MC’s PLOTXY - A scientific plotting and post-processing program

A step-by-step tutorial

NEW: Modelica (Dymola and OpenModelica) compatibility!

See pag. 48.

Table of Contents

About this tutorial

This tutorial shows the basic operations to obtain the first plots from PlotXY. It shows also nearly everything you will need to make also more complex operations with the program.

Please consider that the maximum efforts have been done to make the program as intuitive as possible, therefore the learning curve should be very steep. I imagine that the duration of this tutorial for you will be around 30 minutes.

The figures shown have been made using the Microsoft Windows version of the program. Under Apple Mac and Linux they are slightly different but the elements (buttons, tables, etc.) are exactly the same. Therefore any Mac reader will be as comfortable as Windows users in following this tutorial.

This tutorial is intended for sequential reading. However, in case you strongly want to, you can jump to any of its sections by clicking on the links in the Table of Contents.

What PlotXY is and what can do for you

I propose here just a few Q&A's.

What is PlotXY? It is a program to make plots and post-processing of simulated and measured data, specifically designed for scientists and engineers.

What are the PlotXY strengths as a plotting program? It is built from the ground up having in mind serious scientific usage. For instance:

1) it is at ease with very large data: it loads easily files of large size (tens or hundreds of MB), and plots very fast variables having many samples (several thousands or even hundreds of thousands).
2) It manages effectively the scientific prefixes (“k” for “kilo”, “M, for Mega”, etc). If input data contains the relevant information, it uses the correct unit of measure (“A” for ampere, “V” for volt, etc.)

How PlotXY interacts with my other favourite programs? This program is able to read data from the well-knowns ATP simulation program, as well as from asci files (both its own “ADF” format and common versions of CSV files). For instance it reads smoothly CSV files created from the Modelica simulator OpenModelica (www.openmodelica.org). It can export plots into SVG, PNG, PDF formats.

What kind of post-processing is possible? You can make any kind of algebraic operations among the variables read from input files. You can also make the time-integral of a variable. You can also compute and plot the coefficients of the Fourier polynomial of a variable.

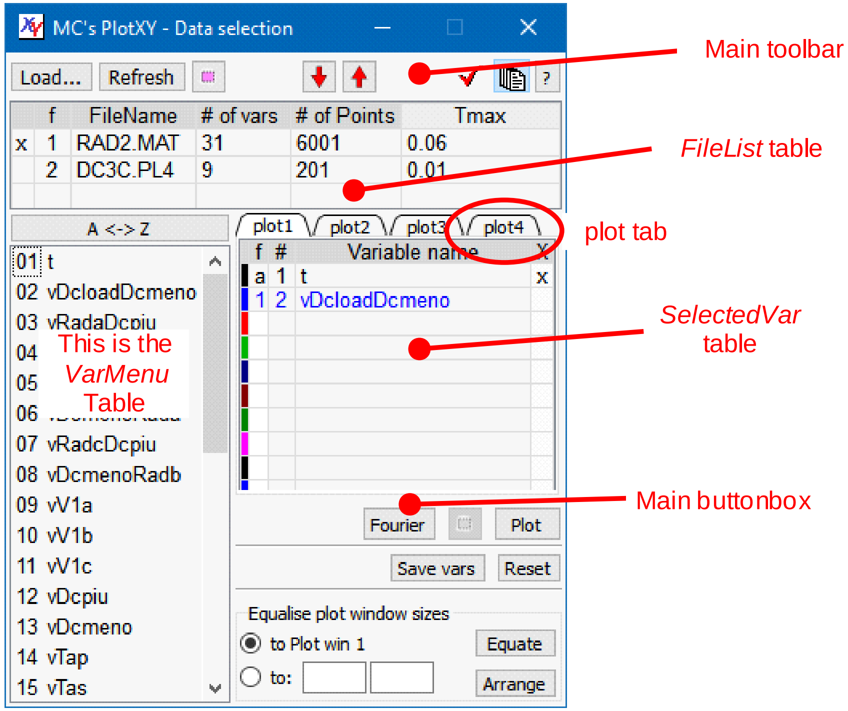
It the program free? Yes, it is completely free. It is distributed under the open source LGPL license.

Getting to know the main windows and giving names to buttons

Before starting doing things, better is to become acquainted with the three program main windows, and to give names to their buttons.

The DataSelection Window

Once PlotXY is started, the first window to appear is the one shown below, that is the main program window.



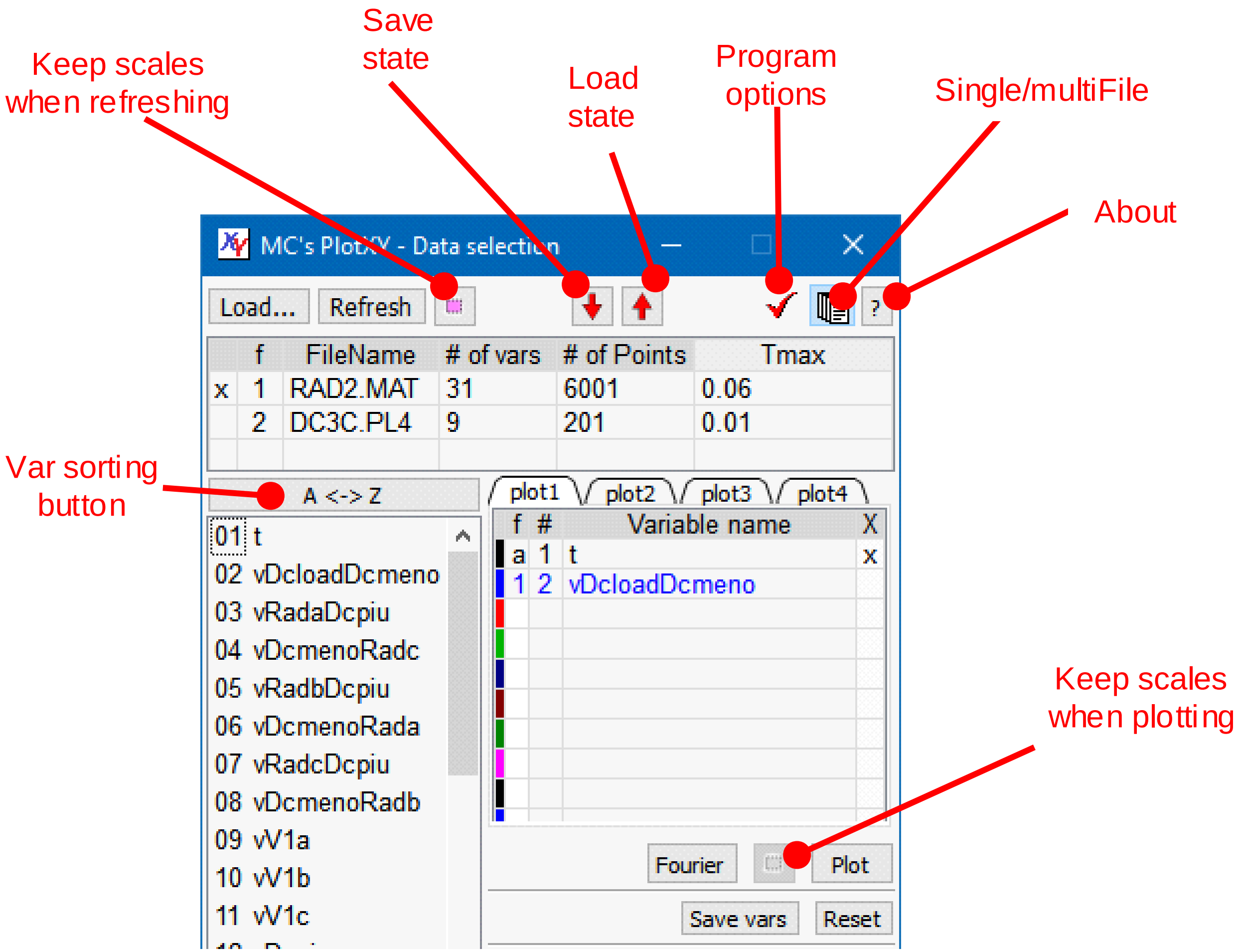
	f	FileName	# of vars	# of Points	Tmax
x	1	RAD2.MAT	31	6001	0.06
	2	DC3C.PL4	9	201	0.01

01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04
05
06
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas

f	#	Variable name	X
a	1	t	x
1	2	vDcloadDcmeno	

● to Plot win 1
○ to:

Below you can find specific names of the toolbar tools



	f	FileName	# of vars	# of Points	Tmax
x	1	RAD2.MAT	31	6001	0.06
	2	DC3C.PL4	9	201	0.01

01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04 vDcmenoRadc
05 vRadbDcpiu
06 vDcmenoRada
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c

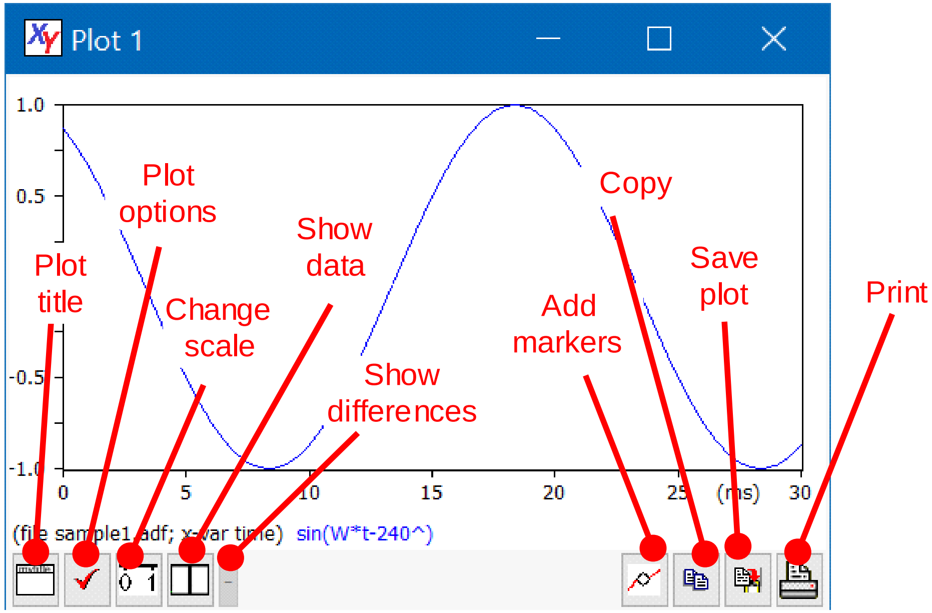
f	#	Variable name	X
a	1	t	x
1	2	vDcloadDcmeno	

The FileList Table is self-explanatory. Some explanation of the SelectedVar table columns is needed instead:

- Column “f” contains the file number. In the example above we have two files; the “u” corresponding to vDcloadDcmeno indicates that this variable comes from the file number 1. The variable on the horizontal axis, here “t”, must be in common with all the considered file, that’s why in the ”f” column, “t” row, we find the indicator “a”, which stands for “all”.
- Column “#” contains the variable number inside the considered file. Here vDcloadDcmeno is variable n. 2.

- Column “X”. This letter stands for “aXis”. In this column, we find a “x” on the variable to be put on the x-aXis, a “r” for variables to be plotted against the vertical y-aXis in case of twin vertical axis (see sect. Special plots: with twin vertical axes or X-Y plots”). If nothing is written on the “x” column for a variable, the default, i.e. left-y axis, applies.

The Plot window

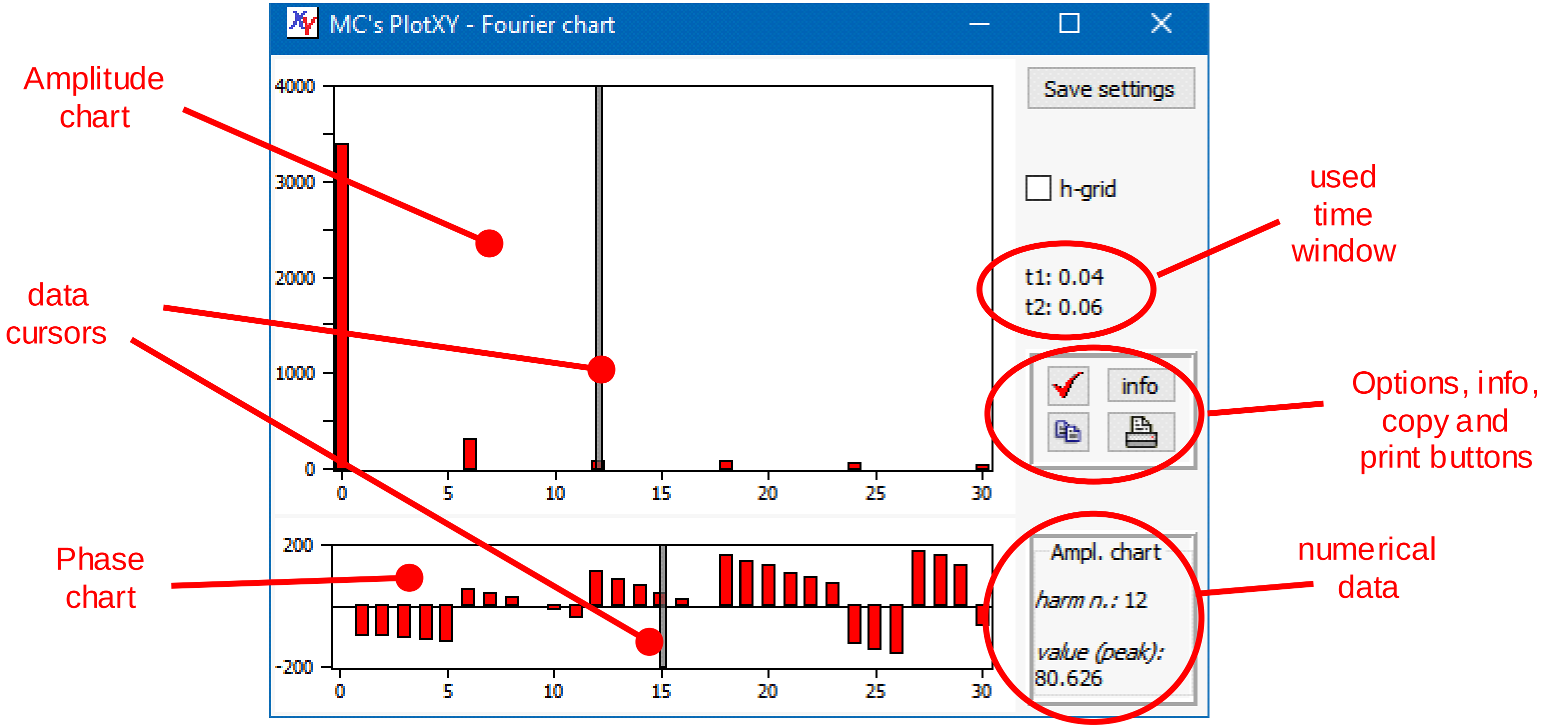


When, in the Data Selection window the plot button is clicked, the plot window is displayed.

It has the appearance shown in the picture aside (in which a simple plot from the supplied file “sample1.adf” is drawn), where names of the toolbar buttons are also given.

The Fourier chart window

The program is also able to make Fourier analysis (more precisely the i.e. Discrete Fourier Transform - DFT) of periodic signals and to display the corresponding amplitude and phase spectra. This is done using the Fourier Chart window, that is shown below along with the name of its main elements.



t1: 0.04
t2: 0.06

Ampl. chart
harm n.: 12
value (peak): 80.626

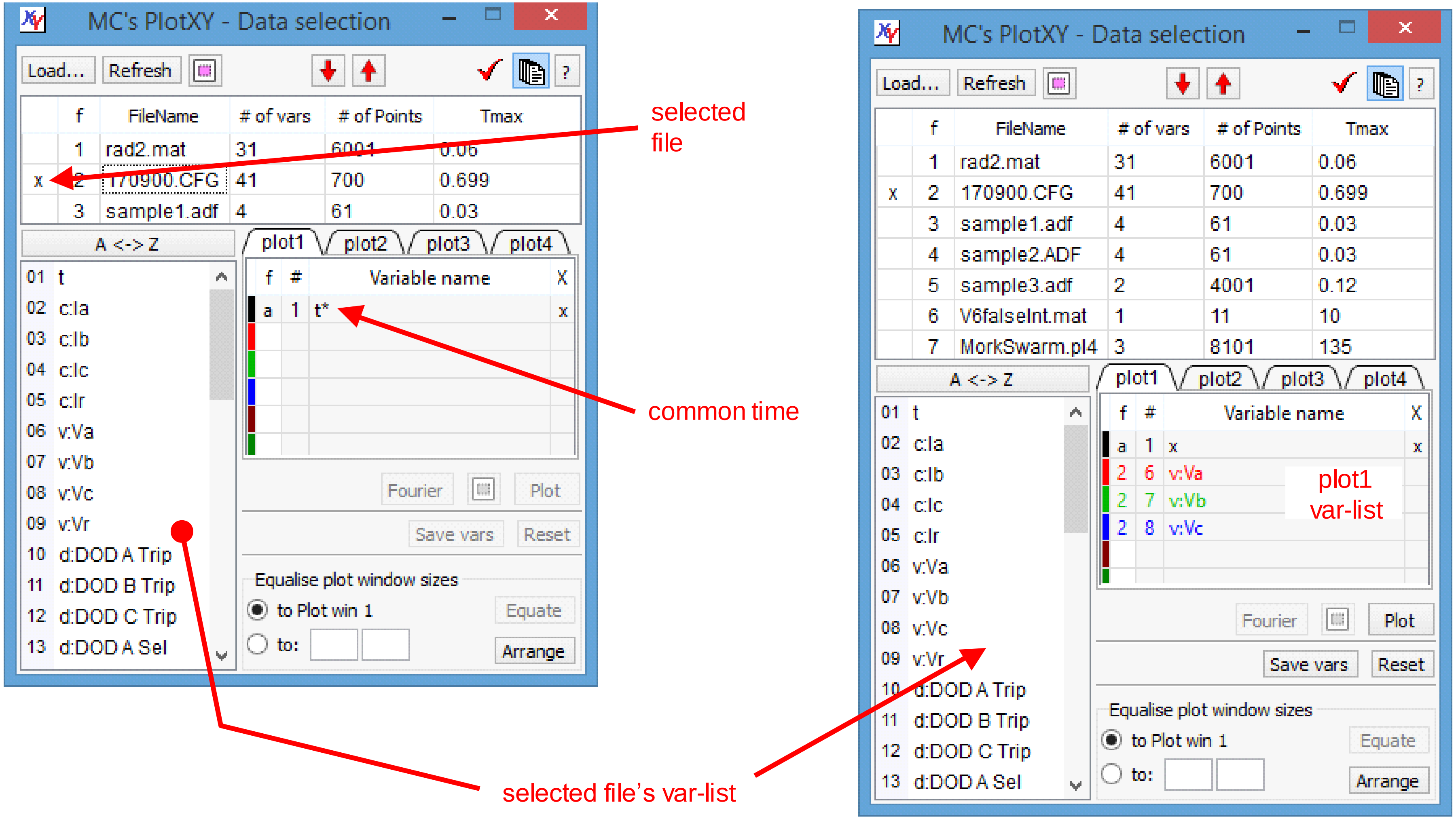
Doing the first things: a “real-life” experience

The first thing to do is to load one or more files. Here I show how it works using my own files, just to give ha hint about a real-life experience with the program. In Working together using the provided files, instead I will propose actions that you can repeat using the sample files provided.

There are two ways of doing this:

1. Using the Load… button
2. Using the drag&drop feature of the operating system: looking at the file name in their own directory, selecting one or more of them, and dragging them onto the main program window.

Once some files have been loaded the main program window can for instance have the appearance shown below at the left side of the figure (three files) or at the right side (seven different files).



	f	FileName	# of vars	# of Points	Tmax
	1	rad2.mat	31	6001	0.06
x	2	170900.CFG	41	700	0.699
	3	sample1.adf	4	61	0.03

01 t
02 c:Ia
03 c:Ib
04 c:Ic
05 c:Ir
06 v:Va
07 v:Vb
08 v:Vc
09 v:Vr
10 d:DOD A Trip
11 d:DOD B Trip
12 d:DOD C Trip
13 d:DOD A Sel

f	#	Variable name	X
a	1	t*	x

Equalise plot window sizes
● to Plot win 1
○ to:

	f	FileName	# of vars	# of Points	Tmax
	1	rad2.mat	31	6001	0.06
x	2	170900.CFG	41	700	0.699
	3	sample1.adf	4	61	0.03
	4	sample2.ADF	4	61	0.03
	5	sample3.adf	2	4001	0.12
	6	V6falseInt.mat	1	11	10
	7	MorkSwarm.pl4	3	8101	135

01 t
02 c:Ia
03 c:Ib
04 c:Ic
05 c:Ir
06 v:Va
07 v:Vb
08 v:Vc
09 v:Vr
10 d:DOD A Trip
11 d:DOD B Trip
12 d:DOD C Trip
13 d:DOD A Sel

f	#	Variable name	X
a	1	x	x
2	6	v:Va	
2	7	v:Vb	
2	8	v:Vc	

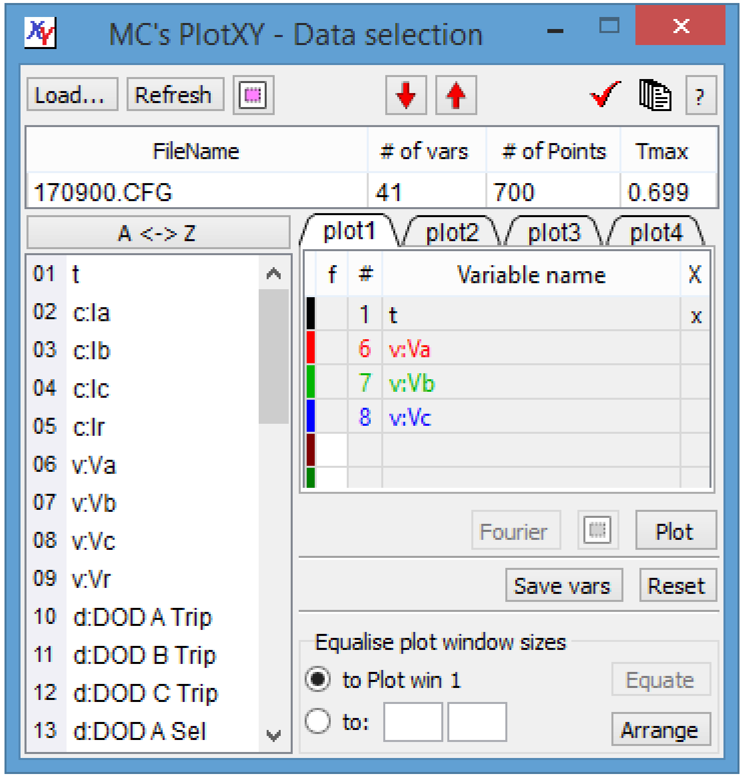
Equalise plot window sizes
● to Plot win 1
○ to:

Note that the program automatically selects the first variable as the “x variable” (the one that should by default correspond to the horizontal axis). If all the loaded files share the same name on the horizontal axis, this one will be chosen.

If all the first names begin with the same first character a name containing that character and ‘*’ will be used. For instance, if all the variables begin with ‘t’ we will have a “common time” variable indicated ad “t*” (see figure left above). Note that based on the first character the horizontal variable will be attributed an unit of measure, according with the following convention:

- Names beginning with ‘t’ will be treated as being time (unit of measure is second)
- Names beginning with ‘f’ will be treated as being frequency (unit of measure is hertz).

It the first names in the selected files var-lists do not have all the same first character, the program gives to this common variable the name “x” and gives up trying to understand which unit of measure it might have,



FileName	# of vars	# of Points	Tmax
170900.CFG	41	700	0.699

01 t
02 c:Ia
03 c:Ib
04 c:Ic
05 c:Ir
06 v:Va
07 v:Vb
08 v:Vc
09 v:Vr
10 d:DOD A Trip
11 d:DOD B Trip
12 d:DOD C Trip
13 d:DOD A Sel

f	#	Variable name	X
⬛	1	t	x
🟥	6	v:Va	
🟩	7	v:Vb	
🟦	8	v:Vc	

● to Plot win 1
○ to:

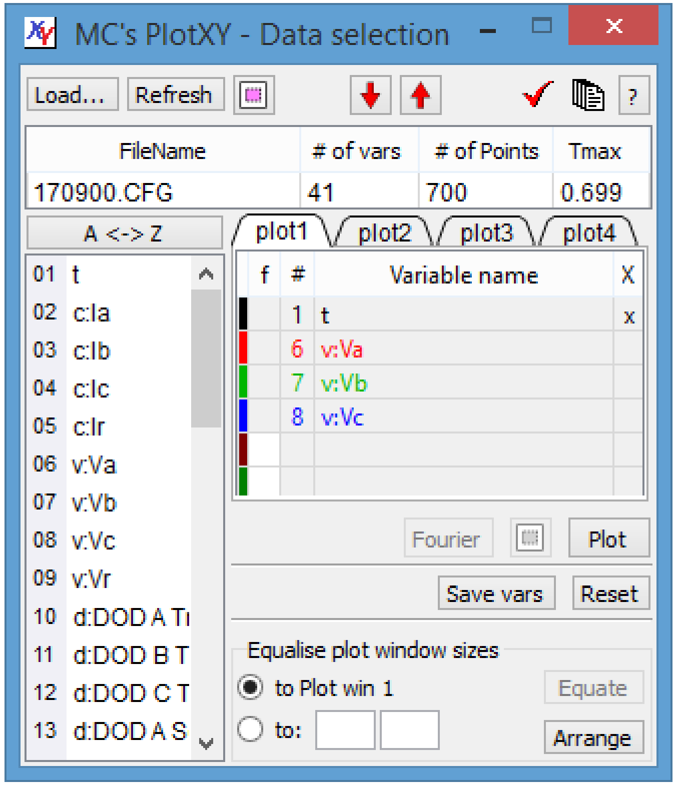
There are times, however, in which the user wants just to deal with a single file. In this case he can switch to the single-file mode, simply by clicking on the second rightmost button on the main toolbar. This way he will save valuable screen space, and have some additional features that are available only in single-file mode.

Switching into *singleFile* mode does not unload any of the already loaded files. When I clicked on the second right-most button of the toolbar from the situation shown in the right window just above here, I got the appearance shown aside (I have also reduced the window’s height by acting as usual on resizable windows).

Now it is time to select and plot variables.

From the figure aside we now select (by clicking onto the respective names) variables #2, #3, #4. The window will now appear as shown below at left.

Once the variables have been selected, they can be plot simply clicking on the “Plot” button. In my example the window shown below at right appears.

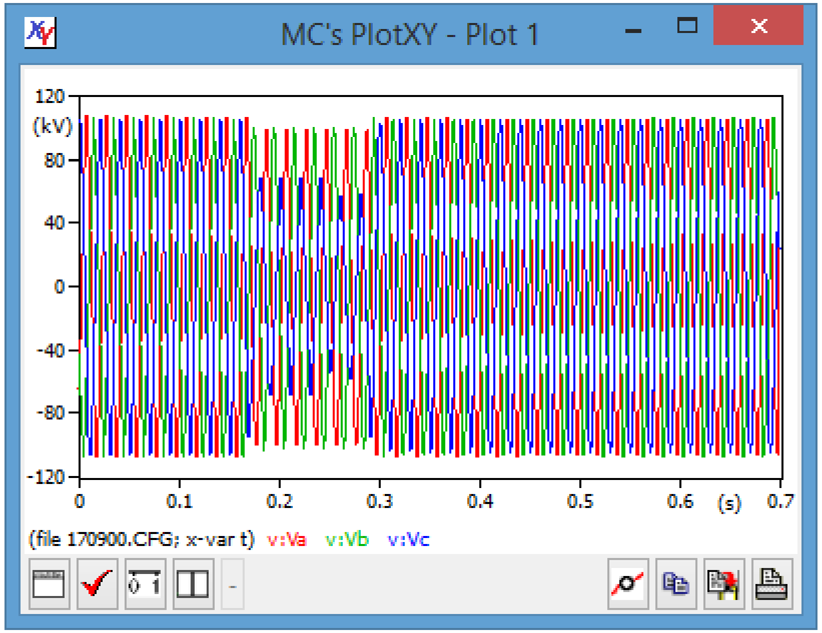


FileName	# of vars	# of Points	Tmax
170900.CFG	41	700	0.699

01 t
02 c:Ia
03 c:Ib
04 c:Ic
05 c:Ir
06 v:Va
07 v:Vb
08 v:Vc
09 v:Vr
10 d:DOD A Tı
11 d:DOD B T
12 d:DOD C T
13 d:DOD A S

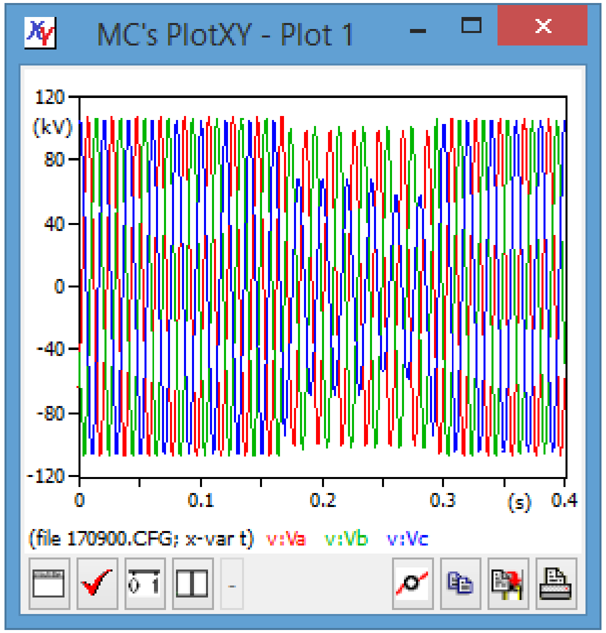
f	#	Variable name	X
	1	t	x
🟥	6	v:Va	
🟩	7	v:Vb	
🟦	8	v:Vc	

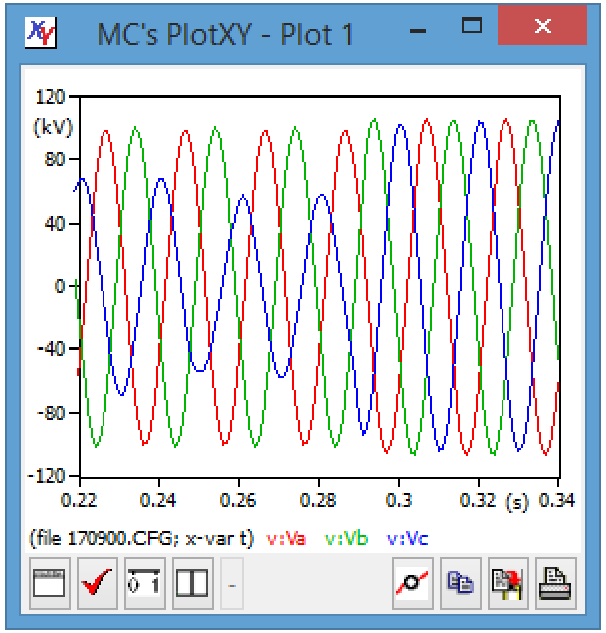
Equalise plot window sizes
● to Plot win 1
○ to:

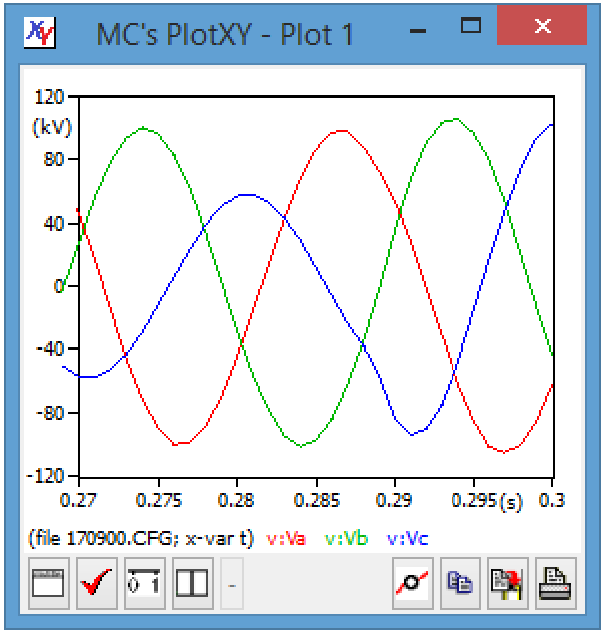


(file 170900.CFG; x-var t) v:Va v:Vb v:Vc

As often, in this case the plot is too cluttered to see details. However, it is very simple in PlotXY to zoom: simply click and drag. To zoom out, right-click and choose whether zoom just one step or totally. Below three progressive zoom levels of the same figure are shown in three windows that, for compactness were been previously horizontally shrunk.







You might have noted a few characteristics of the plots that are produced:

1) Minimum and maximum numerical values on the axes are always "round" numbers. Indeed a rather sophisticated algorithm is present inside do avoid nasty numbers on the axis such as 1.2345 or 100000 or 0.000001
2) The unit of measure of the quantities is automatically set to kV. This is based on the variable names. If the auto-naming feature is not disabled, all variables whose name begins with "v" is taken to be a voltage. Also, time is acknowledged as such and its unit set to seconds.

The rules used for auto-setting of units of measure are as in the following table:

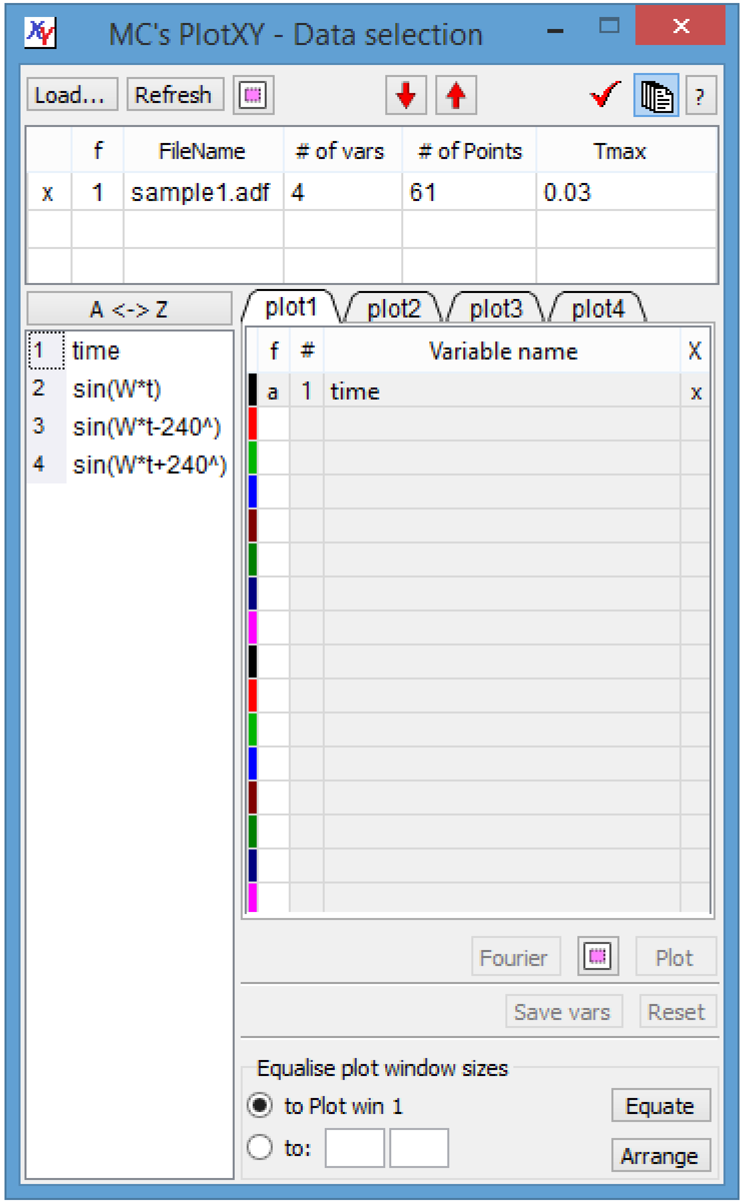
First character of name	*quantity*	*First character of name*	*quantity*
"t"	time	"a"	angle
"v"	voltage	"p"	power
"i" or "c"	current	"e"	energy

When these units of measures are used, also the standard prefixes are used as well (e.g. u for "micro", m for "milli" k for "kilo", M for "Mega", G for "Giga", etc.).

Now that you have a first idea of how the programs work and how the produced plots look like in a real-life case, let us make together some plots step-by-step using the enclosed sample files (from sample1.adf to sample3.adf).

Doing some activity using the provided files

Loading a file and doing first things



	f	FileName	# of vars	# of Points	Tmax
x	1	sample1.adf	4	61	0.03

1	time
2	sin(W*t)
3	sin(W*t-240^)
4	sin(W*t+240^)

f	#	Variable name	X
a	1	time	x

If you have gained access to this tutorial, I can assume you have already installed the program and understood how to launch the program.

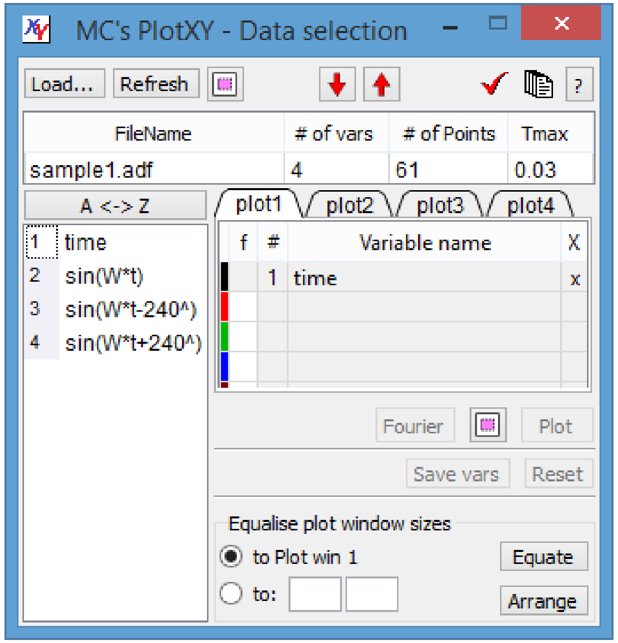
However, to launch the program the best way is to use a link you have created linking to PlotXY.exe (that you've put in a directory of your choice) and click on it.

When the file is loaded, you have two options to load a file (either using the button (Load…) or dragging a file onto the PlotXY window from the directory where it is.

Once you have loaded the program and loaded sample1.adf the program window looks like the one shown aside.

Since the file is compact and just one, we can reduce the window size to reduce its space occupation on the screen. To reduce the window size se use the usual way for resizing windows in computers using the window's borders.

Moreover, we choose to display in the file table only one file we click on the Single/multifile button in the main toolbar



FileName	# of vars	# of Points	Tmax
sample1.adf	4	61	0.03

1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)

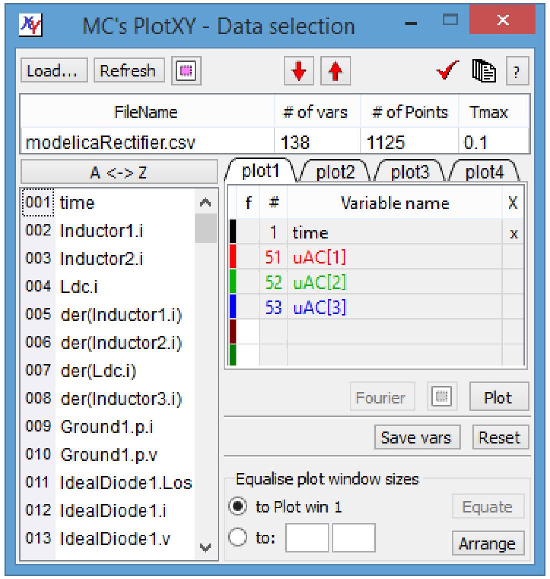
f	#	Variable name	X
	1	time	x

● to Plot win 1
○ to:

The corresponding windows will now look like as shown aside: the window is much smaller and saves much of your precious desktop area.

If you have many variables you might find it useful to sort them alphabetically in ascending or descending order using the *Var Sorting button* (three pictures below). When A<->Z is displayed the variables are listed in the same order in which they appear in their input file.

Note that sorting does not change the number attributed to each variable: for instance the selected variables remained variables N. 51, 52, 53.

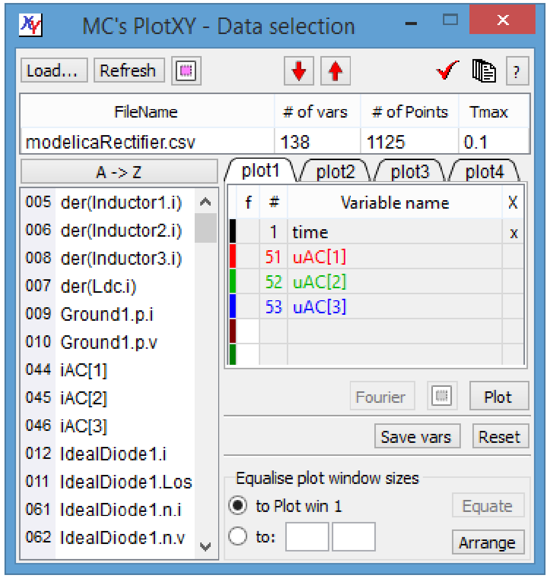


FileName	# of vars	# of Points	Tmax
modelicaRectifier.csv	138	1125	0.1

001 time
002 Inductor1.i
003 Inductor2.i
004 Ldc.i
005 der(Inductor1.i)
006 der(Inductor2.i)
007 der(Ldc.i)
008 der(Inductor3.i)
009 Ground1.p.i
010 Ground1.p.v
011 IdealDiode1.Los
012 IdealDiode1.i
013 IdealDiode1.v

f	#	Variable name	X
	1	time	x
	51	uAC[1]	
	52	uAC[2]	
	53	uAC[3]	

● to Plot win 1
○ to:

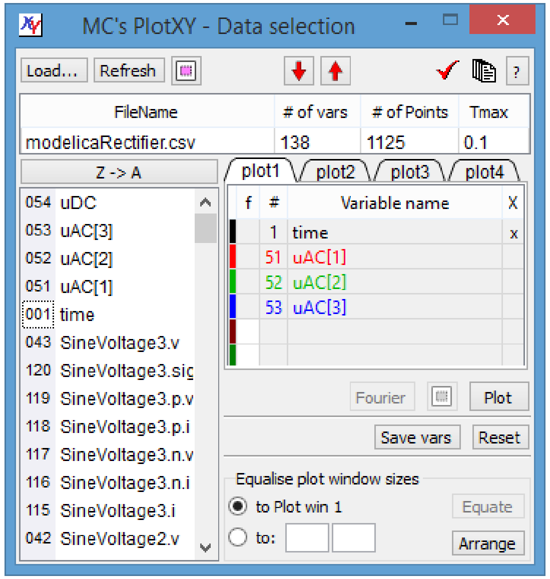


FileName	# of vars	# of Points	Tmax
modelicaRectifier.csv	138	1125	0.1

005 der(Inductor1.i)
006 der(Inductor2.i)
008 der(Inductor3.i)
007 der(Ldc.i)
009 Ground1.p.i
010 Ground1.p.v
044 iAC[1]
045 iAC[2]
046 iAC[3]
012 IdealDiode1.i
011 IdealDiode1.Los
061 IdealDiode1.n.i
062 IdealDiode1.n.v

f	#	Variable name	X
	1	time	x
	51	uAC[1]	
	52	uAC[2]	
	53	uAC[3]	

● to Plot win 1
○ to:

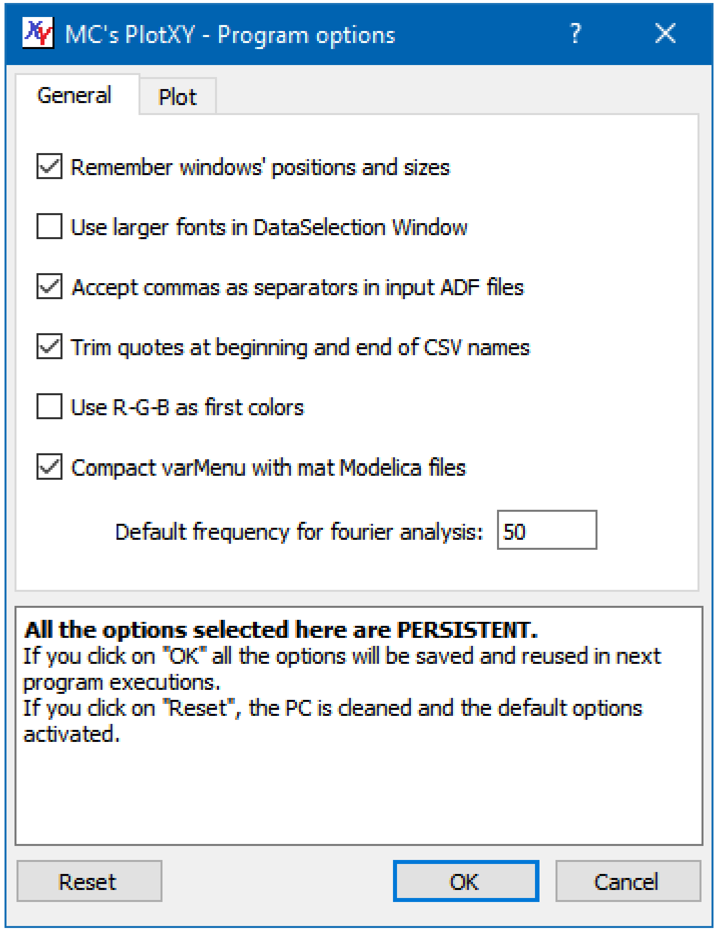


FileName	# of vars	# of Points	Tmax
modelicaRectifier.csv	138	1125	0.1

054 uDC
053 uAC[3]
052 uAC[2]
051 uAC[1]
001 time
043 SineVoltage3.v
120 SineVoltage3.sig
119 SineVoltage3.p.v
118 SineVoltage3.p.i
117 SineVoltage3.n.v
116 SineVoltage3.n.i
115 SineVoltage3.i
042 SineVoltage2.v

f	#	Variable name	X
	1	time	x
	51	uAC[1]	
	52	uAC[2]	
	53	uAC[3]	

● to Plot win 1
○ to:



☑ Remember windows' positions and sizes

☐ Use larger fonts in DataSelection Window

☑ Accept commas as separators in input ADF files

☑ Trim quotes at beginning and end of CSV names

☐ Use R-G-B as first colors

☑ Compact varMenu with mat Modelica files

Default frequency for fourier analysis: 50

All the options selected here are PERSISTENT.
If you click on "OK" all the options will be saved and reused in next program executions.
If you click on "Reset", the PC is cleaned and the default options activated.

Naturally, if we commonly use small numbers of files and variable and plots, we want to use small program windows ad default, without having to resize it anytime we load the program. To do this, we can set the option “remember windows’ positions and sizes”. The program options are accessed clicking on the third button from the right in the man toolbar showing a check-mark as symbol.

Several program options can be set. Here I did show just the one interests us now. Note that the program settings are written as system settings in a region of the computer storage space the operating system chooses. Under Microsoft Windows it is the Windows’ registry.

Although I don’t advise this ;-) you might want to remove the program from your hard disk and leave the computer completely clean. To do so, you have to open the program options window and click on the “Reset” button.

Refreshing files

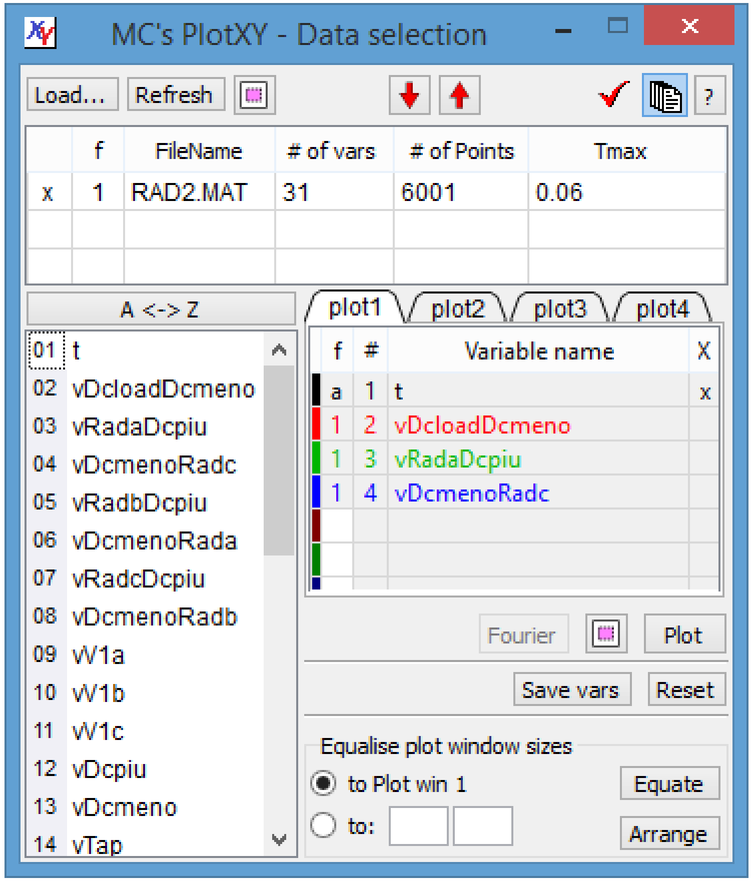
When PlotXY is used in conjunction with other programs, that change data to be displayed or analysed through it, those programs could chance the data while PlotXY is still loaded.

In this case it is useful to “refresh” data: this means that we ask PlotXY to load again from file the data related to the currently selected file. To do this we have to click on the *Refresh* button of the *DataSelection* window. Normally when we refresh data, also the displayed plots are refreshed, by default using automatic determination of horizonal and vertical scale. If one wants to refresh the data while not changing the scale ranges (i.e. min and max values on vertical and horizonal axes), immediately before clicking the refresh button, the small button containing a fuchsia small square on it (having as tooltip “Keep scale ranges while refreshing”), near the refresh button, must be clicked on.

Making, evaluating and managing line plots

Now we want to select a variable and show its plot. To select a variable you can click on its name in the *VarMenu* table, or drag &drop it in the *SelectedVar* table. In the first case, the variable will get the first available colour, in the latter you can choose the variable colour by choosing the appropriate row.

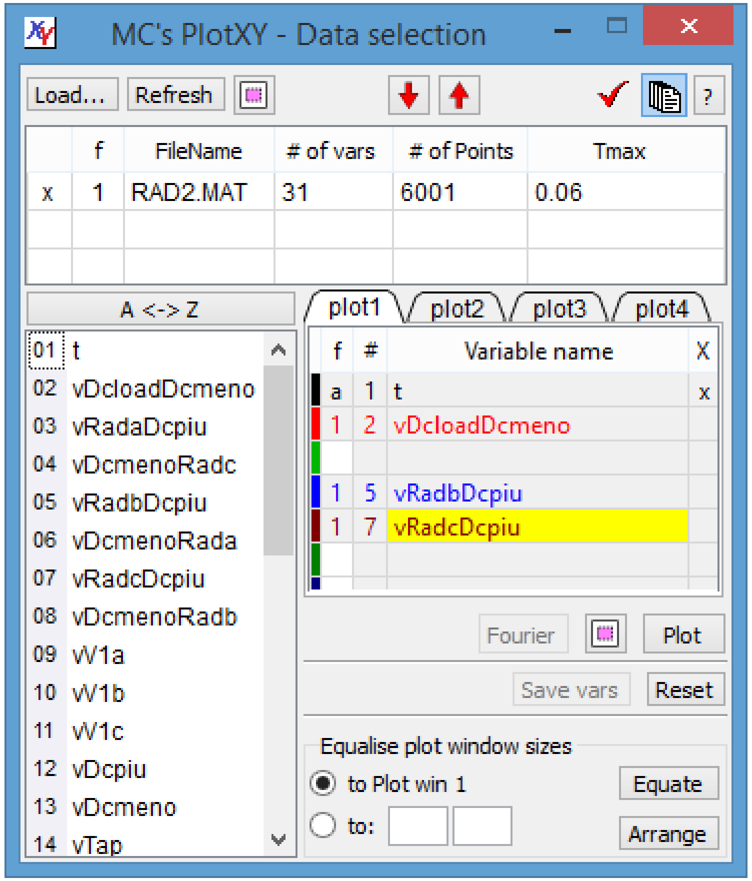
In the figure left below a few variables from the supplied file rad2.mat re elected by clicking on their names; in the one at the right side their names are dragged and dropped, so that the wanted colours are used (the last dropped has a yellow background). The plots will have the same colours as the shown variable names..



	f	FileName	# of vars	# of Points	Tmax
x	1	RAD2.MAT	31	6001	0.06

01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04 vDcmenoRadc
05 vRadbDcpiu
06 vDcmenoRada
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap

f	#	Variable name	X
a	1	t	x
1	2	vDcloadDcmeno	
1	3	vRadaDcpiu	
1	4	vDcmenoRadc	



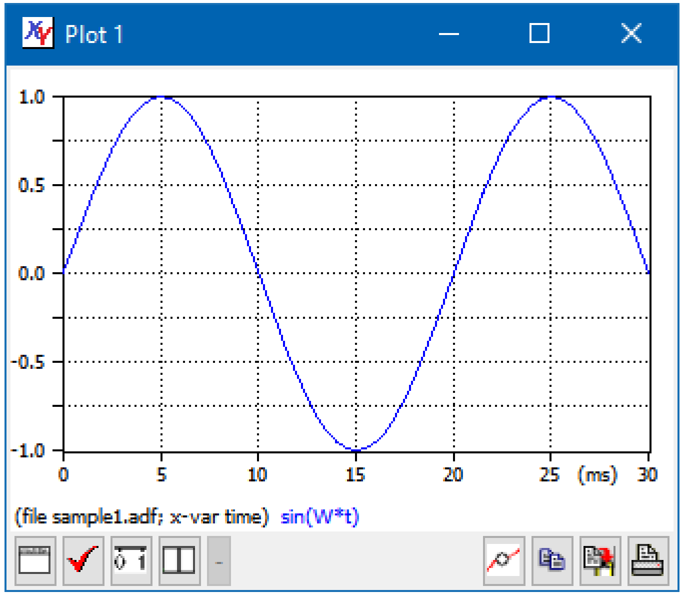
	f	FileName	# of vars	# of Points	Tmax
x	1	RAD2.MAT	31	6001	0.06

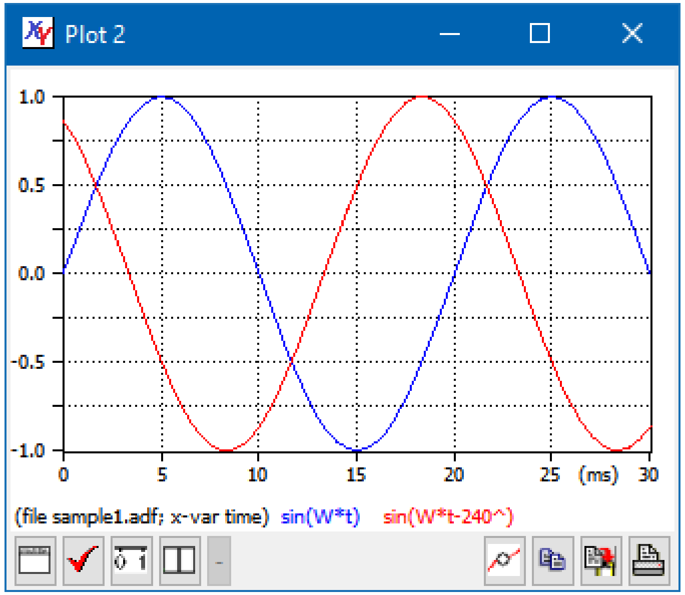
01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04 vDcmenoRadc
05 vRadbDcpiu
06 vDcmenoRada
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap

f	#	Variable name	X
a	1	t	x
1	2	vDcloadDcmeno	
1	5	vRadbDcpiu	
1	7	vRadcDcpiu	

You can also select several adjacent variables at a time: a *group* of variables. You have to keep the shift key down while clicking on the first (the uppermost) variable of the group, and then click normally on the last one.

Once one or more variable(s) is (are) selected, click on the "Plot" button to see the plot. Two examples from file sample1.padf are below: please reproduce them as a very simple exercise.





We see that the plot contain also a legend (shown below) containing the variable names.

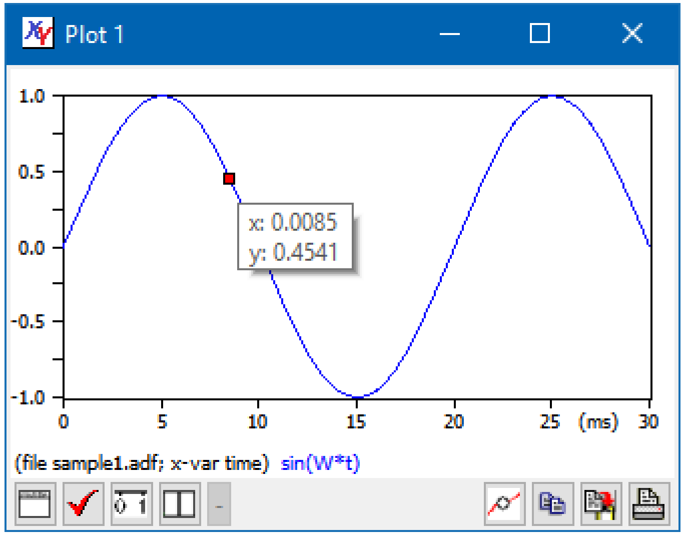
You can zoom the produced plots at will, as we've learned in the Doing the first things: feel a real-life experience.

Now we perform additional actions.

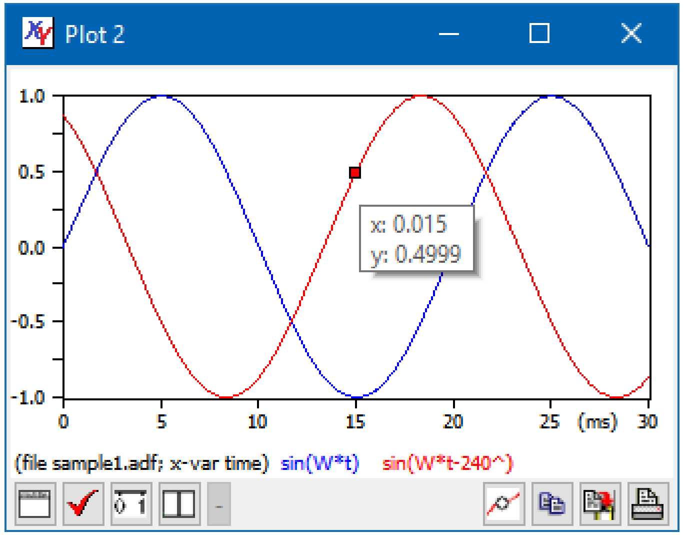
Looking at data values

Consider the plots in the right picture here above. We now want to see some numerical values.

PlotXY offers two ways to do this. The first one is putting the mouse cursor near the curves. The program will find the nearest point in file and show the corresponding value on a tooltip box. Moreover, a small red rectangle indicating what point the value exactly refers to. This is particularly useful when considering plots with only a few points and therefore the actual point on file might be rather far from the mouse pointer. See the two examples below:

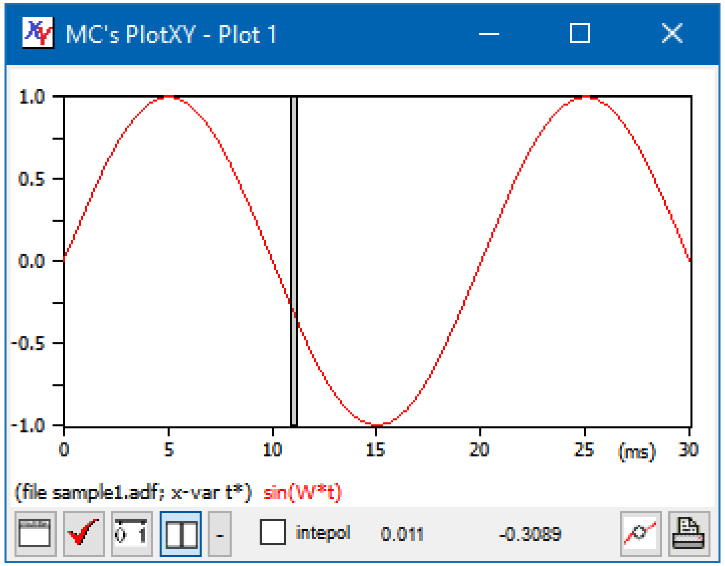


```
x: 0.0085
y: 0.4541
```

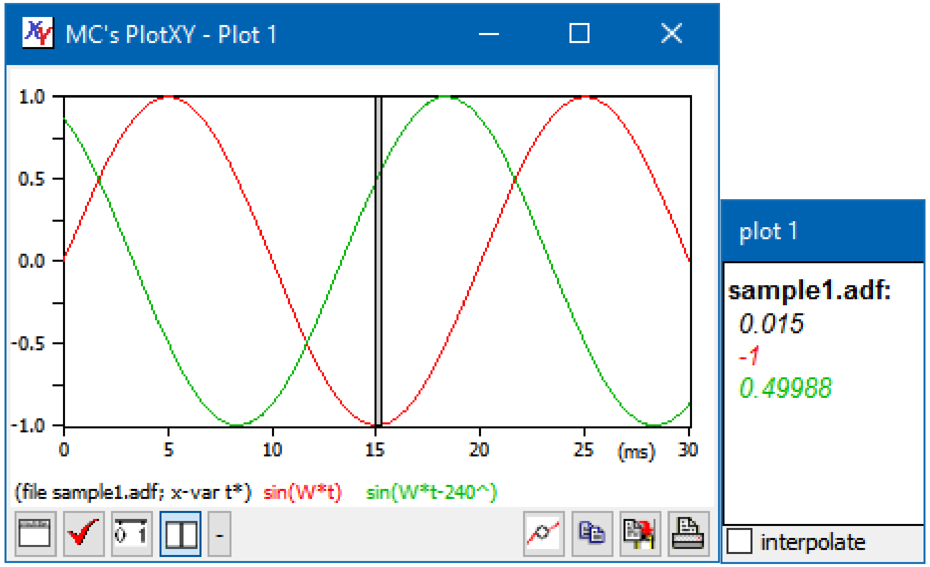


```
x: 0.015
y: 0.4999
```

Another option consists of having simultaneous access to all the values corresponding to a given x-axis value. To do this, click on the *Show data* button. What happens in the cases of a single curve or multiple curves is different. Try! You will get something like what you can see down here.



```
0.011        -0.3089
```

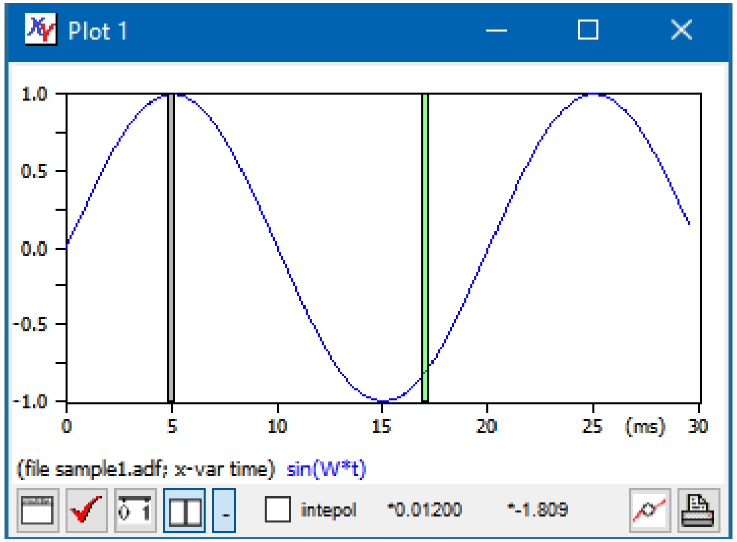


```
sample1.adf:
 0.015
 -1
 0.49988
```

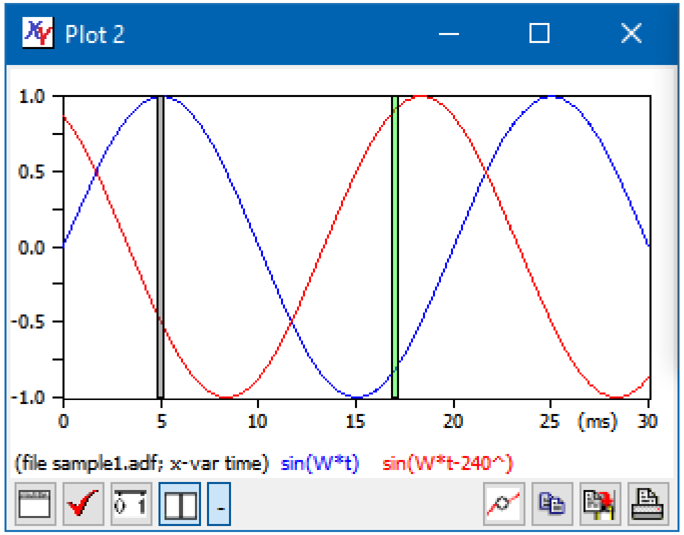
In the left window the values are directly shown in the bottom toolbar (time first), while in the right one, because more space is needed, an additional small window (the ***Data browse window)*** is created to show the numerical values. In the first case the number of significant digits used to display the values is 4, in the second case it is 5. However, if *Exact match* is selected in the scale window (see Modifying the plot scales), in both cases the number of significant digits is rised by one, thus becoming 5 and 6, respectively.

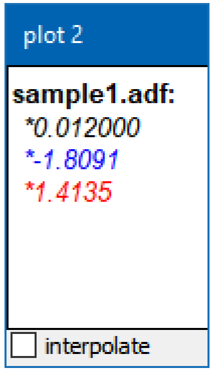
In both cases normally the numbers shown are exactly those present in the inpt file; however, if the "interpolate" box is checked, linear interplation is made between adjacent points, check sample1.adf file: you will see that corresponding to t=0.0105s the sin(Om*t) value is -0.1563.

In case you want to see the difference of the values one or more cuves have at different points of time, you can click on the *Show differences* button. It is located immediately at right of the *show data* button, and carries a sign "-" on it . You will get a second vertical bar such as in the two images below. Whenever you move the grey bar you get actual values; whenever you move the green one, you will get the differences (difference values are marked with an asterisk "*" at the left side of the numbers). As before, in case a single plot is shown the numbers are shown in the Plot window, otherwise in a separatate, small additional window.



*0.01200 *-1.809





sample1.adf:
*0.012000
*-1.8091
*1.4135

☐ interpolate

Hint: you can move cursors using arrow-keys on keyboard instead of mouse. Simple left and right arrows move by one pixel left and right. If CTRL button is hold down when using arrows, movement is faster (three pixels per keypress).

For info about looking at data values when "functions of variables" are used, see sect. Function plots.

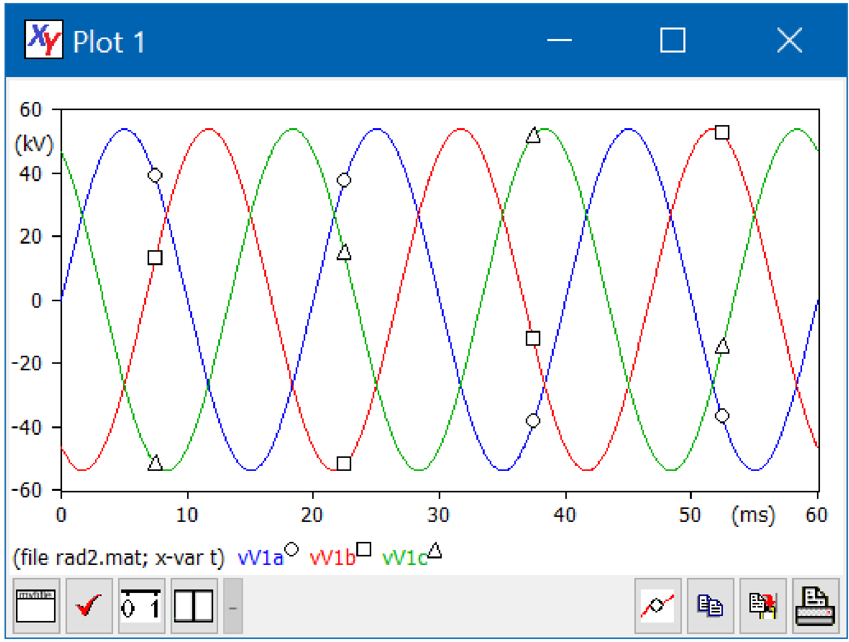
Add markers

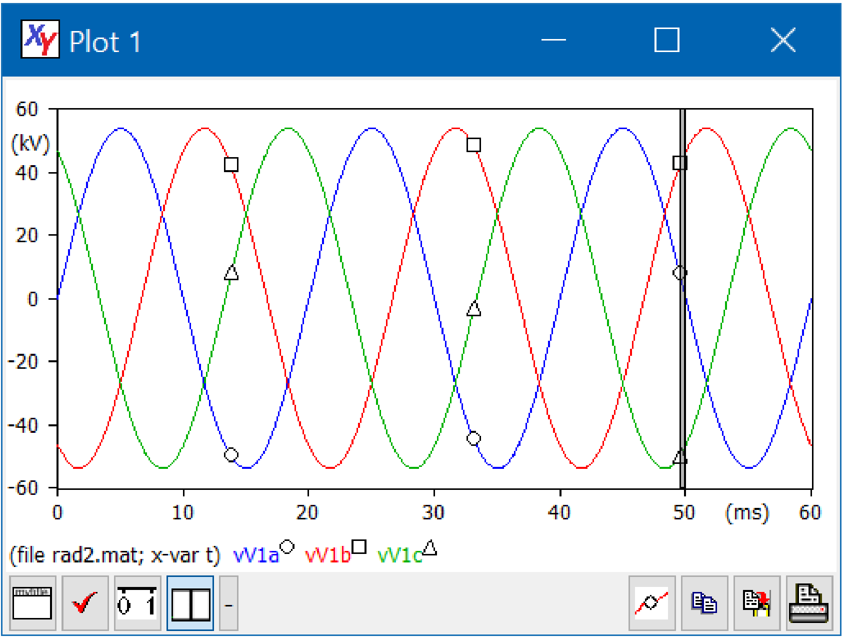
Sometimes one needs to have a multi-curve plot clearly understandable even when printed on a B/W printer.

This can be done using the *Add markers* Button of a Plot Window.

This button has a different behaviour when the *Show data* button is up or down.

When it is up, four markers for each curve are added in automatically determined positions of the plot (see left picture below). If, on the other hand it is down, and therefore the vertical grey bar is visible, they are added in the bar position. Therefore, moving left and right that bar, the user can chose the best position for the markers. This way he can add as many markers as he wants. (See right picture below).

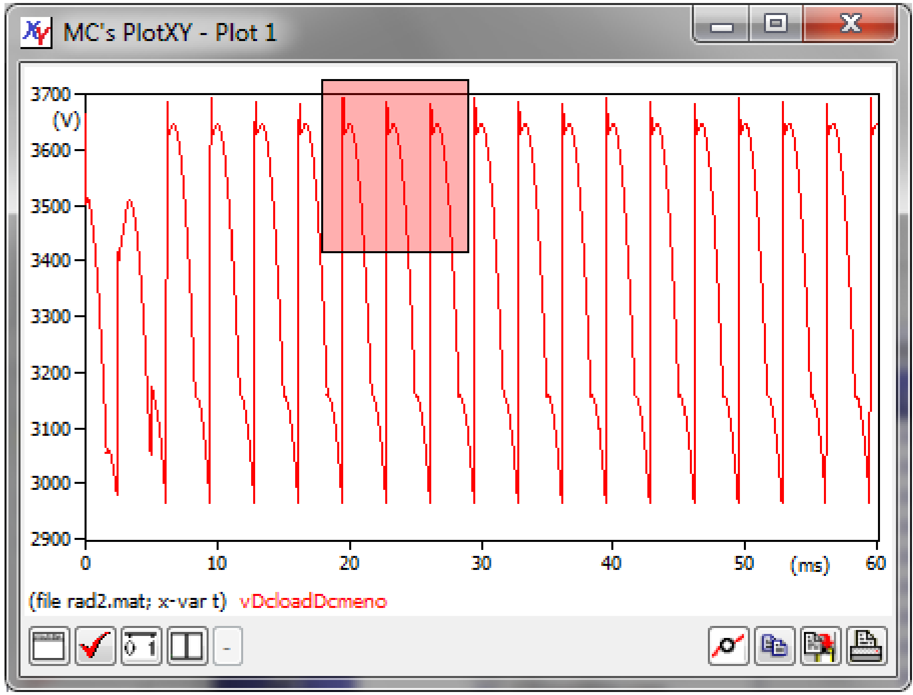


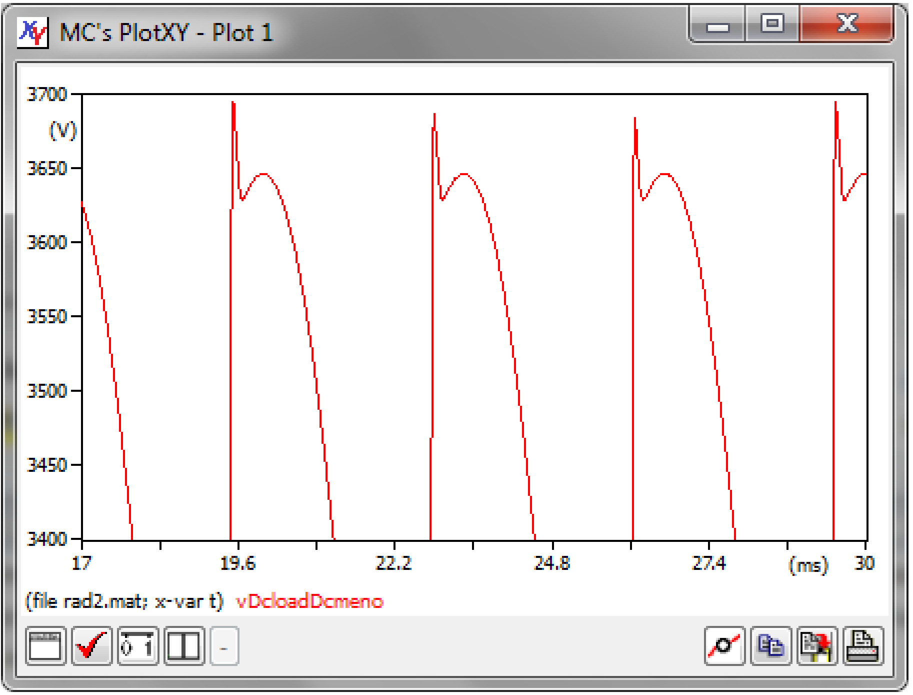


Zooming and keeping zoomed

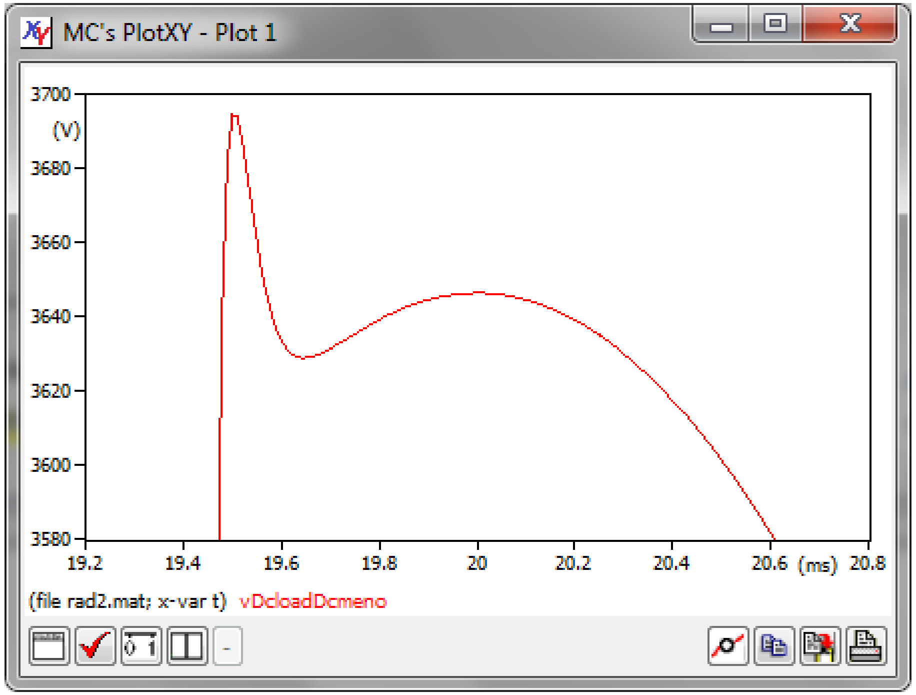
Very often, we need to have a closer look at some plot detail. To do this we want to zoom the plot

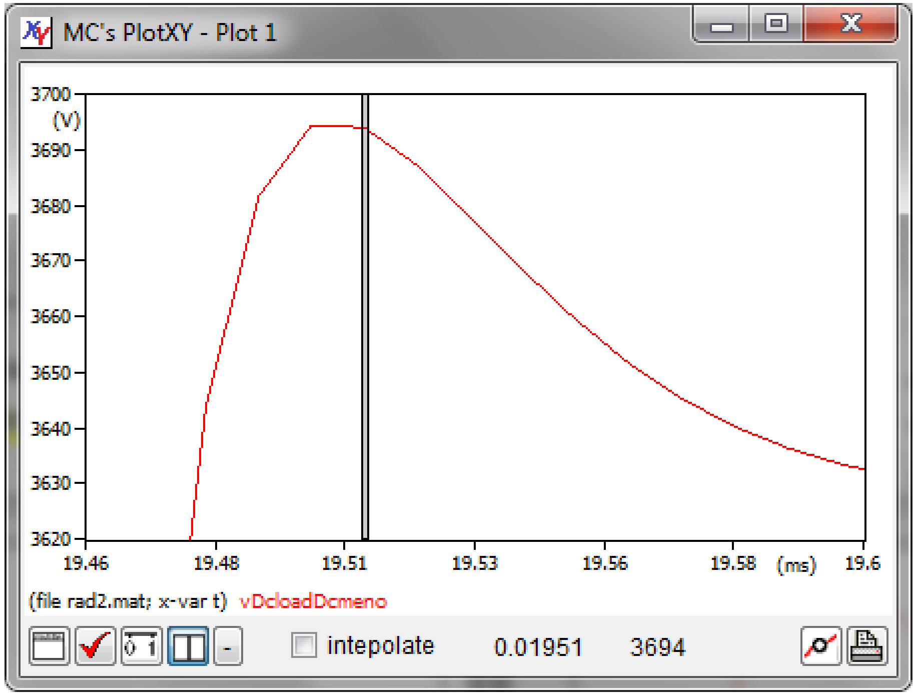
This is straightforward with PlotXY: simply we define the zooming rectangle with the mouse: we push down the left button in the top-left corner of the rectangle, drag the mouse, and leave the button once the rectangle displays what we want. While dragging the selected area is in partly transparent pink that shows what we are selecting. See the two pictures below: the left one shows zooming in action, the right one shows the zoomed plot. Try to repeat this using the supplied file rad2.mat.





We can repeat this many times, to have a look at even tiny details of our plots. See below.





☐ intepolate 0.01951 3694

In a zoomed plot we still can look at the numerical values corresponding to the plot, either focusing on the actual data from the loaded file, or on an interpolation between points (see the right picture above).

Whenever a plot is zoomed, the axes are kept clean: we have “round numbers as minimum and maximum values and on the tic-marks. Only when the zoom is very deep, PloXY gives up choosing clean axes and use whatever it can.

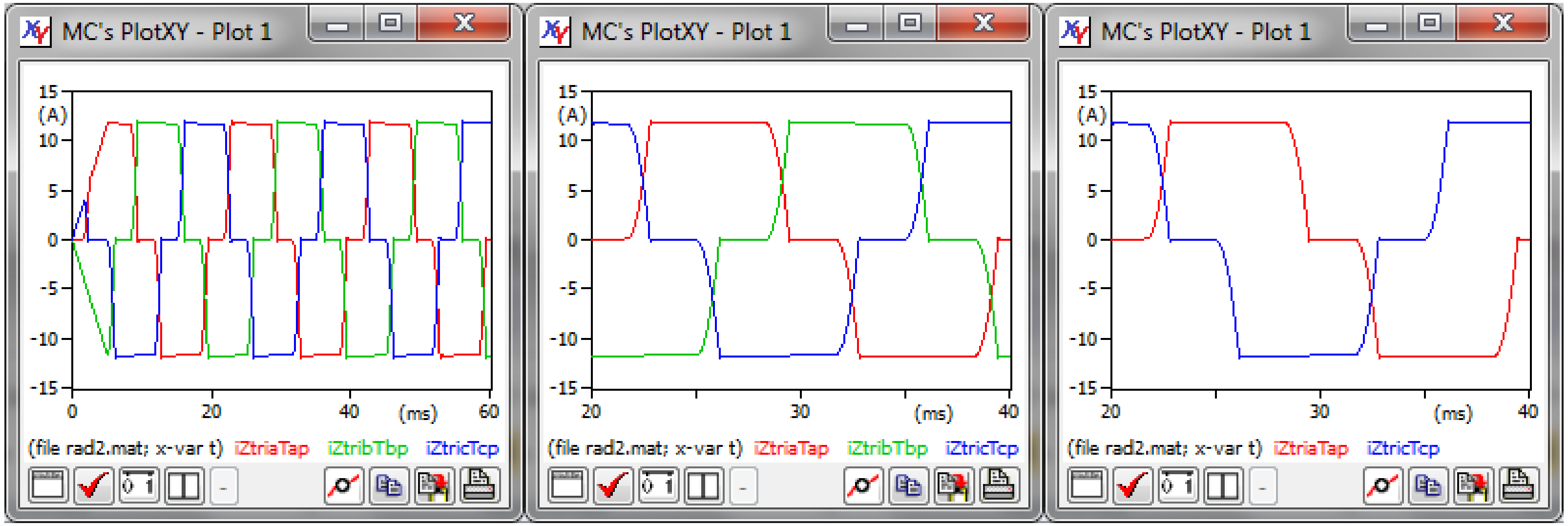
To zoom out, right-click on the plot area: we will have the choice between getting back one level or zoom out the plot completely.

It might also be useful, in some occasions, to add or remove plots from a plot window without updating the zoom range.

Let us do together an exercise. From rad2.mat we can plot variables 16, 17 and 18. The plot will appear as shown below at left. Then manually zoom between 20 and 40 ms. The resulting plot is shown below at centre.

We might now add or remove curves while leaving the axes range unchanced (between 20 and 40 ms horizontaly, and betweeen – 15 and 15 A).

This is done by clicking on the *Keep scale ranges while plotting* button (on the main buttonbox of the DataSelection Window) before clicking on the plot button. If that button is down, the axis scales are left unchanged. Therefore if for instance we remove variable N. 17, and then we plot keeping scales, the plot is as shown at the rightmost picture below.



Special plots: with twin vertical axes or X-Y plots

Rather often the best visualisation is when two different plots are built using the same horizontal scale but they have different vertical sizes. In these cases it may be very convenient to realize plot with two different vertical axes (here called twin vertical axes).

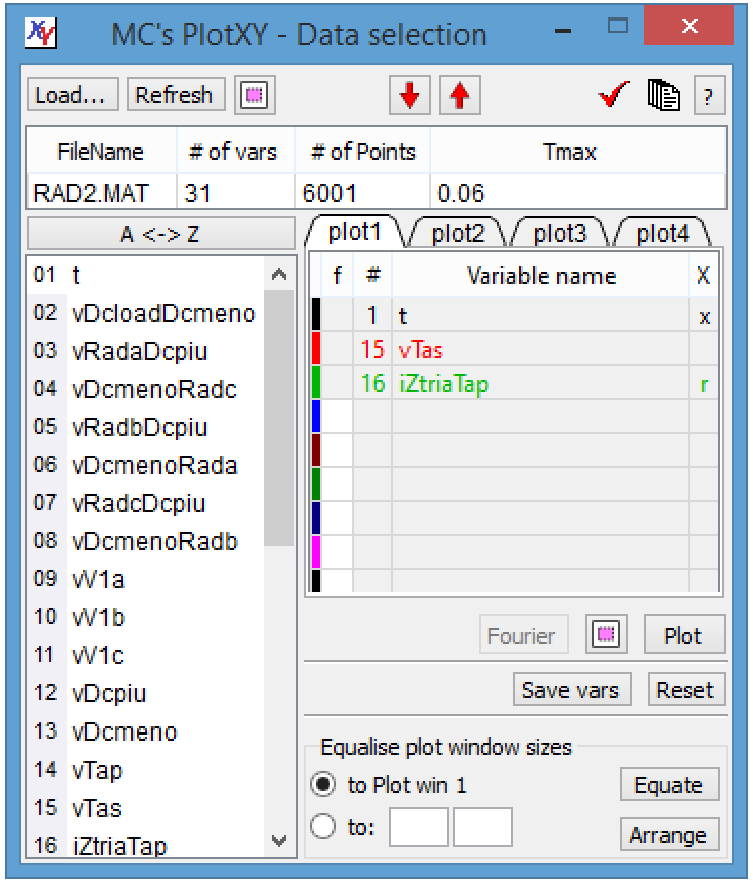
PlotXY allows you to create plots with several curves, while allowing you to select which one corresponds to the left vertical axis and which correspond to the right one.

This can be seen using the enclosed file sample2.mat.

The figures below show what happens if we select a voltage (variable vTas) and a current (iZtriaTap) as variables to be numerically evaluated against the left and right vertical axes respectively.

The first variable has been selected by left-clicking on its name, the second one right-clinking on its one. In the plot window we can easily understand with variables refer to the right vertical axis: their name is shown underlined.

Note: You can also right-click on the name of an already selected variable to toggle between left and right vertical axis usage for that variable.

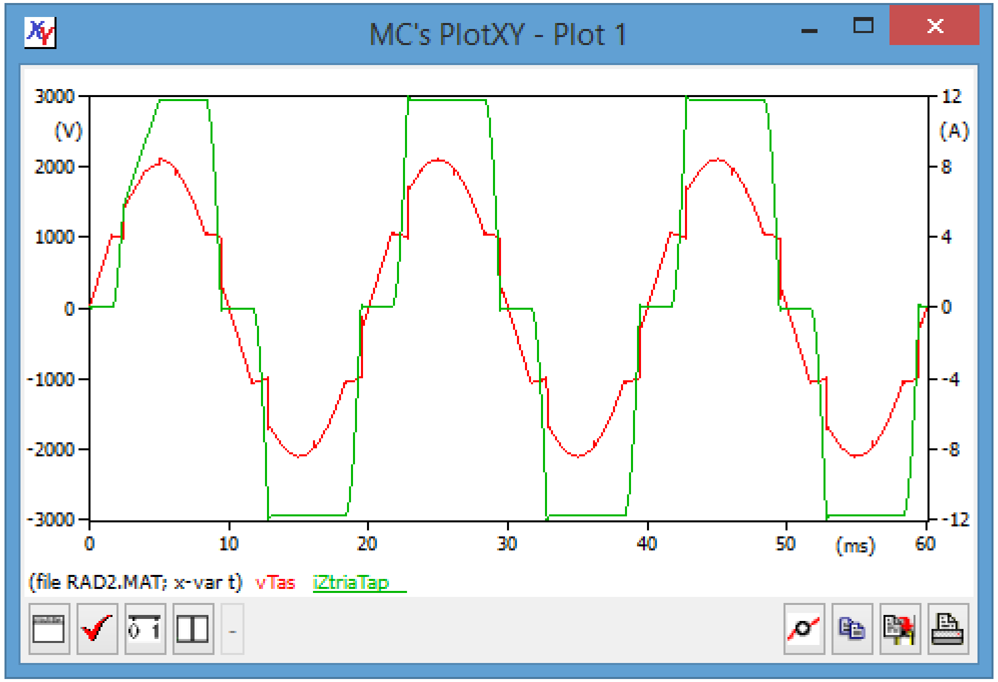


FileName	# of vars	# of Points	Tmax
RAD2.MAT	31	6001	0.06

01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04 vDcmenoRadc
05 vRadbDcpiu
06 vDcmenoRada
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas
16 iZtriaTap

f	#	Variable name	X
	1	t	x
	15	vTas	
	16	iZtriaTap	r

Equalise plot window sizes
● to Plot win 1
○ to:

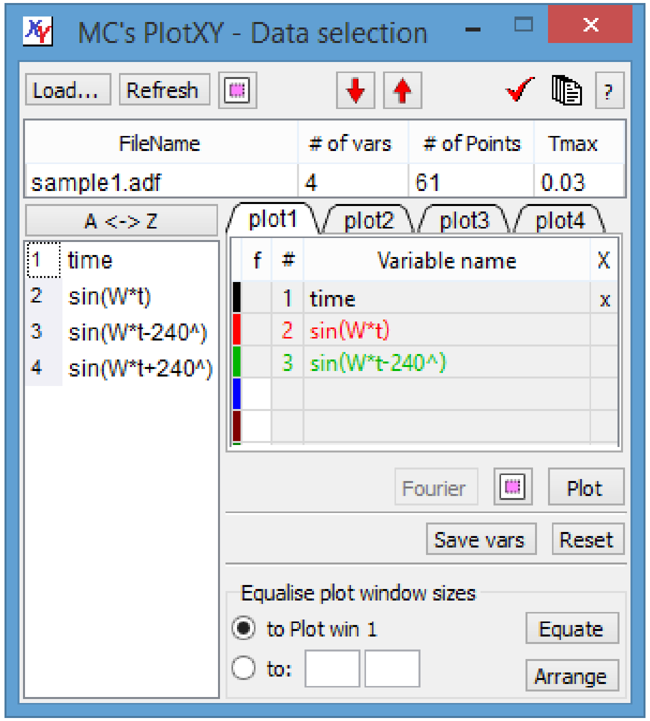


Another useful feature is the possibility to plot a variable *against* another.

This can be done only from SelectedVar tables were the selected variables come all from the same file. To make X-Y plots:

1. select the variables of interest and
2. click on the “X” column in tie plot var-list of the *dataSelection* window.

When step 2 is performed, the row onto which one has clicked becomes the “x” variable for X-Y plots. When this step 2 is performed for the first time after the latest reset of the Plot var-list table the previous x-var is removed from the table, assuming that it represents time, and therefore it is not whished. In case of subsequent clicks, however, the old X variable is not automatically removed and becomes one of the y variables of the X-Y plot.

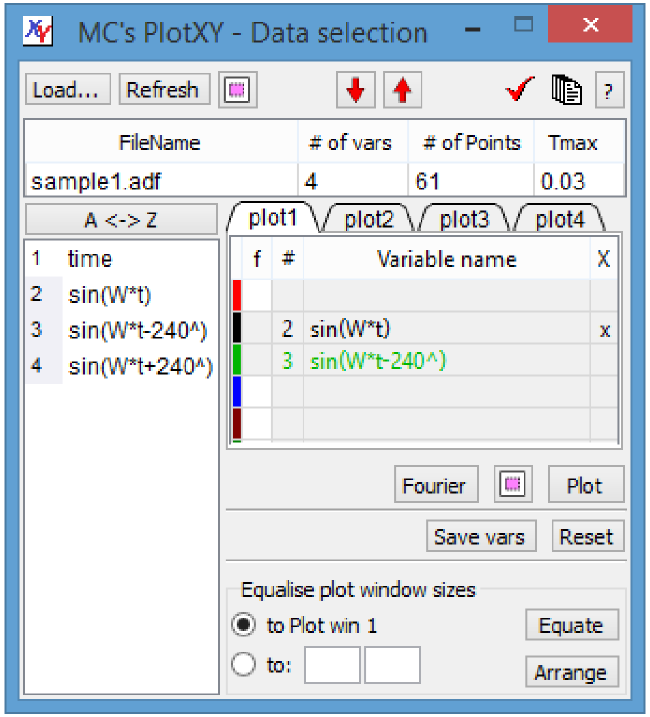


FileName	# of vars	# of Points	Tmax
sample1.adf	4	61	0.03

1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)

f	#	Variable name	X
	1	time	x
	2	sin(W*t)	
	3	sin(W*t-240^)	

● to Plot win 1
○ to:

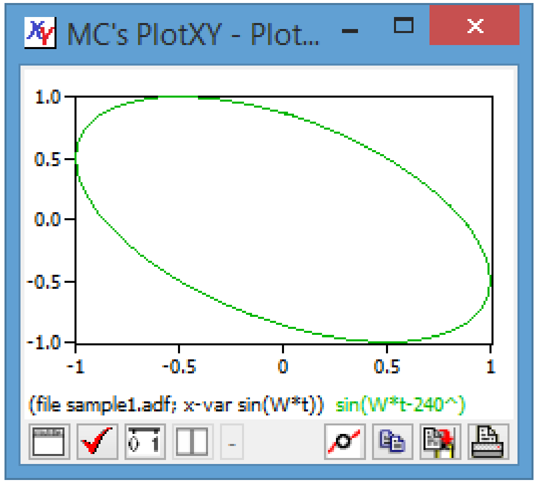


FileName	# of vars	# of Points	Tmax
sample1.adf	4	61	0.03

1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)

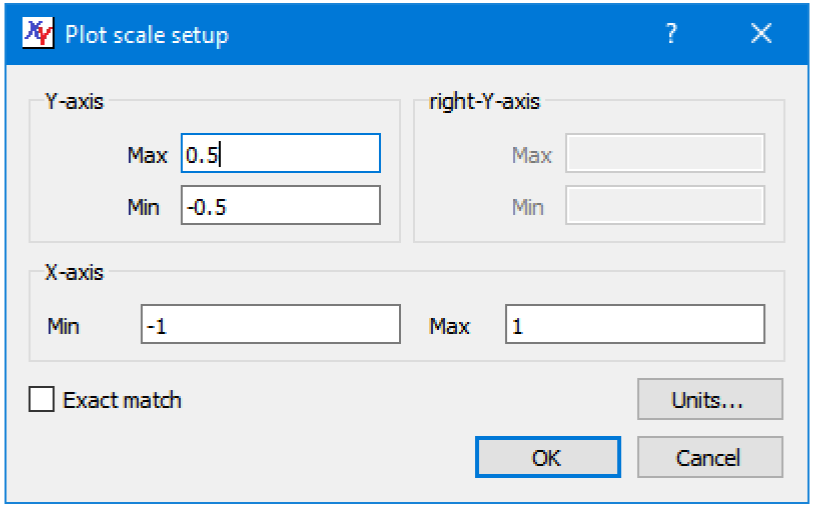
f	#	Variable name	X
	2	sin(W*t)	x
	3	sin(W*t-240^)	

● to Plot win 1
○ to:



Modifying the plot scales and units of measure

We have seen that PlotXY tries to automatically find the most reasonable and readable scales on the axes. Nevertheless, the user often wants to select min and max values himself. This can be done clicking on the *Change Scale* button of the plot window. This will cause the left window below to be displayed:



Plot scale setup

Y-axis

Max 0.5

Min -0.5

right-Y-axis

Max

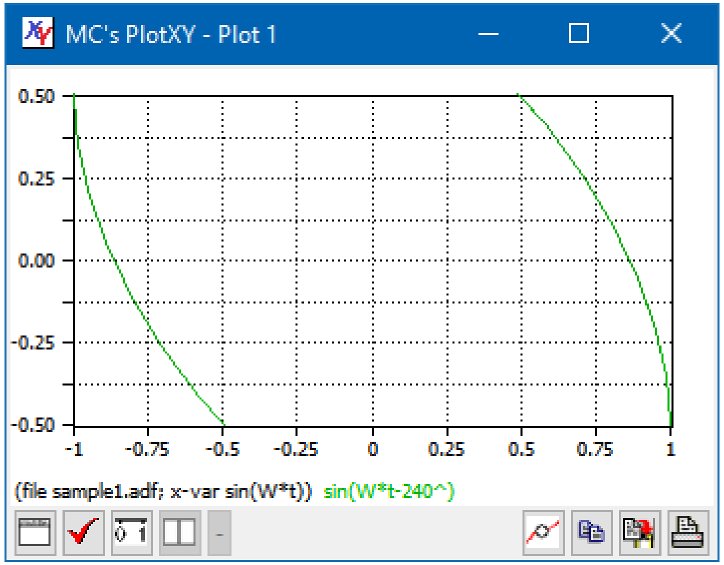
Min

X-axis

Min -1

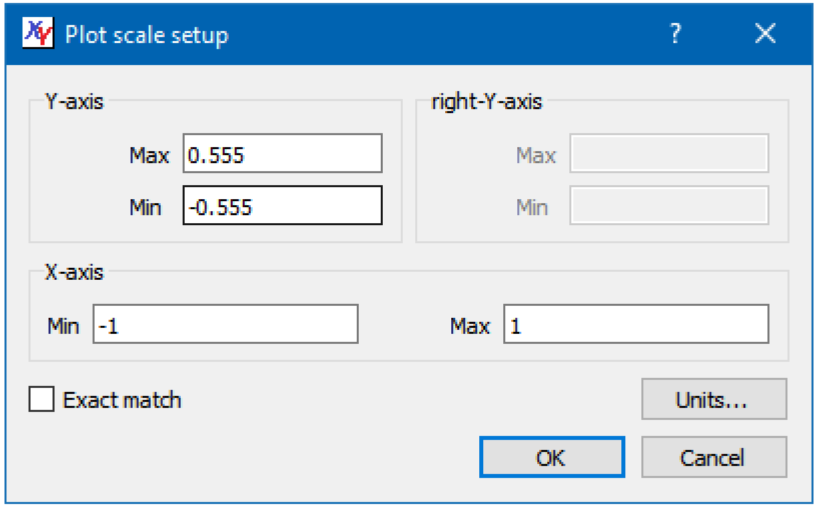
Max 1

☐ Exact match



If one clicks on the ok button, the scales shown in the left plot are used.

If “Exact match” is not selected the given Min and Max are still subject to adaptation by the program to ease readability, otherwise the exact scale ranges chosen by the user are used. Compare the following pictures:



Plot scale setup

Y-axis

Max 0.555

Min -0.555

right-Y-axis

Max

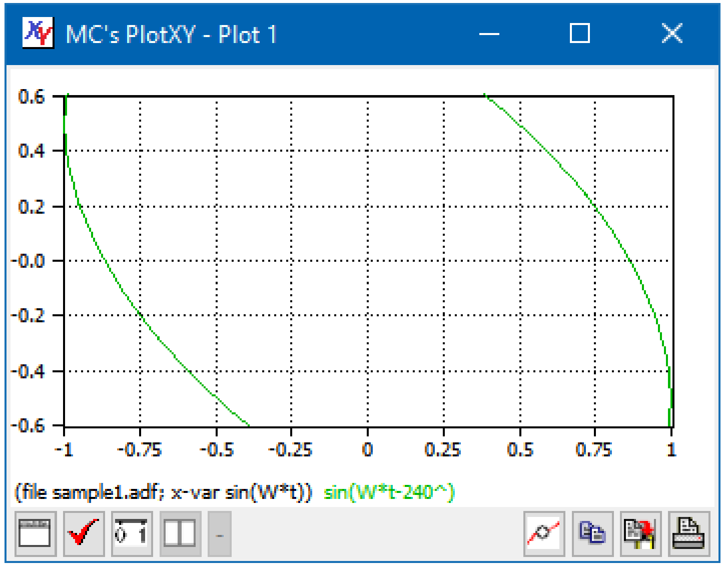
Min

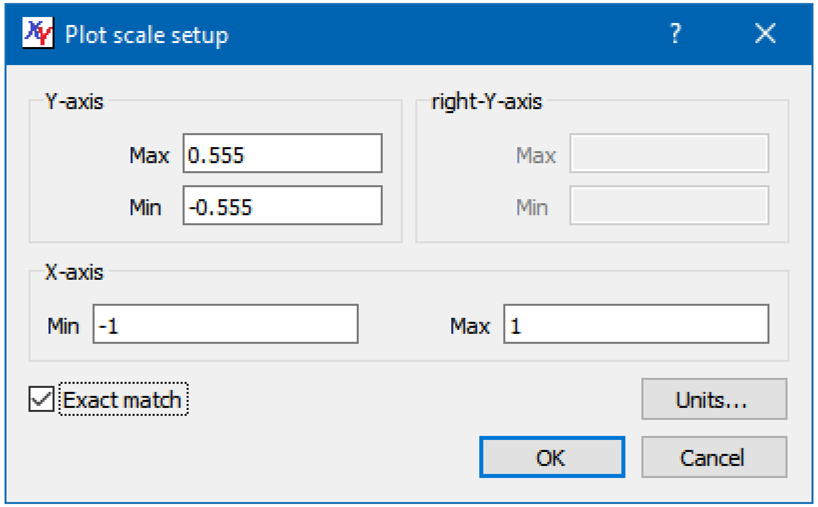
X-axis

Min -1

Max 1

☐ Exact match





Y-axis

Max 0.555

Min -0.555

right-Y-axis

Max

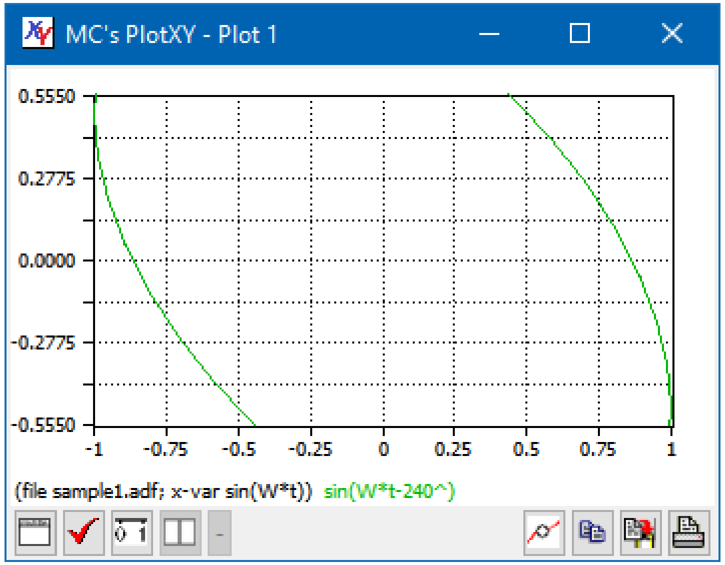
Min

X-axis

Min -1

Max 1

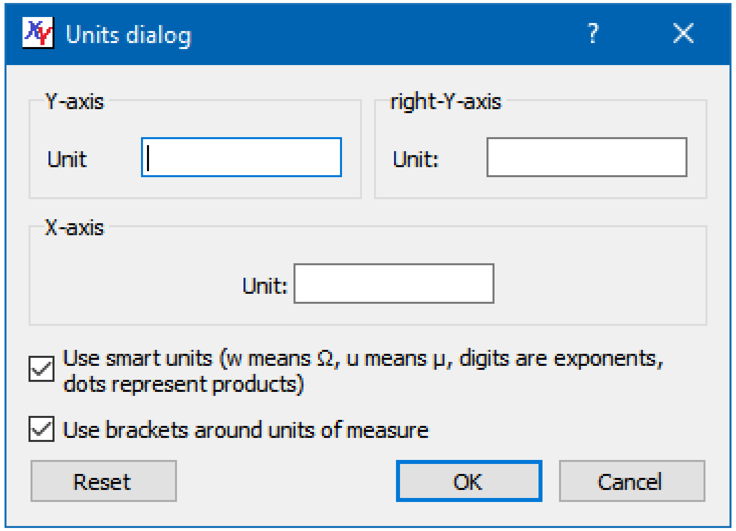
☑ Exact match



When *Exact match* is selected, the number of significant digits used to display values with data cursors (see Looking at data values) is raised by one and brought to 5 or 6, depending on whether the numerical values are displayed in the *Plot window* or in the *Data browse window* (compare sect. “Looking at data values”). This because it is intended that when the user requests Exact match, he wants more precision everywhere.

From some examples above, the reader might have note that the program tries to show reasonable unit of measure on the axes. For instance, variables starting with “v” are interpreted as being voltage, and the chosen unit of measure is “(V)”. This automatic determination of units of measure has important features, which are described in detail in section “Automatic Units of measure and prefixes”. Here I just want to say that the user can manually select his axis labels, clicking on the “Units...” button of the “Plot scale setup” window shown just above here.

When the user clicks on that button, the following *Units dialog* is displayed.



☑ Use smart units (w means Ω, u means μ, digits are exponents, dots represent products)

☑ Use brackets around units of measure

It allows selecting the units for any of the three axes.

When the first checkbox is selected units can be SI compliant. For example it is easy to produce units such as m, Nm, N·m, N/m, Ω, Ωm, μF, N/m^{2}, N·m^{-2}. To write these units, write "w" to get Ω, "u" to get μ; any digit is interpreted as being an exponent (possibly preceded by a sign), any "." as a dot indicating product. It may happen that the user wants to use a non-SI compliant unit of measure, such as "p.u." To do this, it is necessary not to use smart unis, unchecking the first check-box. Thus avoiding the special interpretation of dots and character "u" necessary for smart units.

If a new plot is created using the "Plot" button on the DataSelection window the units selected in the units dialog are not used for it: they might be obsolete! If, however, before clicking on "Plot" the user selects the "Keep scale ranges while plotting", since scale ranges are kept, it is obvious that also the units of measure are expected to be correct, and existing user units are therefore used.

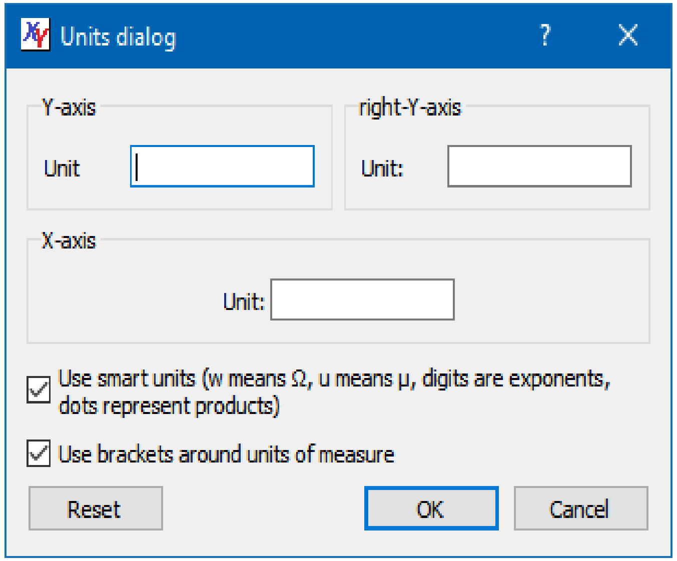
Note that the unit of resistance ohm can correctly be displayed as a Greek uppercase omega.

Changing the plot type and appearance

In the previous section, line plots with linear scale on the axes were shown.

Naturally, there exist several options to change the appearance of the plot. To explore them, try to click on the *Plot options* or *Change scale* buttons.

Click for instance the *Plot options* button. A window like the ones shown below appears. The window has three tabs displaying different options each. The meaning of the options is self-explanatory.


Units dialog

Y-axis

Unit

right-Y-axis

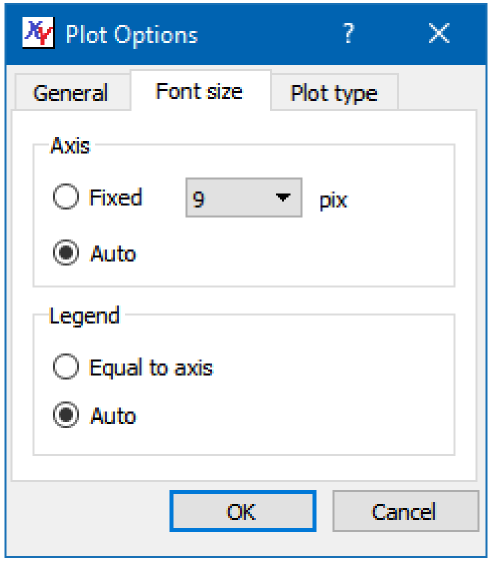
Unit:

X-axis

Unit:

☑ Use smart units (w means Ω, u means μ, digits are exponents, dots represent products)

☑ Use brackets around units of measure


Plot Options

Axis

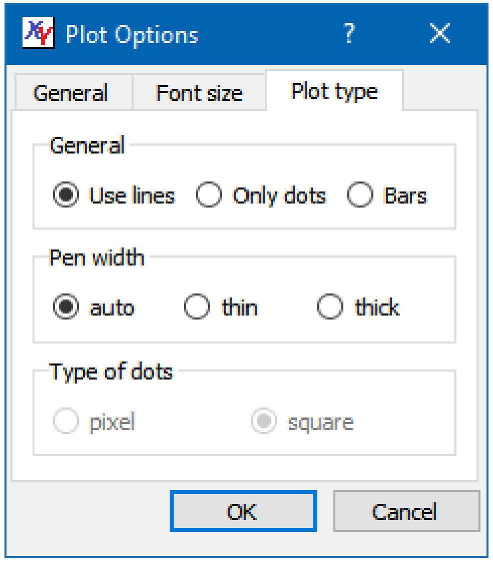
○ Fixed 9 pix

● Auto

Legend

○ Equal to axis

● Auto


Plot Options

General

● Use lines ○ Only dots ○ Bars

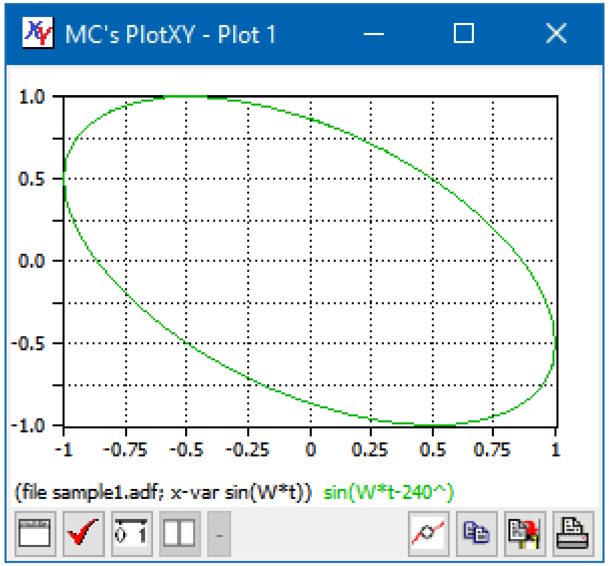
Pen width

● auto ○ thin ○ thick

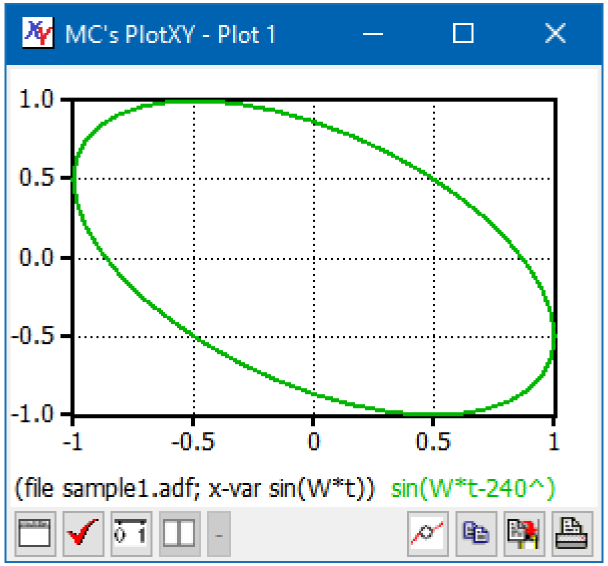
Type of dots

○ pixel ● square

Two examples of the application of some plot options are shown below: General|Display Grid (left), Plot type|pen width|thick and Font size|axis|fixed 12 pts (right).



(file sample1.adf; x-var sin(W*t)) sin(W*t-240^)

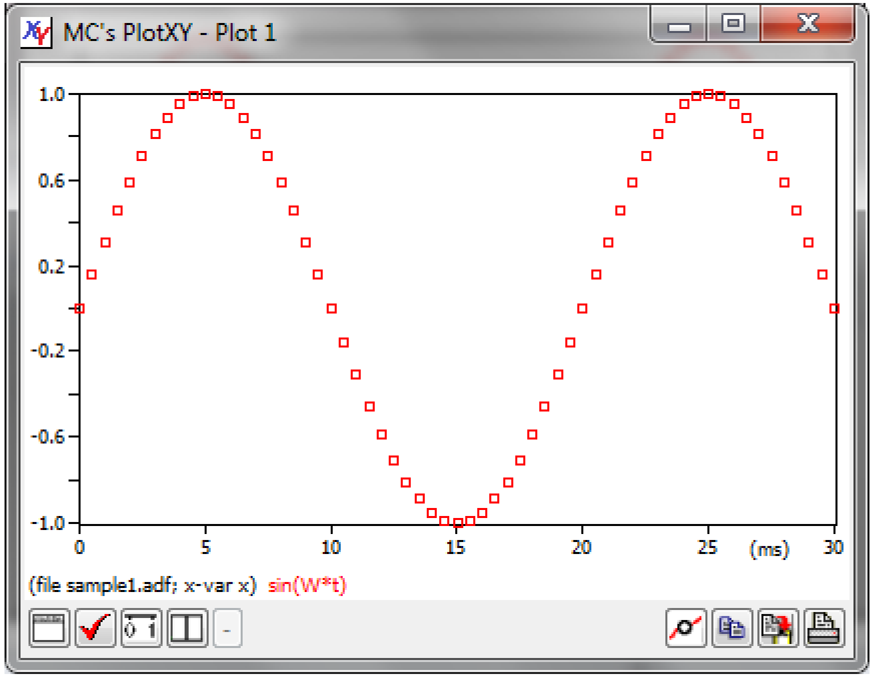


(file sample1.adf; x-var sin(W*t)) sin(W*t-240^)

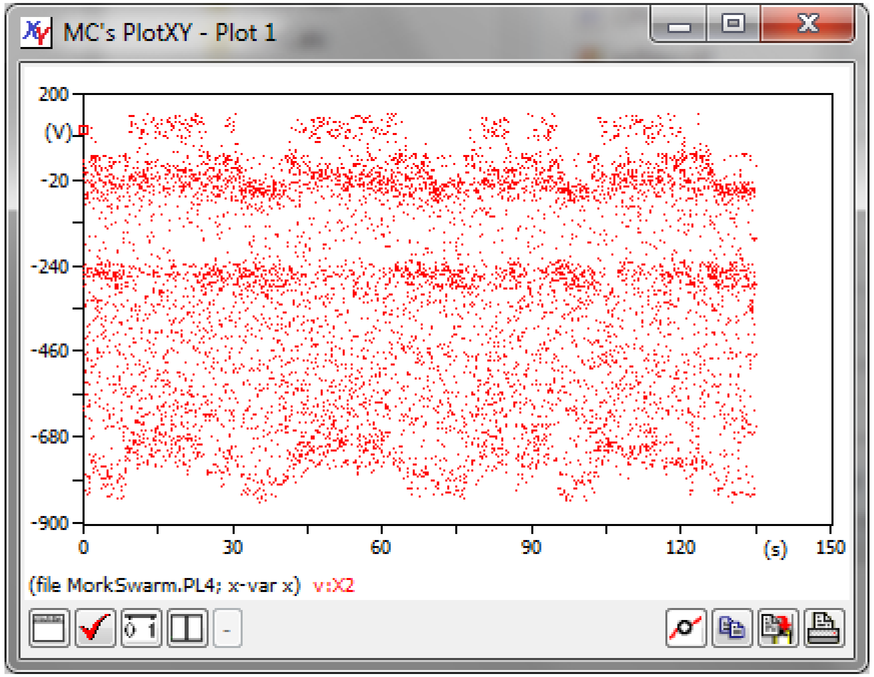
When you use a log axis, the marks on the axes and gridlines will be at 2x, 4x, 8x the main power of ten.

There are cases in which line plots are not the best way to show a plot. If we want to see only the point read from file we can use *General|only dots* option. This is done in the bottom-left plot for the first variable from sample1.adf file.

Finally there are files in which the X-Y correlation is weak; this can be shown using a cloud of points. In this case small squares to show points are too large, and it is better to use single pixels. This is done in the bottom-right plot, using variable X2 from the MorkSwarm.pl4 file

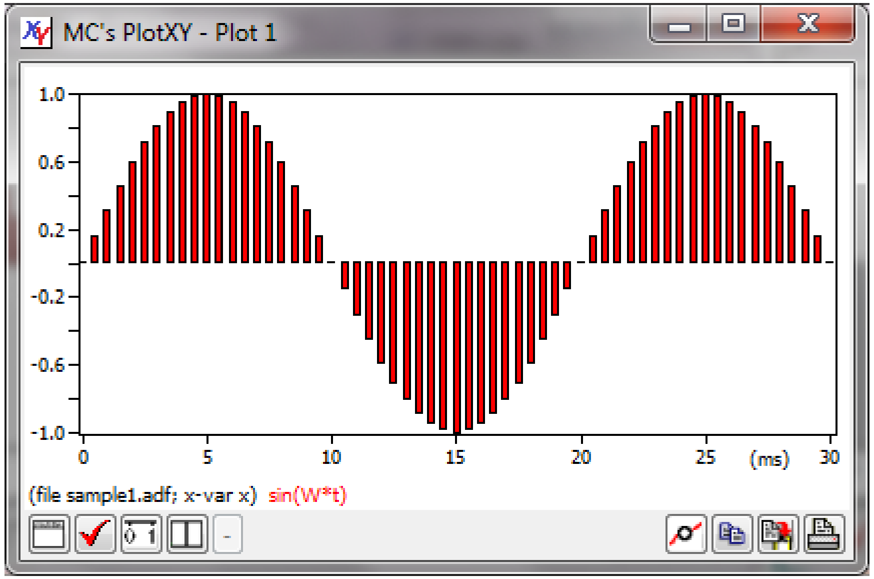


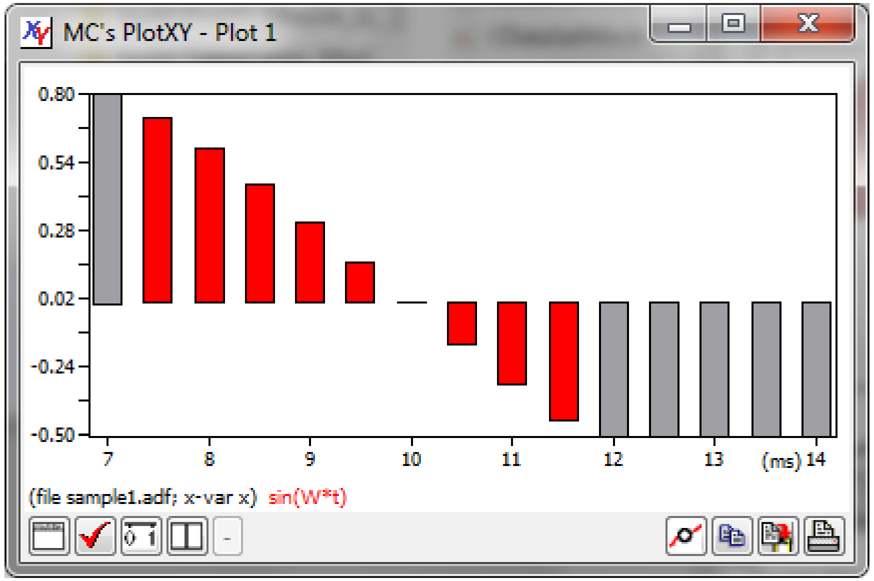
(file sample1.adf; x-var x) sin(W*t)



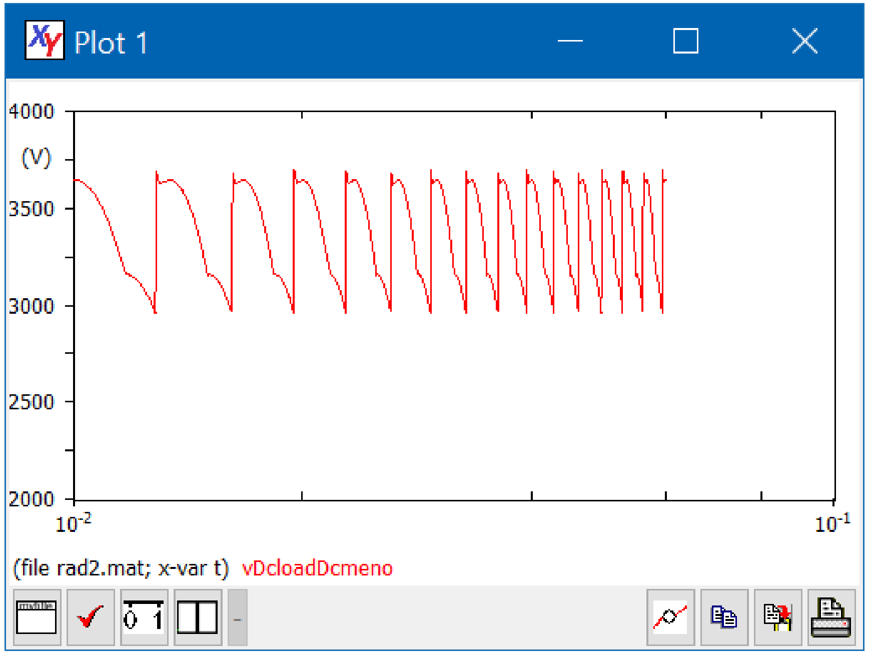
(file MorkSwarm.PL4; x-var x) v:X2

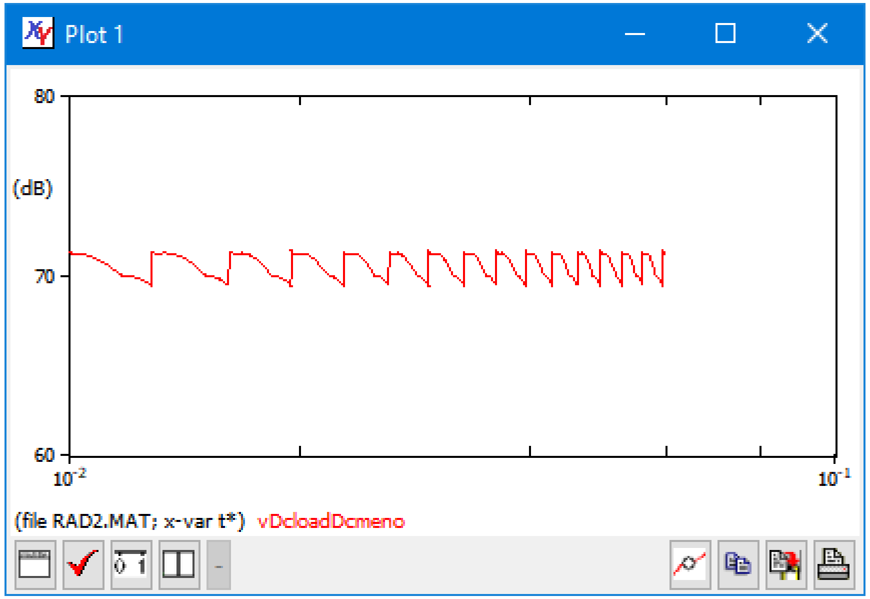
There are also cases in which instead of line plots the user needs to use a bar chart. This is done in the bottom-left plot (from sample1.df) in full scale, and in the bottom-right in a zoomed fashion. The bars whose second end falls outside the zooming window are shown greyed..





Finally scales can be linear, log, and dB. Two examples from sample2.mat are shown below

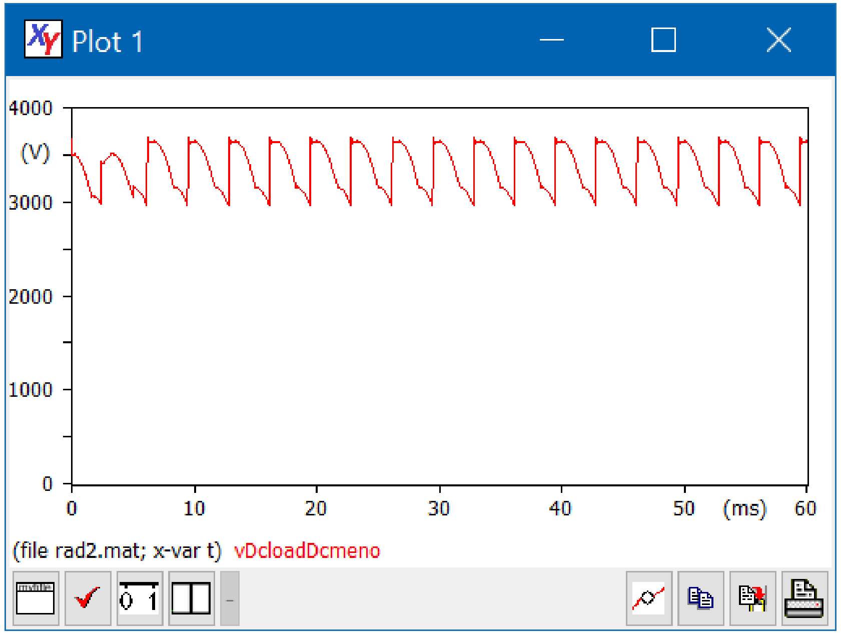


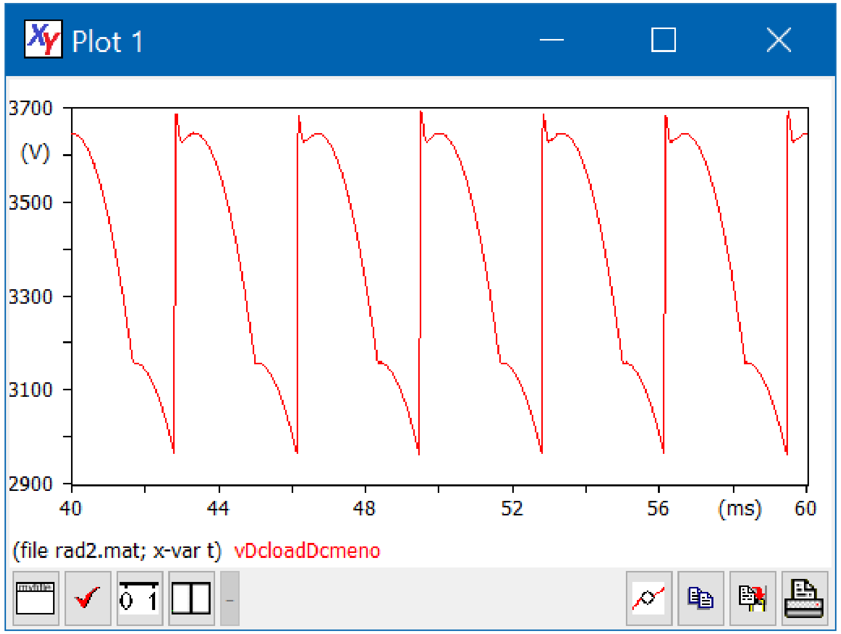


The customisation you do using the *Plot Options* button are valid for the current plot. The default graphical characteristics of the plots, e.g. those valid when plots are first created, decided at program level, using the *Program options* button of the program main window.

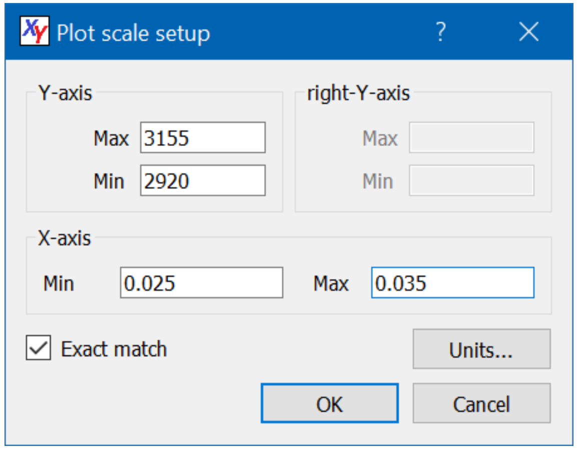
The Change scale button allows to manually change axes scales: this allows more flexibility than zooming: you can pan or choose detailed scaling. In the bottom-left plot the variable vDcLoadDcmeno is shown between 0 and 400 V, overriding the default scales.

In the bottom-right zooming on the horizontal axis is made.





Normally the user selected scales in the *Plot scale setup* dialog are subject to some refining by the scale determination inner engine to allow easy numbers on the numerical tic marks. If, however the user wants total freedom in choosing minimum and maximum values of an axis, he can attain this checking the Exact match checkbox. See the example below.



Y-axis

Max 3155

Min 2920

right-Y-axis

Max

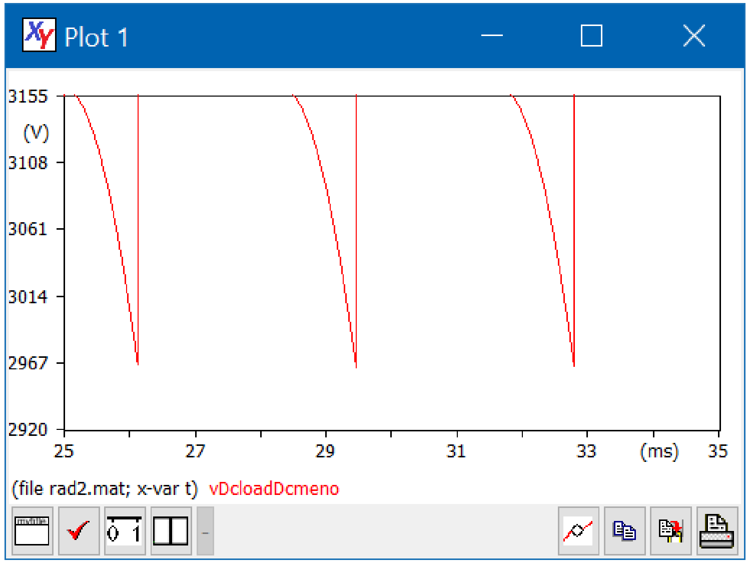
Min

X-axis

Min 0.025

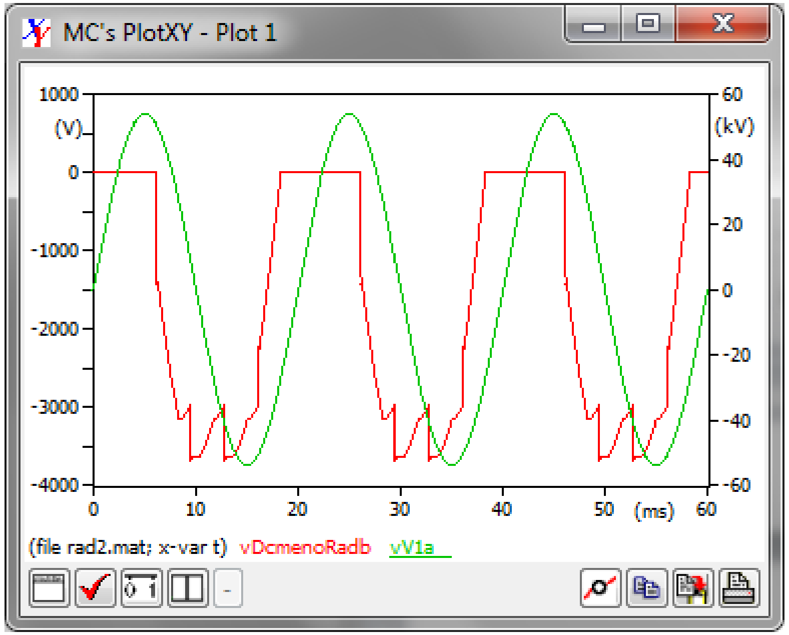
Max 0.035

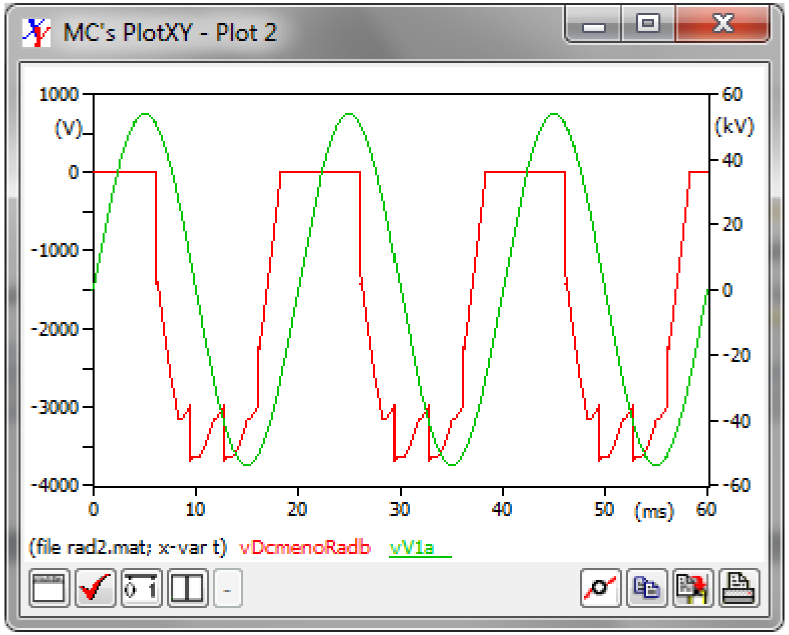
☑ Exact match

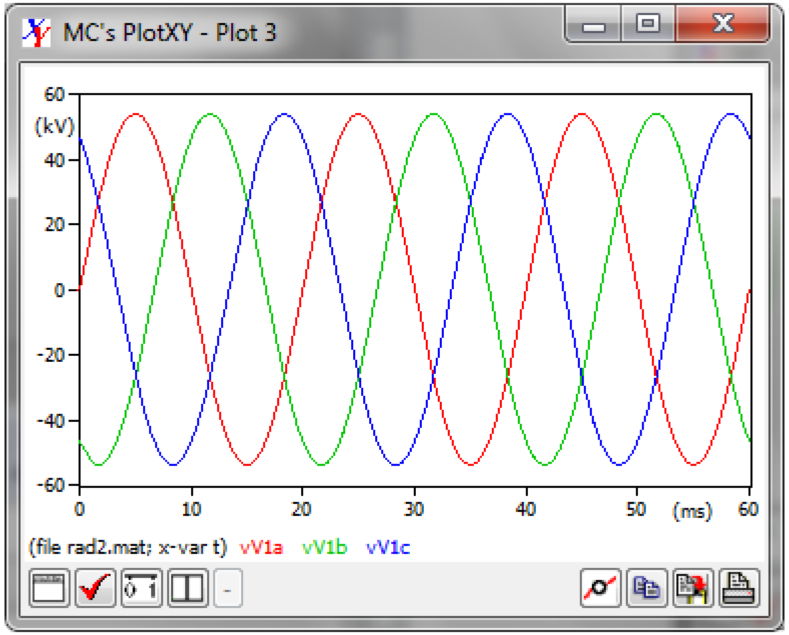


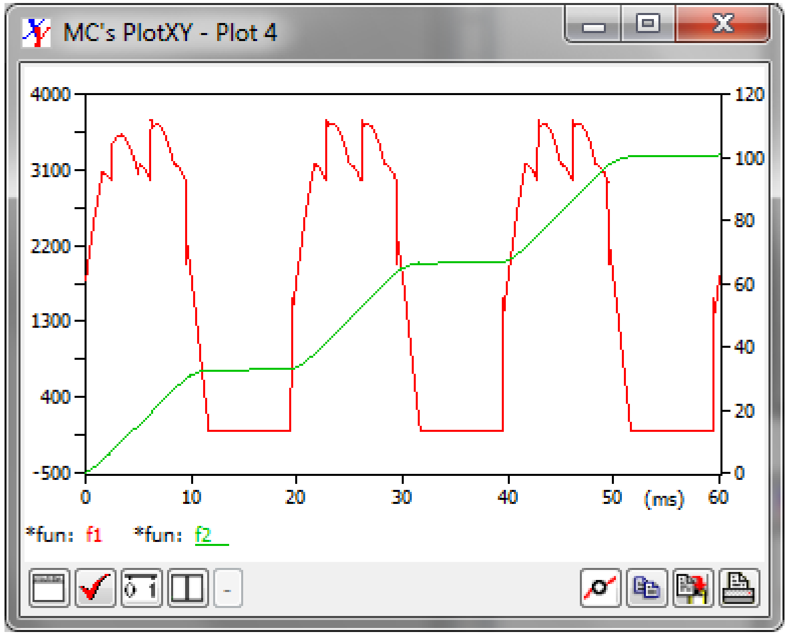
Using several program windows

Starting from the same files and variables you can shoe plots in up to four windows. Each window has its own setups, line style, scale, data-browse window, etc. Example are shown below.









Addition comments on these windows are in sect. “Management of the plot windows.”

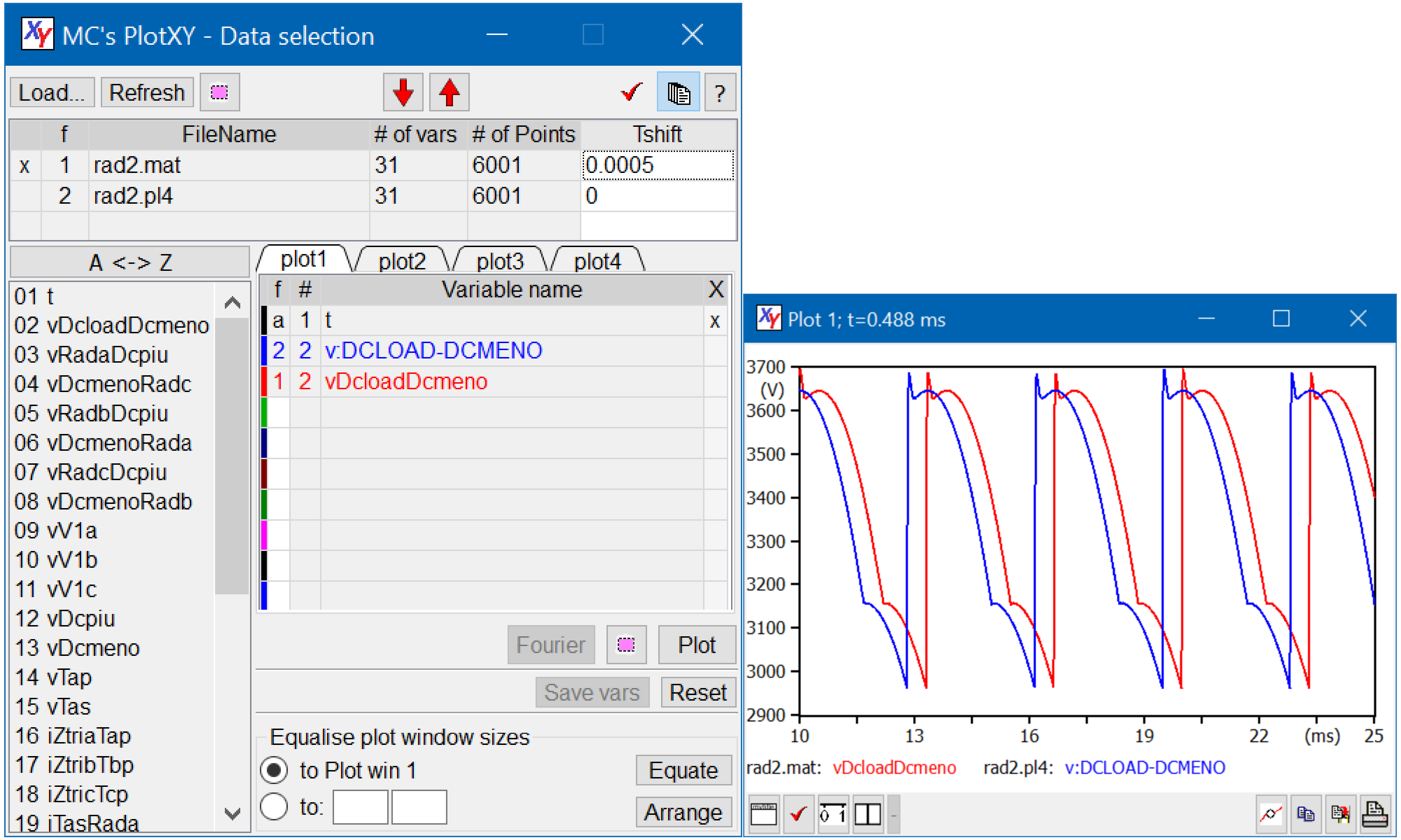
Postprocessing

PlotXY, in addition to plotting data from files, allows some post processing: finding the Fourier polynomial and making the corresponding bar charts, and creating new variables that are algebraic combination of others, or integral of others. These capabilities are discussed in this section.

Shifting time

In some cases, especially when we want to align in time measured and simulated curves, we want to add some offset to the time scale of one or more files. This cam be done clicking on the cell named “Tmax” on the FTable of the DataSelection window. Once that cell is clicked, the situation changes, and “TShift” field is shown in place of Tmax. Furthermore, the corresponding cells get a white background, to indicate that they can be written.

In the example shown below two files different in name but containing the same data are used. Rad2.mat, is retarded by 0.5 milliseconds and the result is shown in the plot at the right side of the DataSelWindow below.



	f	FileName	# of vars	# of Points	Tshift
x	1	rad2.mat	31	6001	0.0005
	2	rad2.pl4	31	6001	0

01 t
02 vDcloadDcmeno
03 vRadaDcpiu
04 vDcmenoRadc
05 vRadbDcpiu
06 vDcmenoRada
07 vRadcDcpiu
08 vDcmenoRadb
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas
16 iZtriaTap
17 iZtribTbp
18 iZtricTcp
19 iTasRada

f	#	Variable name	X
a	1	t	x
2	2	v:DCLOAD-DCMENO	
1	2	vDcloadDcmeno	

● to Plot win 1
○ to:

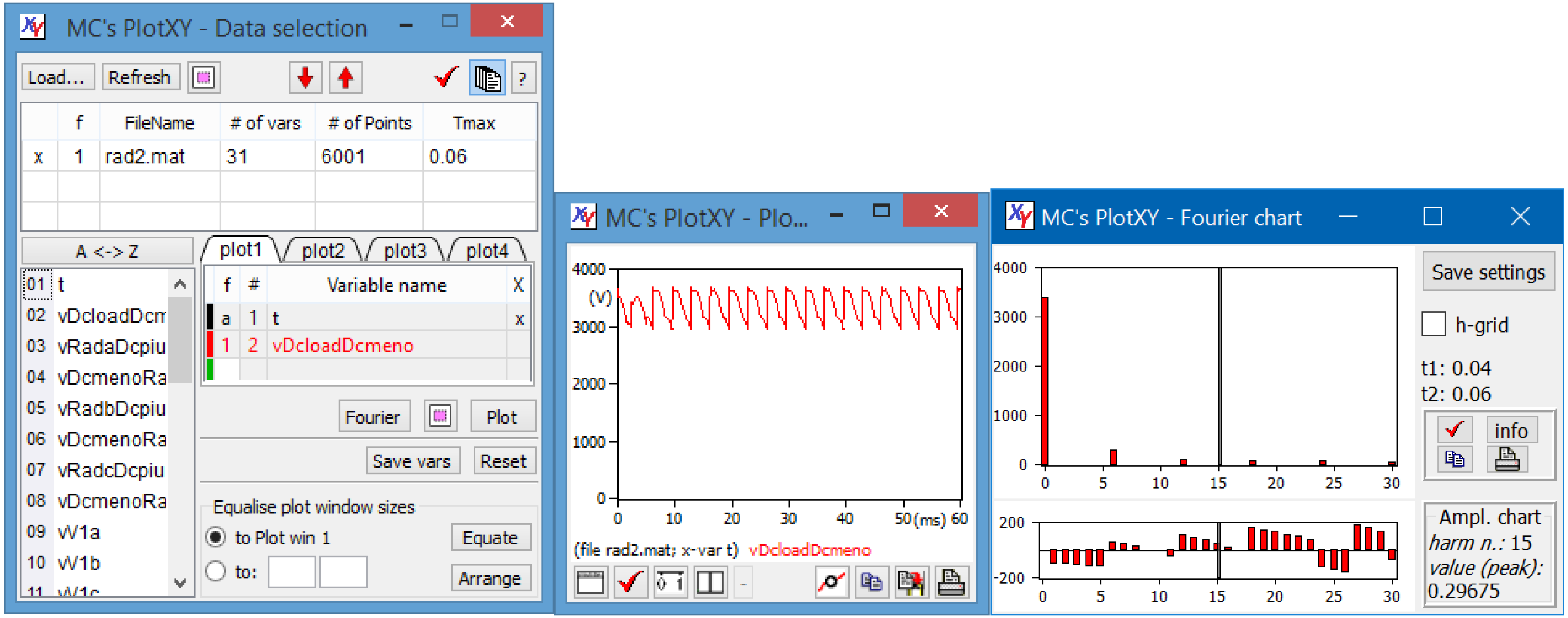
Creating a Fourier Chart

Electric and sound engineers use a lot Fourier decomposition of periodic waves.

This is supported by PlotXY, **assuming that the sampling time is constant**.

To make a Fourier chart only a unique variable must be selected. Then, simply click on the “Four” button. The program will try to choose the best parameters for general usage. In particular, it assumes that the simulation, after a possible initial transient has stabilised. Therefore, using the default frequency (that can be selected by means of the Plot Button of the DataSelection Window) it uses the last period of the available data to make computations.

Take for instance the following pictures, obtained using the supplied rad2.mat file. Here the default frequency is 50 Hz and the simulation goes from zero to 0.06 s, it will use as default the last twenty-millisecond period, for the analysis



	f	FileName	# of vars	# of Points	Tmax
x	1	rad2.mat	31	6001	0.06

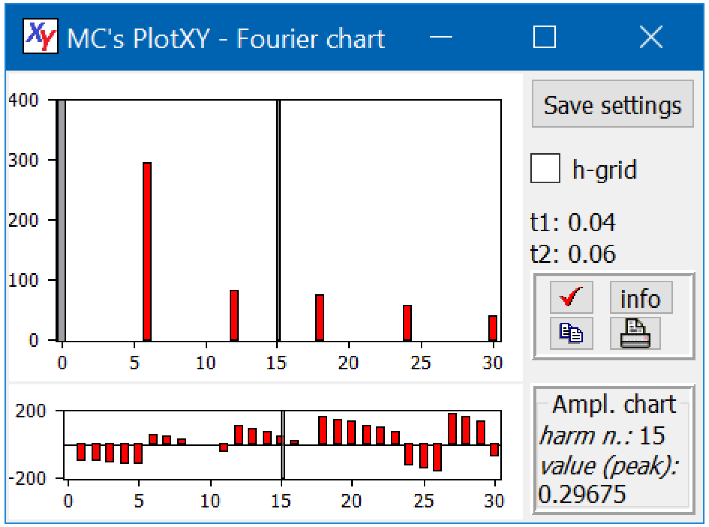
f	#	Variable name	X
a	1	t	x
1	2	vDcloadDcmeno	

t1: 0.04
t2: 0.06

harm n.: 15
value (peak): 0.29675

Once the chart is shown, you can move around the data cursors to see numerical values in the bottom-right part of the window.

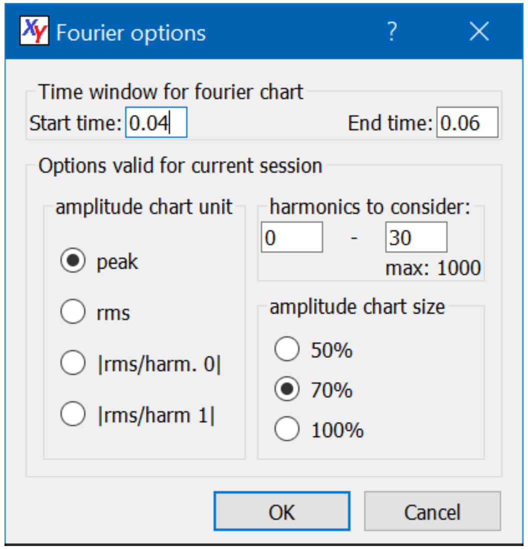
You can zoom the chart. An example is shown below (on the left). The bars that fall into the plot are shown as usually in red (bounded by black rectangles), while the bars that are higher than the available vertical space are shown greyed.



☐ h-grid

t1: 0.04
t2: 0.06

Ampl. chart
harm n.: 15
value (peak):
0.29675



Time window for fourier chart
Start time: 0.04 End time: 0.06

Options valid for current session

amplitude chart unit
● peak
○ rms
○ |rms/harm. 0|
○ |rms/harm 1|

harmonics to consider:
0 - 30
max: 1000

amplitude chart size
○ 50%
● 70%
○ 100%

The Fourier Chart can be customised in several ways clicking on the Fourier Options button. The window that appears has the aspect shown above (on the right). Note that the numerical values can be shown in the Fourier Chart window in four different values:

- peak of sine components (DC component unchanged)
- rms of sine components (DC component unchanged)
- all harmonics as a ratio to the DC component value
- all harmonics as a ratio to the first harmonic value.

By default, the program considers:
- a time-range between the T_f and T_f - 1/F (where T_f is the simulation final time and F is the default frequency as defined in the Program Options window).
- the harmonics between 0 and 30.

These defaults can be overridden clicking on the “Fourier options” button, just left of “info” button. Note that the maximum shown (in the example above max: 1000) is a recommended maximum based on signal’s Nyquist frequency.

The options chosen can be saved into the system data (system registry in windows) by clicking on the “Save settings” button, so that they are used for next Fourier charts, even at subsequent program executions. However, even if times t1 and t2 are saved, the first time per session a Fourier window is displayed the program tries to use for it the last period of signals, based on the default frequency as defined in the Program Options window. Any subsequent Fourier chart will be made using the couple (t1,t2) stored in the system data.

If we write a periodical variable x(t) as $x(t) = A_0 + \sum_1^{\infty} A_k \sin(k\omega t + \alpha_k)$, by default the program and plots the values A_k and α_k k being between 0 and 30. By means of the “Fourier options button these defaults can be overridden. Note that, since A_0 is the signal average in the period considered ($A_0 = \frac{1}{T}\int_t^{t+T} x(\tau)d\tau$), it can be negative.

The “info” button allows also seeing two THD measures, defined as follows:

$THD_0 = 100\sqrt{\sum_1^N A_k^2}/A_0$ $THD_1 = 100\sqrt{\sum_2^N A_k^2}/A_1$, N being the maximum number of harmonic components plotted.

If the minimum harmonic order to be displayed is greater than one, A_0 and A_1 will not be computed and the above formulas are undefined and the two THD’s will not be displayed. If this minimum order is one, only THD_1 will be displayed.

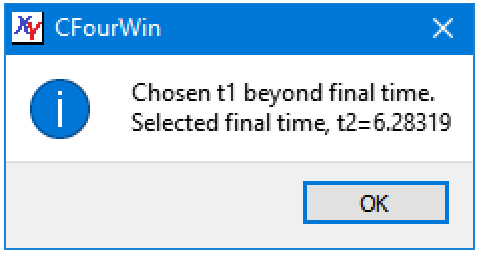
Technical info about times and samples

We set t1 and t2 in such a way that t2-t1 is exactly a period. This is perfectly clear when time is continuous; it must be somewhat interpreted when we have discrete samples. If for instance we have 101 samples with times going from 0 to T (T being the considered period), one of the two extreme samples must be excluded from computation (the values at the two extreme samples should be the same). If we considered all the samples, the error can be significant when the number of samples is small.

PlotXY is expected to be used to analyse transients. When fourier charts are requested, very often all transients disappear in the first part of the time span, and therefore, in these cases, the period to be considered for Fourier analysis is the *last* period T. Because of this, when t1 and t2 are selected, PlotXY considers:

- As the first sample the one corresponding to the smallest time t>t1
- As the last sample the one corresponding to the largest time t≤t2.

If the user wants to select as t2 exactly the last time in the samples, since this is not known up to the last significant digit, he can select as t2 a time larger than the last time present in the input files. A message will be issued indicating that the actual final time is used instead. If. For instance the final time is 2π, we can select t1=0, t2=7 and will receive the following message:



Chosen t1 beyond final time.
Selected final time, t2=6.28319

Using this choice, the Fourier computing algorithm has been checked with well consolidated scientific analysis program: Matlab^{TM} and Scilab, and gives exactly the same results (up to the last digit displayed).

Initial and final fourier times

Fourier charts are to be created over a time interval, which defines a period of the periodic shape to be analysed.

The user has control over these times, with some help from PlotXY, according to the following rules:

- At the first Fourier execution since PlotXY starting, the program tries to define the time interval automatically, based on the program's default frequency freq, as set in the Program Options: t2 is assumed to be equal to the final time of simulation, and t1=t2-1/freq. This rule comes from the assumption that typically simulations have transients, and at the end of transient shapes are stabilised (therefore periodic). If this results to be impossible because t1-1/freq results to be lower that the initial time as read from the input file, a message is issued and full time-range is used for analysis.
- The user can save Fourier window initial and final times when he saves the Four settings by means of the save settings button in the Fourier window, the saved interval is used for any Fourier plot later than the first (see above dot). In case the saved settings are inconsistent, a message is issued and full time-range is used for analysis.

Function plots

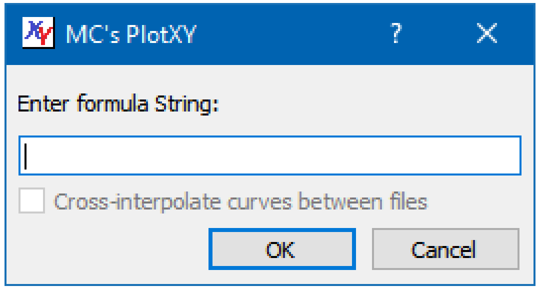
PlotXY not only allows power display of the contents of your variables, but also can manipulate your variables to obtain new ones and plot them. You can for instance sum, multiply, divide your variables. You can also take a mathematical function of a variable or an expression: available functions are sin(), cos(), abs(), exp(), sqrt(). Parentheses are allowed. In a way, we can say that we can create a function of your variables and plot that function. Because of this, this feature has been called "Function plot"

You can also request the time integral of a variable or an expression. In this case, the string you enter to define the function must begin with "int" and end with ")".

To define functions, you access your variables by a *compact name* built according to the following conventions:

- In general, the compact name indicates the file number and variable number with the scheme "f#v#". For instance, variable number 3 from file number 5 has as compact name "f3v5"
- To indicate Variables from the *Selected file* (that is indicated by a "x" in the first column of the *File list* table) the file indication can be dropped. For instance, the 5th variable from the Selected file can just be indicated ad "v5".

Functions are defined by clicking on a cell in the "f" column of the *File list table* in the *Data selection* window. A small window will appear prompting you to introduce the string defining your function. It will have the following aspect:



Enter formula String:

☐ Cross-interpolate curves between files

Note that the check-box named "cross-interpolate curves between files" in the current version will always remain unchecked. In the current version of the program, that disabled checkbox has been introduced to remind the user that *no cross-interpolation will be done.* *This means that the program assumes that in case of function plots involving variables from different files, the program assumers that all the samples refer to the same time instants. Necessary (but not sufficient) for this is that the numbers of points are exactly the same for all the files involved in the same function plots.*

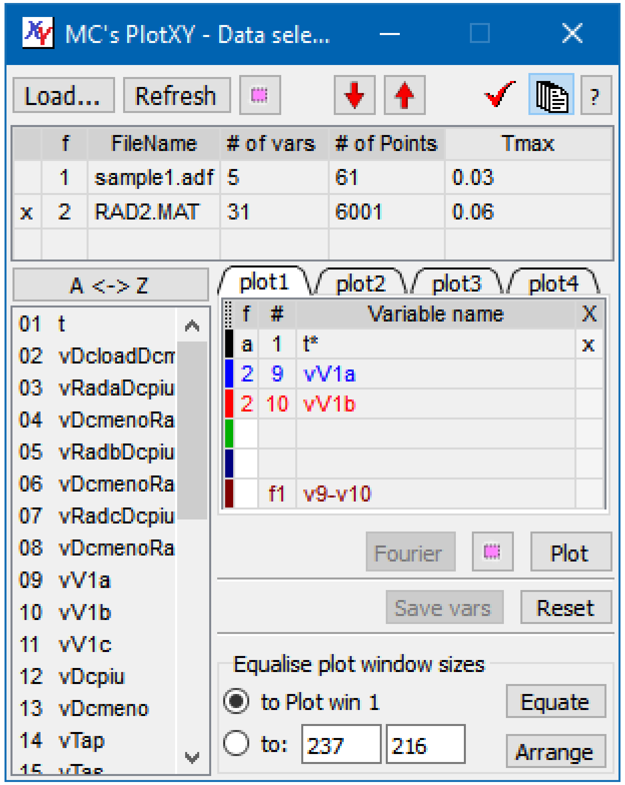
The allowed operators between variables are the following ones: +, -, *, /. As usual, * and / take precedence over + and -. Obviously division requires care, since there might be division by zero. If division by zero is detected a dialog box will be issued and computation of the function plot will be stopped.

Valid strings for instance are: "v9-v10", "2*v1-v2/5", "V1+1e3*v2", "1e-3*v1", "f1v1+f2v2+f1v3", "v1-(v2+v3)", "10*v9+20", "int(v2+v3)",. The latter example requests the integral between t=0 and the generic time *t* of v2+v3.

The first and last strings are used in the examples are shown below. The integral is shown in the same plot along with the function to be integrated for the maximum clarity.

NOTE. The name of the function is just indicated as "f1", "f2", etc. However, to ease analysis, if the mouse is left above the variable name shown in the plot window, a full representation of the formula is shown as a tooltip.

For instance, in in the picture right below the mouse is left on "f1" in the plot window the user will see in a tooltip: "vV1a-vV1b"

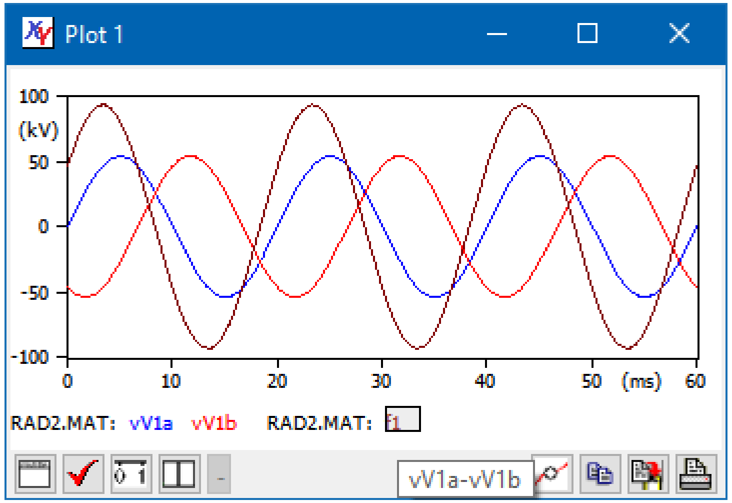


	f	FileName	# of vars	# of Points	Tmax
	1	sample1.adf	5	61	0.03
x	2	RAD2.MAT	31	6001	0.06

01 t
02 vDcloadDcm
03 vRadaDcpiu
04 vDcmenoRa
05 vRadbDcpiu
06 vDcmenoRa
07 vRadcDcpiu
08 vDcmenoRa
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas

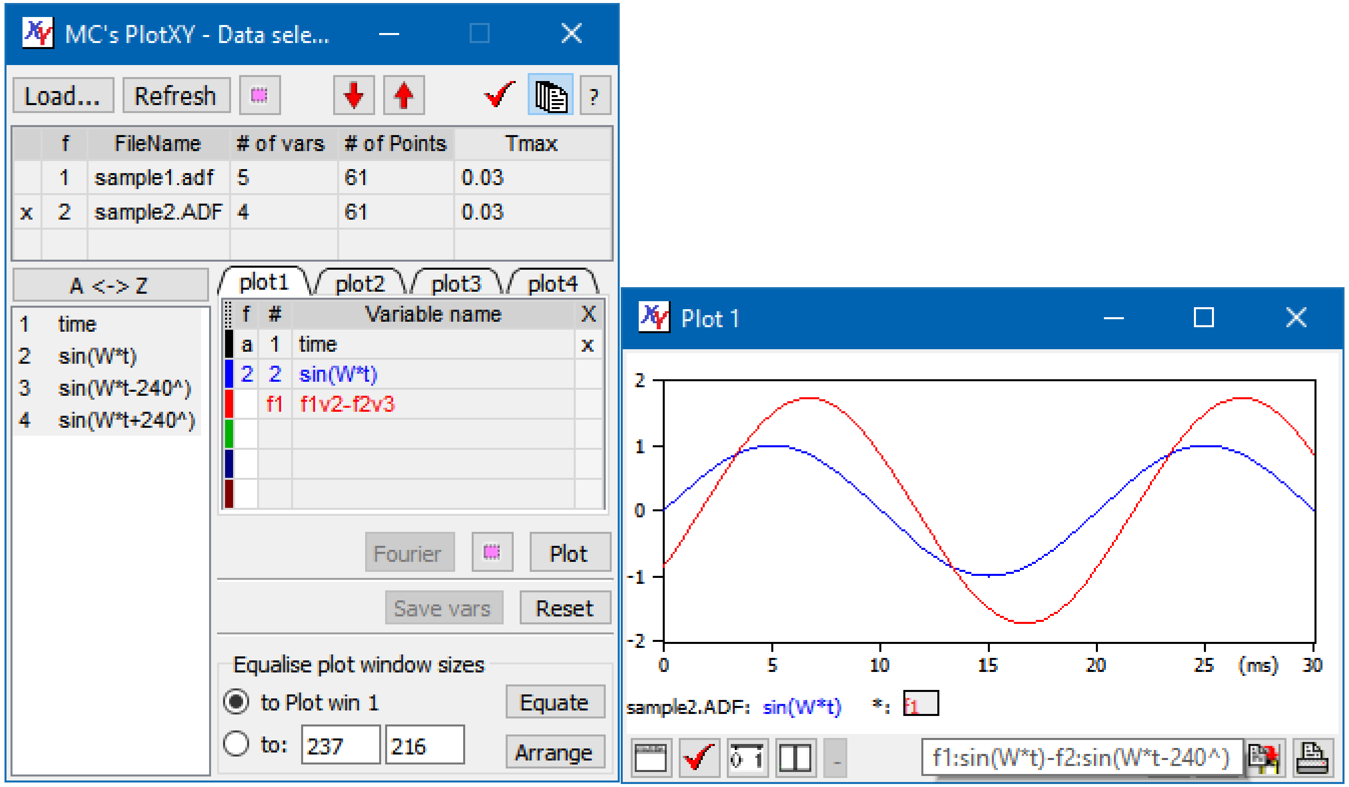
f	#	Variable name	X
a	1	t*	x
2	9	vV1a	
2	10	vV1b	
	f1	v9-v10	

Equalise plot window sizes
● to Plot win 1
○ to: 237 216



Note that If you hold the mouse pointer onto the function name in the plot window you get a full indication of what that function means (here “vV1a-vV1b”).

In case a function plot mixes variables from different files, each file is referred to as f#: f1 for file num 1, f2 for file num 2, etc. See next figure.



	f	FileName	# of vars	# of Points	Tmax
	1	sample1.adf	5	61	0.03
x	2	sample2.ADF	4	61	0.03

1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)

f	#	Variable name	X
a	1	time	x
2	2	sin(W*t)	
	f1	f1v2-f2v3	

Equalise plot window sizes

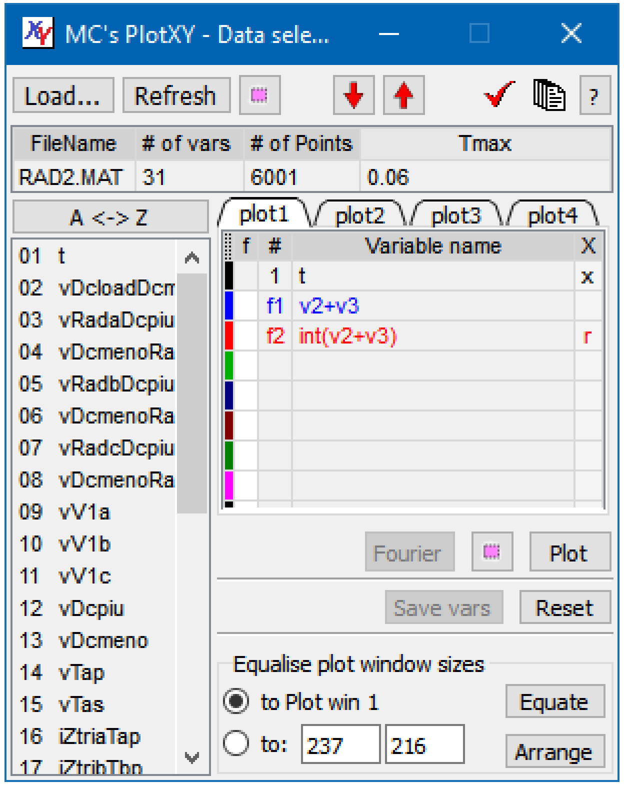
● to Plot win 1

○ to: 237 216

sample2.ADF: sin(W*t) *: f1

f1:sin(W*t)-f2:sin(W*t-240^)

Another interesting example of function plots is shown below:



FileName	# of vars	# of Points	Tmax
RAD2.MAT	31	6001	0.06

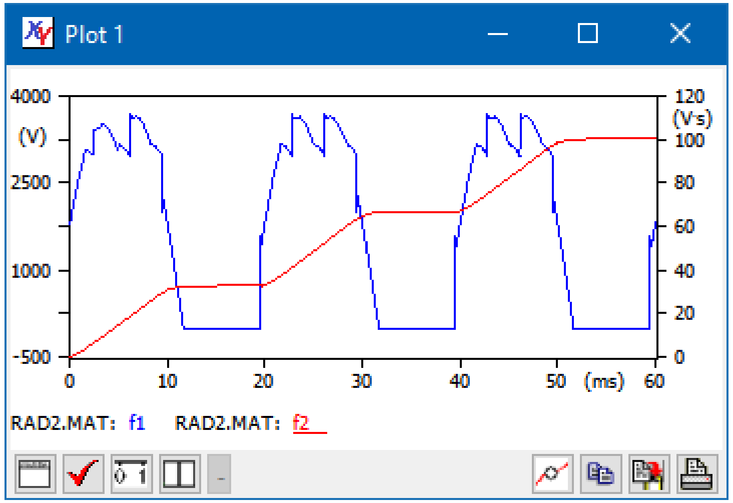
01 t
02 vDcloadDcm
03 vRadaDcpiu
04 vDcmenoRa
05 vRadbDcpiu
06 vDcmenoRa
07 vRadcDcpiu
08 vDcmenoRa
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas
16 iZtriaTap
17 iZtribTbp

f	#	Variable name	X
	1	t	x
	f1	v2+v3	
	f2	int(v2+v3)	r

Equalise plot window sizes

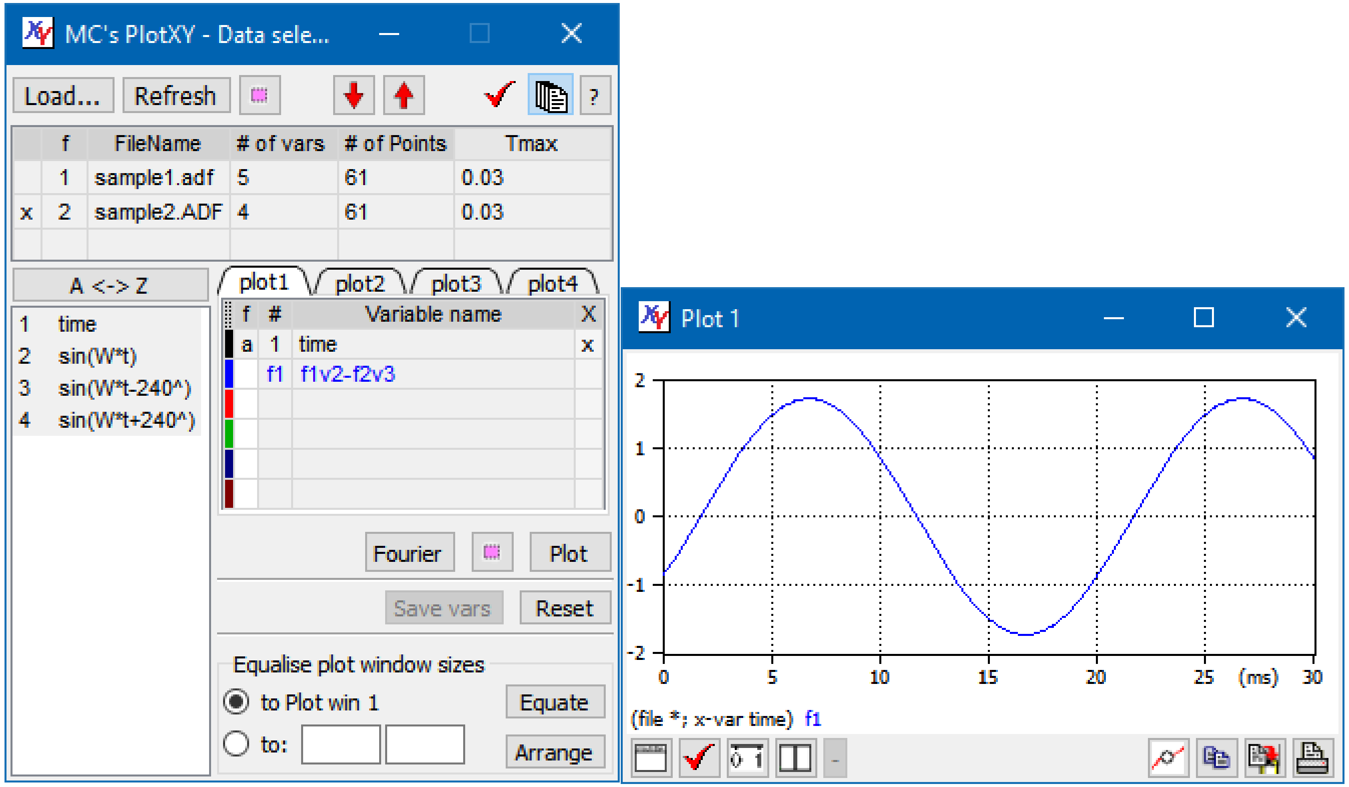
● to Plot win 1

○ to: 237 216



NOTE: integrals are useful to calculate energy flows. For instance, int(f*v) and int(u*i) will calculate the mechanical energy and electrical energy if f, v, u, i stand for force, speed, voltage, current respectively.

In the following picture an example of usage of function plots involving data from two different files is shown. Remember, as already noted, that the two files must have the same time instants at which the function is to be evaluated, and the program just verifies that the numbers of points are the same for all the involved files.



	f	FileName	# of vars	# of Points	Tmax
	1	sample1.adf	5	61	0.03
x	2	sample2.ADF	4	61	0.03

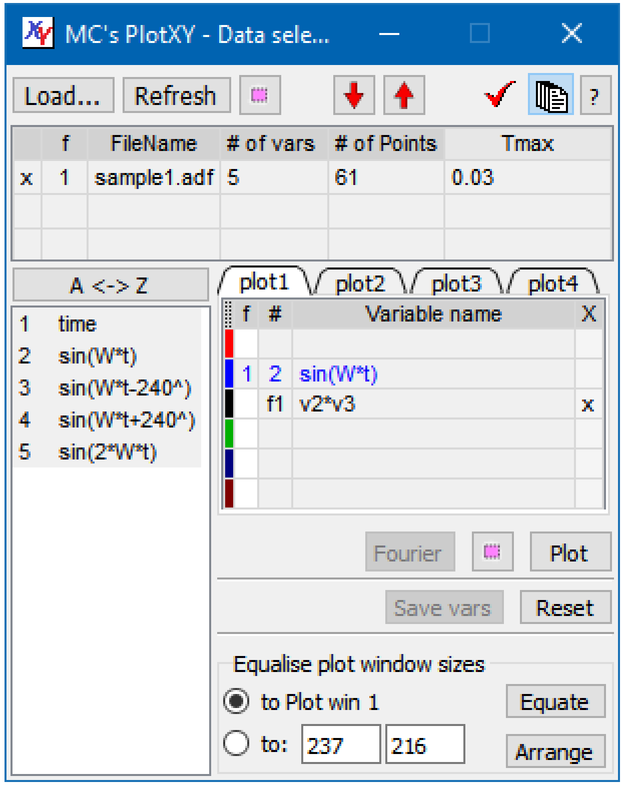
1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)

f	#	Variable name	X
a	1	time	x
	f1	f1v2-f2v3	

● to Plot win 1
○ to:

(file *; x-var time) f1

A function plot can be used as the x variable. Compare the two plots below:

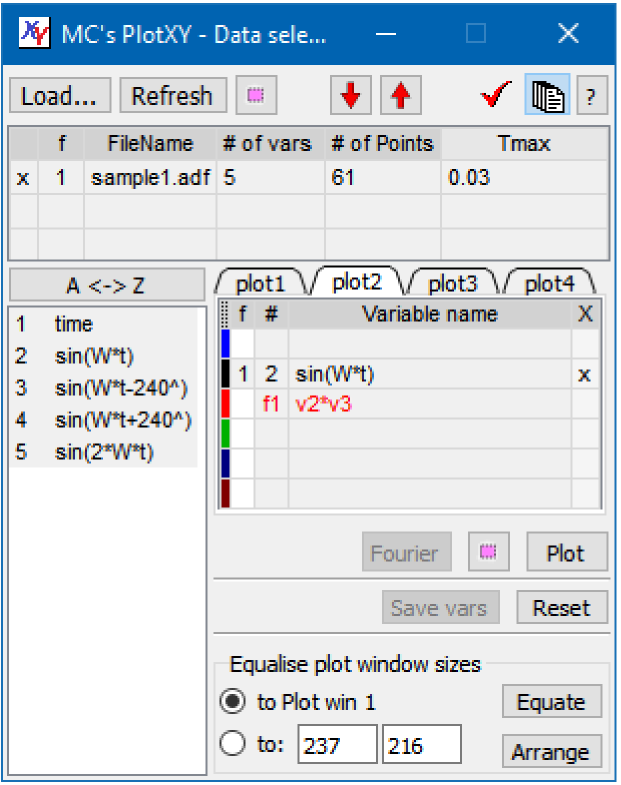


	f	FileName	# of vars	# of Points	Tmax
x	1	sample1.adf	5	61	0.03

1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)
5 sin(2*W*t)

f	#	Variable name	X
1	2	sin(W*t)	
	f1	v2*v3	x

● to Plot win 1
○ to: 237 216

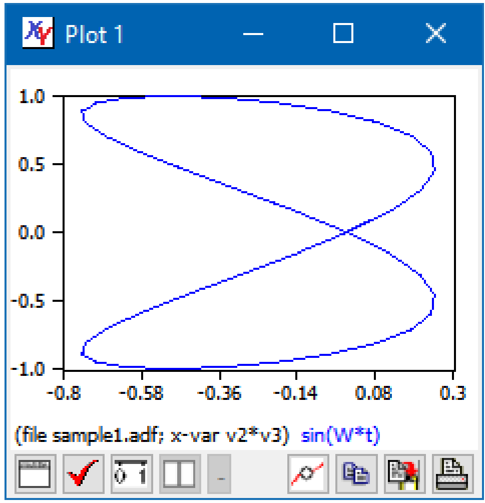


	f	FileName	# of vars	# of Points	Tmax
x	1	sample1.adf	5	61	0.03

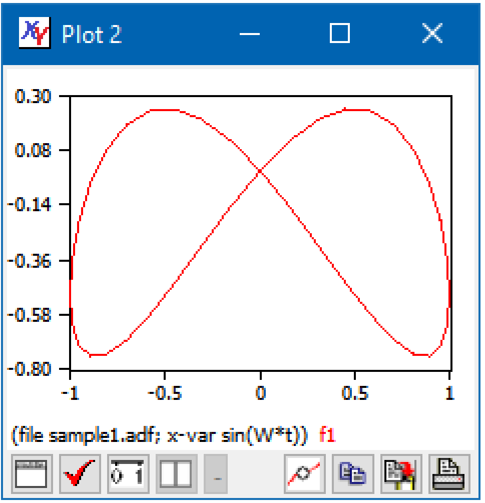
1 time
2 sin(W*t)
3 sin(W*t-240^)
4 sin(W*t+240^)
5 sin(2*W*t)

f	#	Variable name	X
1	2	sin(W*t)	x
	f1	v2*v3	

● to Plot win 1
○ to: 237 216



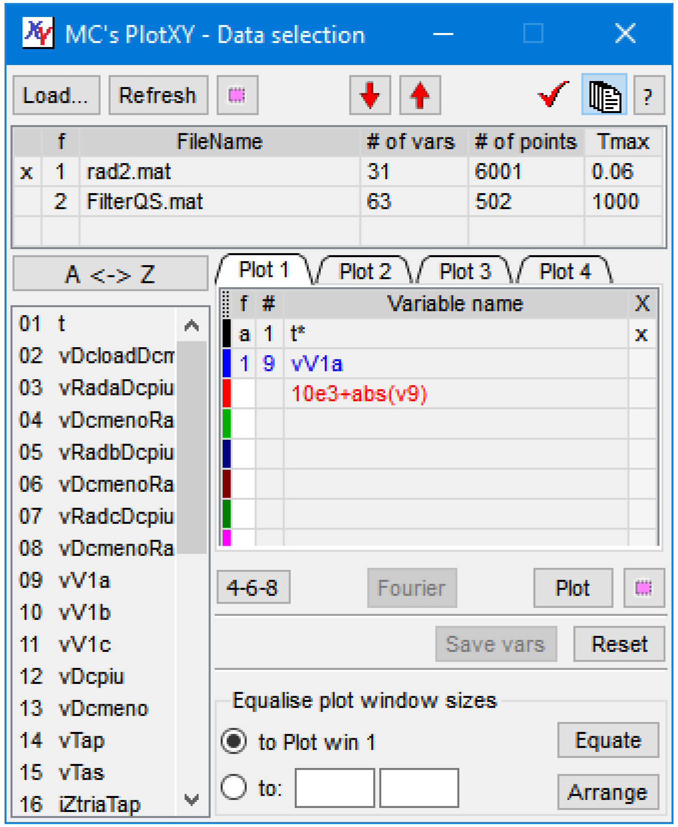
(file sample1.adf; x-var v2*v3) sin(W*t)



(file sample1.adf; x-var sin(W*t)) f1

Usage of mathematical functions in function plots

Finally, two examples of usage of mathematical functions inside function plots (that can be replicated using the supplied sample files; plots below the corresponding DataSelWindows):

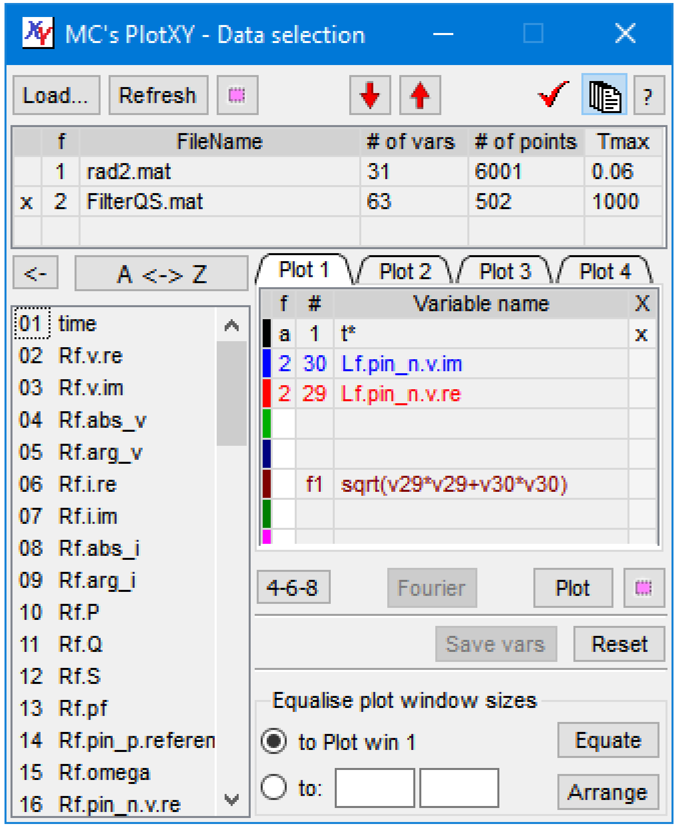


	f	FileName	# of vars	# of points	Tmax
x	1	rad2.mat	31	6001	0.06
	2	FilterQS.mat	63	502	1000

01 t
02 vDcloadDcm
03 vRadaDcpiu
04 vDcmenoRa
05 vRadbDcpiu
06 vDcmenoRa
07 vRadcDcpiu
08 vDcmenoRa
09 vV1a
10 vV1b
11 vV1c
12 vDcpiu
13 vDcmeno
14 vTap
15 vTas
16 iZtriaTap

f	#	Variable name	X
a	1	t*	x
1	9	vV1a	
		10e3+abs(v9)	

● to Plot win 1
○ to:

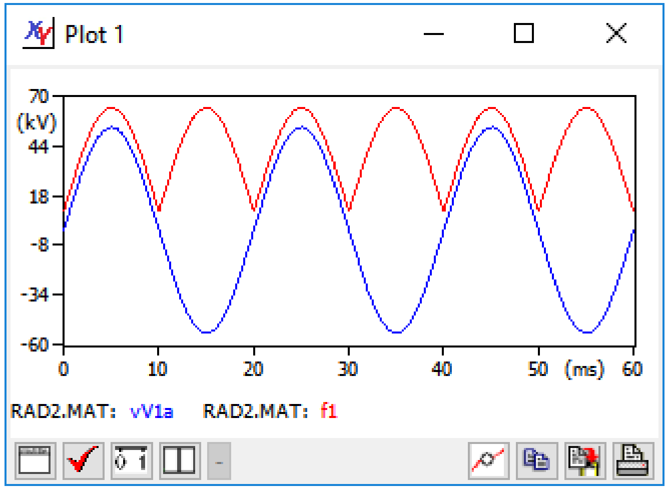


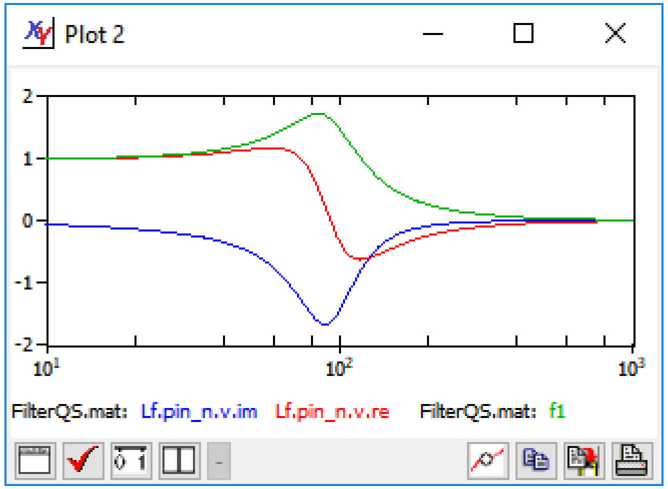
	f	FileName	# of vars	# of points	Tmax
	1	rad2.mat	31	6001	0.06
x	2	FilterQS.mat	63	502	1000

01 time
02 Rf.v.re
03 Rf.v.im
04 Rf.abs_v
05 Rf.arg_v
06 Rf.i.re
07 Rf.i.im
08 Rf.abs_i
09 Rf.arg_i
10 Rf.P
11 Rf.Q
12 Rf.S
13 Rf.pf
14 Rf.pin_p.referen
15 Rf.omega
16 Rf.pin_n.v.re

f	#	Variable name	X
a	1	t*	x
2	30	Lf.pin_n.v.im	
2	29	Lf.pin_n.v.re	
	f1	sqrt(v29*v29+v30*v30)	

● to Plot win 1
○ to:





In the first case we use the "abs function" within a function plot that requires also algebraic operations. In the second case we computed the complex amplitude of curves containing the corresponding real and imaginary parts. In this case, the horizontal axis was chosen to be logarithmic, for the maximum graphical effectiveness.

Working across sessions

PlotXY helps to re-enter rapidly the preferred working environment in new sessions with the possibility to save program state, to save Fourier settings, and to automatically load files. This is detailed in this section.

Customising the program behaviour allows more consistent experiences across sessions as well.

Saving Program State

When we work on PlotXY we load files, select variables, define functions, make plots.

When we close it, all this setup is in general lost. To avoid this, one can "save Program State" , i.e. the list of files loaded, variables and functions per file and plots shown, so that he can reload it again at a later time. The state is automatically stored in the Operating System's program setting area (in Windows this is the Windows registry) and can be recalled at will.

To save the program's state click on the "Save state" button on the main toolbar of The DataSelection Window.

Note that when working with PlotXY it may happen that a plot shows contents that does not correspond to the corresponding SelectedVar table. For instance we could select in the Plot1 SelectedVar table variables X and Y; plot them, and select a different set of variables without hitting the plot button again. When the program state is restored (clicking on the Load state on the Data Selection window), however plots always correspond to the corresponding SelectedVar table contents, as if the user has just hit the plot button.

As regards Fourier plot, PlotXY keeps trace of which plot table (e.g. “plot1”) is displayed when the fourier chart was last clicked. When reloading the state, if a single variable is present on that table, that variable’s fourier chart is drawn. This is done simulating a click on the “Fourier” button: therefore Fourier chart is done using the default settings (initial and final time, harmonics range, etc.).

NOTES:

- Fourier plots are not automatically generated when restoring the program state.
- The plot’s appearance (line vs bars vs dots, zoom state) is not stored.

Saving Fourier chart settings

While *Saving Program State* means to save which files are loaded and which variables and function plots (see section “Function plots”) are selected for each file, there are settings specific to Fourier Chart that can be saved from within the Fourier window itself.

When Fourier chart’s window “Save Settings” button is clicked the following settings are saved:

- whether the h-grid box is selected
- all the options selected in the “Fourier options” except Start time and End time.

Start and end time are not stored because these times are automatically computed when a Fourier chart is requested from the last part of the simulation: a period is estimated from the default frequency as specified in the Program Options window.

Automatic loading of files

It may happen that you always (or very often) load the same files or files having the same name

In this case, in addition to the “save state” feature you have another option: specifying the files at command line.

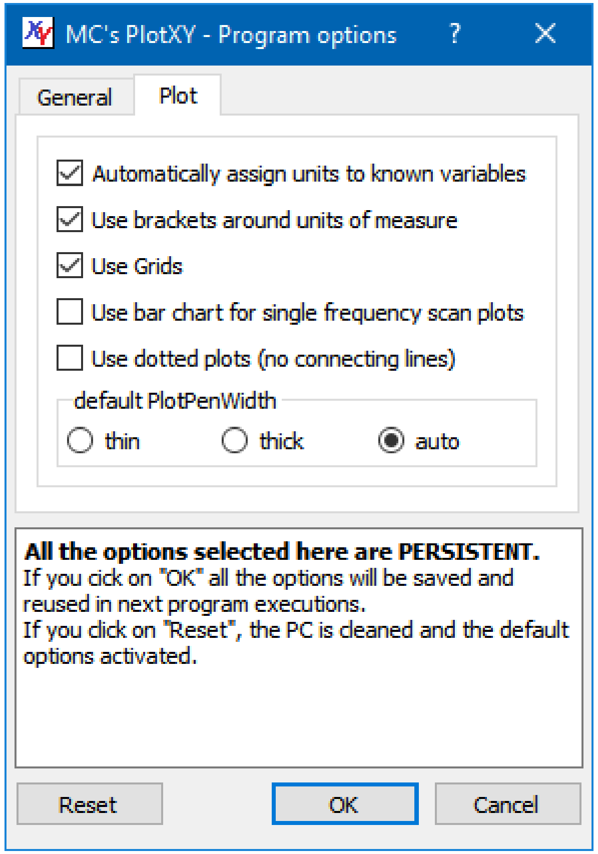
If, for instance, in Microsoft windows you can create a link to PlotXY.exe in your folder containing “sample1.adf” and “rad2.mat” and writing in the program name after “PlotXY.exe” (without quotes) “rad2.mat sample1.adf” (without quotes). Your PlotXY will open and automatically load the two files.

You can list up to eight files as command-line parameters. They will all automatically load.

Customizing the program behaviour

We have already met in section named “Loading a file and doing first things” the “Program Options window.

There we saw the appearance of the “General” tab sheet. We also have other options to choose from in tab sheet named “Plot”, that is shown below here.



☑ Automatically assign units to known variables
☑ Use brackets around units of measure
☑ Use Grids
☐ Use bar chart for single frequency scan plots
☐ Use dotted plots (no connecting lines)

default PlotPenWidth
○ thin ○ thick ● auto

All the options selected here are PERSISTENT.
If you cick on "OK" all the options will be saved and reused in next program executions.
If you click on "Reset", the PC is cleaned and the default options activated.

Their meaning is rather straightforward. Only the “Automatically assign units to known variables” might need some explanation and this is done in the following “Automatic Units of measure and prefix” section.

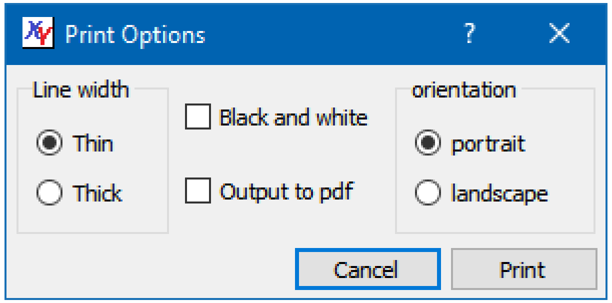
Note that when we hit the “Reset” button all program options stored on the PC, including program state (see “Saving Program State”)

And Fourier settings (see”Saving Fourier chart settings”) are reset, and the PC cleaned. At this point if PlotXY.exe is deleted, the PC is left totally cleaned of everything PlotXY has written on it.

Printing and exporting

Printing on paper or pdf

Plots can be printed on paper or pdf. The options at your disposal are those shown in the following diagram, that appears when you click on the “print” button of the plot window.



Print Options

Line width
● Thin
○ Thick

☐ Black and white
☐ Output to pdf

orientation
● portrait
○ landscape

Exporting selected variables into a new data file

Very often files containing output from simulations or measure contain a lot of variables, while only a few of them are of interest for a given purpose.

PlotXY is able to create new files with such a subset of variables. Currently, however, saving only can be made from variables coming from the same input file. Thus, variables from different files or variables created using Function plots , cannot be saved.

To export the selected variables into a new data file chick on the *Save vars* button of the in the *Data Selection* window.

Note that pl4 as output file type is only allowed when saving variables already belonging to a pl4 file. In this case the pl4 file created is of the so-called “pisa” format (or newpl4=2 format).

Exporting plot into system clipboard or SVG/PNG

Once you are satisfied with your plot you may want to export it somewhere.

Maybe you want to include it into a document file (e.g. Microsoft Word or OpenOffice writer): to do this you click on the *Copy* button on a Plot window, and then paste it into your favourite program. This way you get a pixmap copy (once called bitmap copy) of the plot.

As an alternative you can save you plots onto disk. In this case, you can obtain this result clicking, still in the plot window, on the *Save plot* button. You will get both a Portable Network Graphic (“PNG”) and a scalable Vector Graphic (“SVG”) plot copies on your hard disk. The PNG is a pixmap while SVG is a vector copy . The SVG has the advantage that it can be manipulated using a SVG editing program (such as the freely available *Inkscape* from www.inkscape.org or the commercial Microsoft's Visio). In this way you will find that curves will remain curves (can be moved around as a single object, text items will remain text items, etc. Moreover zooming can be done without loss of visual quality.

SVG files can be drag&dropped in Word 2016 or PowerPoint 2016. In this case, Word and PowerPoint keep a local copy of the created SVG, which can be edited using your favourite SVG editing program (e.g. Inkscape). If you edit embedded SVG plots, the edited plots are embedded independently on the original drag&dropped file.

Note that another way to have a picture on file of your plots is to use the Printing on paper or pdf feature.

NEW: Modelica compatibility!

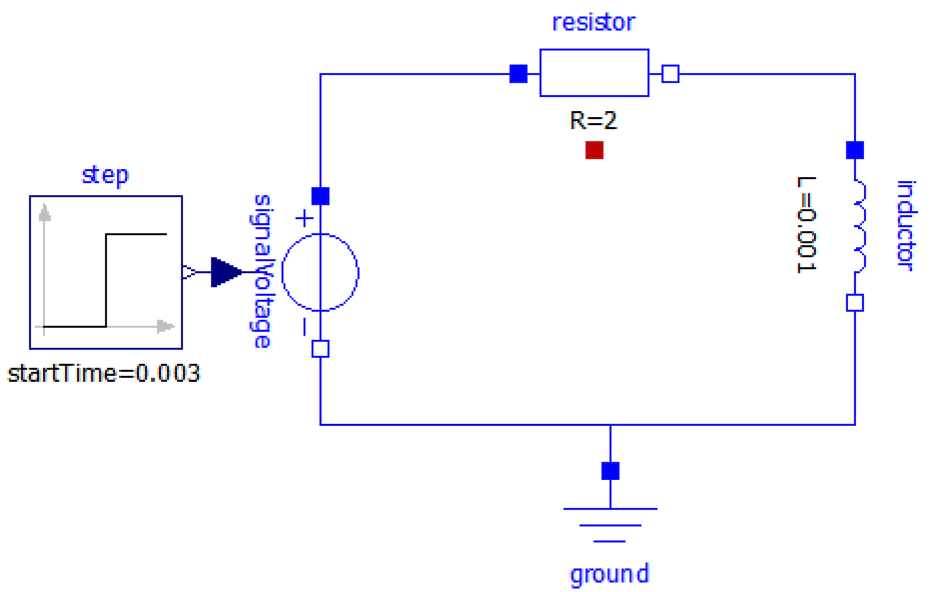
Modelica is a non-proprietary, object-oriented, equation based language to conveniently model complex physical systems containing, e.g., mechanical, electrical, electronic, hydraulic, thermal, control, electric power or process-oriented subcomponents.

Models written in Modelica language are converted into executable files and run through Modelica tools. Modelica tools are listed in the Internet site of the modelica association: https://www.modelica.org/tools

Two very much used tools are Dymola (commercial) and OpenModelica (free). Both, as default generate output files in the same binary format: a special version of Matlab® V4 files. It will be called here ModelicaMat format

Typically, to see outputs from Dymola and Modelica the user should have one of these tools installed.

It may, however, be useful to distribute output results to other people, what don't have (and don't like) to install any of these huge programs just to look at results and make some post-processing. This is where MC's PlotXY can be useful. You can load output files created with either Dymola and OpenModelica, in binary form, with PlotXY, and to all the actions you can do with it: watching plots, comparing curves, creating function pots, Fourier chart, exporting selected curves, etc.



The data in the binary Modelica files contains useful information in addition to the raw data and variable names. This allows MC’s PlotXY to do special things, which are explained here.

Note that the electric components are shown in their IEC standard symbols, but carry explicit terminals: one filled-blue (the positive terminal, named “è” and one empty blue (the negative one, named “n”

This is general for all Electric Analog Modelica components.

The left part of the circuit is a control part, with a signal output that control the generated signal voltage.

Every terminal has its own voltage (indeed potential), every two-terminal (one-port) component has its own voltage (difference of terminal potentials). The choice of signs is that the one-ort current is equal to the current entering it through its positive terminal. This richness of variables can make the list of variables rapidly grow. For instance in the above circuit it is

`resistor.p.i=resistor.i=-resistor.n.i` (resistor.p.i is the current “i” of terminal “p” of “resistor”)

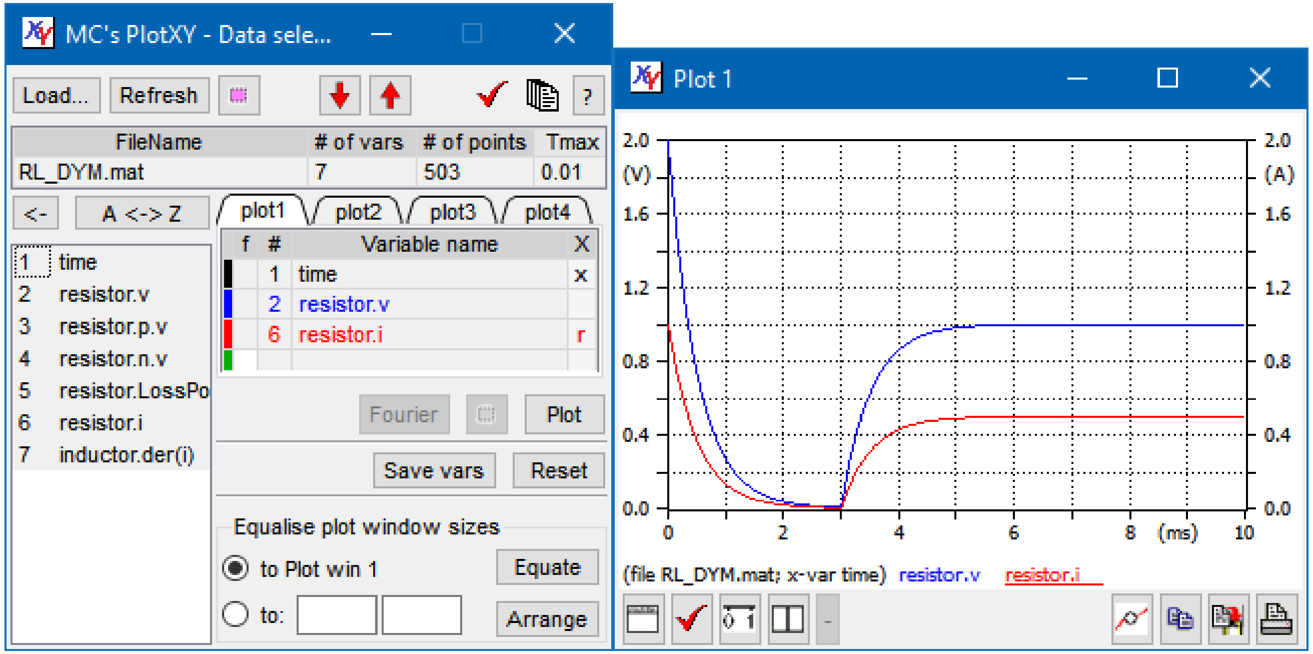
In the simple case of this circuit there are many variables that are equal to each other, or to the opposite of another variable. Namely:

```
signalVoltage.i=signalVoltagee.p.i=-signalVoltage.n.i=
resistor.i     =resistor.p.i      =-resistor.n.i=
inductor.i     =inductor.p.i      =.inductor.n.i
```

In the example above we can call one of the above nine quantities *base* variables, and the other *alias*. To reduce the simulation output size, ModelicaMat format allows storing only base quantities, while providing indication of the corresponding aliases. Dymola and OpenModelica exploit this characteristic, although not exactly in the same way. This way, there could be a number of aliases in OpenModelica slightly lower than in Dymola.

MC's PlotXY is able to take advantage of this special feature of ModelicaMat file structure. It can thus propose to the user either the full list of variables or just the base ones. Depending on his preferences.

This can be seen in the pictures below, taken from the supplied binary file RL_DYM, obtained running the circuit shown above.



FileName	# of vars	# of points	Tmax
RL_DYM.mat	7	503	0.01

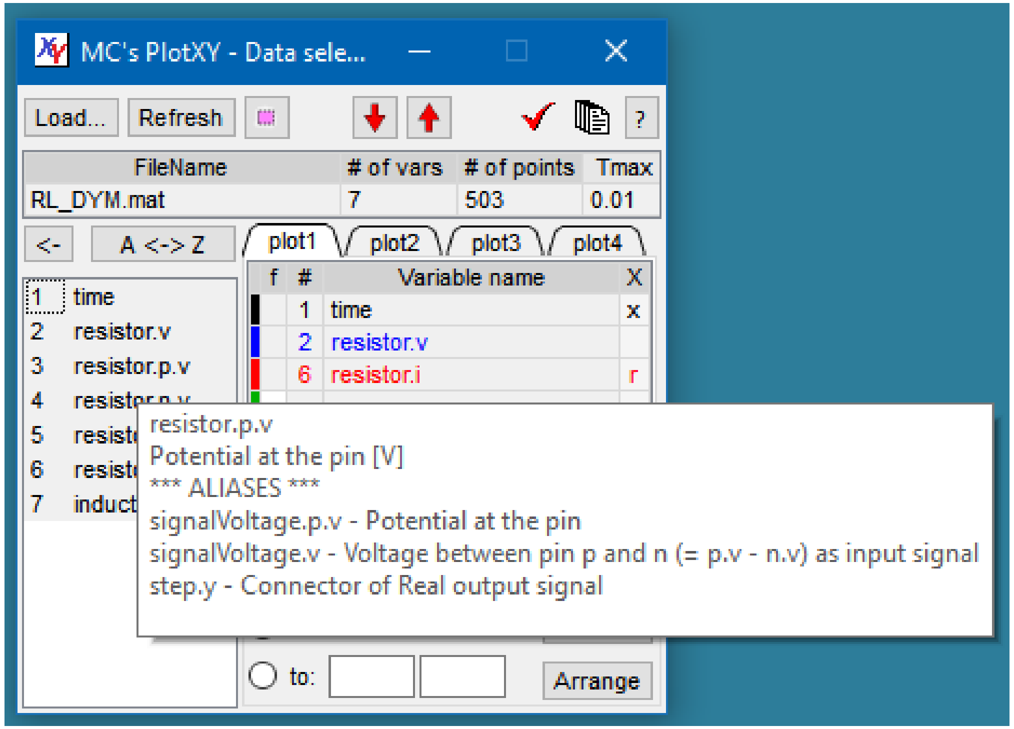
1 time
2 resistor.v
3 resistor.p.v
4 resistor.n.v
5 resistor.LossPo
6 resistor.i
7 inductor.der(i)

f	#	Variable name	X
⬛	1	time	x
🟦	2	resistor.v	
🟥	6	resistor.i	r
🟩			

Note that the unit of measure of quantities is directly taken from the ModelicaMat file, and thus does not require explicit definition. *However, currently (Oct 2017) unit of measure are not available in ModelicaMatFile produced by OpenModelica.*

Also note that the plot is a twin vertical axis plot, which is not currently (Oct 2017) available in OpenModelica.

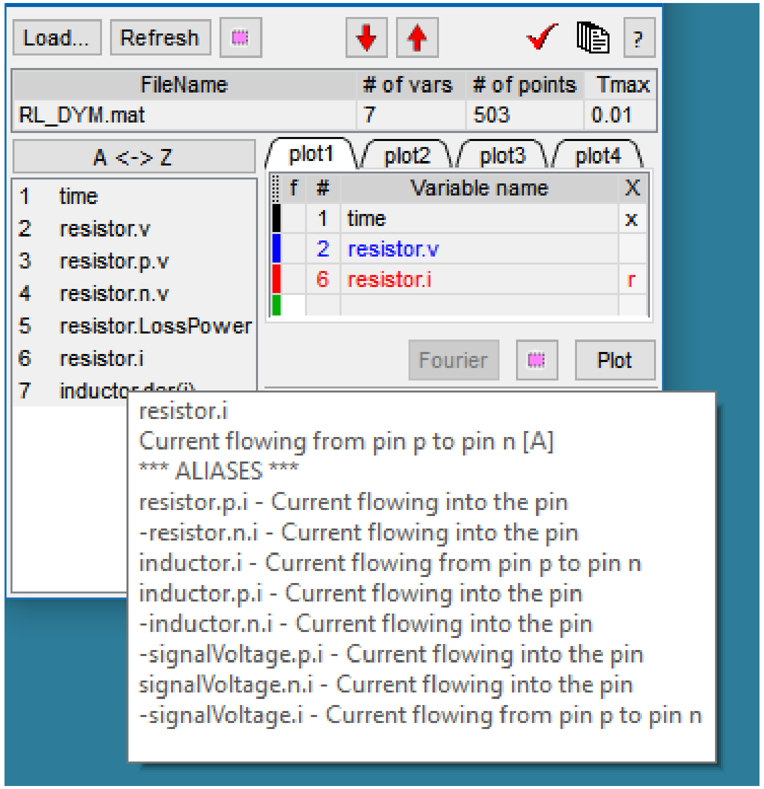
PlotXY supplies, as a hint a list of the alias variables. Two examples are shown below.



FileName	# of vars	# of points	Tmax
RL_DYM.mat	7	503	0.01

f	#	Variable name	X
	1	time	x
	2	resistor.v	
	6	resistor.i	r

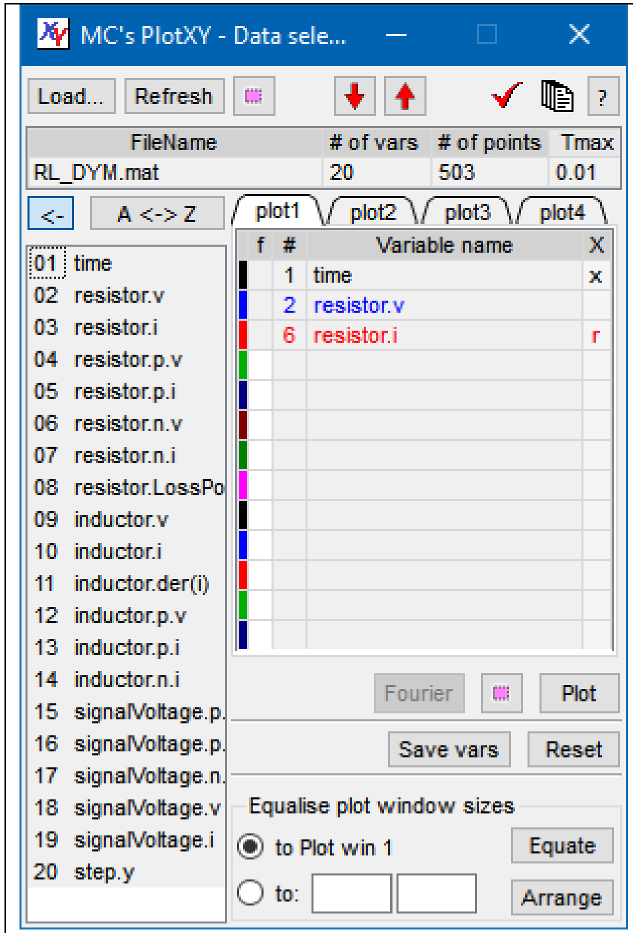
resistor.p.v
Potential at the pin [V]
*** ALIASES ***
signalVoltage.p.v - Potential at the pin
signalVoltage.v - Voltage between pin p and n (= p.v - n.v) as input signal
step.y - Connector of Real output signal



FileName	# of vars	# of points	Tmax
RL_DYM.mat	7	503	0.01

f	#	Variable name	X
	1	time	x
	2	resistor.v	
	6	resistor.i	r

resistor.i
Current flowing from pin p to pin n [A]
*** ALIASES ***
resistor.p.i - Current flowing into the pin
-resistor.n.i - Current flowing into the pin
inductor.i - Current flowing from pin p to pin n
inductor.p.i - Current flowing into the pin
-inductor.n.i - Current flowing into the pin
-signalVoltage.p.i - Current flowing into the pin
signalVoltage.n.i - Current flowing into the pin
-signalVoltage.i - Current flowing from pin p to pin n



FileName	# of vars	# of points	Tmax
RL_DYM.mat	20	503	0.01

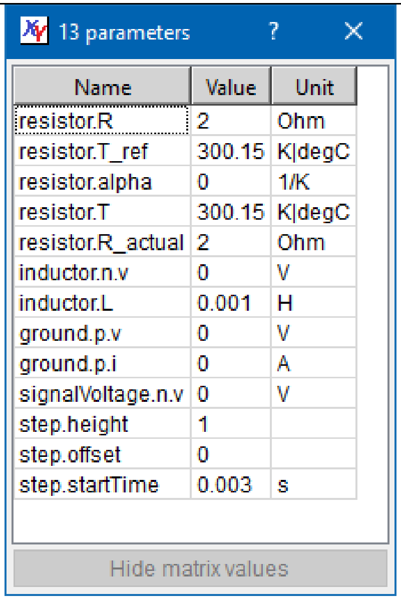
01 time
02 resistor.v
03 resistor.i
04 resistor.p.v
05 resistor.p.i
06 resistor.n.v
07 resistor.n.i
08 resistor.LossPo
09 inductor.v
10 inductor.i
11 inductor.der(i)
12 inductor.p.v
13 inductor.p.i
14 inductor.n.i
15 signalVoltage.p.
16 signalVoltage.p.
17 signalVoltage.n.
18 signalVoltage.v
19 signalVoltage.i
20 step.y

f	#	Variable name	X
	1	time	x
	2	resistor.v	
	6	resistor.i	r

Note that the hint contains:

- description of the base variable with its unit of measure
- list of alias variables, with their descriptions and indication of their sign.

Just in case the user prefers not to have alias variables he can unselect among the program options “Compact VarMenu with MatModelica files” and the full list of variables will be available. In this case 20 variables, as shown aside.



Name	Value	Unit
resistor.R	2	Ohm
resistor.T_ref	300.15	K|degC
resistor.alpha	0	1/K
resistor.T	300.15	K|degC
resistor.R_actual	2	Ohm
inductor.n.v	0	V
inductor.L	0.001	H
ground.p.v	0	V
ground.p.i	0	A
signalVoltage.n.v	0	V
step.height	1	
step.offset	0	
step.startTime	0.003	s

As a final feature of Modelica compatibility is the option to display constant quantities.

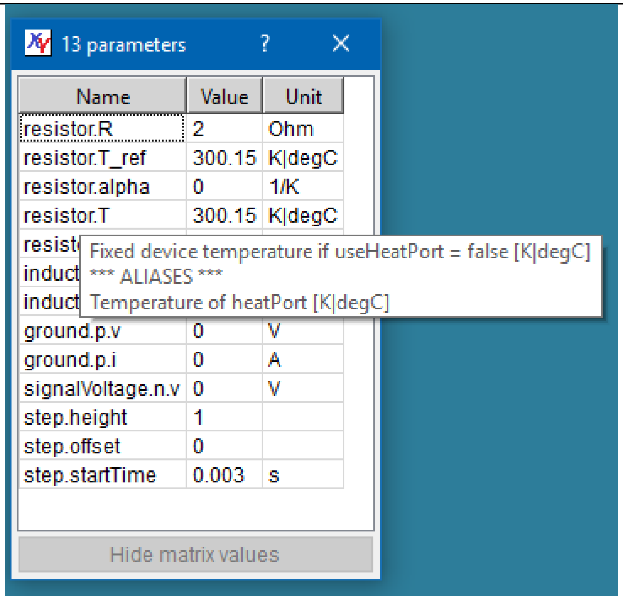
In the ModelicaMat files a single number can be stored for constant quantities, i.e.

- parameters, such as resistance and inductance in the circuit above
- quantities that in principle arefunction of time but that stay constant because of physical constraints. For instance ground voltage and current in the circuit above.

These constant quantities are called “parameters” in PlotXY. The Parameters window shows the numbers of parameters in its title.

The values of these constant variables can be accessed through the tiny button, labelled “<-“ that appears left of “A<->Z” in the header of the VarMenuTable in the DataSel window.

Aside you can see what is displaiyed for the above mentioned RL circuit.



Name	Value	Unit
resistor.R	2	Ohm
resistor.T_ref	300.15	K|degC
resistor.alpha	0	1/K
resistor.T	300.15	K|degC
resist		
induct		
induct		
ground.p.v	0	V
ground.p.i	0	A
signalVoltage.n.v	0	V
step.height	1	
step.offset	0	
step.startTime	0.003	s

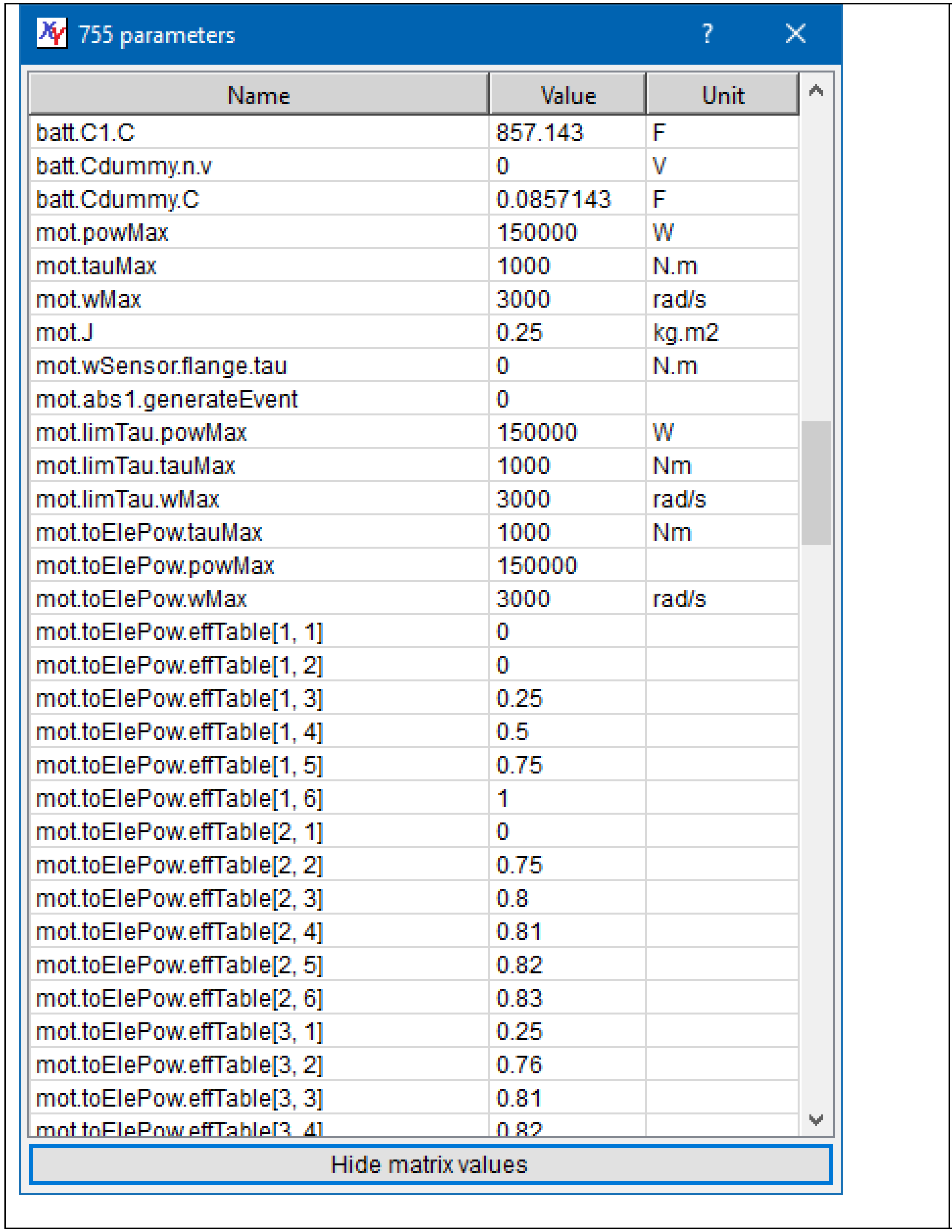
Fixed device temperature if useHeatPort = false [K|degC]
*** ALIASES ***
Temperature of heatPort [K|degC]

Also parameters might have aliases, which are not shown explicitly, but quoted in the tooltip. For instance the already mentioned RL_DYM.mat file has an alias of the value of resistor.T, which is shown with its tooltip as shown asode.

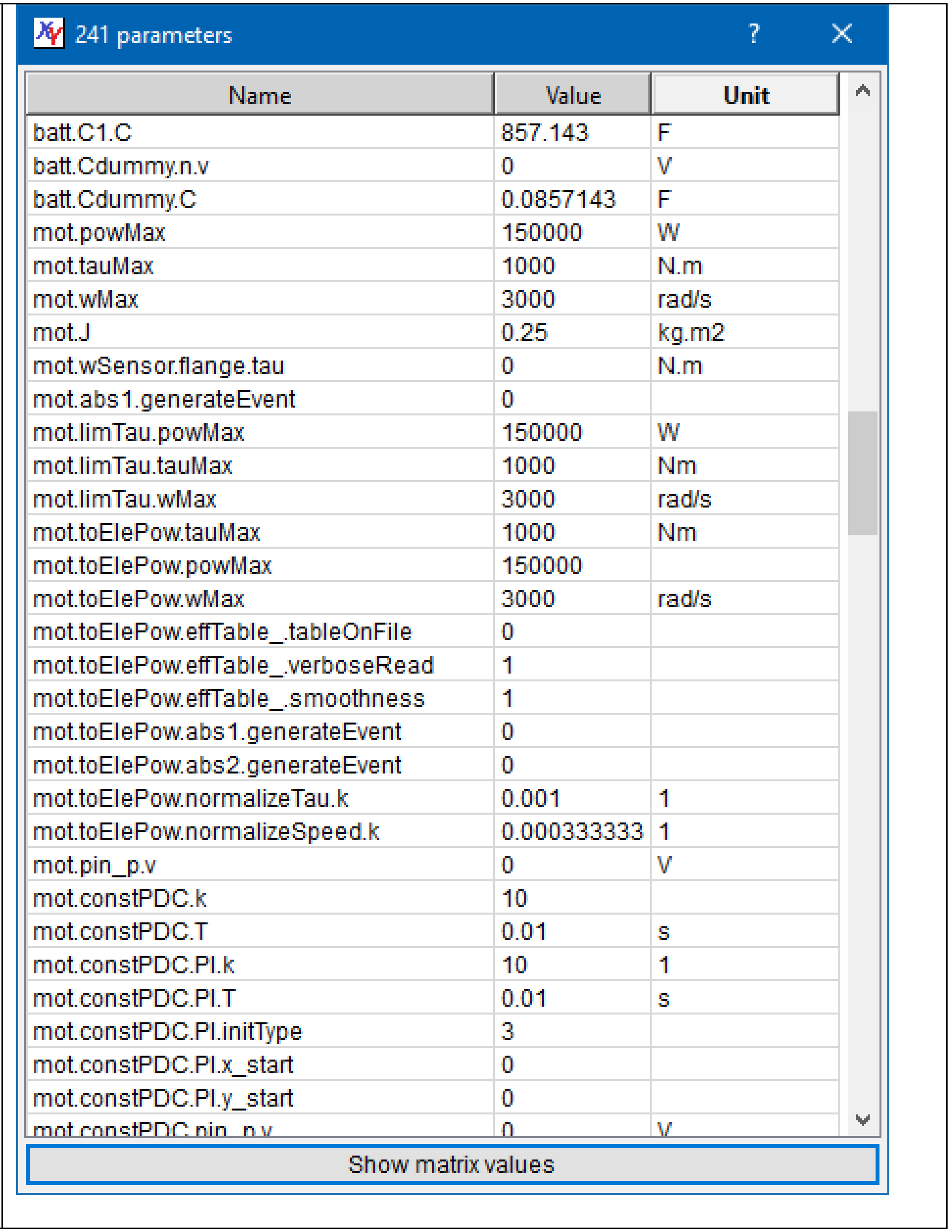
There are models in Modelica in which matrices of parameters are used, which can clutter the parameters table, making it difficult to look as the scalar ones. In these cases the “Hide matrix values” at the bottom of the window can do an useful job. Check the tow images below.

Finally, it must be noted that in some cases the models have matrix parameters that dominate the parameter list making it difficult to see the values of scalar ones. In these cases some help can be obtained by hiding the matrix parameters, taking advantage of the button shown in the bottom of the parameter windows.

An example is shown below.



Name	Value	Unit
batt.C1.C	857.143	F
batt.Cdummy.n.v	0	V
batt.Cdummy.C	0.0857143	F
mot.powMax	150000	W
mot.tauMax	1000	N.m
mot.wMax	3000	rad/s
mot.J	0.25	kg.m2
mot.wSensor.flange.tau	0	N.m
mot.abs1.generateEvent	0	
mot.limTau.powMax	150000	W
mot.limTau.tauMax	1000	Nm
mot.limTau.wMax	3000	rad/s
mot.toElePow.tauMax	1000	Nm
mot.toElePow.powMax	150000	
mot.toElePow.wMax	3000	rad/s
mot.toElePow.effTable[1, 1]	0	
mot.toElePow.effTable[1, 2]	0	
mot.toElePow.effTable[1, 3]	0.25	
mot.toElePow.effTable[1, 4]	0.5	
mot.toElePow.effTable[1, 5]	0.75	
mot.toElePow.effTable[1, 6]	1	
mot.toElePow.effTable[2, 1]	0	
mot.toElePow.effTable[2, 2]	0.75	
mot.toElePow.effTable[2, 3]	0.8	
mot.toElePow.effTable[2, 4]	0.81	
mot.toElePow.effTable[2, 5]	0.82	
mot.toElePow.effTable[2, 6]	0.83	
mot.toElePow.effTable[3, 1]	0.25	
mot.toElePow.effTable[3, 2]	0.76	
mot.toElePow.effTable[3, 3]	0.81	
mot.toElePow.effTable[3, 4]	0.82	



Name	Value	**Unit**
batt.C1.C	857.143	F
batt.Cdummy.n.v	0	V
batt.Cdummy.C	0.0857143	F
mot.powMax	150000	W
mot.tauMax	1000	N.m
mot.wMax	3000	rad/s
mot.J	0.25	kg.m2
mot.wSensor.flange.tau	0	N.m
mot.abs1.generateEvent	0	
mot.limTau.powMax	150000	W
mot.limTau.tauMax	1000	Nm
mot.limTau.wMax	3000	rad/s
mot.toElePow.tauMax	1000	Nm
mot.toElePow.powMax	150000	
mot.toElePow.wMax	3000	rad/s
mot.toElePow.effTable_tableOnFile	0	
mot.toElePow.effTable_verboseRead	1	
mot.toElePow.effTable_smoothness	1	
mot.toElePow.abs1.generateEvent	0	
mot.toElePow.abs2.generateEvent	0	
mot.toElePow.normalizeTau.k	0.001	1
mot.toElePow.normalizeSpeed.k	0.000333333	1
mot.pin_p.v	0	V
mot.constPDC.k	10	
mot.constPDC.T	0.01	s
mot.constPDC.PI.k	10	1
mot.constPDC.PI.T	0.01	s
mot.constPDC.PI.initType	3	
mot.constPDC.PI.x_start	0	
mot.constPDC.PI.y_start	0	
mot.constPDC.pin_p.v	0	V

You can see that the huge matrix mot.toElePow.effTable is not displayed anymore. The number of total parameters displayed, in this case, has changed from 755 to 241, as shown in the parameter table title.

Additional information

Management of the plot windows.

PlotXY has a special architecture, which creates individual windows for individual plots, which are also separated from the main window, the DataSelection Window. When working on a cluttered desktop, it may happen that some of the plot windows finish below other windows (from other programs running).

To put on top of the desktop all the active plot windows, select the DataSelection window, and hover with the mouse on its area.

On the other hand, to have in the DataSelection window the tab corresponding to a plot window to be displayed, select a plot window and hover in the plot area with the mouse.

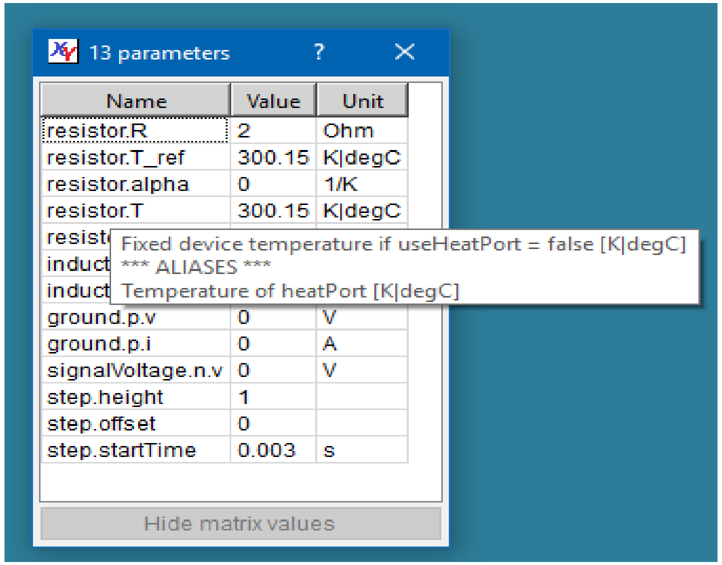
MC's PlotXY is multiple-screen aware. If the system on which it operates has an extended screen, windows can be put on it, and saved. However, if when the program is restarted the extended screen is not available anymore, windows are moved into the primary screen so that they are visible and manageable.

Automatic determination of scale span

PlotXY contains an algorithm to determine automatically the span of the variables on the axes in such a way that the actual plot covers a large part of the plot rectangle (at least 80%) and the number on the axes are "round", i.e. they contain the least possible significant digits.

The following table, for instance, contains the axes extrema, which are computed with given minimum and maximum values of the displayed variable.

Actual extrema		***Rounded extrema***	
Min	*Max*	***Min***	***Max***
0	1.05	**0**	**1.2**
-1.05	1.05	**-1.2**	**1.2**
-1000	10	**-1000**	**200**
0.555	0.6	**0.55**	**0.6**
-0.01	100.3	**0**	**100**

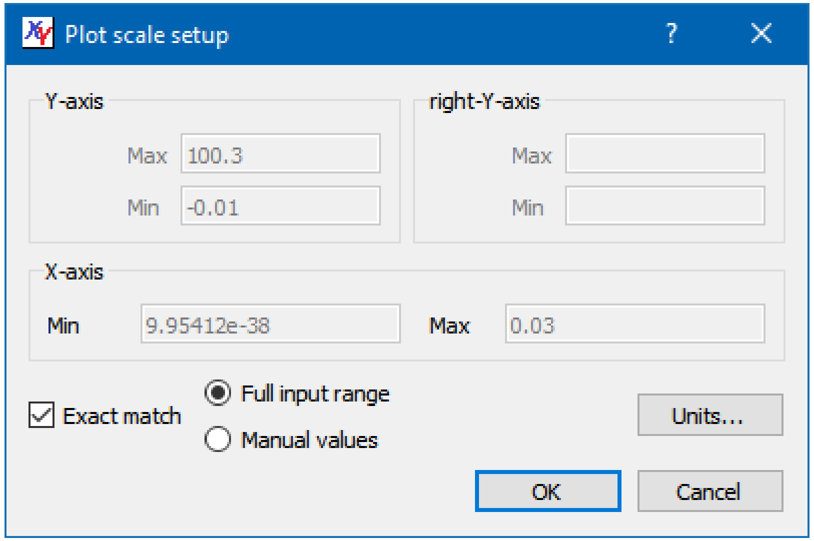


Name	Value	Unit
resistor.R	2	Ohm
resistor.T_ref	300.15	K|degC
resistor.alpha	0	1/K
resistor.T	300.15	K|degC
resist		
induct		
induct		
ground.p.v	0	V
ground.p.i	0	A
signalVoltage.n.v	0	V
step.height	1	
step.offset	0	
step.startTime	0.003	s

Fixed device temperature if useHeatPort = false [K|degC]
*** ALIASES ***
Temperature of heatPort [K|degC]

The last row in the table is a special case: the software cannot find round extrema to contain the whole plot, but finds very nice ones that contains the (by far) large majority of it. In this case, he makes this choice, but signals this fact, using a grey-tick line for the plot rectangle, as shown aside here.

In this case, however, if the user clicks on the "Change Scale" button, he will have the opportunity of seeing the plot with the full ranges on the axes (figure below – left). However, in the case the purpose of showing "round" numbers on the axes is given up. (figure below right).

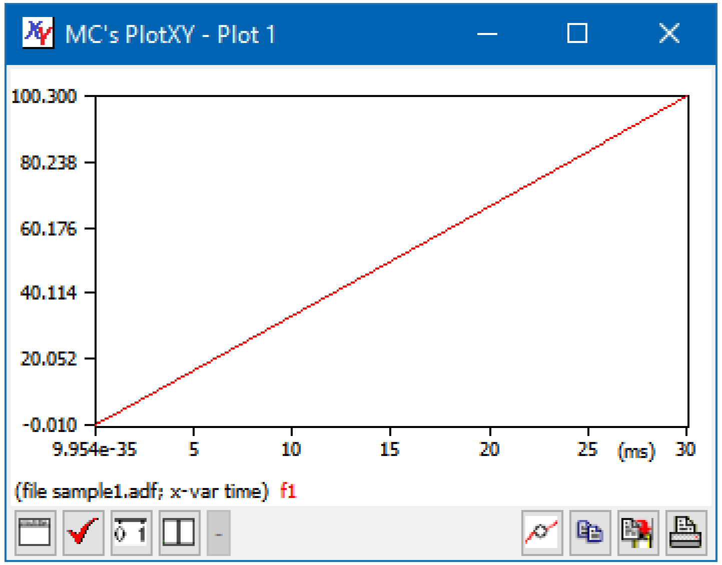


Y-axis
Max 100.3
Min -0.01

right-Y-axis
Max
Min

X-axis
Min 9.95412e-38
Max 0.03

☑ Exact match
● Full input range
○ Manual values



Note that not only the program chooses round numbers on the extrema, but also selects the number of tic-marks in such a way that the numbers on them are as round as possible and the number of them is reasonable, depending on the window's size. The user is prompted to experiment with this feature resizing the same plot and seeing how the intermediate tic-marks change in number. A tiny plot can have numerical labels just at the two extreme values.

Automatic Units of measure and prefixes

You may have noticed that in come plots units of measure (such as (V) or (kV)) have appeared.

This by default is automatically done by the program based on the following simple rules:

The "prefix", substitutes the powers of ten, according to the standard notation, shown for completeness in the following table:

prefix	*meaning*	*prefix*	*meaning*
p	$x10^{-12}$	k	$x10^{3}$
n	$x10^{-9}$	M	$x10^{6}$
u	$x10^{-6}$	G	$x10^{9}$
m	$x10^{-3}$	T	$x10^{12}$

The units of measure are chosen as a function of the first character of the variable name, according to the following table:

name begins with	*assumed unit*	*symbol*	*name begins with*	*assumed unit*	*symbol*
'v'	volt	V			
'c'	ampere	A	'i'	ampere	A
'p'	power	W	e	energy	J
't'	time	s	'f'	frequency	Hz

Note that the correspondences on the last row are used for horizontal-axis variables only

You can disable this feature unchecking the "Automatically assign units to known variables" check box in the Program Options|Plot tab sheet (cf. section "Customizing the program behaviour". Note that even if you leave "Automatically assign units to known variables" active, units can always be overridden by manual setting units through the "Units" windows, which is accessed first clicking on the "Change Scale" button in the Plot window (cf. sect. "Modifying the plot scales and units of measure").

More on loading files

In section Loading a file and doing first things it was said that loading files can be done either using the Load… button, or dragging files on the Data Selection window. Here I supply some more details.

With both techniques, you can load individual files or groups of files. The program will adapt to different situations and behaves as expected. At the end of load operation more than one file is loaded, the program is in multi-file mode (it goes into this mode if it was in single-file mode).

However, when individual files are loaded starting from single-file mode, the program behaviour depends on whether the action is initiated from single-file or multi-file mode:

- when starting from multi-file mode the new loaded or dragged file is added to the already loaded ones
- when starting from single-file mode, the new loaded or dragged tile replace the currently displayed one.

It is recommended to try these differences: once you get accustomed, you can take advantage of them.

Command-line options

We've seen in section "Automatic loading of files" that file names can be passed as command-line parameters. Before any file name, special command-line options can be specified. They are rather exoteric flags, and useful only for advanced users. They are listed before:

1	/set	"show elapsed time" It requests the time needed to generate a plot to be displayed in the plot window header (the uppermost row, containing the string "Plot 1", "Plot 2", etc.)
2	/dtQtF (deprecated)	requests plotting using to represent pixels floating-Qt point numbers
3	/dtQtI (deprecated)	requests plotting using to represent pixels floating-Qt point numbers
4	/dtQtP (deprecated)	requests plotting using to represent pixels floating-Qt Polynomials

Options 2, 3, and 4 show only rarely minor advantages, and are deprecated.

In fact, PlotXY uses a very effective algorithm to eliminate visually redundant points, that reduces plotting times, and output (pdf, svg) sizes, without any practical effect. Minor differences can be seen when deeply zooming the created SVGs; they are not important and therefore not discussed here. The user, however, can play with these options to see them. Basically his result will be that these options do not give advantages, since the default plotting and outputting algorithm is by fare the best.

Note that options 2, 3, 4 contrast to each other, If more than one of them is specified, the latter one (i.e. the rightmost one) is used.

Examples

- "PlotXY.exe rad2.mat sample1.adf" (without quotes) automatically requests to load rad2.mat and sample1.adf files
- "PlotXY.exe \set" does not load any file, but when files are loaded and plots made, the elapsed time during plotting activities is shown

- “PlotXY.exe \set \dtQtP” does not load any file, but shows elapsed times and uses built-in Qt drawing algorithm, without dropping visually redundant points. The user can play with this to see that the standard PlotXY plotting routine, which drops redundant points, is faster even though no visual difference can be appreciated
- “PlotXY.exe \set \dtQtP rad2.mat sample1.adf” does the same as before, but automatically loads of two files, after processing the command-line options.

What else?

In this tutorial you have learnt all the major things you can do with PlotXY. You can learn the rest just by trying.

However, there are specific details on the structure of adf file or the rules that the program use to convert variable names into the supplied formats: for instance not all the names that are valid as ADF file or pl4 file names can be used as such in matlab files.

This more advanced documentation is provided document: “Input formats and naming conventions.pdf”