ReadMe FitSCLC

Version 1.05, October 04, 2019
Martijn Kemerink, Linköping University, Sweden
mail: martijn.kemerink@liu.se

Contents

0. Purpose

FitSCLC is written to facilitate the extraction of mobilities and the underlying disorder and hopping parameters from temperature-dependent space charge limited conductivity (SCLC) data. The goal was to make a program that is about as easy as 'slope 2 fitting' to $j = \frac{9}{8}\varepsilon_0\varepsilon_r\mu V^2/L^3$ but that, in contrast, does give reliable and meaningful results. The underlying concepts are extensively described in Felekidis et al., Organic Electronics, Vol. **61**, pp. 318-328 (2018), but those that just need a mobility value can typically use the default settings and the 'Murgatroyd/Gill' jV model and 'GDM' for mu model and press 'Fit!'. After pressing 'Save', zero-field zero-density mobilities are stored on disk, see point 9 in section '2. Running the program' below. Reading this manual is warmly recommended in all cases, though.

1. Required documents

To run the application, the following documents are needed:

Stand-alone application:

- GUI_FitSCLC.exe
- MatLab runtime (not included here, can be downloaded for free from the MatLab website, it's about 3 GB large but does not require a license. See readme – executable.txt for details, especially regarding the runtime version that is needed).

In MatLab:

- FitSCLC.fig (figure file containing the GUI)
- FitSCLC.m (main program handling all actions in the GUI)
- cGDM.m
- eGDM.m
- ET-GDM.m
- FitSCLCerror.m

- FreeFit.m
- Gill.m
- MakeLookup.m
- mu0_T.m
- ODDD.m
- SCLC.m
- SetActive.m.

2. Running the program

The exe-file runs like a normal executable. To run the program in MatLab, all required files should sit in a single directory that is made the active directory of MatLab. Either opening the FitSCLC.fig file in GUIDE or opening FitSCLC.m in the editor and pressing the green arrow should get you going. In both cases this GUI will appear (Figure 1):

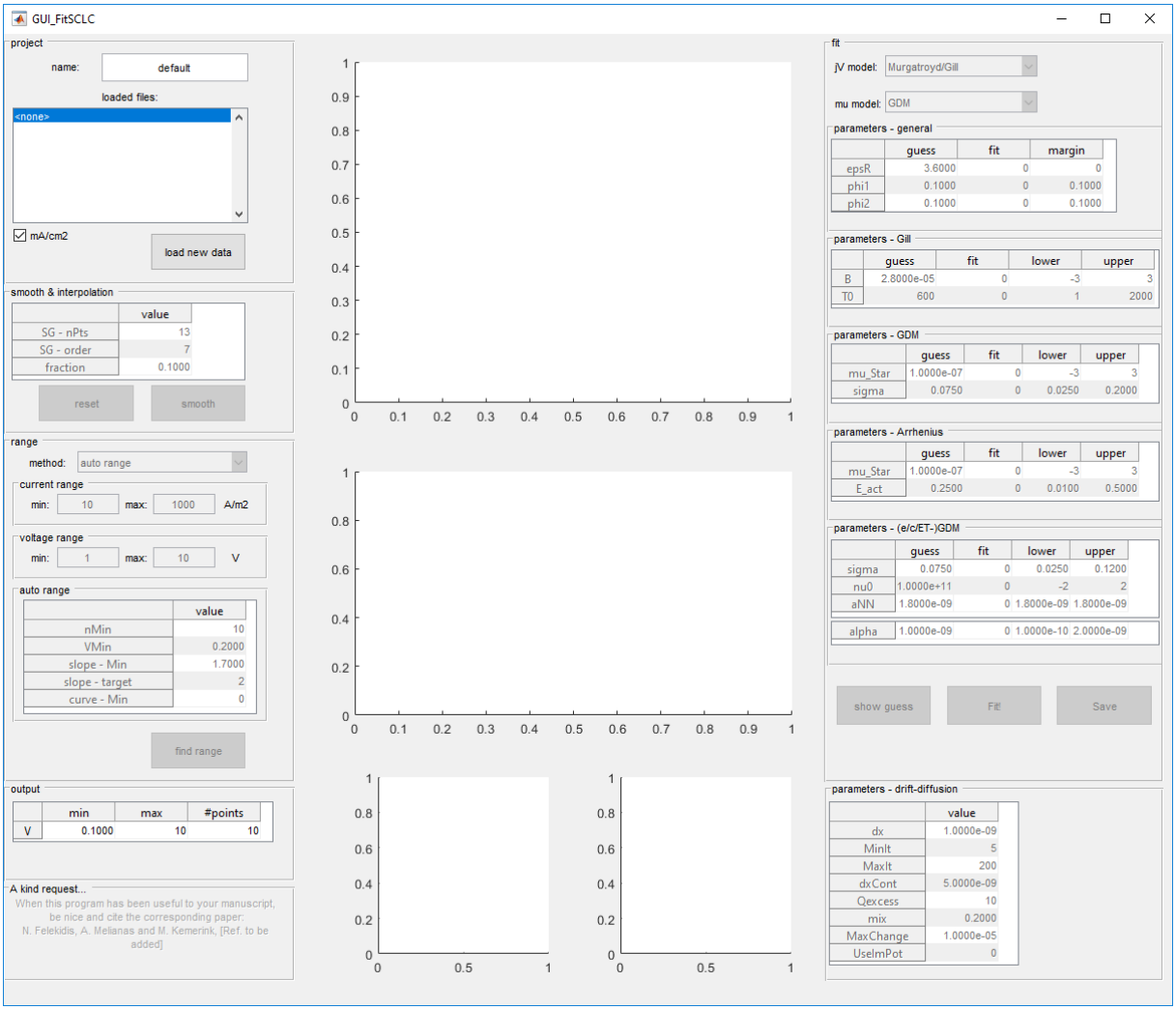

Figure 1. GUI as shown on opening the program

To fit your JV data, you'll have to go through a number of steps that are organized top-down, starting on the upper left of the GUI:

1. Define a project name. The name will also be used to store all output in the same folder as from where the data are loaded, see also steps 2, 9.

2. Load some data. The program expects a CSV file with V (in Volts) in the first column and J (in A/m^2 or mA/cm^2, use the toggle in the GUI to indicate) in the second. **The filename should end in …xxxnm_yyyK.txt, with xxx and yyy both being three (!) digits defining the active layer thickness in nm and T in K.**
   The loaded data will be displayed in the upper two graph windows, the upper being the JV curves, whereas the middle panel shows the corresponding power law slope $d(\log(J))/d(\log(V))$. Interpreting these slope curves is a bit tricky but worth the effort, see Organic Electronics, Vol. **61**, pp. 318-328 (2018).
   IMPORTANT: each JV can contain up to 10 000 data points, but more points lead to slower fitting.
3. Optional: smooth the data. This only affects/helps the auto-range procedure. The results are shown as thin dashed lines. The smoothing procedure first performs a Savitzky-Golay moving average filter to smoothen the experimental data before taking the numerical derivative to calculate the power law slope that gets smoothed with a robust loess filter.
   The parameters 'SG – nPts' and 'SG – order' refer to the span (must be odd) and the order (must be less than the span) of the Savitzky-Golay filter, 'fraction' is the fraction (0-1) of the total number of data points used in the weighted moving average loess filter. See Matlab online help on data smoothing for further details.
4. Select the data range to be fitted using one of the methods in the pull-down menu and the corresponding parameters.
   Auto-range parameters: 'nMin' - minimum number of points required for meaningful SCLC fit; 'VMin' – minimum voltage to be used; 'slope – Min' and 'slope – target' - minimum and preferred lowest slope of JV; 'curve – Min' - minimum acceptable curvature of power law slope ($d(slope)/dV$). Auto range will not work well with noisy slope data. In case the offered methods do not yield the desired ranging for your data, you will need to cut them outside the GUI.
5. Set the voltage range for which the fitted JV curves are calculated. When 'speed mode' is used (only active/visible for Drift-Diffusion), #points also sets the spacing of the points at which trial JV's are calculated, and therefore has influence on both the speed and the accuracy of the fit (speed mode enables much faster fitting at marginally lower accuracy).
6. Select the model to fit the data to from the upper pull-down menu in the 'fit'-box and select the mobility model in the lower pull-down menu. Then set the trial parameters and the allowed range of variation. To help with that, the 'show guess' button plots the JV's corresponding to the chosen guess parameters.
   IMPORTANT: for some parameters that usually take either very small or very large values (B, mu_Star, nu0), the upper and lower limits are to be read as exponents. E.g. -2 and +2 imply the upper and lower limit are 10^{-2} and 10^{2} times the guess value.
   IMPORTANT: to avoid landing in some local minimum, it is a good idea to run the fit several times, each time with different guess parameters.

7. Fit and inspect the result. When pressing fit, a new window will open, showing the progress of the least squares routine. More concretely, the plot will show the least-squares error:

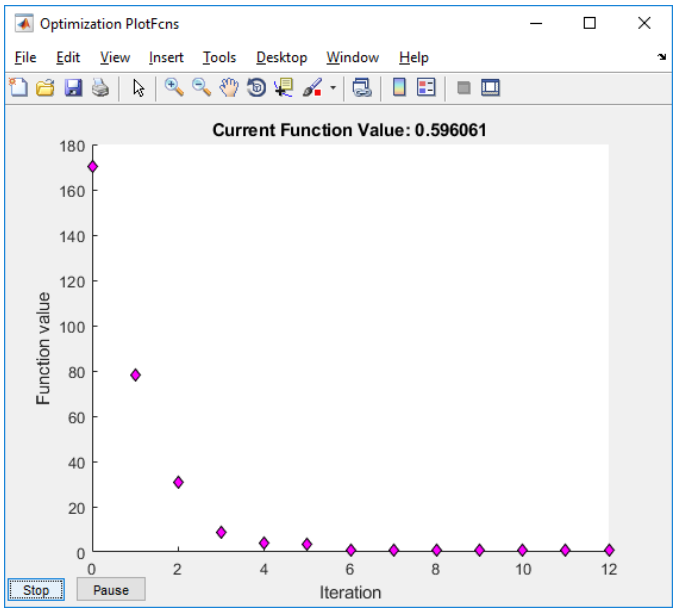

Figure 2. Example of a least squares sum reaching convergence.

8. At any time, the procedure can be stopped with the 'stop' button and the program will use the best (not converged!) fit till then. Otherwise, another small window will appear when the fitting is complete. The fitted JV's and slopes will be plotted, and the corresponding parameters will be added to the relevant table(s). For Murgatroyd/Gill fits, also an analysis of the μ_0 and γ vs. *T* will be made and plotted in the lower two graph windows.
   IMPORTANT: check that the fitted parameter values are not constrained by the upper and lower limits, unless this is intended.
9. Save the data. The program saves the following in the same directory as from which the JV curves come:
   - '[project name] par.txt': text file with guess and fit values of the parameters used in the last saved fit
   - '[project name] mu_gamma.txt': text file with the fitted zero-field, zero-density mobility μ_0 and (when applicable) the field-enhancement factor γ vs. *T*.
   - '[project name] jV fit.txt': text file with fitted JV curves, first line contains *T*.
   - '[project name] slope fit.txt': text file with corresponding slope curves, first line contains *T*.
   - '[project name] all.mat': MatLab file containing 3 structures with experimental data, the used input parameters (from the GUI) and the fitting results.
10. It seems that sometimes the program freaks out. Usually it indicates bad input, but if nothing helps, restarting will.
    The drift-diffusion model is not as stable (and not as fast) as the analytical models, especially when the injection barriers are low and/or when the image potential is activated and/or when devices are thick. It might show up as a failure to make progress in the fitting routine or failure to plot (parts of) the JV curves when pressing 'show guess'. To solve the problem, either deactivating the image potential, or (a) reducing the weight factor of old and new solutions in the self-consistent loop or (b) changing the allowed excess charge or (c) changing the mesh spacing might help. See below for full list of parameters.

After this procedure the GUI might look something like shown in Figure 3:

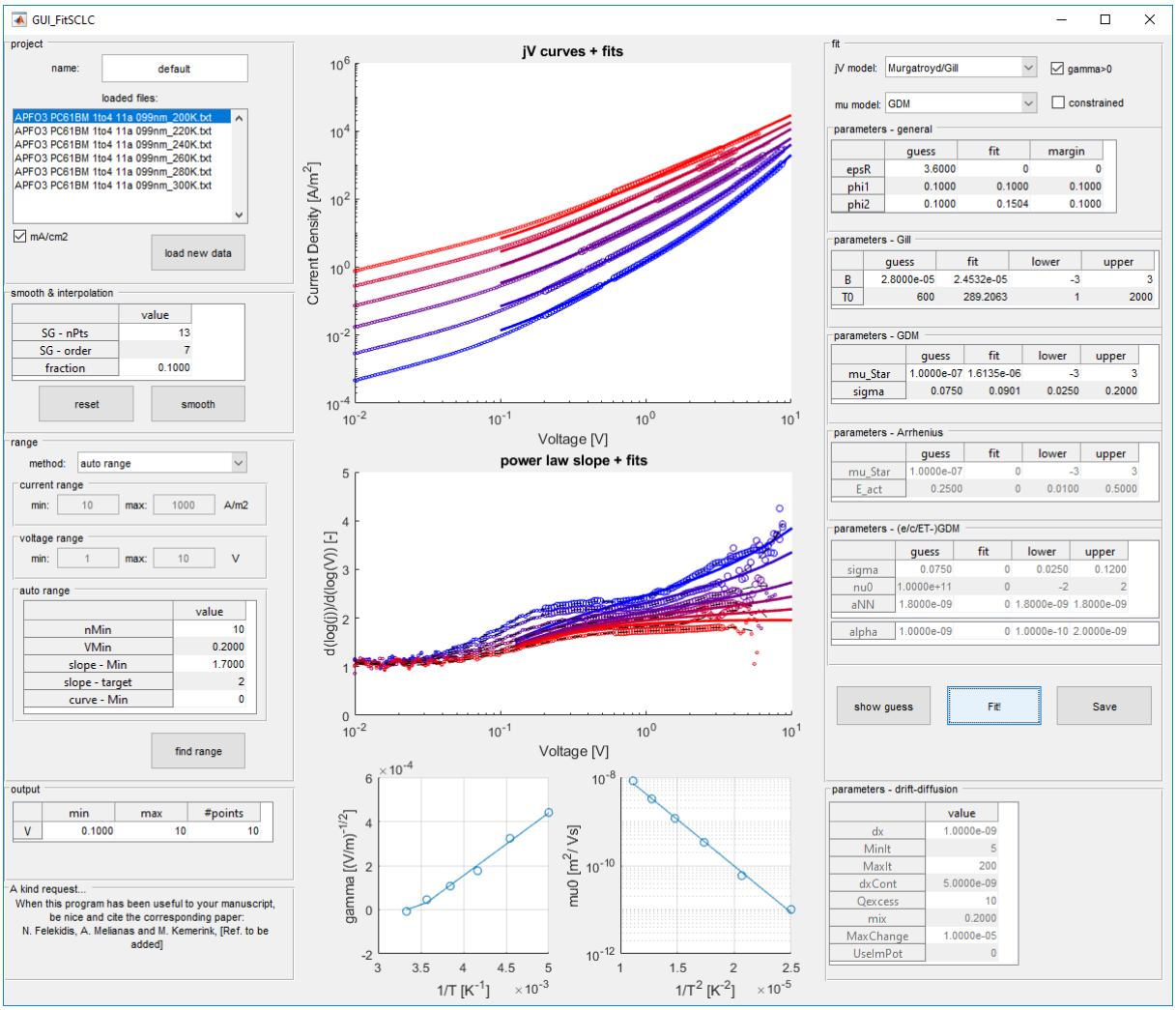

Figure 2. GUI after a completed fitting procedure.

3. Some general remarks

- The program uses SI units throughout the GUI, with energies being measured in eV. Parameter units are not shown for lack of space.
- You cannot directly specify a guess value for the mobility at one or more specific temperatures since the program is primarily meant for T-dependent fitting, where there is supposed to be some physics governing this T-dependence. This physics is captured by the chosen model and the corresponding parameter (guess) values; together these predict the mobility at any T. However, fitted mobilities are saved for all T.
- Hovering the mouse over buttons and input fields will bring up a tooltip.
- The Murgatroyd/Gill model uses a built-in voltage V_{bi}, the drift-diffusion model uses two input barriers (phi1, phi2). To keep the GUI consistent, only φ_1 and φ_2 are shown. They are related to V_{bi} via $V_{bi} = \varphi_1 - \varphi_2$. For the Murgatroyd/Gill model only the difference matters, for drift diffusion the actual values matter!
- To make a new executable in MatLab (requires the compiler ‘mcc’), type: “mcc -e FitSCLC” in the MatLab command window.

- Meaning of the parameters in the drift-diffusion menu:
  dx – mesh spacing
  MinIt – minimum number of iterations in self-consistent loop
  MaxIt – maximum number of iterations in self-consistent loop
  dxCont – thickness of contact area that is disregarded in calculation described at 'Qexcess'
  mix – weight factor of old and new solutions in the self-consistent loop
  Qexcess – is the maximum allowed ratio of the actual charge in the device and the charge expected on basis of the geometrical capacitance; higher space charge densities are truncated
  mix – weight factor of old and new solutions in the self-consistent loop
  MaxChange – convergence criterion, maximum allowed relative change in potential in the self-consistent loop
  UseImPot – toggles the use of the image potential

4. A kind request…

If this program was helpful to your paper, please be nice and cite the following work:
"*Automated open-source software for charge transport analysis in single-carrier organic semiconductor diodes*" by Nikolaos Felekidis, Armantas Melianas and Martijn Kemerink, Organic Electronics, Vol. **61**, pp. 318-328 (2018).

5. Things to improve & disclaimer

This program has been tested quite extensively, but of course it can always be improved. In case you have requests that you feel might benefit general users, do not hesitate to contact me. Keep in mind though that this is just a fitting program, not a general-purpose drift-diffusion solver or a plotting program. And making the program dual carrier to deal with solar cells and LEDs is not an option as the number of parameters will explode, making the thing completely underdetermined.

Although this program is written with great care, I do not take any responsibility for mistakes, errors, damages etc. that occur due to the direct or indirect use of this program.