Manual for program "2/3-Diode Fit" by Stephan Suckow

Contents

1 Introduction

Current version: 2.3.12, released: 18 August 2014.

1.1 Overview

This program allows to simulate or fit current-voltage (I-V) measurement data of solar cells (or more generally p-n junctions) using different 2- or 3-diode models. Different file formats and batch processing are supported as well as different modes: dark or illuminated I-V curve fitting, variable fixed ideality factors, Suns-V_{OC} and an extended 2-diode model taking into account the distributed nature of a part of the series resistance according to a model developed by Breitenstein and Rißland [1]. Alternatively the 3-diode model can take increased series resistance in the path of the 3rd diode into account and was first proposed by Hernando [2] and later popularized by McIntosh [3]. This functionality is complemented by the ability to plot and fit the local ideality factor for further cell analysis.

The program features a graphical user interface (GUI) which is - in my humble opinion - intuitive and easy to use. The current calculation algorithm is fast and quite stable. Nevertheless, feel free to send any feedback, improvement suggestions or errors to suckow@iht.rwth-aachen.de.

The algorithm to calculate $I(V)$ is described in detail in [4]. All functionality discussed therein can be found in the file *diodeI.m*.

To get started you may want to consult section 3.1. There, the two methods to install the program, using MATLAB or as stand-alone executable, are explained. Depending on which way you choose you should download different files and you should not need to download the entire 1 GB.

Updates with minor version changes, like going from version 2.3.10 to 2.3.12, are likely simple bugfix releases. In this case you can probably keep your customized *config.xls* instead of the default one provided with the download. Upon major version changes like going from 2.2.8 to 2.3.12 there are very likely changes to the *config.xls*. If you want to keep your previous settings you can edit the Excel file directly and copy them into the proper fields in the new file.

1.2 Copyright

This program is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 unported License. A human-readable form and the full license are available at
http://creativecommons.org/licenses/by-nc-sa/3.0/

In short: this means you are allowed to use, share and adapt the work for non-commercial purposes under a similar license, as long as you attribute its origin. Anything else requires permission from the author. If you publish work based on this program, acknowledge its author by citing:

Stephan Suckow, 2/3-Diode Fit (2014). http://nanohub.org/resources/14300.

1.3 Credits

Thanks go to E. Stegemann for the first version of the program and to O. Breitenstein, S. Rißland, N. Wilck, M. Thore and D. Nielinger for enduring beta-testing, as well as various other people for finding and reporting smaller bugs. Special thanks go to M. Raval for bringing up and testing the resistance limited recombination model. Further thanks go to J. Blass, A Gorodnichev, K. Razavidinani and J. Xie for the first implementation of the Breitenstein-Rißland model and to T. Pletzer for continued support and feedback.

2 Physical models

2.1 The regular 2-diode model

Describing the *I*-*V* characteristic of a solar cell by a single diode is a good first approximation. Upon closer inspection more circuit elements are needed, leading to the 2-diode equivalent circuit depicted in figure 1.

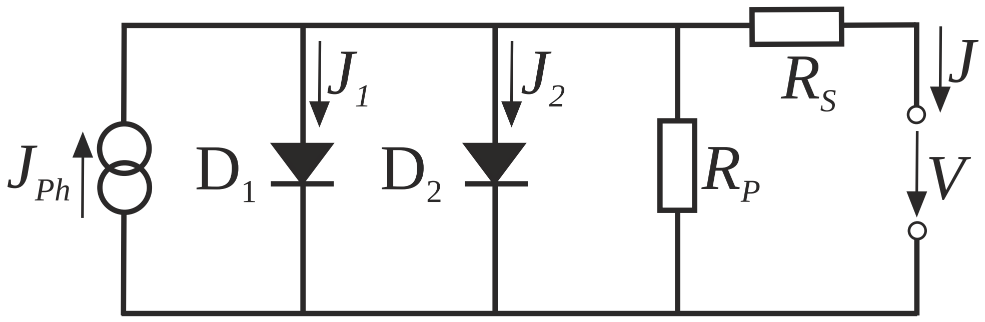

Figure 1: Equivalent circuit of a solar cell according to the 2-diode model.

J and V are the current and voltage at the contacts, R_S is the lumped series resistance of the entire device, R_P a parallel resistance (shunt), D_1 is the diode to describe the diffusion current J_1, D_2 is the diode representing the recombination current J_2 and J_{Ph} is the photo current. The terminology used throughput the program is:

- $R_S = R\ ser$
- $R_P = R\ par$
- $J_{Ph} = J\ photo$
- The pre-exponential coefficient of J_1 is J 01.
- The pre-exponential coefficient of J_2 is J 02.
- The ideality factor of diode D_1 is n 1.
- The ideality factor of diode D_2 is n 2.

The 2-diode model has been pioneered by Wolf and Rauschenbach in 1963 [5] and since then has become a valuable standard tool for solar cell characterization. It contributed significantly to a better understanding of the processes inside the cells and related efficiency loss mechanisms [6, 7, 8, 9]. Compared to the simpler 1-diode model it provides better accuracy, especially in the vicinity of the maximum power point of conventional Si solar cells (see e.g. [10] and references therein). Further information can be found in standard solar cell text books.

2.2 Distributed series resistance (Breitenstein-Rißland model)

An extended version of the standard two-diode model is available. It is based on the work of Breitenstein and Rißland [1] and takes the distributed nature of a part of the series resistance into account. By applying this model as proposed, a set of cell parameters can be derived which allows to describe the cell consistently under all illumination conditions (except for J 02 and n 2). It is available by checking *Breitenstein-Rißland model* in the *Configuration* menu.

Activating this mode switches *R ser* to *R hom* to describe the homogenous series resistance present at every point of the solar cell. An additional fit parameter *R dis* appears, which is the distributed series resistance, depending on the distance of the generated charge carriers to the bus bars. For a full understanding of the physical meaning of these parameters and their derivation please consult the original paper [1].

Breitenstein and Rißland propose to use the model in the following way:

- Fit the illuminated curve using a fixed *R dis* $= 0$ in the Breitenstein-Rißland model (equivalent to the standard model). Write down the value of V_{OC} and don't touch the fit result of *J photo*.
- Load a dark measurement of the same cell, use the right *Current in* mA/cm^2 edit field to limit the current to a maximum of 3 mA/cm^2 for bulk Si cells, so that the distributed series resistance can still be neglected, and fit.
- At the top of the plot V_{OC} is estimated for these fit parameters based on the value of *J photo* (which is still correct from the illuminated fit). Now repeat "adjust *n* 1 and fit" until the estimated V_{OC} matches the measured value. Keep this *n* 1 fixed for the remaining fit procedure.
- Extend the voltage range as far as meaningful data is present, set *R dis* as fit parameter and fit. The results should already be the final cell parameters.
- Load an illuminated measurement and set only *R par*, *J* 02, *J photo* and *n* 2 as fit parameters. The fit should work very well, keeping all other parameters fixed. *J* 02 and *n* 2 are expected to increase and *R par* to decrease somewhat under illumination, see [1] and references therein for further insight.

2.3 3-Diode model with resistance limited enhanced recombination (RLR)

This model is an extension of the classical 2-diode model and was first proposed by Hernando [2] and later popularized by McIntosh [3]. The equivalent circuit is shown in figure 2. Compared to the regular 2-diode model a 3rd diode D_3 connected via an additional series resistance R_H is added.

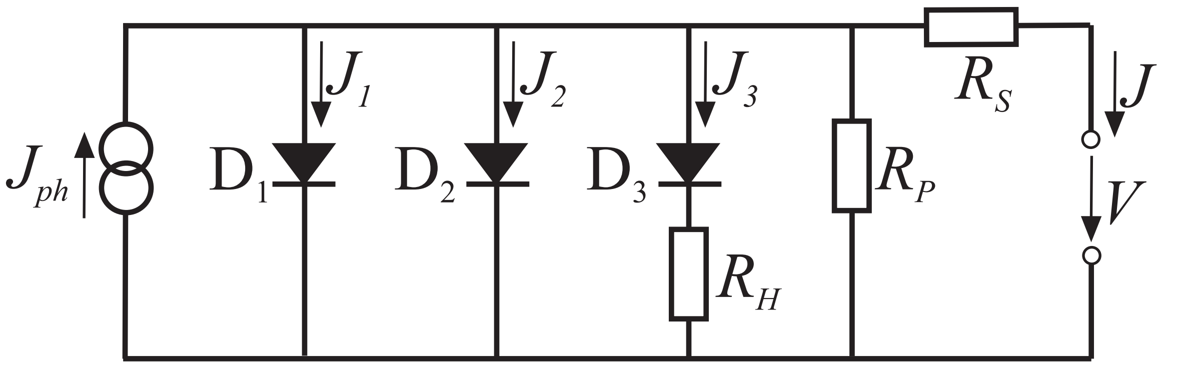

Figure 2: Equivalent circuit of a solar cell according to the 3-diode model with resistance limited recombination.

The terminology for the new elements used throughput the program is:

- $R_H = R\ rec$
- The pre-exponential coefficient of J_3 is *J* 03.
- The ideality factor of diode D_3 is *n* 3.

Diode D_3 represents recombination currents flowing through regions with increased series resistance. This resistance must be due to local defects, because if it was homogenously distributed it would already be included in R_S. It results in a "hump" in the local ideality factor n(V) (see section 3.4.3 for the definition of this quantity), as shown in figure 3. Hence it has historically been called R_H, but is called $R\ rec$ in this program to emphasize that it's the resistor limiting this recombination current.

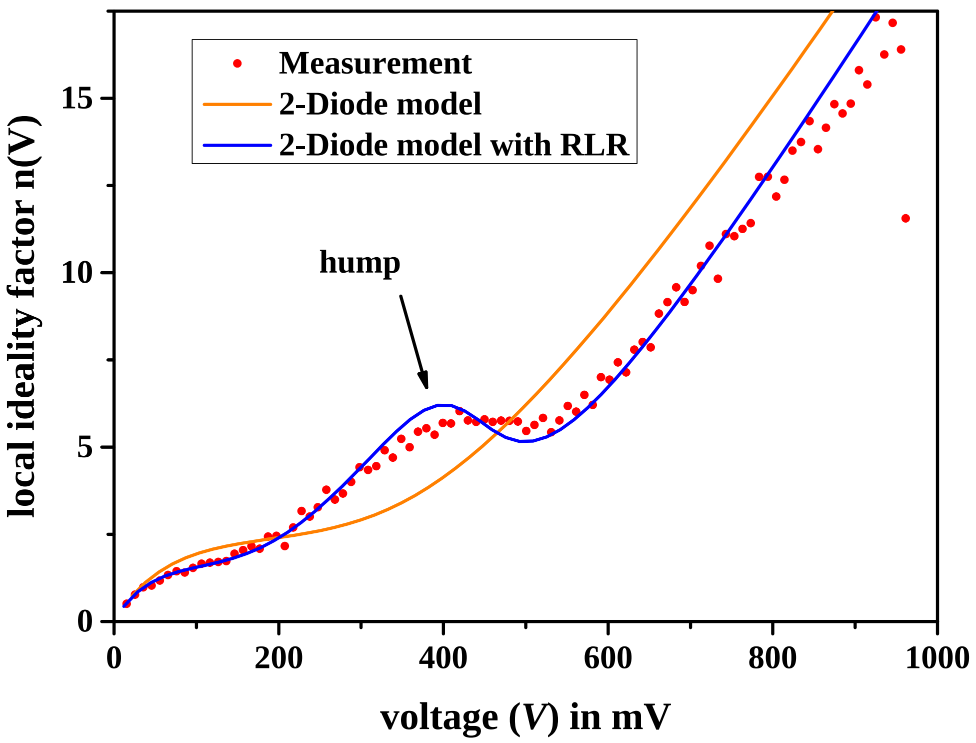

Figure 3: Sample data from Example4.drk fitted by the regular 2-diode model and with RLR using diodes D_1 and D_3. Only positive bias for $n(V)$.

The sample curve fit of the same data shown in figure 4 demonstrates the far superior fit, which can be obtained using this extended model for appropriate data. Note that in this case only diodes D_1 and D_3 have been used, whereas $J\ 02$ was fixed at 0. The excellent fit without diode D_2 means that in this cell RLR dominates any possible recombination via D_2. Adding D_2 will undoubtly improve the fit further, but will likely not help to gain any physical insight since the main features of the measurement curve are already reproduced well. In the spirit of Ockham's razor the simpler solution should be preferred. Similarly, if D_2 is used it is probably a good idea to start with $n\ 2$ fixed at 2 rather than using it as a free fit parameter.

There are several indications that the regular 2-diode model can't fit this data. From 0 to 500 mV I-V is never a straight line, as it would be for a pure exponential term and a logarithmic scale. $R\ par$ can't help either, as otherwise the reverse current would become too high. The only way left for this model to continually reduce the slope is via the global series resistance R_S. However, increasing R_S also applies to diode D_1 and decreases the slope too much at high forward bias. These conditions force a mismatch at reverse bias, too. The fit balances these contradicting requirements by choosing values in between extremes, starting with a slope too high, ending with it being too low and never matching the data or its slope for more than single voltage points.

Viewing the plot of $n(V)$ in figure 3 helps to better visualize the same issues. Here the change

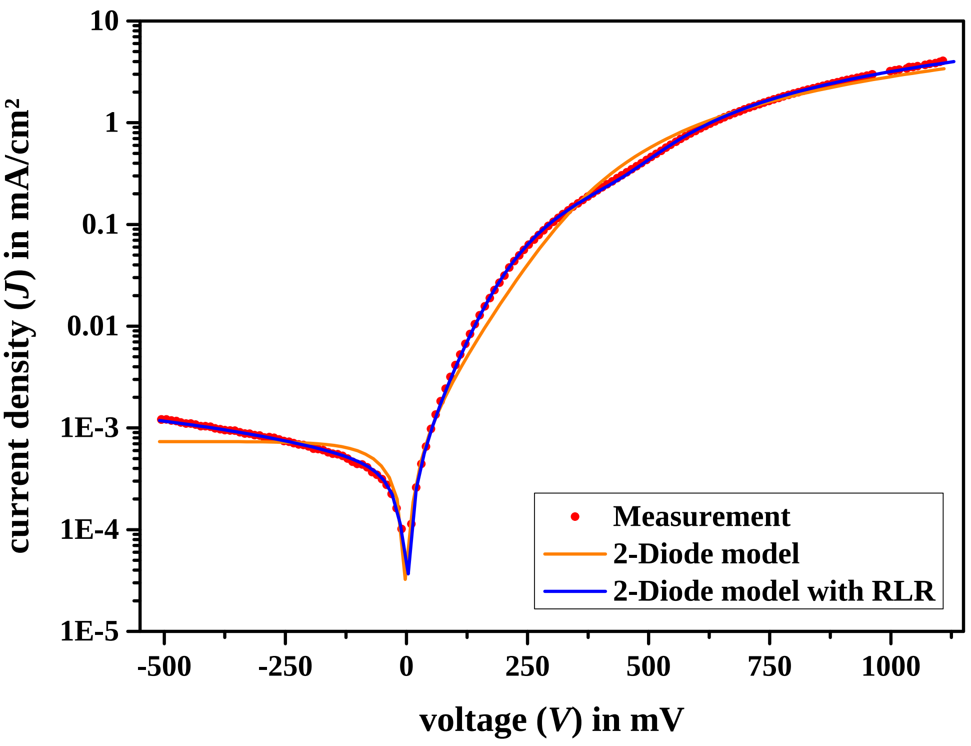

Figure 4: Sample data from Example4.drk fitted by the regular 2-diode model and with RLR using diodes D_1 and D_3.

in slope of the experimental data produces a hump around 400 mV, which the regular 2-diode model can not fit. On the other hand the 2-diode model with RLR can fit a single hump. The full 3-diode model with RLR can even fit 2 humps in $n(V)$.

A possible cause of RLR is edge recombination or non-linear “J02-type” edge shunts, as described in chapter 4.3 in McIntosh’s dissertation [3]. In this case n 3 ≥ 2 is expected. Alternatively it can be caused by damaged tips of the pyramids constituting the anti-reflection texturing [2], which yields similar n 3 values. Another possible source of RLR is the formation of local Schottky contacts bypassing the pn-junction. It is described in further detail in chapter 4.4 in McIntosh’s dissertation [3] and leads to n 3 ≈ 1.4. This unique value can be used to distinguish this mechanism from others. Further mechanisms causing a hump in $n(V)$ but unrelated to RLR are listed chapter 4.5 in McIntosh’s dissertation [3], as well further advice regarding cell diagnosis using this tool.

3 Using the program

3.1 Installation and prerequisites

All files (except this manual) can be found by clicking *Additional materials available* at nanoHUB, beneath the button *Download (PDF)*.

There are 2 different ways to run the program: as a stand-alone executable and using MATLAB. The advantage of using MATLAB is that you can freely adapt the program to your needs. On the other hand the stand-alone executable can run without a MATLAB license and provides access to the full functionality of the program. How to install the program and which files to download depends on which method you choose. Both are explained in the following.

The full program functionality also requires Microsoft Excel (tested with version 2003 and 2010). If Excel is not found, the fit results are saved into a text file rather than an Excel file and the data become harder to read. You also can't save your configuration automatically. In order to adjust the program to the values and options you typically use you'll have to edit the file *config.xls* manually instead. OpenOffice or LibreOffice can be used for this.

3.1.1 MATLAB

Extract all contents from the file *MATLAB files.zip* into the folder you want to work in. Do not rename any files.

The current release was tested with MATLAB 2014a and 2012a, but it should work with moderately older versions as well. The *optimization toolbox* is required to perform the actual curve fit.

Some functionality is disabled in the publicly released version due to license concerns. This applies to the parallel computation: clicking the corresponding button will open a MATLAB Pool, if you have access to the *Parallel Computing Toolbox*, but otherwise has no effect. If this toolbox is not installed the button will be disabled.

3.1.2 Stand-alone executable

The executable files are compiled with the *MATLAB Compiler*. Therefore you need to have the *MATLAB Compiler Runtime* installed in order to run the program. It's available from nanoHUB or the Mathworks: http://www.mathworks.com/products/compiler/mcr/

You need exactly the version used to compile the program, which is 7.17 (R2012a). Other versions will probably not work. These versions are placed in the files *MATLAB Compiler Runtime *.zip* for either Windows 32/64 or Linux 64.

Windows
Extract all contents from the file *Standalone - WinXX.zip* into the folder you want to work in. Here XX represents 32 or 64, depending on whether you have a 32 or 64 bit Windows. You also need the Visual Studio 2010 Runtime redistributable, which is included in the file *MATLAB Compiler Runtime *.zip* and which is already installed on most PCs.

Linux
Extract all contents from the file *Standalone - Linux64.zip* into the folder you want to work in. Binaries have been compiled for 64 Bit Linux, but could not be tested. Let me now if it works or not.

3.2 Starting the program

3.2.1 MATLAB

Start MATLAB and browse to the directory into which you extracted the program files. Start by executing *zweidiodenmodell.m*, e.g. by double clicking it or by typing "zw" + *tab* + *Enter* into the console.

3.2.2 Stand-alone executable

Windows
Browse to the directory into which you extracted the program files and simply double click *START.bat*. This opens a command line window in which various helpful status messages are displayed. Double clicking *zweidiodenmodell.exe* also works, but doesn't show the messages.

Linux
Navigate to the directory into which you extracted the program files and execute the shell script *run_zweidiodenmodell.sh* on the command line. You may need to set *zweidiodenmodell* as executable.

3.3 Basic use

After installing (ref. section 3.1) and starting (ref. section 3.2) the program you must accept the license terms before the main GUI is opened.

3.3.1 Initialization

Click the *Input* button to open a dialog to choose the file containing the *I-V* curve you want to fit. Important is to choose the proper file type, as this determines how the data is interpreted. The supported formats are described in detail in section 3.5.

The current path and the chosen file type are saved in the program settings, so you don't have to navigate to the same path each time you load a new measurement file. You can save all current settings to disk (file: *config.xls*) for the next session by clicking *Configuration* (top menu) followed by *save settings*.

If the file is loaded successfully the measurement data is plotted immediately. Depending on the current at zero bias compared to forward bias, the program automatically selects between dark and illuminated characteristic. Make sure the entries cell *Area*, measurement *Temperature* and *illumination* are correct. The voltage and current range is automatically set to the maximum range where data is available, but in special cases you may want to limit or extend this range by editing the fields *Voltage in V* or *Current in mA/cm*^{2} manually.

3.3.2 Performing the fit

Choose the parameters to fit by checking the box in front of the parameter name. You can edit the initial guess in the left column, right next to the parameter name. For normal use cases it's often a good idea to let the program find meaningful initial guesses (see section 3.4.5) and only intervene if necessary. Press *Fit* to begin fitting using this initial guess. The fit continues until a tolerance criterion is satisfied (see section 3.4.7).

Afterwards the resulting parameter values are updated in the left column. You can change them individually, change which parameters to fit and/or continue fitting by pressing *Fit* again. The previous starting guess is shown in the right non-editable column for quick comparison. You can reapply these (and hence undo a fit) by pressing the *Back* button. In case you get stuck at strange parameter values you can also revert everything to initial values by pressing the *Reset* button. It's a good idea to watch the command line for unusual messages from the fit or from the program in general.

3.3.3 Judging fit quality: RMSE

The fit routine finds the optimum parameters by calculating the so-called *Root Mean Squared Error (RMSE)*, which is a measure for the difference between fit and measurement. It's value is calculated and shown in the lower right field *Root mean square error* as soon as data is loaded. This section explains how the RMSE is calculated and when to use which mode. Refer to section 3.4.7 for the relation between the *RMSE* and the fit termination criteria.

Absolute Residuals
The simplest RMSE is based on absolute differences between measured and fitted current I, called residuals, and can be used by clicking *Absolute* in the *Residuals* panel. The residuals are squared for every data point j, summed and subject to various normalizations:

$$RMSE = \sqrt{\frac{1}{N_V}\sum_{j=1}^{N_V}\left(\frac{\Delta V_j}{\Delta\tilde{V}}(I_{j,measured} - I_{j,fit})\right)^2}\,, \tag{1}$$

with N_V as the number of voltage points, ΔV_j as the individual voltage intervals and $\Delta\tilde{V}$ being the average voltage interval

$$\Delta\tilde{V} = \sum_{j=1}^{N_V}\frac{\Delta V_j}{N_V}\,. \tag{2}$$

Weighting by ΔV_j helps fitting measurements with non-equidistantly spaced voltages, whereas the normalization to the number of voltage points in equation 1 ensures that the apparent fit quality is independent of the measurement voltage resolution and voltage range fitted. This mode works well for illuminated I-V curves.

Relative Residuals
If the current spans more than one order of magnitude, as is customary for dark I-V measurments, the absolute residuals do not work well and you should check *Relative* in the *Residuals* panel. Thereby the residuals are normalized to the current, so that the physical meaning of the *RMSE* changes from equation 1 into

$$RMSE = \sqrt{\frac{1}{N_V}\sum_{j=1}^{N_V}\left(\frac{\Delta V_j}{\Delta\tilde{V}}\left(\frac{I_{j,measured} - I_{j,fit}}{I_{j,measured}}\right)\right)^2}\,. \tag{3}$$

If your dark I-V data at low currents is too noisy, switching back to absolute residuals may improve the fit. Otherwise you should exclude that voltage range.

The program's default setting is to use absolute residuals for illuminated I-V and relative ones for dark measurements. This default setting is applied when ever you open a file of the other type, assuming that if you changed the mode manually and open a file of the same type you want to keep the previous mode. You can change the defaults to your current settings by clicking *Save settings* in the *Configuration* menu.

Fitting the local ideality factor
The *Residuals* panel also provides the option to fit I-V and the local ideality factor $n(V)$ (see section 3.4.3) simultaneously by activating *Fit n(V).* In this case the relative RMSE is extended from equation 3 to

$$RMSE = \sqrt{\frac{1}{N_V}\sum_{j=1}^{N_V}\left(\frac{\Delta V_j}{\Delta\tilde{V}}\sum_{i=I,n}\Lambda_i\left|\frac{i_{j,measured} - i_{j,fit}}{i_{j,measured}}\right|\right)^2}\,. \tag{4}$$

Here the relative residuals for every data point j are calculated for I and n (represented by i) in the same way, weighted by respective Λ_i and summed up. Λ_i can be adjusted by the slider right to the checkbox *Fit n(V).* It is automatically set so that $\Lambda_I + \Lambda_n = 1$ and thus the magnitude of the RMSE remains comparable to regular relative residuals.

Calculating $n(V)$ assumes an exponential $I(V)$, hence no meaningful values can be obtained at reverse bias. To account for this equation 4 is only applied at forward bias, whereas for reverse bias equation 3 is used unmodified.

3.3.4 Exporting the results

You can save the current fit parameters by pressing *Save* and/or all calculated curves being displayed by pressing *Save graph*. Calling either function triggers the output file name choice dialog, if you did not already define an output file (button *Output*).

If Microsoft Excel is present on your system, clicking *Save* appends the fit parameters to a *.xls file of the name given in the *Output* dialog. If the file does not yet exist, a new one is created. There are 3 header lines: a general category and the actual values followed by the corresponding units. If Excel is not present the same data is put into a simple text file in the same way. The only drawbacks are missing entries for cell name and fit data, as well as reduced readability. But the data is delimited by tabs and can thus easily be imported into programs such as Origin or Open/LibreOffice.

Clicking *Save graph* writes all calculated curves into simple text files with the names given in the *Output* dialog and their legend label plus *.dat* appended. The 1st column contains the voltage in V and the 2nd one the current in mA/cm^2. The current sign is set according to the solar convention, i.e. technical current direction. The voltage resolution of the fit curve can be set in the *Configuration* menu by clicking either *Plot: #voltage points* or *Plot: voltage resolution*. The measurement data is also saved, which might seem redundant but can come in handy to diagnose file import errors or if you want to plot fit and measurement in an external program in the same way as it was shown in this program. A screenshot of the GUI in its current state is also saved.

3.4 Advanced functionality

3.4.1 Simulation mode

You can easily simulate *I-V* characteristics by not loading measurement data. The resulting *I-V* and individual currents can be saved into text files for further processing or plotting (see 3.4.2 for further details). Note that since the current depends on the voltage you can limit and extend the voltage range, but by editing the minimum and/or maximum current you can only limit the voltage range, but not extend it.

3.4.2 Graphing

The graph window always contains an *I-V* curve calculated based on the current parameter settings and a marker for the mean power point *MPP* for illuminated *I-V* curves. This curve is called *Simulation* until a data file is loaded, which switches the name to *Fit*. Afterwards *Previous Fit* in the *Show:* panel at the bottom of the GUI can activate a plot with the previous fit parameters for quick comparison. A plot of the voltage dependent local ideality factor *n(V)* can also be activated in the same panel (see section 3.4.3 for a definition of this quantity).

Additional components can be plotted by activating check boxes in the *Show:* panel to help visualize the different contributions to *Simulation/Fit*. For *Previous Fit*, *Diode 1*, *Diode 2*, *Diode 3*, and *R par* $I(V)$ is plotted as well as $n(V)$ if the corresponding box is checked. *J photo* simply adds a line at the photo current.

The current sign can be switched between the solar convention (technical direction) and physical current direction by toggling *Current sign: solar convention* in the *Configuration* menu. The axis scaling can be toggled between linear (default for illuminated *I-V*), logarithmic current (default for dark *I-V*) and double-logarithmic mode by repeatedly clicking the button which is named *Logarithmic*, *Double logarithmic* or *Linear* depending on the current plot status.

The voltages at which the *Simulation/Fit* curve is plotted is independent of the measurement data. It can be controlled by setting in the *configuration* menu either the number of voltage points (*Plot: #voltage points*) or the voltage resolution (*Plot: voltage resolution*). The default setting is to use 100 voltage points for a quick and sufficiently smooth plot, but you can set higher precision if neccessary. Set less than 2 voltage points or a resolution of 0 to use the same voltage points as the measurement.

For dark *I-V* near 0 V the calculated currents can become very small and disturb the plot. In this case either limit the voltage range, e.g. starting from 0.01 V, or use a more coarse voltage resolution.

3.4.3 Plotting and fitting the local ideality factor n(V)

The local ideality factor *n(V)* of a pn junction is calculated by assuming a simple exponential dependence of the current I on the voltage V with ideality factor n

$$I \propto \exp(\frac{V}{nk_BT}) \, , \tag{5}$$

with k_B as Boltzman's constant and T as the absolute temperature [3]. This can be rearranged to yield [3]

$$n(V) = \frac{I}{k_BT}\frac{\partial V}{\partial I} \, , \tag{6}$$

which is used to calculate $n(V)$ by approximating the derivative by pairwise differences. This differentiation makes *n(V)* prone to measurement noise, which can be counteracted by smoothing data prior to importing them. No automatic smoothing is performed in this program in order not to disturb user data when this might be unwanted. Equation 5 generally only holds true for forward bias of the pn junction, hence *n(V)* is not calculated at reverse bias.

Fitting *I-V* data sometimes fails to reproduce the derivative correctly, e.g. humps in *n(V)* appear at slightly offset voltages or are missing completely. The latter case hints at the model being inadequate to describe the measurement data and you might want to activate RLR (see section 2.3 for a description of this model). In the former case the model should be able to reproduce the measurement, but the regular residual calculation yields the minimum RMSE for a solution which is obviously not physically optimal.

To counteract such behaviour the option is provided to fit *n(V)* in addition to *I-V*. To use it you have to activate *Relative* in the *Residuals* panel, which will reveal the option to *Fit n(V)*. Activating this mode shows a slider to set the weight between *I-V* and *n(V)*. Values further to the right put more emphasis on *n(V)*, which helps to match the experimental *n(V)* curve better. You can even set pure *n(V)* fitting, although this does not work well since it usually fails to reproduce the absolute currents. Trying to fit too noisy *n(V)* data does not work well either. For further information about how this option changes the RMSE consult section 3.3.3.

3.4.4 Dark I-V

Whether you are using a dark or a illuminated *I-V* curve is detected automatically and the default graph scaling is set accordingly (see section 3.4.2). If this fails un/-check *Illuminated I-V*. For dark *I-V* measurements *J photo* is automatically treated as 0 and can not be a fit parameter. If a different value is set manually it is used to estimate the open circuit voltage V_{OC} based on the calculated dark *I-V* curve and *J photo*.

The measured current at 0 V is determined via interpolation and subtracted from the data (otherwise the fit would not work well). An example of such an *I-V* curve is *Example1.dat* in the directory *File Examples*.

3.4.5 Automatic parameter estimates

The *Configuration* menu provides the option *Suggest x*, with *x* representing most 2 or 3 diode model parameters. If activated the program tries its best to automatically generate good initial guesses for these parameters (until the first fit is performed). These estimates work pretty well, have noise reduction mechanisms built in and mostly help the fit to converge faster or to find a solution at all.

Nevertheless, there are obviously cases where this can go wrong, hence these suggestions are optional. They can also be overridden manually by editing a value. This disables the estimate for the edited parameter until the next data file is loaded, but keeps updating the other parameters in accordance with the manually set value.

The estimates are generated in the following way:

- *J photo* is interpolated at 0 V, or extrapolated if necessary.
- *R par* is estimated from the slope at reverse bias, if sufficient data is present. Regions where reverse bias breakdown is detected are excluded.
- *R ser* is extracted from a single-diode equation at forward bias over a range of up to 0.2 V.
- *J* 01 is again estimated at forward bias using the single-diode approximation and the previously estimated *R ser*, taking a range of 0.1 V into account to treat measurement noise.
- To estimate the parameters of the 2nd diode a voltage range is selected where the current through this diode dominates, or the closest possible forward bias range: $J_2 > 2J_1$ and $J_2 > 2J(R\ par)$.
- *n* 2 is determined from the slope in this voltage range.
- Finally *J* 02 can be calculated in the same voltage range as *n* 2.

If the 3-diode model with RLR is chosen it is assumed that there is a good reason for this, i.e. the 3rd diode with RLR is clearly visible in the data. Its parameters *J* 03, *n* 3 and *R rec* are estimated by assigning all current to the 3rd diode which is not either *J photo*, J_1 or $J(R\ par)$ in the following procedure:

- A voltage range is selected where the relative strength $J_3/(J_1 + J(R\ par))$ of the 3rd diode is large at forward bias.
- *R rec* is determined from the change of slope in this voltage range.
- *n* 3 is determined from the slope in this voltage range.
- *J* 03 is calculated in this voltage range.

Any remaining current is then attributed to the 2nd diode with *n* 2 fixed at 2 (see section 2.3) and *J* 02 is calculated from this current.

3.4.6 Boundaries

You can check the box *Boundaries* to expand the GUI and reveal further options. Here you can freely set minimum or maximum parameter values. However, doing so will switch the MATLAB fit routine from the default Levenberg-Marquardt to a trust-region algorithm, which performs considerably worse. You will need a good initial guess and minimum function (RMSE) tolerance for this.

3.4.7 Fit termination criteria

There are several stopping criteria for the fit. Besides the obvious one, the number of iterations (adjustable under *Max. iter.*), there are also function value and step size tolerances. Which stopping criterion was triggered is written to the command line.

The function tolerance refers to the *RMSE*, as defined for illuminated *I-V* in equation 1 and for dark *I-V* in equation 3. In the latter case setting a *Relative RMSE tolerance* of e.g. 10^{-3} in the *Set precision* menu means the fit will continue until one fit iteration changes the *RMSE* by less than 10^{-3} of the previous value. A resulting *RMSE* of e.g. 0.02 then means that fit and measurement differ on average by 2 %. The tolerance setting has the same meaning for illuminated *I-V*, but the *RMSE* is given in absolute current rather than being relative.

The step size tolerance is controlled by the *Relative parameter tolerance* in the *Set precision* menu and terminates the search if the change in the fit parameters relative to their previous values falls below this threshold. MATLAB actually uses the 2-norm of all parameter changes, which ensures that no parameter changes by more than the set threshold.

3.4.8 Suns-Voc

In order to fit data from Suns-V_{OC} measurements, simply deactivate *R ser* as fit parameter and set it to 0.

3.4.9 1-diode model

The current program can easily be used for fits according to a 1-diode model: simply deactivate *J* 01 or *J* 02 as fit parameter and set either to 0. A voltage-dependent ideality factor often used in conjunction with such models can be displayed along the fit (see section 3.4.2).

3.4.10 Variable ideality factors

According to classic theory *n* 2 should be equal to 2 [11], since the most recombination active defects are expected to be located in the middle of the bandgap. However, in practice higher values are common for defect rich cells or lower values for high quality cells (see e.g. [6, 12] and references therein). Closer spatially resolved inspection reveals strong local variations of *n* depending on local defects [12]. This can not be accounted for in the two diode model, yet treating *n* 2 as a free fit parameter undoubtly generates better results than fixing it at 2.

If one desires, *n* 1 can also be changed to fixed values other than 1, e.g. to match the measured open circuit voltage in dark *I-V* fits, or it might be treated as a free fit parameter.

3.4.11 Batch processing

You can fit any number of files automatically using *run job* in the *Batch processing* menu. Whatever is currently set in the GUI will be used for all files and all results are written into the specified output file. The automatic parameter suggestions (see 3.4.5) will come in very handy if the *I-V* curves differ considerably (e.g. data from a firing optimization). On the other hand for very similar cells a good static initial guess should work very well. For fitting large batches the *Parallel computation* option should help (see section 3.4.12).

Quality control is important after a batch job. Check the RMSE, required number of iterations and the extracted parameters in the result file for anomalies and treat questionable data manually. Additionally you can save the fit curves by activating *Save plot* in the *Batch processing* menu and check these in some post-processing tool.

Note: the current GUI state is literally applied to all batch fits. This can be problematic if you performed a dark *I-V* fit and want to fit illuminated data in batch mode. In this case *J photo* is still inactive and can't be set as fit parameter even for the illuminated fits.

3.4.12 Parallel computing

Checking *Parallel computation* in the *Configuration* menu to open a MATLAB Pool session enables execution acceleration if your machine has multiple CPU cores and the *Parallel Computing toolbox* if you're using MATLAB. Enabling this options requires anywhere from 5 to 60 s depending on your machine, but afterwards the acceleration is available for the entire session.

In batch mode you can expect very good scaling, approaching a speedup equal to the number of physical CPU cores, and the best load balancing using as many workers as you have logical cores. For single fits the stand-alone executable version can also parallelize this workload. This parallelization generates additional overhead (see [4] for timing results), so that the benefit is larger the longer each fit iteration takes, i.e.

- the more voltage points your measurement contains.
- the more complex the physical model is. Sorted in ascending complexity: 2-diode model, Breitenstein-Rißland-model, RLR.
- the more fit parameters you check.

3.4.13 Save configuration

When ever you fit similar *I-V* curves repeatedly it is convenient to save your current settings as default by clicking *Save settings* in the *Configuration* menu. This includes any checked/unchecked boxes and options, as well as initial parameter values. This only works if Microsoft Excel is installed. Otherwise you can manually edit the file *config.xls* via e.g. OpenOffice or LibreOffice. There are error checks for the *config.xls* file, so if anything goes wrong the defaults will be used and you should see an error message in the command line output.

3.4.14 Configure excel input (files)

If you want to import measurement data contained in .xls files, you have to tell the program where the data is located. Click *Configure Excel input* in the *Configuration* menu. Enter the proper values and save those settings if you need them repeatedly.

3.5 Supported file formats

Currently 5 different file formats are supported. Examples of each type are included in the file *File Examples.zip.* Common to all formats is that the separator between the columns should be a space or tab, while commas are OK for some and are converted to decimal points in others. Table 1 gives an overview over the formats.

extension	header lines	comma interpeted as	information from header
.dat	0	decimal point	-
.mes	1	decimal point	-
.TxT	29	delimiter	area, name
.drk	30	delimiter	area, temperature, name
.xls	flexible	-	-

Table 1: Overview of supported file formats

Note: upon import the data is sorted by voltage, so that files can include a forward and backwards sweep. However, in case of significant hysteresis this will look like noise or two different overlapping curves. Such cases should be avoided by the user.

3.5.1 *.dat

This is a simple text file. There is no header or other meta-data included. The data is arranged in 2 columns: the 1st one contains the voltage in V, the 2nd one contains the current in A. The decimal point can be “.“ or “,”. This format is probably ideal if you convert your data from some other format.

3.5.2 *.mes

This format is similar to .dat, but contains an additional header of 1 line. Such files are generated e.g. by an HP 4156A semiconductor parameter analyzer.

3.5.3 *.TxT

This file format contains 29 header lines, generated by a custom solar simulator software. Some meta data is extracted from this and displayed in the fitting tool. The cell area is automatically read from the header. The actual data is arranged in 3 columns: the 1st one contains the voltage in V, the 2nd one the current in A and the 3rd one a monitor value, which is disregarded in this tool.

3.5.4 *.drk

Such files are generated e.g. by a PV Tools EQE measurement setup. They consist of 30 header lines, followed by the voltage in V and the measured current in A. The cell name, area and average measurement temperature are extracted automatically from the header.

3.5.5 *.xls

For Excel files the worksheet and data range can be configured freely in the *Configuration* menu (option *Configure Excel input*).

4 Extending the program

This section mainly describes how additional languages can be added to the program. This requires MATLAB and few programming skills. Furthermore the steps necessary to add support for additional file formats are listed. This does require some programming skills. Further modifications probably require serious knowledge of how the program works. Advice is given on how the language support works, should such modifications be attempted.

4.1 New languages

Adding support for additional languages to the program is relatively easy. Load the file *language.mat* into MATLAB, edit the 5 variables contained therein and save the results. They are of the type cell and must contain strings, i.e. add values as *'value'*. First add another row to the variable *LanguageDefinition* and place the language name to be displayed therein. In the variables *GUILabels*, *Tooltips* and *OtherLabels* simply add another column and place the translations there. Editing *Special* works in the same way, except that some cells contain single-column cell arrays themselves for multi line text labels.

Note: changing the 1st column in any variable (except *LanguageDefinition*) breaks the program and will make it default to hard-coded German.

4.2 New file formats

In order to support new formats one has to add a function to *zweidiodenmodell.m* analogous to e.g. the function *read_IV_Textfile*. It has to be called in *LoadData_Callback* and *Menu-RunBatch_Callback*, analog to the other cases in the switch statement. Furthermore *handles.OpenDialogLabel* has to be extended by one element to show the new option. This is done in function *SetLanguage*, before the *catch* statement. The new label text must be placed into *language.mat*, variable *Special* following the other file format descriptions. Note that the row numbers of any elements in *Special* are hard coded in *SetLanguage* and must thus be updated.

4.3 Extending the 2-diode model

Extensions already implemented are the distributed series resistance (see section 2.2) and the 3-diode model for RLR (see section 2.3). Remarkably the actual current calculation algorithm also works with these models, with minor adjustments. This suggests that other components might be added to the equivalent circuit in a similar manner - as long as the behavior of the resulting equation does not change fundamentally. Otherwise the search algorithm to calculate the current will probably break.

4.4 Other extensions

Further modifications of the program probably require real programming effort. The only further advice given here concerns the language: if you add elements to the GUI you likely need to add new text labels to the file *language.mat*. Static labels belong into the variable *GUILabels* and tooltips into the variable *Tooltips*. Add the exact tag (without "handles.") of the additional element in the first empty row of the 1st column and the actual text labels in the following columns. The program will automatically detect the number of entries and treat them all.

Care must be taken for labels, which do not belong into either category. A good example is the graph scale button: it can show 3 different labels, depending on the situation. These can not be set statically, but are rather saved into a field of *handles* in the function *SetLanguage* and are assigned later on as needed. If you add elements which require this special treatment, you should add them and their field name to the variable *OtherLabels* in *language.mat*. You may also want to define default values in the *catch* statement within the same function, in case *language.mat* can not be loaded.

Elements which require any other treatment should be placed into the variable *Special* in *language.mat* and treated at the before the *catch* statement in the function *SetLanguage*.

5 Version history

2.3 by Stephan Suckow (2014)

- Custom Levenberg-Marquardt, Jacobi option not needed any more (removed).
- Can plot individual components of local ideality factor.
- Can fit n 2 when using RLR.
- Added parameter estimate for J 02 when using RLR.
- Performance improved through analytic derivatives and delayed I/O.
- GUI adjusts to window size.

2.2 by Stephan Suckow (2014)

- Rearranged residual control: can also fit illuminated IV with relative residuals.
- Can fit I-V and local ideality factor.
- Enhanced parameter estimates.
- Improved batch performance by reduced overhead and parallel computation.
- Moved language definition to MATLAB file for better stability.
- *Save graphs* also saves screenshot.

2.1 by Stephan Suckow (2014)

- Weighting residuals with their voltage interval for better fit of non-equistantly spaced data.
- Added update check.
- Can show local ideality factor.
- Added 3-diode model for RLR.
- Automatic estimate of 3rd diode added.

2.0 by Stephan Suckow (2013)

- Made left and right parameter edit/text fields more intuitive and consistent.
- Added undo fit.
- Added Simulation mode.
- Added options to display previous fit and individual currents.
- Can set illumination intensity.
- Can set data range by limiting current.
- Initial guess for J 02 and n 2 added.
- Tolerances enhanced and passed to current calculation.

1.7 by Stephan Suckow (2013)

- Breitenstein-Rißland model implemented.

- Additional series resistance in the current path of the 2nd diode removed.
- Can set fit tolerances.
- Better initial guess for $R\ par$, significantly better for $R\ ser$.
- Initial guess for $J\ 01$ added.
- Automatic detection of dark / illuminated I-V curves with automatic plot scaling.
- Support for PV Tools EQE measurement device added.

1.6 by Stephan Suckow (2013)

- n 1 adjustable.
- Support for 32 bit Windows added.
- Batch processing added.

1.5 by Stephan Suckow (2012)

- Help menu, about screen and copyright notice added.
- Language can be switched between German and English, more can easily be added into language.xls.
- Voltage resolution or number of voltage points in plot can be chosen via menu.
- Series resistance in the current path of the 2nd diode can be activated in the menu.

1.4 unpublished (2010)

1.3 by Stephan Suckow (2009)

- Tolerances for Levenberg-Marquardt set more generous.
- Can choose plot scale: linear, logarithmic and double logarithmic.
- Can save fit curve.
- Can reset plot manually.
- Don't need to restart tool for next I-V file.
- Current calculation significantly faster and more robust.
- Can choose voltage range manually.
- Can choose between illuminated and dark I-V curves.

1.2 by Stephan Suckow (2008)

- Added option for parallel computation.
- Added Jacobi-option for fit.
- Unified diodeI.m and diodeII.m into diodeI.m.
- Current calculation improved.
- Can set boundaries for parameters.
- Can choose fit parameters.
- Can read .mes files.

1.1 by Elmar Stegemann

References

[1] Breitenstein O., Rißland S. A two-diode model regarding the distributed series resistance. *Solar Energy Materials & Solar Cells* 2012; **110**: 77-86, DOI: 10.1016/j.solmat.2012.11.021.

[2] Hernando F., Gutierrez R., Bueno G., Recart F., Rodriguez V. Humps, a surface damage explanation. *Proc. 15th EC PVSEC* Vienna, 1998; pp. 1321-1323.

[3] McIntosh K.R. Lumps, Humps and Bumps: Three Detrimental Effects in the Current-Voltage Curve of Silicon Solar Cells. Dissertation, Centre for Photovoltaic Engineering, University of New South Wales, Sydney; 2001.

[4] Suckow S., Pletzer T.M., Kurz H. Fast and reliable calculation of the two-diode model without simplifications. *Progress in Photovoltaics: Research and Applications* 2012; accepted, DOI: 10.1002/pip.2301.

[5] Wolf M., Rauschenbach H. Series resistance effects on solar cell measurements. *Advanced Energy Conversion* 1963; **3(2)**: 455-479, DOI: 10.1016/0365-1789(63)90063-8.

[6] Wolf M., Noel G.T., Stirn R.J. Investigation of the Double Exponential in the Current-Voltage Characteristics of Silicon Solar Cells. *IEEE Transactions on Electron Devices* 1977; **24(4)**: 419-428, DOI: 10.1109/T-ED.1977.18750.

[7] Stutenbaeumer U., Belayneh M. Equivalent model of monocrystalline, polycrystalline and amorphous silicon solar cells. *Renewable Energy* 1999; **18**: 501-512, DOI: 10.1016/S0960-1481(98)00813-1.

[8] Greulich J., Glatthaar M., Rein S. Fill factor analysis of solar cells' current-voltage curves. *Progress in Photovoltaics: Research and Applications* 2010; **18**: 511-515, DOI: 10.1002/pip.979.

[9] Enebish N., Agchbayar D., Dorjkhand S., Baatar D., Ylemj I. Numerical analysis of solar cell current-voltage characteristics. *Solar Energy Materials & Solar Cells* 1993; **29**: 201-208, DOI: 10.1016/0927-0248(93)90035-2.

[10] Ishaque K., Salam Z., Taheri H. Simple, fast and accurate two-diode model for photovoltaic modules. *Solar Energy Materials & Solar Cells* 2011; **95**: 586-594, DOI: 10.1016/j.solmat.2010.09.023.

[11] Sah C.T., Noyce R.N., Shockley W. Carrier generation and recombination in P-N junctions and P-N junction characteristics. *Proceedings of the IRE* 1957; **45**: 1228-1243, DOI: 10.1109/JRPROC.1957.278528.

[12] Breitenstein O. Nondestructive local analysis of current-voltage characteristics of solar cells by lock-in thermography. *Solar Energy Materials & Solar Cells* 2011; **95**: 2933-2936, DOI: 10.1016/j.solmat.2011.05.049.