PVMOS manual

B.E. Pieters*
Institut für Energie und Klimaforschung - IEK5 Photovoltaik,
Forschungszentrum Jülich, 52425 Jülich, Germany
(Dated: June 10, 2014)

I. INTRODUCTION

This is (or rather will be as the current state of this document is far from finished) the manual for the PhotoVoltaic MOdule Simulator (PVMOS). PVMOS is an ordinary differential equation solver using finite-differences specifically designed to electrically model solar modules. For more information on how that works I refer the reader to []. The purpose of this document is to document how PVMOS is operated.

The input for PVMOS is a plain text file with commands. To solve a problem PVMOS is typically called from the command line with as an argument the filename describing the problem. With these input files you can specify the geometry of your solar cell/module including the local properties such as electrode sheet resistances and solar cell properties. You can also specify which calculations you want PVMOS should perform and what data to save. A call to PVMOS from the command line looks like this:

```
pvmos [verbose-option] <input-file>
```

where `[verbose-option]` is an option which how much inpormation PVMOS outputs to stdout, and `<input-file>` is the plain text input file. We first describe the mesh data structure in more detail in the next section.

II. THE PVMOS MESH DATA STRUCTURE

A large part of the commands implemented in PVMOS are to manipulate meshes. The following describes the data stored in a mesh structure in PVMOS. The mesh contains rectangular elements. For each element we store the coordinates of the lower left corner (x_1, y_1) and of the upper right corner (x_2, y_2). Furthermore each node has a unique ID number. In order to save memory space we do not store the electrical properties for every element. Instead, we store the electrical parameters per "area". To this end a mesh contains a list of area data-structures whcih contain parameters such as the electrode sheet resistances and the solar cell properties. Each element now refers to a certain area by means of an area-ID number. Each area also has a name for easy referencing by the user.

In PVMOS we can define more than one mesh at the same time. In order to reference one particular mesh, each mesh has a name. To reference an area within a mesh you can refer to `<mesh-name>.<area-name>`, i.e. the name of the mesh followed by a dot and the name of the area within that particular mesh. Sometimes we need to select elements in a mesh (for example when we want to assign a set of elements to a certain area). To this end each mesh has a list of selected elements. If you select elements in a mesh the list is occupied by the element ID's, after which you can do operations on the selected elements. Note that elements are selected on a per mesh basis. In the following section we discuss the PVMOS-input file syntax.

III. PVMOS-SYNTAX

In the input file you can define meshes. Each mesh you define is a variable andthus has a name. You can define several meshes and join them together in new meshes. This way it becomes fairly simple to, for example, put several cells in series by defining a single cell and joining several single cells together to get them series connected.

The input file should contain commands. Each command takes a certain number of input parameters. In the following table all commands are listed.

Creating Meshes

*Electronic address: `b.pieters@fz-juelich.de`

Keyword	Description
`newmesh`	Create a new, rectangular mesh. The command takes six arguments: 1. `x1`, x-coordinate of the lower left corner 2. `y1`, y-coordinate of the lower left corner 3. `x2`, x-coordinate of the upper right corner 4. `y2`, y-coordinate of the upper right corner 5. `Nx`, Number of elements in x direction 6. `Ny`, Number of elements in y direction 7. `mesh-name`, Name of the new mesh
`joinmesh`	Create new mesh by joining two meshes. Make sure the meshes touch but do not overlap. The function takes offset values as input which allow you to "shift" the second mesh to align it to the first. The command takes 5 arguments: 1. `x_off`, x-offset in coordinate system of the second mesh 2. `y_off`, y-offset in coordinate system of the second mesh 3. `mesh1-name`, Name of the first mesh 4. `mesh2-name`, Name of the second mesh 5. `mesh3-name`, Name of the resulting mesh
`joinmesh_h`	Create new mesh by joining two meshes. Make sure the meshes touch but do not overlap. The function takes a y-offset value as input which allow you to "shift" the second mesh in the y-direction to align it to the first. The x-offset value is the maximal x-value found in the first mesh. The command takes 4 arguments: 1. `y_off`, y-offset in coordinate system of the second mesh 2. `mesh1-name`, Name of the first mesh 3. `mesh2-name`, Name of the second mesh 4. `mesh3-name`, Name of the resulting mesh
`joinmesh_v`	Create new mesh by joining two meshes. Make sure the meshes touch but do not overlap. The function takes an x-offset value as input which allow you to "shift" the second mesh in the x-direction to align it to the first. The y-offset value is the maximal y-value found in the first mesh. The command takes 4 arguments: 1. `x_off`, x-offset in coordinate system of the second mesh 2. `mesh1-name`, Name of the first mesh 3. `mesh2-name`, Name of the second mesh 4. `mesh3-name`, Name of the resulting mesh

Selecting Elements

Keyword	Description
`select_rect`	Select a rectangular area in a mesh. The command takes five arguments: 1. `x1`, x-coordinate of the lower left corner of the selected rectangle 2. `y1`, y-coordinate of the lower left corner of the selected rectangle 3. `x2`, x-coordinate of the upper right corner of the selected rectangle 4. `y2`, y-coordinate of the upper right corner of the selected rectangle 5. `mesh-name`, Name of the mesh
`select_circ`	Select a circular area in a mesh. The command takes four arguments: 1. `x_c`, center x-coordinate of the selected circle 2. `y_c`, center y-coordinate of the selected circle 3. `r`, radius of the selected circle 4. `mesh-name`, Name of the mesh

`select_poly` Select an area within a polygon-contour. In order to use this command you must first load a polygon from file with the `load_poly` command. The command takes one argument.

1. `mesh-name`, Name of the mesh

`load_poly` Load a polygon from file. This command is used in conjunction with the `select_poly` command. The command takes one argument.

1. `mesh-name`, Name of the mesh

Manually changing the mesh topology

Keyword	Description
`split_x`	Split selected elements in x-direction. If no elements are selected, all elements are split. As the topology of the mesh changed all selected nodes in the mesh are un-selected after this command. The command takes one argument

1. `mesh-name`, Name of the mesh

`split_y` Split selected elements in y-direction. If no elements are selected, all elements are split. As the topology of the mesh changed all selected nodes in the mesh are un-selected after this command. The command takes one argument

1. `mesh-name`, Name of the mesh

`split_xy` Split selected elements in both x- and y-direction. If no elements are selected, all elements are split. As the topology of the mesh changed all selected nodes in the mesh are un-selected after this command. The command takes one argument

1. `mesh-name`, Name of the mesh

`split_long` Split selected elements in thier longest direction. If no elements are selected, all elements are split. As the topology of the mesh changed all selected nodes in the mesh are un-selected after this command. The command takes one argument

1. `mesh-name`, Name of the mesh

`simplify` Attempt to simplify a mesh. If elements are selected they are un-selected as the topology of the mesh changed. The command takes one argument

1. `mesh-name`, Name of the mesh

Saving andloading meshes

Keyword	Description
`savemesh`	Save a mesh to file in the PVMOS binary format, so it can be loaded again at a later time (see the `loadmesh` command). The command takes two arguments.

1. `mesh-name`, Name of themesh to be saved.
2. `file-name`, filename to save the mesh to.

`loadmesh` Load a mesh saved to file in the PVMOS binary format (see the `savemesh` command). The command takes two arguments.

1. `file-name`, filename of the file containing the mesh data.
2. `mesh-name`, Name to assign to the loaded mesh

Element-wise export of data

Keyword	Description
`printmesh`	Export the mesh in a manner that is plottable with the gnuplot program (www.gnuplot.info/). The resulting plot draws the contour of each element in the mesh. The command takes two arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `file-name`, filename to save the data in.

The output file will contain coordinates in columns. For each element the file contains the coordinates of the lower left- and the upper right corners empty line:

```
x1    y1
x2    y1
x2    y2
x1    y2
<empty line>
```

`printconn` Print lateral connections in the electrodes in a format plottable with gnuplot (www.gnuplot.info/). When plotting the file (with vectors) a vector is drawn between the center of each element to the center of the adjacent elements to which it is connected. This routine may be useful when inspecting generated meshes. The command takes two arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `file-name`, filename to save the data in.

The output is coordinates in columns. For each element the file contains the following data where `xc`, `yc` is the center of the current element and `xca_i`, `yca_i` is the center coordinate of the i-th adjacent element:

```
xc yc xca_1 yca_1
xc yc xca_2 yca_2
...
```

`printarea` Print the geometry of the mesh which identifies each element and the area it belongs to. The fileformat is laid out such that it is plottable with a surface plot in gnuplot (www.gnuplot.info/). The command takes two arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `file-name`, filename to save the data in.

The output file will contain data in columns. The file contains coordinates, the element ID and the corresponding area ID. Note that the parameters for each area can be exported with the `printpars` command. For each element it plots 2 times 2 data lines with an empty line inbetween. Between the data of two elements are two empty lines. This file is formatted such that when plotted with "splot" in gnuplot you can plot a surface for each element in the mesh, which allows you to see the areas in the defined geometry. For each element the folowing data is printed to the file:

```
x1    y1    element-ID area-ID
x1    y2    element-ID area-ID
<empty line>
x2    y2    element-ID area-ID
x2    y1    element-ID area-ID
<empty line>
<empty line>
```

`printV` Print the front and back electrode element voltages for each stored solution. The output is formatted for gnuplot's splot command, such that a surface plot plots each element individually. The command takes two arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `file-name`, filename to save the data in.

The output file will contain data in columns. The file contains coordinates followed by the potential in the positive and negative electrodes for each solution.For each element it plots 2 times 2 data lines with an empty line inbetween. Between the data of two elements are two empty lines. This file is formatted such that when plotted with "splot" in gnuplot you can plot a surface for each element in the mesh. For each element the folowing data is printed to the file:

```
x1    y1    V_+^1 V_-^1 V_+^2 V_-^2 ...
x1    y2    V_+^1 V_-^1 V_+^2 V_-^2 ...
<empty line>
x2    y2    V_+^1 V_-^1 V_+^2 V_-^2 ...
x2    y1    V_+^1 V_-^1 V_+^2 V_-^2 ...
<empty line>
<empty line>
```

`printpar` Print a summary of the parameters per area, including both area-name and area-ID. The command takes two arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `file-name`, filename to save the data in.

`printmesh_sel` Same as `printmesh` except that it only exports a selected area specified by its lower left and upper right corners. The command takes six arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `x1`, x-coordinate of the lower left corner of the selected rectangle
3. `y1`, y-coordinate of the lower left corner of the selected rectangle
4. `x2`, x-coordinate of the upper right corner of the selected rectangle
5. `y2`, y-coordinate of the upper right corner of the selected rectangle
6. `file-name`, filename to save the data in.

`printconn_sel` Same as `printconn` except that it only exports a selected area specified by its lower left and upper right corners. The command takes six arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `x1`, x-coordinate of the lower left corner of the selected rectangle
3. `y1`, y-coordinate of the lower left corner of the selected rectangle
4. `x2`, x-coordinate of the upper right corner of the selected rectangle
5. `y2`, y-coordinate of the upper right corner of the selected rectangle
6. `file-name`, filename to save the data in.

`printarea_sel` Same as `printarea` except that it only exports a selected area specified by its lower left and upper right corners. The command takes six arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `x1`, x-coordinate of the lower left corner of the selected rectangle
3. `y1`, y-coordinate of the lower left corner of the selected rectangle
4. `x2`, x-coordinate of the upper right corner of the selected rectangle
5. `y2`, y-coordinate of the upper right corner of the selected rectangle
6. `file-name`, filename to save the data in.

`printV_sel` Same as `printV` except that it only exports a selected area specified by its lower left and upper right corners. The command takes six arhuments:

1. `mesh-name`, Name of the mesh mesh
2. `x1`, x-coordinate of the lower left corner of the selected rectangle
3. `y1`, y-coordinate of the lower left corner of the selected rectangle
4. `x2`, x-coordinate of the upper right corner of the selected rectangle
5. `y2`, y-coordinate of the upper right corner of the selected rectangle
6. `file-name`, filename to save the data in.

IV. EXAMPLE:MONOLITHICALLY SERIES CONNECTED CELLS AND A DEFECT

```
#############################################################################################
# This example demonstrates the simulation of a monolithically interconnected thin-film module
# The following meshes are created
# cell_a: mesh for the active part of a cell stripe (i.e., not including dead area)
# p1: mesh for the p1 laser line
# da: mesh for the area's between laser lines (for simplicity this mesh will be used
#               between p1 and p2 aswell as between p2 and p3)
# p2: mesh for the p2 laser line
# p3: mesh for the p3 laser line
# cp: mesh for a contact trip connected to the positive electrode
# cn: mesh for a contact strip to the negative electrode
#############################################################################################
# x1  y1 x2  y2 Nx Ny name
newmesh  0.0 0.0  0.5 30  50  30  cell_a
newmesh  0.0 0.0  8e-3 30  1  30  p1
newmesh  0.0 0.0  12e-3 30  1  30  da
newmesh  0.0 0.0  5e-3 30  1  30  p2
newmesh  0.0 0.0  5e-3 30  1  30  p3
newmesh  0.0 0.0  8e-3 30  1  30  cp
newmesh  0.0 0.0  8e-3 30  1  30  cn

#############################################################################################
# A mesh consists of elements. The elements in a mesh belong to areas. When a mesh is
```

```
# initialized, all elements are part of the samedefault area. All areas that are part of a mesh
# must have a name. The nameof the default area is the same as the name of the mesh when it is
# initialized. You can change the properties of an area by using "set" commands. You can also
# define new areas and assign elements to these new areas. More on that later. We first define
# the properties of the default areas for each mesh. To refer to a certain area you must specify
# the mesh and the area name like so: <mesh-name>.<area-name>
# Here we set the diode properties of the area cell_a in the mesh cell_a:
##############################################################################################
# mesh.area J0 n Jph Rs Rsh Temp Eg
set_1DJV  cell_a.cell_a  1e-8  1.5  0.03  1e-3  1e4  300  1.12
# here we set the sheet resistances for the same area, for the positive and negative electrodes
set_Rp cell_a.cell_a 20
set_Rn cell_a.cell_a 1

# here follow various areas for the various meshes
# set_R sets the resistance between front and back electrode, here we remove the contact
set_R  p1.p1    1e90
set_Rp p1.p1 1e9
set_Rn p1.p1 1
set_SplitX p1.p1 0

set_1DJV  da.da  1e-8  1.5  0.03  1e-3  1e4  300  1.12
set_Rp da.da 20
set_Rn da.da 1
set_SplitX da.da 0

set_R  p2.p2    1e-5
set_Rp p2.p2 20
set_Rn p2.p2 1
set_SplitX p2.p2 0

set_R  p3.p3    1e90
set_Rp p3.p3 1e-3
set_Rn p3.p3 1e9
set_SplitX p3.p3 0


##############################################################################################
# We have two contact strips. Within the strips we remove the solar cell (define it as a
# resistors with a very high resistance). Either the positive or the negative electrode is made
# very conductive, and connected to the positive applied voltage or 0V, respectively.
# we do not split elements in this mesh in the x direction.
##############################################################################################

set_R  cp.cp    1e90
set_Rp cp.cp 1e-3
set_Rpvp  cp.cp  1e-8
set_SplitX cp.cp 0

set_R  cn.cn    1e90
set_Rn cn.cn 0.1
set_Rpvn  cn.cn  1e-8
set_SplitX cn.cn 0

##############################################################################################
# Now we start to assemble the complete mesh out of the various parts. When joining meshes we
# will use the coordinate system of the first mesh. The coordinate system of the second mesh can
# be shifted by using an x and y offset value. Note that you have to take care not to make the
# meshes overlap I have not (yet) inplemented error checking for this! Here I will use
# jointmesh_h, which joins meshes horizontally and keeps track of the x-offset value for me
# (i.e. the x-offset valueis computed as the maximum x-value in the first mesh).
##############################################################################################
#  xoff y_off mesh1 mesh2 mesh_out
joinmesh_h  0.0  cell_a p1 cell_p1
```

```
# 0.5+0.008=0.508
joinmesh_h  0.0  cell_p1 da cell_p1da
# 0.508+0.012=0.52
joinmesh_h  0.0  cell_p1da p2 cell_p1dap2
# 0.52+0.005=0.525
joinmesh_h   0.0  cell_p1dap2 da cell_p1dap2da
# 0.525+0.012=0.537
joinmesh_h 0.0  cell_p1dap2da p3 cell_p1dap2dap3
# 0.537+0.005=0.542

#############################################################################################################
# Note that we now have many meshes defined:
# 1.  cell_a
# 2.  p1
# 3.  da
# 4.  p3
# 5.  p3
# 6.  cp
# 7.  cn
# 8.  cell_p1
# 9.  cell_p1da
# 10. cell_p1dap2
# 11. cell_p1dap2da
# 12. cell_p1dap2dap3
# The last mesh is cell stripe including dead area. We can easily series connect several cells
# by joining several instances of the last mesh in a row. Here we series connect five cells
#############################################################################################################
joinmesh_h 0.0  cell_p1dap2dap3 cell_p1dap2dap3 2cells
# 0.542+0.542=1.084
joinmesh_h 0.0  2cells cell_p1dap2dap3 3cells
# 1.084+0.542=1.626
joinmesh_h 0.0  3cells cell_p1dap2dap3 4cells
# 1.626+0.542=2.168
joinmesh_h 0.0  4cells cell_p1dap2dap3 5cells
# 2.168+0.542=2.710

#############################################################################################################
# Now we add the contacts. Our final mesh will be called: cells
#############################################################################################################
joinmesh_h 0.0  cp 5cells cp5cells
# 8e-3+2.710=2.718
joinmesh_h 0.0  cp5cells cn cells
# 2.718+8e-3=2.726



#############################################################################################################
# Now we have three perfectly homogeneous cells. That is boring so we add a shunt in the middle
# cell. First we create a new area. This should be part of the cells mesh. We can create a new
# area simply by starting to specify its properties, a new area will automatically be created.
#############################################################################################################
# set_JV cells.shunt shunt.dat
set_R  cells.shunt  1e-3
set_Rp cells.shunt 20
set_Rn cells.shunt 1

#############################################################################################################
# So far the creation of this area has no effect as no nodes are assigned to the area. However,
# before we assign nodes to this area we need a better resolution as the nodes in the cell area
# are now 0.01x1.0 cm^2, which is not sufficient to specify a small defect. I will add a defect
# at the coordinate (0.7,15). Refining the mesh will be done by selecting nodes and subsequently
# splitting the nodes. As the nodes are long in the y direction we first split the nodes only in
# the y-direction. The select_circ creates a circular selection, Only nodes whose center fall
# within the specified circle are selected. The circle is specified by its center coordinates
```

```
# (xc,yc) and the radius, r. The split commands automatically use the selected nodes. Beware
# that if no nodes are selected prior to giving a split command, all nodes in the mesh are split
#############################################################################################
#           xc  yc r mesh
select_circ 1.5 15 4 cells
split_y cells

select_circ 1.5 15 3 cells
split_y cells

select_circ 1.5 15 2 cells
split_y cells

select_circ 1.5 15 1 cells
split_y cells

select_circ 1.5 15 0.1 cells
split_y cells

select_circ 1.5 15 0.07 cells
split_y cells

select_circ 1.5 15 0.06 cells
split_y cells

#############################################################################################
# Note that I gradually reduced the radius of the circle to get a denser and senser mesh around
# the coordinate (0.7,15). The nodes at this coordinate are now about 0.01x0.0078 cm^2 large. As
# the nodes are now close enough to quadratic we now split the nodes three times in both x- and
# y- direction to get a satifactory resolution with nodes of about 10m x 10m. Note that in a
# regular, equidistant mesh such node sizes would result in meshes of almost 50 milion nodes
# where the mesh here is about 20000 nodes large.
#############################################################################################
select_circ 1.5 15 0.05 cells
split_xy cells


#############################################################################################
# Now we have at the location of the shunt nodes of about 0.001x0.001 cm^2, this suffices for
# a defect with a radius of about 0.01 cm and up. As we already defined the shunt properties
# of the area "shunt" we only need to assign nodes to this area. To this and we select the nodes
# and then assign them (note if you do not select nodes before you do this, all the nodes in the
# mesh will be assigned to the shunt area.
#############################################################################################
select_circ 1.5 15 0.04 cells
assign_properties cells.shunt



#############################################################################################
# Now we print some stuff so we can check whether we made no mistakes.
# printmesh prints the mesh (i.e. contour of each node)
# printpars prints the name, id number and parameters for each area in a mesh
# printarea prints the meah and adds element id and area id for each element
# printconn prints the connections between nodes (verctors from center to center of a node
# (bi-directional)
#############################################################################################
printmesh cells mesh_initial.dat
printpars cells pars.dat
printarea cells areas.dat
printconn cells conn.dat

#############################################################################################
# Finaly do some simulations, we start by adapting the mesh a few iterations (5). The adapttive
```

```
# solve command expects as arguments:
# 1: the mesh to adapt
# 2: the applied voltage
# 3: adaption factor, af,  (lower value means more nodes are adapted each iteration)
# 4: number of iterations for the mesh adaption
# We also output the refined mesh
#########################################################################################################
#  mesh V af iterations
adaptive_solve  cells  2.4  0.3  5
printmesh cells mesh_refined.dat

#########################################################################################################
# Now we simulate an IV curve, save it to IV.dat and plot the spatial potential distribution
# for the available simulation that is closest to 4.0 V
#########################################################################################################
solve cells -0.5 4.0 46
printIV cells IV.dat
printV cells checkpot.dat
surfVplot cells 0.008 14.0 2.718 16 300 400 4.0 thin_film_detail.dat
surfVplot cells 0.008 0 2.718 30 300 400 4.0 thin_film_wholedevice.dat
```