ElecSus GUI – User Guide

Contents:

Last updated: 2015/09/04

1. Introduction

ElecSus is a program to calculate the *Elec*tric *Sus*ceptibility of an atomic vapour, which determines many of the optical properties of the vapour. ElecSus currently supports the alkali-metal atoms, in the vicinity of their D1 and D2 resonance lines ($nS_{1/2}$ to $nP_{1/2}$ and $nP_{3/2}$ transitions). The optical properties are calculated as a function of laser detuning – the laser frequency relative to a line-centre frequency.

For more details on the physics, and the back-end of ElecSus, see our paper in Computer Physics Communications. If the use of this program contributes to any research output, please cite this paper:

M. A. Zentile *et. al.*, Comp. Phys. Commun. **189**, 162 (2015).

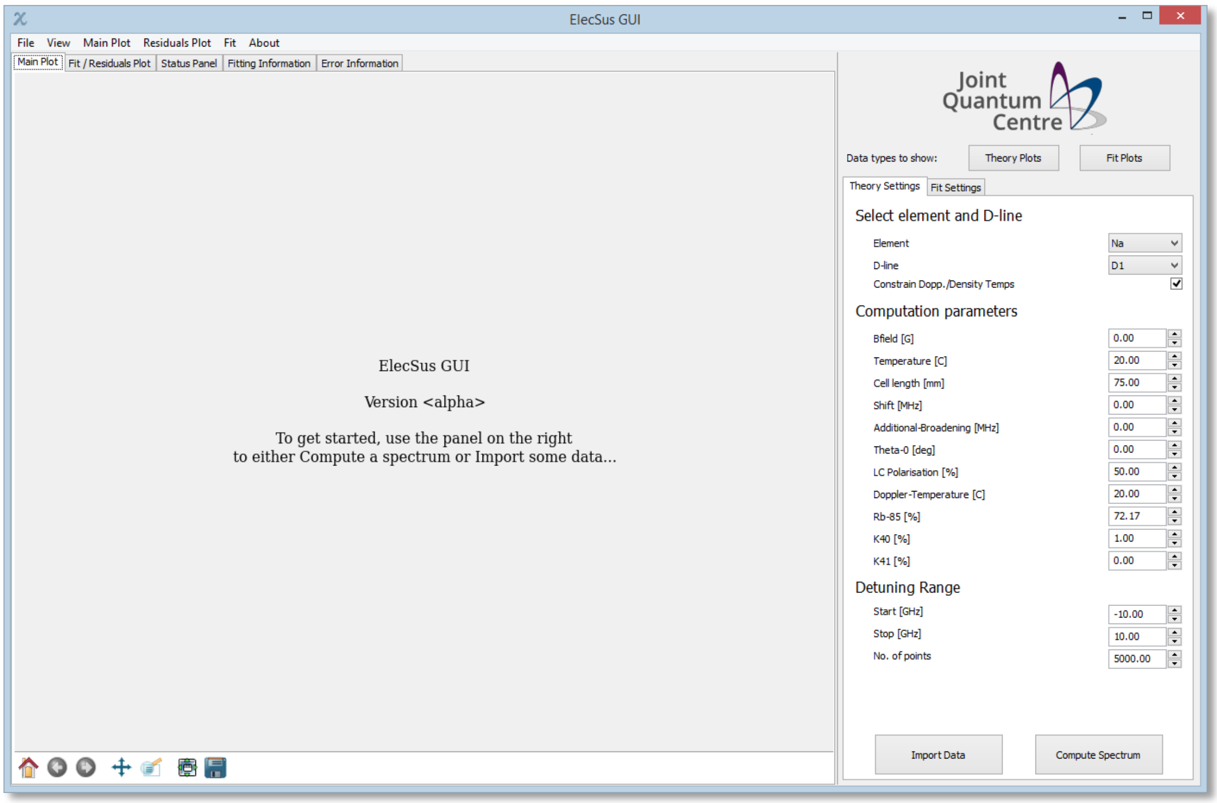
This user guide concerns the graphical user interface (GUI) to ElecSus. The GUI is designed to:

- Speed up the process of calculating and displaying spectra
- Allow pseudo-real-time parameter adjustment to look at how altering a particular parameter, or set of parameters, affects the spectrum
- Expedite the process of fitting experimental data, allowing quantitative comparison between experiment and theory.

This guide will outline each element of the GUI and it's operation.

2. Elements of the front panel

The main panel has several elements which we will go through separately.



ElecSus GUI

Version <alpha>

To get started, use the panel on the right
to either Compute a spectrum or Import some data...

Select element and D-line

Element	Na
D-line	D1
Constrain Dopp./Density Temps	✓

Computation parameters

Bfield [G]	0.00
Temperature [C]	20.00
Cell length [mm]	75.00
Shift [MHz]	0.00
Additional-Broadening [MHz]	0.00
Theta-0 [deg]	0.00
LC Polarisation [%]	50.00
Doppler-Temperature [C]	20.00
Rb-85 [%]	72.17
K40 [%]	1.00
K41 [%]	0.00

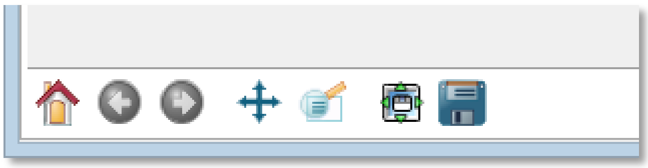
Detuning Range

Start [GHz]	-10.00
Stop [GHz]	10.00
No. of points	5000.00

Along the top of the window are the menus (in Ubuntu this bar appears on the top of the screen). The menu bar items will be discussed in later sections.

a. Plot panel tabs

Just under the menu bar is a list of tabs. The first two of these will display plot windows. The main plot shows theory curves and experimental data, if it has been loaded. The fit/residual plot shows the last loaded experimental plot and the current theory curve, and the difference between the two (the residuals). This plot will be updated when a fit is run. Both of the plots are interactive, allowing the user to pan, zoom, save data and adjust the subplots (figure panels) and are controlled via the matplotlib toolbar, shown below:

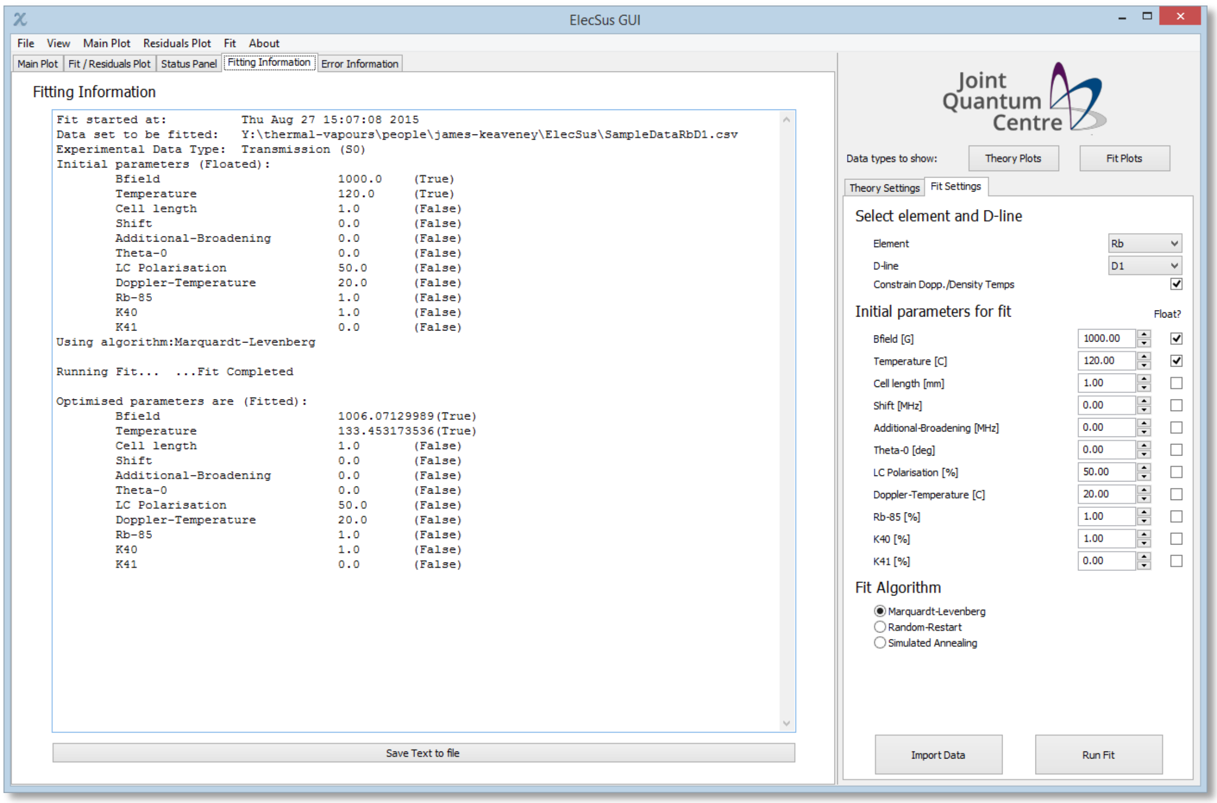


If you're not familiar with how this toolbar works, see the matplotlib documentation for details.

http://matplotlib.org/users/navigation_toolbar.html

Note that the navigation keyboard shortcuts are not functional in the GUI.

b. Status panel tabs



Fitting Information

```
Fit started at:          Thu Aug 27 15:07:08 2015
Data set to be fitted:   Y:\thermal-vapours\people\james-keaveney\ElecSus\SampleDataRbD1.csv
Experimental Data Type:  Transmission (S0)
Initial parameters (Floated):
        Bfield                        1000.0     (True)
        Temperature                   120.0      (True)
        Cell length                   1.0        (False)
        Shift                         0.0        (False)
        Additional-Broadening         0.0        (False)
        Theta-0                       0.0        (False)
        LC Polarisation               50.0       (False)
        Doppler-Temperature           20.0       (False)
        Rb-85                         1.0        (False)
        K40                           1.0        (False)
        K41                           0.0        (False)
Using algorithm:Marquardt-Levenberg

Running Fit...   ...Fit Completed

Optimised parameters are (Fitted):
        Bfield                        1006.07129989(True)
        Temperature                   133.453173536(True)
        Cell length                   1.0        (False)
        Shift                         0.0        (False)
        Additional-Broadening         0.0        (False)
        Theta-0                       0.0        (False)
        LC Polarisation               50.0       (False)
        Doppler-Temperature           20.0       (False)
        Rb-85                         1.0        (False)
        K40                           1.0        (False)
        K41                           0.0        (False)
```

Data types to show:

Select element and D-line

Element	Rb
D-line	D1
Constrain Dopp./Density Temps	✓

Initial parameters for fit

		Float?
Bfield [G]	1000.00	✓
Temperature [C]	120.00	✓
Cell length [mm]	1.00	☐
Shift [MHz]	0.00	☐
Additional-Broadening [MHz]	0.00	☐
Theta-0 [deg]	0.00	☐
LC Polarisation [%]	50.00	☐
Doppler-Temperature [C]	20.00	☐
Rb-85 [%]	1.00	☐
K40 [%]	1.00	☐
K41 [%]	0.00	☐

Fit Algorithm

[✓ Marquardt-Levenberg] Random-Restart Simulated Annealing

The three tabs next to the plot tabs are status information panels. They are all read-only.

The first, simply labelled ‘Status Panel’, simply redirects python’s normal stdout to a text box (stdout is where any ‘print’ statement is shown, usually in the command-line window where python is run from).

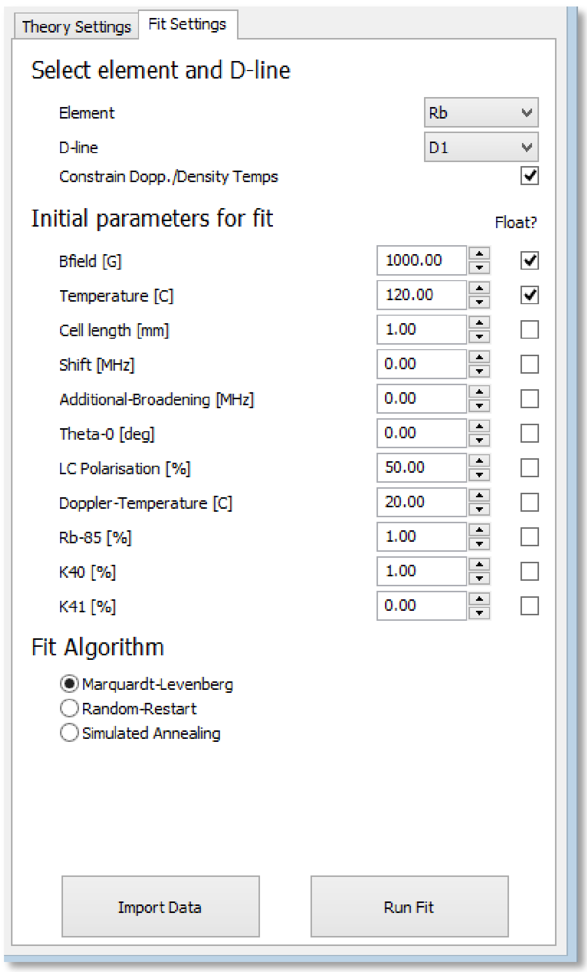
The second, labelled ‘Fitting Information’, shows information on the fits that have been run since the program started. An example is shown above. This panel details starting parameters, final (fitted) parameters and other information that may be useful, such as fit quality indicators (RMS error, chi-squared values), fit start time, the file and data type to be fitted etc. Note this is blank until a fit is run.

The final tab is labelled ‘Error Information’ and in an ideal world will be blank! However, this tab redirects pythons ‘stderr’ output, so any errors can be traced and debugged.

All three panels have a ‘Save text to file’ button at the bottom, which allows all the data contained in the text box to be saved as a .txt file. In addition, they support text highlighting, copy and paste etc.

c. *Theory/Fit settings tabs*

These tabs contain the parameters for calculating spectra / running fits.



Select element and D-line

Element	Rb
D-line	D1
Constrain Dopp./Density Temps	✓

Initial parameters for fit

		Float?
Bfield [G]	1000.00	✓
Temperature [C]	120.00	✓
Cell length [mm]	1.00	☐
Shift [MHz]	0.00	☐
Additional-Broadening [MHz]	0.00	☐
Theta-0 [deg]	0.00	☐
LC Polarisation [%]	50.00	☐
Doppler-Temperature [C]	20.00	☐
Rb-85 [%]	1.00	☐
K40 [%]	1.00	☐
K41 [%]	0.00	☐

Fit Algorithm

[✓ Marquardt-Levenberg] Random-Restart Simulated Annealing

Hovering over each parameter for a short time brings up a ToolTip describing in more detail what each parameter is. For more details, see the ElecSus paper.

For fitting, the ‘float?’ tick-boxes select the parameters that will be allowed to vary in the fits.

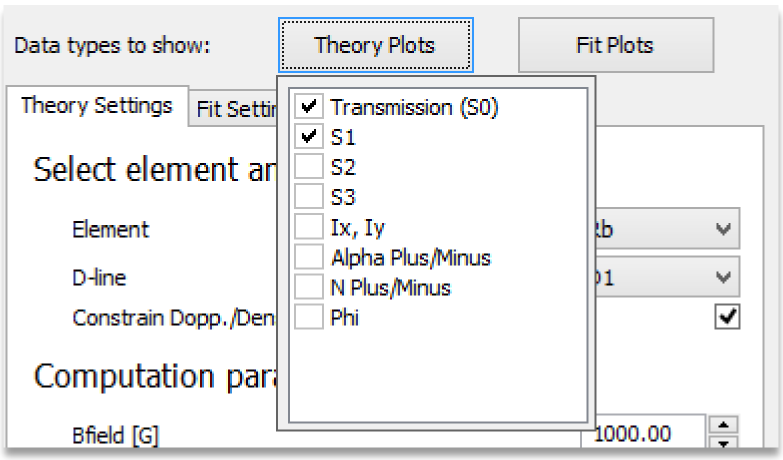
Note: The parameters can be copied from the Theory Settings to Fit Settings and vice-versa via the buttons on the Edit Menu.

3. Calculating a spectrum

a. Adding / removing plot types

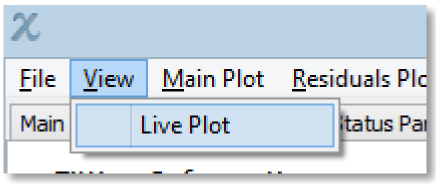
To actually calculate the spectrum, input the computation parameters that are required and click the ‘Compute Spectrum’ button at the bottom right of the window.

By default, only the transmission signal will be displayed (S0 in Stokes Parameter shorthand). However, more data can be displayed by clicking on the buttons next to ‘Data types to show:’, which will bring a popup window with check boxes for selecting the various data types to display. An example is shown below:



b. Live Plotting

Sometimes it is instructive to vary parameters to ‘get a feel’ for how spectral properties change. Instead of clicking ‘Compute Spectrum’ many times, in the ‘View’ Menu, click ‘Live Plot’, and the plot will update automatically as parameters are varied.



c. Quickly changing parameters

Each parameter in the list can be changed by clicking into the text control box and typing a new value, or alternatively using the either the spin control buttons (up/down arrows next to the control) or mouse wheel to increment the parameter values by a set amount. There are some shortcuts for these controls that make it easy to quickly spin through values:

(source: http://wxpython.org/Phoenix/docs/html/lib.agw.floatspin.html)

- The initial value is remembered - press Esc to return to it;
- Shift + arrow = 2 * increment (or Shift + mouse wheel);
- Ctrl + arrow = 10 * increment (or Ctrl + mouse wheel);
- Alt + arrow = 100 * increment (or Alt + mouse wheel);
- Combinations of Shift, Ctrl, Alt increment the FloatSpin value by the product of the factors;
- PgUp & PgDn = 10 * increment * the product of the Shift, Ctrl, Alt factors;
- Space sets the control’s value to it’s last valid state.

4. Fitting experimental data

a. Importing data

Clicking on either the 'Import Data' button at the bottom-right of the main panel, or File->Open (Ctrl+O shortcut) brings up a dialog box asking which data type is to be imported.

The file should be in csv format, 2 columns with no header line. The detuning axis should be in GHz. Included with the ElecSus files are three example data files, copy the format of these if anything is not clear.

Currently only one experimental trace per data type is allowed – if a second file of the same data type is imported it will simply overwrite the first.

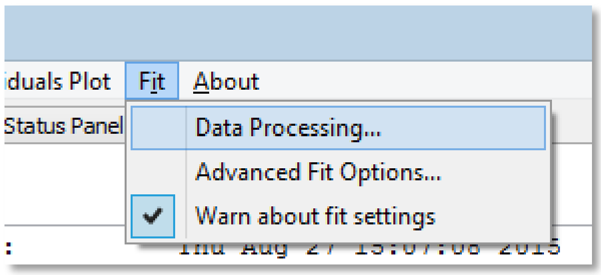
b. Data processing

Quite often the data file will have a large number of elements, typically in the range 10^4 to 10^5. Whilst there is in principle nothing preventing the program from using all these points, fitting will be very slow since the spectrum has to be evaluated at each of the points. A considerable speed-up can be gained by reducing the amount of data points by locally averaging the data (known as binning the data).

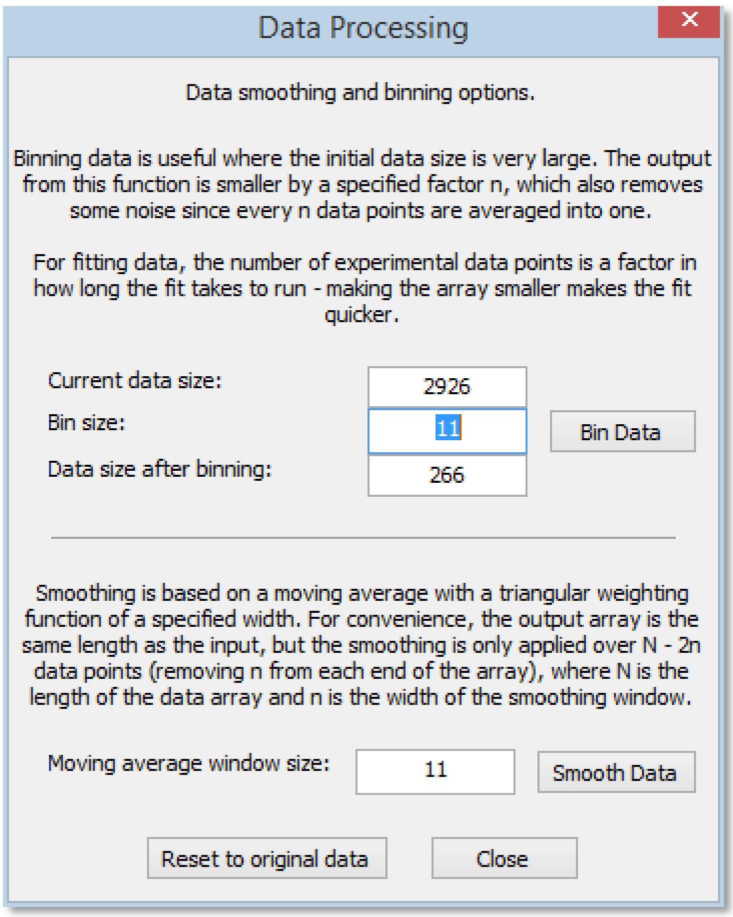
We recommend for reasonable speed using < 5000 data points, especially if Random-Restart or Simulated Annealing fit algorithms are used. By default the program warns the user if the fit is expected to take a long time – these warnings can be turned off via the check-box button on the menu, shown below.

Alternately if the data is particularly noisy but not a long array, the data may be smoothed using a moving-average method.

In the ElecSus GUI this is possible via the 'Data Processing' menu item, located in the 'Fit Options' menu, shown below:



This will bring up a dialog box allowing both binning and smoothing to be done.


Data Processing

Data smoothing and binning options.

Binning data is useful where the initial data size is very large. The output from this function is smaller by a specified factor n, which also removes some noise since every n data points are averaged into one.

For fitting data, the number of experimental data points is a factor in how long the fit takes to run - making the array smaller makes the fit quicker.

Current data size: 2926

Bin size: 11

Data size after binning: 266

Smoothing is based on a moving average with a triangular weighting function of a specified width. For convenience, the output array is the same length as the input, but the smoothing is only applied over N - 2n data points (removing n from each end of the array), where N is the length of the data array and n is the width of the smoothing window.

Moving average window size: 11

The dialog shows the current data size and the size after binning. Note nothing happens unless either the ‘Bin Data’ or ‘Smooth Data’ buttons are pressed. The dialog can be dragged around and the result of binning/smoothing is shown immediately on the main plot.

The ‘Reset to original data’ button is fairly self-explanatory – it simply undoes any binning or smoothing while the dialog box is open. However, once the dialog box is closed, the new data arrays are stored and bringing up the Data Processing dialog box again will start from the now-processed arrays.

c. Choice of fitting routine

ElecSus provides 3 fitting routines, which vary in speed and complexity. Check the paper for references to the exact algorithms used and more details, but here we quickly summarise their relative merits.

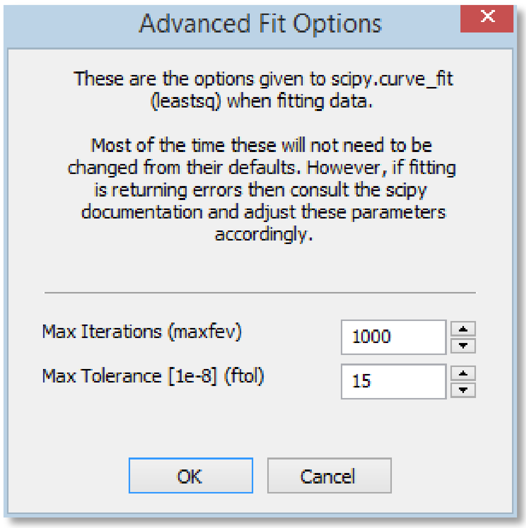
- Marquardt-Levenberg (ML) is the simplest fitting algorithm, and involves a simple downhill method to find a minimum in parameter space. Its simplicity makes it relatively fast, and a fit on a data set with around 5000 points takes around 15 seconds, depending on computer speed and choice of starting parameters. It runs on a single processing core. However, for large numbers of free parameters, it is known to find local minima and hence often returns non-optimal parameters. When the number of free parameters is more than 3, we recommend using either Random-Restart or Simulated Annealing instead.

- Random-Restart (RR) is the next most complex algorithm. It uses all the computer's processing cores (by default) by utilising python's 'multiprocessing' module. Since the ML method is known to find local minima in parameter space, the RR method takes a spread of starting parameters, centred around the initial parameters. The ML method is called separately on each of these starting parameters in order to find a global minimum. The speed of this method depends on the number of processing cores and number of free parameters in the fit. Typically this method takes 1-5 minutes.

- Simulated Annealing is the most complex algorithm included with ElecSus, and is therefore the most computationally intensive. However, it is most likely to return the global minimum, assuming a sensible starting condition. This fit is also capable of using all processor cores, but the method is slow, and should therefore only be used for the most complex fitting problems, where RR fails to find a good solution, or to check the results of an RR fit. Typical time to compute is around 10-15 minutes.

d. Advanced fit options

If the fit is failing to complete, it may be improved by changing the keyword argument options that are passed to the scipy.optimize.curve_fit (leastsq) method (the method used by ML fitting). All three fit algorithms use the ML method at some stage. These options can be found in the 'Fit Options' menu.


Advanced Fit Options

These are the options given to scipy.curve_fit (leastsq) when fitting data.

Most of the time these will not need to be changed from their defaults. However, if fitting is returning errors then consult the scipy documentation and adjust these parameters accordingly.

Max Iterations (maxfev) 1000

Max Tolerance [1e-8] (ftol) 15

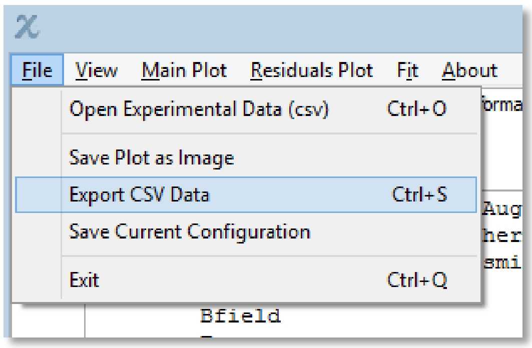
(to update as more options are added…)

These options are intended for experts. Details on these parameters can be found on the scipy documentation webpages:

http://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.leastsq.html

5. Saving results

Results can be saved in many forms – all from the File menu.



(to update when load button is present!)

Save plot as image allows the user to save the current plots as image files. The file formats allowed are any format that matplotlib supports – currently that is png, ps, eps, pdf and svg.

Export CSV Data exports the data from the main plot – all output types are saved, whether or not they are displayed. The csv file has a header line with the output types, then the data is arranged in columns.

Save Current Configuration (and the reverse – Load Previous Configuration) saves/loads the current program settings from a data file, so that the user can quickly pick up from where they left off. This may be useful when data is fitted so the user can quickly reproduce the fit/plot without re-calculating or entering in all the parameters by hand. (Note: this is currently not implemented, but is intended for a future release).

6. Reference

a. How to cite this work

This work should be cited using the ElecSus *Computer Physics Communications* publication. If this work has been valuable in your research, please add a citation to:

M. A. Zentile *et. al.*, Comp. Phys. Commun. **189**, 162 (2015)

b. Requirements for running the GUI

The GUI is written in wxPython, a python wrapper for the wxWidgets C++ library.

On windows, we recommend using Enthought Canopy – all packages required are included (as of September 2015) with the free version.

On Ubuntu Linux, Enthought does not support wxPython and therefore we recommend installing the required packages (numpy, scipy, matplotlib, wx) via apt-get install <package>, to the standard python installation (/usr/bin/python). The required packages on Ubuntu are:

```
python-numpy, python-scipy, python-matplotlib,
python-wxgtk2.8, python-wxtools, wx2.8-i18n
```

Other operating systems have not been tested.

To install, rather than running locally, in a terminal/command prompt window, navigate to the elecsus top directory with the setup.py file and type:

```
python setup.py install
```

After installation, elecsus can be used (and the GUI started) by simply importing it, either in scripts or from the python interpreter:

```
python

>>>     import elecsus
>>>     elecsus.elecsus_gui.start()
```

or

```
>>>     from elecsus import elecsus_gui
>>>     elecsus_gui.start()
```

c. Good starting parameters for the sample data

There are three sample files provided with ElecSus, which are located in the elecsus/sample_data folder. Depending on the install method, these files may be in your python package installation directory (<>/site-packages/elecsus), or just from the folder where ElecSus has been unzipped to.

To familiarise yourself with the program, we recommend using the sample data and fitting theoretical spectra. To aid with this, in the table below are some suggested starting parameters for the sample data. The checks indicate recommended parameters to float.

File name:		**SampleDataRbD1.csv**		**SampleData_S1_RbD2.csv**		**SampleData_Ix_RbD2.csv**
Data Type:		S0		S1		Ix
Element:		Rb		Rb		Rb
Dline:		D1		D2		D2
Constrain:		True		True		True
B field:	✓	1000 G	✓	5500 G	✓	250 G
Temperature:	✓	130 C	✓	65 C	✓	92 C
Cell length:		1 mm		1 mm		5 mm
Shift:		0 MHz		0 MHz		0 MHz
Add. Broad:	✓	5 MHz	✓	5 MHz		0 MHz
Θ_0:		0°	✓	45°		0°
σ^-:		50 %		50 %		50 %
Dopp. Temp:		N/A		N/A		N/A
^{85}Rb:		1 %		1 %		72.17 %
^{40}K:		N/A		N/A		N/A
^{41}K:		N/A		N/A		N/A

james.keaveney@durham.ac.uk

Durham University, UK