Lisa software – tutorial

Warning: if you spot a mistake in this tutorial, please contact me.

Disclaimer:

The author does not make any warranty, expressed or implied, or assume any legal liability or responsibility for the accuracy, completeness or usefulness of any information contained in this work. Use the information and instructions contained in this work at your own risk.

Requirements

To run Lisa, one needs:

1. Python 3.
2. Numerical python – numpy.
3. Matplotlib library to plot the results.
4. Scipy library.

Example run – single simulation

1 Adjust the parameters in the parameters.py file

In particular, set the output flags to "True" for a single calculation.

Comment

If **jv_flag** is set to true, Lisa calculates the full JV characteristic and saves it to a file. This can take much longer than standard simulations. When this flag is set to false, Lisa uses algorithm to find maximum of the power density curve and Voc, and therefore needs much less voltage points to calculate the solar cell performance. The calculations are much faster.

2 Run the solver

In the console type:

```
python lisa.py [thickness in um] [SRV in cm/s]
```

For example:

```
python lisa.py 1 0
```

The results are printed in the console. The output should look like:

```
============ Parameters of the structure ============

Built-in potential of the junction (V):          0.846
Cell thickness (um):                             1.000
Emitter thickness (um):                          0.005
SCR thickness (um):                              0.336
Thickness of the quasi-neutral n-type region (um):    0.003

============ Results ============

Thickness (um): 1.000
Bottom SRV (cm/s): 0.000
Open-circuit voltage (V): 0.833
Short-circuit current density (mA/cm2): 33.364
Fill Factor: 0.902
Efficiency (%): 25.050
```

3 Plot JV curve

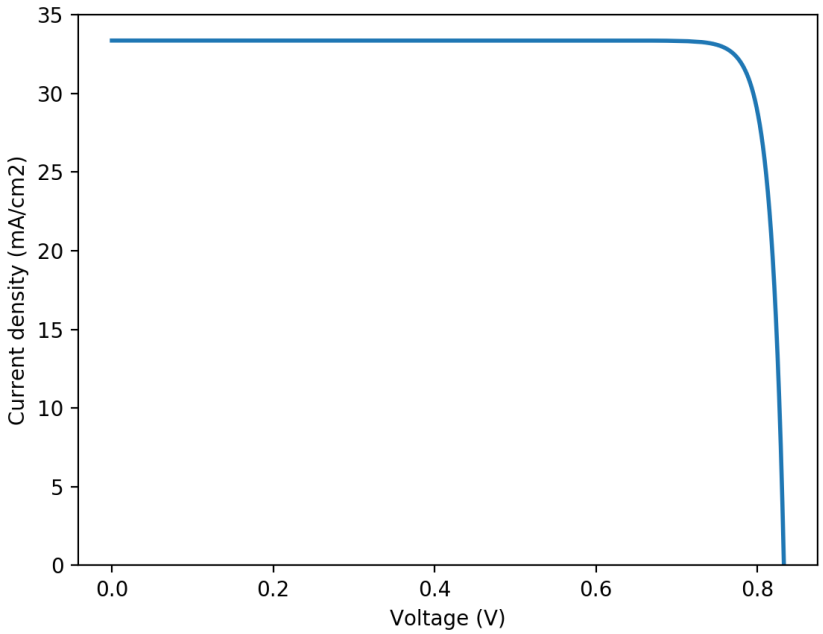
In the console type

python plots.py jv [filename]

The file should be in the **Results/** directory. For example:

python plots.py jv JV.dat

You should get a plot similar to this one:



4 Example run – batch simulation

For example, we want to calculate efficiency limits in a given thickness range.

Adjust the parameters in the parameters.py file

In particular, set all output flags to "False".

Prepare loop

Edit loop.py file. E.g., set thickness range. Then run it:

```
python loop.py
```

File loop.sh will be generated. Execute it by typing:

```
sh loop.sh
```

The calculations will start, and the results will be printed on a screen. If you want to save the results to file, instead of the command above, you type:

```
sh loop.sh >> [filename]
```

E.g.,

```
sh loop.sh >> ./Results/loop.dat
```

The output will be saved to the given file.

5 Plot the results

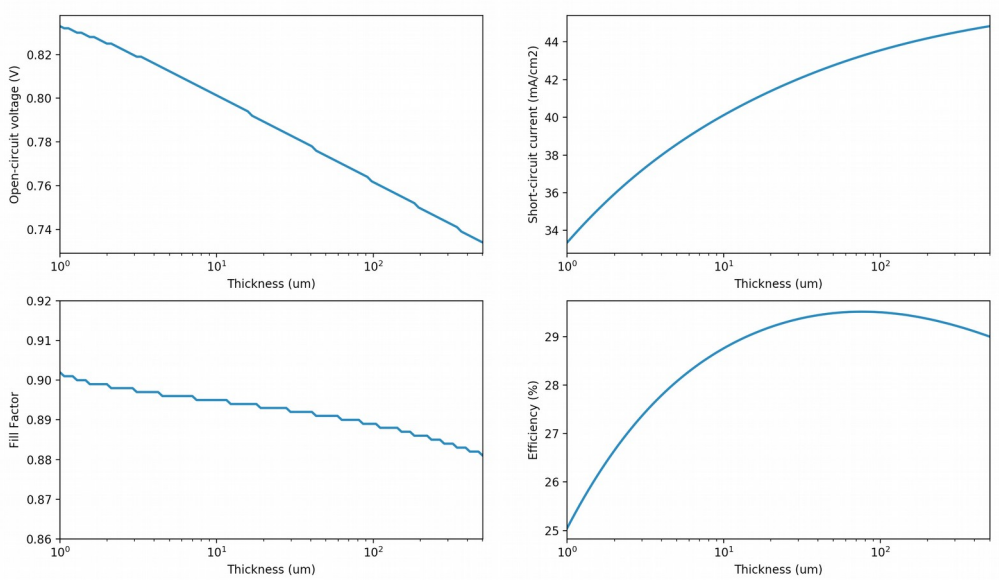
Now you can plot parameters of the cells. Type:

```
python plots.py panel [filename]
```

For example:

```
python plots.py panel loop.dat
```

You should get a plot similar to this one:



6 Plot the results – comparison

It is often useful to compare our results with a different data set. For example, we want to compare the results obtained using the Hovel model with the results obtained using the diode equation. To do so, type:

```
python plots.py panel_comparison [filename 1] [filename 2]
```

For example:

```
python plots.py panel_comparison loop.dat loop_diode.dat
```

You should get a plot similar to this one: