pytoolsPlot.py – Manual (version 0.6)

by ullix, April 2018

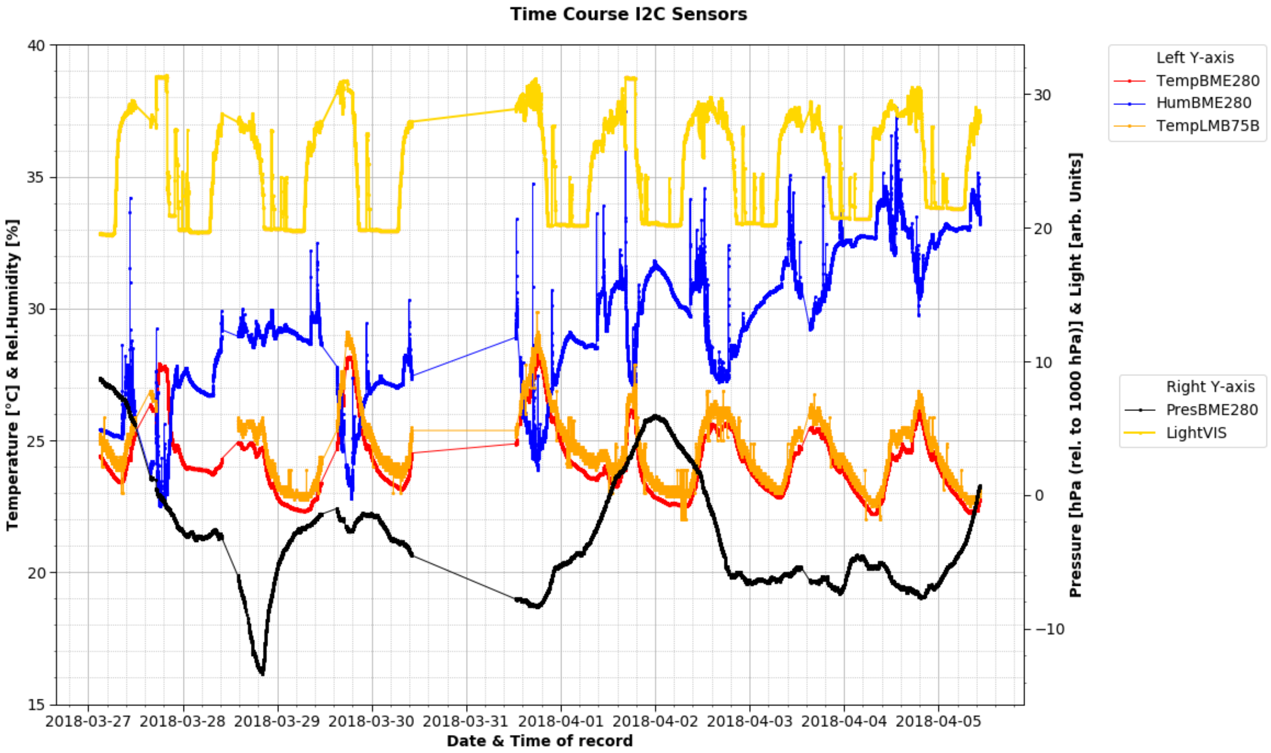
Software and documents available for download at https://sourceforge.net/projects/i2cpytools/

pytoolsPlot.py is software to plot data from any file in CSV (Comma Separated Values) format. It is based on Python3 scripts.

This is an example for data in this format (lines starting with '#' are comments, which are ignored for plotting); shown are 7 columns of data, with 3 records of data points in each:

```
#   Index, Date&Time,            T,      P,        H,       T,      L
    12990, 2018-03-14 06:17:22,  21.89,  1004.59,  31.51,   23.08,  103.06,
    12991, 2018-03-14 06:17:25,  21.90,  1004.62,  31.47,   23.08,  102.96,
    12992, 2018-03-14 06:17:28,  21.89,  1004.62,  31.48,   23.09,  103.13,
```

Such data might result in this plot:



Note: the spikes in the blue curve (humidity from BME280 sensor) and yellow curve (light from TSL2591 sensor) are real; the flutter on the orange curve (temperature from LM75B sensor) is not. For the latter, see below.

The above data are taken from the log files of **I2Cpytools**, and while **pytoolsPlot.py** is a helpful companion to I2Cpytools, **pytoolsPlot.py** is stand-alone software.

Beyond a standard Python3 installation, **pytoolsPlot.py** needs only the modules matplotlib and numpy, and therefor should run on any operating system Linux, Windows, Mac and else.

Starting

In a terminal window enter:

```
path/to/pytoolsPlot.py  /path/to/mycsvfile
```

When the datafile has become very big it may take some time to get the plot, while it might suffice to show only the last N points. This can be had by limiting the plot using the ‘-l N’ (l as in last) option; e.g. to show the last 5000 records only, enter:

```
path/to/pytoolsPlot.py -l 5000 /path/to/mycsvfile
```

Stopping

The program is stopped with CTRL-C from the terminal from where the program was started, or by closing the graph window or by CTRL-W or Q (like quit) when in the graph window.

Manipulating the Graph

The graph shows a set of icons. Rest your mouse pointer over an icon to read the description of the possible action. You can pan and zoom the graph, change its layout, edit the X and Y axis (e.g. change form Lin to Log!), and save the graph as a graphic file in whatever format available on your system.

Configuration

The layout of the plot created by **pytoolsPlot.py** can be configured in the accompanying default configuration file **pytoolsPlot.cfg** in subdirectory **cfg.**

Using the configuration file is not a requirement, but without it the plot result may be satisfactory only with few data columns and few data points.

Any number of configuration files can be defined and used for individual plots by selecting them with the ‘-c configfile’ option on the command line:

```
path/to/pytoolsPlot.py -c /path/to/myconfigfile /path/to/mycsvfile
```

More details on the configuration options are found in the configuration file itself; its content is shown in the appendix.

Two more configuration files are provided as examples: The **I2Csensors.cfg** configuration file is used for a run with **I2Cpytools** using three I2C sensors LM75B, BME280, TSL2591, plus one LED matrix HT16K33. The figure above is produced with this config file.

The file **TempVsTemp.cfg** is used to plot one variable against another one from the same records. An example plot is shown below.

Without a configuration file:

- Default labels will be used, like ‘Title’, ‘Xlabel’, etc

- All data columns will be plotted, as long as they can be interpreted as numbers.
  In the example above, the Date&Time column would not be shown, as without the info from the configuration file, it will not be known how to interpret this text.
- The X-axis will be taken as consecutive numbers 1,2,3, … N, where N is the total number of records.
- Odd columns will be plotted on the left Y-axis, even ones on the right Y-axis.
- The labels for the data columns will be created as Value1, Value2, …, ValueN.
- The colors of lines and markers will be chosen from the set of 10 default colors.
- Line width and marker size are defaults.

With a configuration file:

- The labels for graph and axis can be defined with meaningful names like ‘Temperature’.
- Data columns can be excluded from plotting.
- An X-axis can be chosen.
- A Date&Time column can be used as X-axis. The axis can be labeled with a Date&Time stamp, or it can be labeled e.g. as time since first record in sec, min, hours, days.
- For each data column it can be chosen: to plot on left or right Y-axis, label text, color, line width and marker size.
- The marker size can be auto-scaled. The markers are bigger with few data points and very small with many, which is helpful to handle large data sets.
- The scale of both Y-axis can be preset, or kept at auto-scale
- Values of the data columns can be scaled with a math formula, like
  ‘col – 1000’, or ‘SQRT(col) * 5 + 100’, or ‘ABS((LOG10(COL)+1000)/3.14) + 10’.
  See paragraph ‘scale’ in the configuration file for more details.
- The size and position of the window can be set.

Missing Values

pytoolsPlot.py recognizes missing values and by default ignores them in the plots. The missing value default is ‘nan’ (without the quotes, stands for: not-a-number) in the records. You can redefine the missing value in the configuration file under ‘missingvalue’. Remember to provide them in the exact format they appear in the CSV file; e.g. ‘-999’ may be saved as ‘-999.000’!

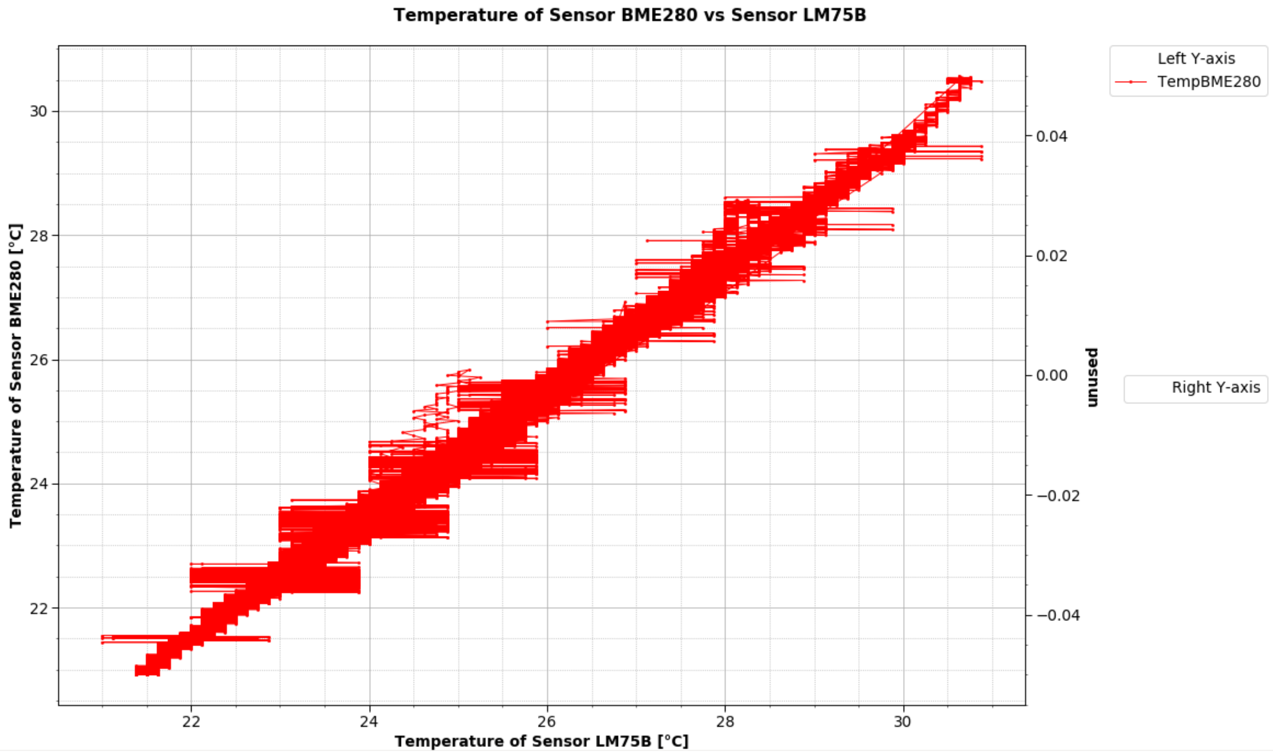
If your missing values are numbers – like -999 – and you want them not ignored but plotted, then change ‘missingaction’ in the configuration file from ‘ignore’ to ‘none’.

This is a segment from a CSV file created with **I2Cpytools** with highlighting of the missing values:

```
2018-03-25 09:23:47,    24.7500,    26.9400,      1.9916,   4041.0000,   2029.0000
2018-03-25 09:24:07,    24.7500,    27.1200,      1.9912,   4068.0000,   2043.0000
2018-03-25 09:24:27,    25.4710,    24.8750,         nan,         nan,         nan
```

Example Plot Variable versus Variable

While data produced with a logger will typically be shown versus a time axis, it is sometimes helpful to show one of the variables versus another, as shown in the next figure:



The data are the same as used in the first figure, but this time plotted with config file **TempVsTemp.cfg.** The plot of the variable ‘temperature from sensor BME280’ versus the ‘temperature from sensor LM75B’ reveals a problem of the latter sensor: whenever the temperature measured by the LM75B is at an integer number (22,23,24,…) it expresses a strange flutter; probably due to some bug in its firmware.

Package

The content is:

- **pytoolsPlot.py** and its configuration default file **pytoolsPlot.cfg**, plus example files **I2Csensors.cfg** and **TempVsTemp.cfg**
- **pytoolsPlot.py-Manual.pdf**
- data file **segment.log**, a subset of what was used here

Help Info

To get help info on the program enter on the command line:

```
path/to/pytoolsPlot.py  -h
```

You will get a printout like:

```
Usage:  pytoolsPlot.py [Options] Datafile
Options:
    -h, --help          Show this help and exit
    -V, --Version       Show version status and exit
    -c, --config name   Set the configuration file to name;
                        name is the configuration file
                        including path.
                        Default is 'cfg/pytoolsPlot.cfg'
    -l, --last N        Plot only the last N records

Datafile                File must be of type CSV
                        (Comma Separated Values)
                        If data file is not in same
                        directory as program, a relatve
                        or absolute path must be given:
                        e.g.: path/to/csvfile4plotting
```

Appendix: Content of Configuration File

```
# Configuration file for CSV plotter 'pytoolsPlot.py'
#
# Download from:  https://sourceforge.net/projects/i2cpytools/
# author:         ullix
# copyright:      Copyright 2016
# license:        GPL
#
#
# All options are needed. If they are not specified here, then a default
# will be created in the program and used.
#
# title           A title for the figure
#                 Content:    <text>
#                 Default:    Title
#
# xlabel:         A label for the X-axis
#                 Content:    <text>
#                 Default:    XLabel
#
# ylabelleft:     A label for the left Y-axis
#                 Content:    <text>
#                 Default:    YLabelLeft
#
# ylabelright:    A label for the right Y-axis
#               Content:    <text>
#               Default:    YLabelRight
#
# yscaleleft:   Min and max limits for the left Y-axis if not none
#               Both values must be specified, or the setting will be ignored
#                           Ymin    , Ymax
#               Content:    <number>, <number>
#               Default:    none, none
#
# yscaleright:  Min and max limits for the right Y-axis if not none
#               Both values must be specified, or the setting will be ignored
#                           Ymin    , Ymax
#               Content:    <number>, <number>
#               Default:    none, none
#
# window:       Geometry of the window: X-position, Y-position, X-width, Y-height
#               X/Y-Position 0, 0 is the top left corner of the desktop
#               Content:    <number>, <number>, <number>, <number>
#               Default:    0, 0, 1200, 750
#
# markerscale:  Allows scaling of the marker size depending on the number of
#               data points. With few points the line color is hard to
#               distinguish; with many points, the markers obscure the lines.
#               This provides a compromise, which can be switched off
#               Content:    auto|none
#               Default:    auto
#
# missingvalue: This value is a placeholder for a real value, which actually
#               could not be measured. The default is 'nan' (not a number).
#               Content:    <number>|<text>
#               Default:    nan
#
# missingaction:The action to take when a missing value is encountered. An
#               'ignore' will exclude the data point from being shown.
#               Content:    none|ignore
#               Default:    ignore
#
# 1: ... N:     Index for the value-series extracted from the CSV file.
#               1st value is #1 ('0' (zero) has a special purpose; don't use!)
#               Options specify the values' function (X,Y axis, date/time-axis),
#               any scaling, names, and color and size property of lines.
#               Content: purpose, scale, range, name, color, linewidth, markersize
#               Default: y<l|r>, none,  none,  ValueI, <colorI>,    2,    2
#                        <l|r>    : is alternating between 'l' and 'r'
#                        I        : is the number 1 ... N of the series
#                        <colorI> : uses successive color names from default list
#
# Options for the data series:
#
#  purpose:     x|time|date|yl|yr|ig[nore]
#               Specifies series to be used as one of the X or Y-axis, or be
#               ignored.
#               If no axis is designated x or time or date, then a sequence
#               1, 2, 3, ... will be used as X-axis.
#               If time or date is used as X-axis, the preferred format is
#               "2018-03-08 10:22:33". The parser will attempt to recognize
#               other formats, but it may fail.
#               If date is used, the options scale and ref will be ignored.
#
#  scale:       An option to scale the data with a math formula. May also be
#               useful to shift the graph up or down. The formula may include
#               math functions:
#               + - * / :   basic math
#               **:         raise to the power, e.g. 2**8  (=256)
#               COL:        the data of that column
#               LOG:        log to base e; natural log
#               LOG10:      log to base 10
#               LOG2:       log to base 2
#               SIN:        sine
#               COS:        cosine
#               TAN:        tangent
#               SQRT:       square root
#               CBRT:       cube root
#               ABS:        absolute value
#               example:    col - 1000
#               example:    SQRT(col) * 5 + 100
#               example:    ABS((LOG10(COL)+1000)/3.14) + 10
#               Content:    none | <math formula>
#               Default:    none
#
#  ref:         Relevant mostly for time data. Subtracts 1st data point from all
#               others; the time becomes relative to the start time. The X-axis
#               may be labelled 'time since 1st record'.
#               Content:    none | rel
#               Default:    none
#
#  name:        Any text to be shown as label for the data series. No more
#               than 10 characters wiil be displayed.
#               Content:    <text>
#               Default:    ValueI (with I being the series number 1 ... N)
#
#  color:       The name of the color to be used for lines and markers. A list
#               of allowed color names is found at
#               https://matplotlib.org/examples/color/named_colors.html
#               In addition, colors can be given as values in hexadecimal
#               notation, like #FEBA75 (in RGB order).
#               Internally these 10 color names are used in this order: black,
#               red, orange, gold, green, blue, cyan, magenta, purple, pink
#               Content:    <colorname>
#               Default:    ColorIM from list for variable I, IM = I mod 10
#
#  width:       The width of the line. 1 is very thin, 0 is invisible
#               Content:    <number>    (integer or floating point)
#               Default:    2
#
#  marker:      The size of the marker for a datapoint. A marker size of x has
#               the same diameter as a line of with x is wide, i.e. the marker
#               would not be seen on the line!
#               If markerscale is the to 'auto', the marker size will be
#               adjusted, see markerscale.
#               Content:    <number>    (integer or floating point)
#               Default:    2
#
# Note: the more data need to be plotted, the thinner the lines and markers
#       should be. For datapoints up to about 100 a line=2 & marker=5 setting
#       may work. For very many data points line=1 and marker=0 may be better
#
#
# Example configuration for Time, Pressure and Humidity Values
# as used for sensors BME280, LM75B, TSL2591  for T, P, H, Light
#
# a sample from a shortend CSV file:
#   Index, Date&Time,            T,       P,          H,        T,      L
#   12990, 2018-03-14 06:17:22,  21.89,   1004.59,   31.51,    23.08,  103.06,
#   12991, 2018-03-14 06:17:25,  21.90,   1004.62,   31.47,    23.08,  102.96,
#   12992, 2018-03-14 06:17:28,  21.89,   1004.62,   31.48,    23.09,  103.13,

title:          Title
xlabel:         Date
ylabelleft:     ValuesLeft
ylabelright:    ValuesRight
yscaleleft:     none, none
yscaleright:    none, none
window:         0,    0,    1200,  750
markerscale:    auto
missingvalue:   nan
missingaction:  ignore


#  purpose,  scale,   ref,      name                   color   width   Marker
#1:  x,      none,    none,     none,                  black,    2.0,     2
1:  ig,      none,    none,     none,                  black,    2.0,     2
#2:  time,   /3600,    rel
#2:  time,    none,    rel
2:  date
3:  yl,      none,    none,     Value1,                black,    2.0,     2
4:  yr,      none,    none,     Value2,                red,      2.0,     2
5:  ig,
6:  ig,
7:  ig,
8:  ig,
9:  ig,
10: ig,
```