            filename = line[line.index(" '") + 2:line.index(".txt'")]
            date, temp, electrode = filename.split('_')
            temp, potential = temp.split('m')
            try:
                potential = int(potential)
            except ValueError:
                potential = int(filter(str.isdigit, potential))
            potential = (potential * -1) / 1000.

            # build filenames for a and for b
            out_a_filename = '_'.join((date, temp, electrode, 'a'))
            out_b_filename = '_'.join((date, temp, electrode, 'b'))

            # ignore empty lines until the actual data is read
            line = f.readline()
            while not line.startswith('='):
                line = f.readline()

            line = f.readline() # this line is empty

            line = f.readline().split() # line with a values
            _, _, a, _, a_error, _ = line

            line = f.readline().split() # line with b values
            _, _, b, _, b_error,_ = line

            # append values to further agreggate per filename
            data[out_a_filename].append([potential, float(a), float(a_error)])
            data[out_b_filename].append([potential, float(b), float(b_error)])
```

```
        line = f.readline()
    f.close()

    # write values to filenames
    for filename, values in data.iteritems():
        f = open(filename + '.txt', 'w')
        values.sort()
        for value in values:
            f.write('%.10f %.10f %.10f\n' % tuple(value))
        f.close()

if __name__ == '__main__':
    SUFFIX = 'Data'
    import os

    all = filter(lambda f: f.endswith(SUFFIX + '.txt'), os.listdir('.'))

    for archivo in all:
        fit_data(archivo)

    parse_fit_log()

    sys.exit(0)

\end{python}
```