POZNAN UNIVERSITY OF TECHNOLOGY ACADEMIC JOURNALS
No 69 Electrical Engineering 2012

Andrzej KSIĄŻKIEWICZ*

USE OF FREE AND OPEN-SOURCE SOFTWARE IN ANALYSING EXPERIMENTAL RESEARCH DATA

During research process there is often large amount of experimental data acquired. These data can be obtain from all kinds of measurement devices, such as oscilloscopes for example. Most of the time the raw data collected cannot be used directly but has to be processed. That process can involve filtering the data, numerical computations to acquire new information and finally making a visual representation of the results in form of a graph. All those tasks can be carried out with the uses of specialized, scientific computer software which is free and open-sourced. Examples of such programs are Scilab, used for numerical calculations, and gnuplot, used for creating both simple and complex graphs. Both of the programs are described in this article with examples of how to used them.

1. WHAT IS FREE AND OPEN-SOURCE SOFTWARE

Every computer programs comes with a licences that describes how the user may uses that program, that is his rights. For example there are programs that can be installed only on one computer by one user and there are program that allow the user unlimited amount of installations of the program. Over rights may describe the freedom to change the program to the user expectations. For a program to meet the requirements of free software the licence of the program must uphold four freedoms as described by Free Software Foundation. In this context free software refers to the freedom of uses and not the software price which doesn't have to be distributed at no charge. The four essential freedoms are [1]:

– The freedom to run the program, for any purpose (freedom 0).
– The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
– The freedom to redistribute copies so you can help your neighbour (freedom 2).
– The freedom to distribute copies of your modified versions to others (freedom 3). By doing this you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

* Poznan University of Technology.

There is a different approach to the freedom of software which is realized by The Open Source Initiative. The OSI is actively involved in Open Source community-building, education, and public advocacy to promote awareness and the importance of non-proprietary software [2]. The Open Source Definition describes user rights in a different manner but with the essential freedoms stay the same.

2. EXAMPLES OF SCIENTIFIC FREE AND OPEN-SOURCE SOFTWARE

There are many examples of free software in many fields of science and engineering. In this article two programs are described, that can be used in a researcher tools – Scilab and gnuplot. Scilab if developed and maintained by the Scilab Consortium [3], which currently includes 24 members (both industrials and academics). The latest version of this software is governed by the CeCILL license (GPL compatible) abiding by the rules of distribution of free software. Gnuplot is a portable command-line driven graphing utility for Linux, OS/2, MS Windows, OSX, VMS, and many other platforms [4]. The source code is copyrighted but freely distributed (i.e., free of charge). It was originally created to allow scientists and students to visualize mathematical functions and data interactively, but has grown to support many non-interactive uses such as web scripting.

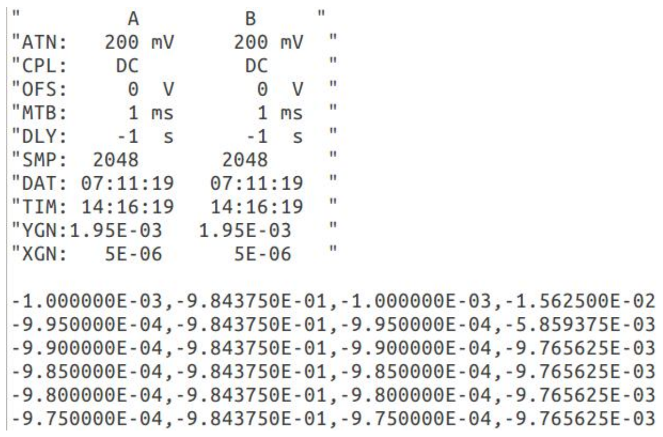
3. SCILAB

Key features of Scilab are: maths and simulation, 2-D and 3-D visualization, optimization, statistics, control system design and analysis, signal processing and application development. In this article usage of this software to calculate the let-through energy (eq. 1) during a short circuit current in a circuit protected by a modular circuit breaker (MCB) is presented. This example includes reading of multiple raw data files, data filtration, numerical calculations and creating a file containing the results. The test circuit will not be described here because it is not relevant here.

$$J = \int i^2 dt \tag{1}$$

In Figure 1 an example of the data file obtained from the oscilloscope used in the research is presented. At the beginning of this file there are informations about the oscilloscope settings used in the test. Later there are four columns containing raw data gathered during the test. The first and third column are identical and represent the time, for the first and second channel of the oscilloscope. The second column is voltage and the fourth is the current measured. Twenty-nine test were carried out and the same amount of raw data files were created. In order to process such amount of data a script for Scilab was created, that controlled the work flow of the program.

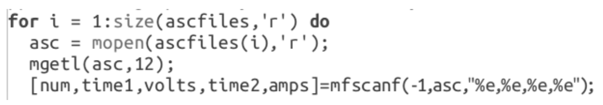
The first step was to read the gathered raw data (Fig. 2). It was done in a for loop, starting at file 1 and ending with the last file in the directory. Files containing the data were listed in ascfiles variable and the function size(ascfiles,'r') returned the file count. Next each files was opened and the first twelve lines, containing the oscilloscope settings, were read and discarded. And last by using the mfscanf function the data was read from the file and put into four variables – time1, time2, volts, amps. Each variable corresponds with the proper data file column.



```
"          A           B      "
"ATN:   200 mV      200 mV  "
"CPL:     DC          DC      "
"OFS:      0  V        0  V  "
"MTB:      1 ms        1 ms  "
"DLY:     -1  s       -1  s  "
"SMP:  2048        2048      "
"DAT: 07:11:19    07:11:19  "
"TIM: 14:16:19    14:16:19  "
"YGN:1.95E-03    1.95E-03   "
"XGN:   5E-06       5E-06   "

-1.000000E-03,-9.843750E-01,-1.000000E-03,-1.562500E-02
-9.950000E-04,-9.843750E-01,-9.950000E-04,-5.859375E-03
-9.900000E-04,-9.843750E-01,-9.900000E-04,-9.765625E-03
-9.850000E-04,-9.843750E-01,-9.850000E-04,-9.765625E-03
-9.800000E-04,-9.843750E-01,-9.800000E-04,-9.765625E-03
-9.750000E-04,-9.843750E-01,-9.750000E-04,-9.765625E-03
```

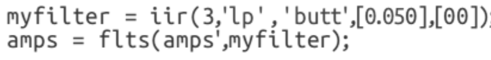
Fig. 1. Raw data file obtained from the oscilloscope



```
for i = 1:size(ascfiles,'r') do
  asc = mopen(ascfiles(i),'r');
  mgetl(asc,12);
  [num,time1,volts,time2,amps]=mfscanf(-1,asc,"%e,%e,%e,%e");
```

Fig. 2. Reading of raw data files

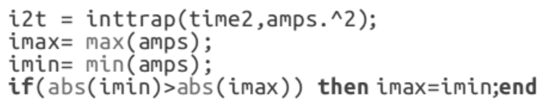
Later the input data had to be filter by a low-pass filter. In order to do it two functions were used as shown in figure 3. The first function creates a low-pass 'lp' Butterworth 'butt' digital filter with a 50 Hz cut-off frequency. The second function applies the filter to the raw data.



```
myfilter = iir(3,'lp','butt',[0.050],[00]);
amps = flts(amps',myfilter);
```

Fig. 3. Filtering the input data

Last the proper calculations are executed (Fig. 4). The first function performs an integration of experimental data by trapezoidal interpolation. Current values stored in the amps variable are raised to the second power according to equation 1. The next three lines are used to calculate the peak current.



```
i2t = inttrap(time2,amps.^2);
imax= max(amps);
imin= min(amps);
if(abs(imin)>abs(imax)) then imax=imin;end
```

Fig. 4. Calculating the let-through energy and maximum current

Finally the results are written to an output file and can be processed further. In this example only the most important parts of Scilab script used where presented.

4. GNUPLOT

Gnuplot supports many types of plots in either 2D and 3D. It can draw using lines, points, boxes, contours, vector fields, surfaces, and various associated text. It also supports various specialized plot types. Chart presenting results form computer simulation of light intensity in a room is discussed in this example (Fig. 5). Gnuplot uses special commands that control the chart generation process and they are stored in a plain text file. File used to generated the example charts has for instance 155 lines of code. These commands control every aspect of the chart starting from the output file format, font used, line types and their colours, labels, axis, plot styles and many more.

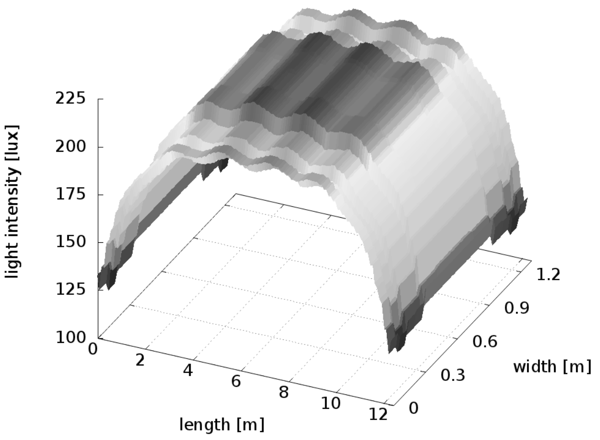


Fig. 5. 3D chart created from the simulation data

In this article only some chosen most important command are presented. For a complete list of available command please refer to programs manual [5]. An important part of the chart is the labels and rang for the three axis. Setup of the x an y axis is presented on Figure 6.

```
setxlabel"length [m]"
setxlabeloffset character 1, -1, 0 font "" textcolor lt -1 norotate
setxrange[ 0.00000 : 12.4000 ] noreverse nowriteback
setylabel"width [m]"
setylabeloffset character 1.5, -1, 0 font "" textcolor lt -1 rotateby90
setyrange[ 0.00000 : 1.30000 ] noreverse nowriteback
```

Fig. 6. Setting the x and y label text, coordinates and scale range

Setxlabel command with a following text sets the x axis label, *setxlabeloffset* is used to set the exact location of the label, font used and rotation of the label. The *setxrange* command sets the maximum and minimum range of the axis. The y and z labels are controlled with similar commands. The chart presented in figure 5 can be presented in a top-down view with a colorbox representing the light intensity displayed (Fig. 7).

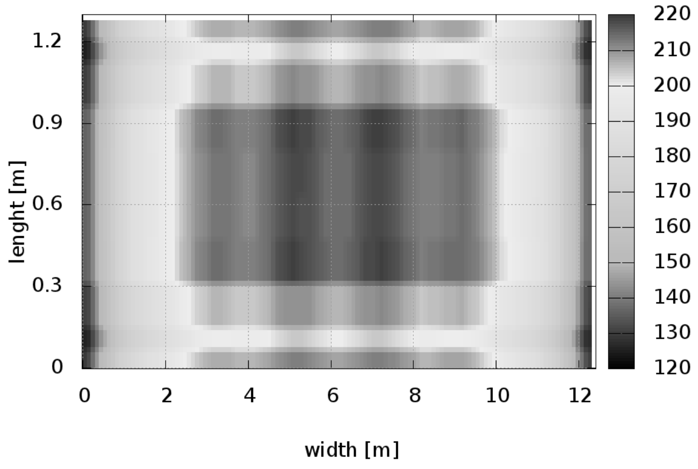


Fig. 7. 2D view from top of previous chart with colobox

To set-up the proper palette and scale the *setpalette* command is used, as shown in figure 8. This definition consists of four digits which refer to the red, green and blue values of colours used for the coloured scale. The *setcolorbox* sets the size and place of the colorbox used in the chart, as shown on the right side of Figure 7.

```
setpalette defined (0 0 0 0, 0.3 0.7843 0.7843 0, 0.8 1 10,\
    1 1 0 0 )
setcolorbox verticalorigin screen 0.9, 0.2, 0 size screen0.05,0.6,0frontbdefault
```

Fig. 8. Setting both the palette of the chart and colorbox position

5. CONCLUSION

Presented in this article are two selected programs that are free (in terms of price and freedom of use) and open-sourced. As shown in the examples both programs are feature-rich and can meet the requirements of many scientists. This programs can be used to analyse raw research data, perform complex calculations and create a print-ready graphics to present the results. Although it is necessary to devote some time in order to learn this programs the benefit for the scientist is considerable.

REFERENCES

[1] http://www.gnu.org/philosophy/free-sw.html
[2] http://www.opensource.org/about
[3] http://www.scilab.org/aboutus/consortium/members
[4] http://gnuplot.info/
[5] http://gnuplot.info/docs_4.4/gnuplot.pdf

WYKORZYSTANIE OTWARTEGO I DARMOWEGO OPROGRAMOWANIA W ANALIZIE DANYCH DOŚWIADCZALNYCH

Podczas procesu badawczego często gromadzone są znaczne ilości danych doświadczalnych. Dane te mogą być uzyskanie z wykorzystaniem wszelkiego rodzaju urządzeń pomiarowych, na przykład oscyloskopów. Przeważnie zebrane surowe dane nie nadają się do bezpośredniego wykorzystania i muszą zostać przetworzone. Proces ten może obejmować filtrowanie danych, obliczenia numeryczne, z których uzyskuje się nowe informacje, i wreszcie tworzenie wizualnej reprezentacji wyników w postaci wykresów. Wszystkie te zadania mogą być wykonywane z zastosowaniem specjalistycznego, naukowego oprogramowania, który może być darmowe i otwarto-źródłowe. Przykładami takich programów są Scilab, używany do obliczeń numerycznych, i gnuplot, używany do tworzenia zarówno prostych, jak i skomplikowanych wykresów. W tym artykule opisano oba te programy wraz z przykładami ich wykorzystania.