PyUedge

November 14, 2019

1 PyUedge Jupyter Notebook

1.0.1 This notebook is to help get users started using the Python version of UEDGE.

Start off by importing uedge. If installed you should see the CVS tag for your installed version. Note that because of the method used to bind the compiled fortran code to Python, the variables and functions contained within the packages is hidden. (To execute code blocks select and then type control-return)

```
In [1]: from uedge import *
        print(bbb.uedge_ver)
        print(dir(bbb))
        from case_setup import *          # sets up sizing variables and allocate space for case,
```

```
[b'$Name: V7_08_04 $                                                                ']
['__class__', '__delattr__', '__dir__', '__doc__', '__eq__', '__format__', '__ge__', '__getattri
```

If the previous cell fails uncomment the the following cell (remove the #) and run.

```
In [2]: ! #pip install uedge --upgrade
        ! pip install wurlitzer
        %load_ext wurlitzer
```

```
Requirement already satisfied: wurlitzer in /mfe/local/anaconda3_5.0.1/lib/python3.6/site-packag
```

As of this version most of PyUedge cases are translated from ones originally run with the Basis version of the code. Because of this it is worth a short description of issues encountered doing this conversion.

There are a few differences between the Python and Basis environments the user must be aware of.

1. Python variables must reference the package while Basis has a priority based namespace. In Basis uedge_ver can be used without specifying “bbb” while in Python the package is required as in bbb.uedge_ver. To convert an existing Basis script start by using the application “**bas2py** basis_file python_file” to get started. This will add the package references for you.

2. Basis arrays may have variable index start or stop values. Basis variables start at 1 and use parentheses; Python variables always start at 0 and uses square brackets. If values assigned are indices then they should probably follow Fortran rules: start with 1 and row major.
   - nxleg(1,1)=4 # basis
     - com.nxleg[1,]=[12,10] # python
   - kelighi(igsp) = 5.e-16 # basis
     - bbb.kelighi[bbb.igsp-1] = 5.e-16 # python
   - Basis initialization with do-loop (Note that index ranges are both ends inclusive. (1:3) is 1,2,3)

```
do ijk = nhsp+1, nhsp+6
 difniv(0:ny+1,ijk) = pd
enddo
```

   - Python initialization with for-loop (Note that index ranges are only left side inclusive. [1:4] is 1,2,3)

```
for ijk in arange(com.nhsp,com.nhsp+6):
bbb.difniv[0:com.ny+2,ijk] = pd
```

2 Save/Restore

2.1 PDB (PFB) Files - The Basis save file format.

This requires that the pact python module has been installed . This is supported on a limited number of systems. The GA cluster Iris and the LLNL cluster Singe are the main platforms that support this.

```
In [21]: try:
             from uedge.pdb_restore import * # if this fails pact is likely not installed
             pdb_restore('d3d.pdb')          # variable tgs is not included, no error

         except:
             print "pact not installed?"
```

```
      File "<ipython-input-21-657d738a1c0a>", line 6
    print "pact not installed?"
                              ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("pact not installed?
```

2.2 HDF5 Files - The Python save file format.

This requires that the h5py python module has been installed. This is part of the Anaconda distribution.

```
In [3]: try:
            from uedge.hdf5 import *         # if this fails h5py is likely not installed
            hdf5_restore('d3d.hdf5')         # variable tgs is not included and don't worry arout

        except:
            print("h5py not installed?")
```

```
Old style hdf5 file
```

2.3 Run a case

- Basis - exmain
- Python - bbb.exmain()

```
In [4]: from uedge import *
        bbb.exmain()    #
        print("Usual iteration output should be in terminal window running Jupyter Notebook")
        print("nksol ---  iterm = ",bbb.iterm)
```

```
 UEDGE $Name: V7_08_04 $
 Wrote file "gridue" with runid:    EFITD    09/07/90      # 66832 ,2384ms

 ***** Grid generation has been completed
  Updating Jacobian, npe =                      1
 iter=    0 fnrm=     0.8836028059824554     nfe=      1
  Updating Jacobian, npe =                      2
 iter=    1 fnrm=     0.7830357704942191     nfe=     27
 iter=    2 fnrm=     0.6987551326282868     nfe=     57
 iter=    3 fnrm=     0.6451626053526677     nfe=     92
 iter=    4 fnrm=     0.5564541870508536     nfe=    129
 iter=    5 fnrm=     0.4650795163520757     nfe=    166
  Updating Jacobian, npe =                      3
 iter=    6 fnrm=     0.4047963244535194     nfe=    174
 iter=    7 fnrm=     0.3101544995105658     nfe=    184
 iter=    8 fnrm=     0.1425256655297171     nfe=    196
 iter=    9 fnrm=     0.9692888388847173E-01 nfe=    233
 iter=   10 fnrm=     0.6472862782303432E-01 nfe=    268
  Updating Jacobian, npe =                      4
 iter=   11 fnrm=     0.5981843989990394E-01 nfe=    273
 iUsual iteration output should be in terminal window running Jupyter Notebook
nksol ---  iterm =  4
ter=   12 fnrm=     0.2510534292033969E-01 nfe=    283
 iter=   13 fnrm=     0.2493542404082094E-01 nfe=    307
 iter=   14 fnrm=     0.2597513210544682E-01 nfe=    324
 iter=   15 fnrm=     0.2309583170038645E-01 nfe=    341
  Updating Jacobian, npe =                      5
```

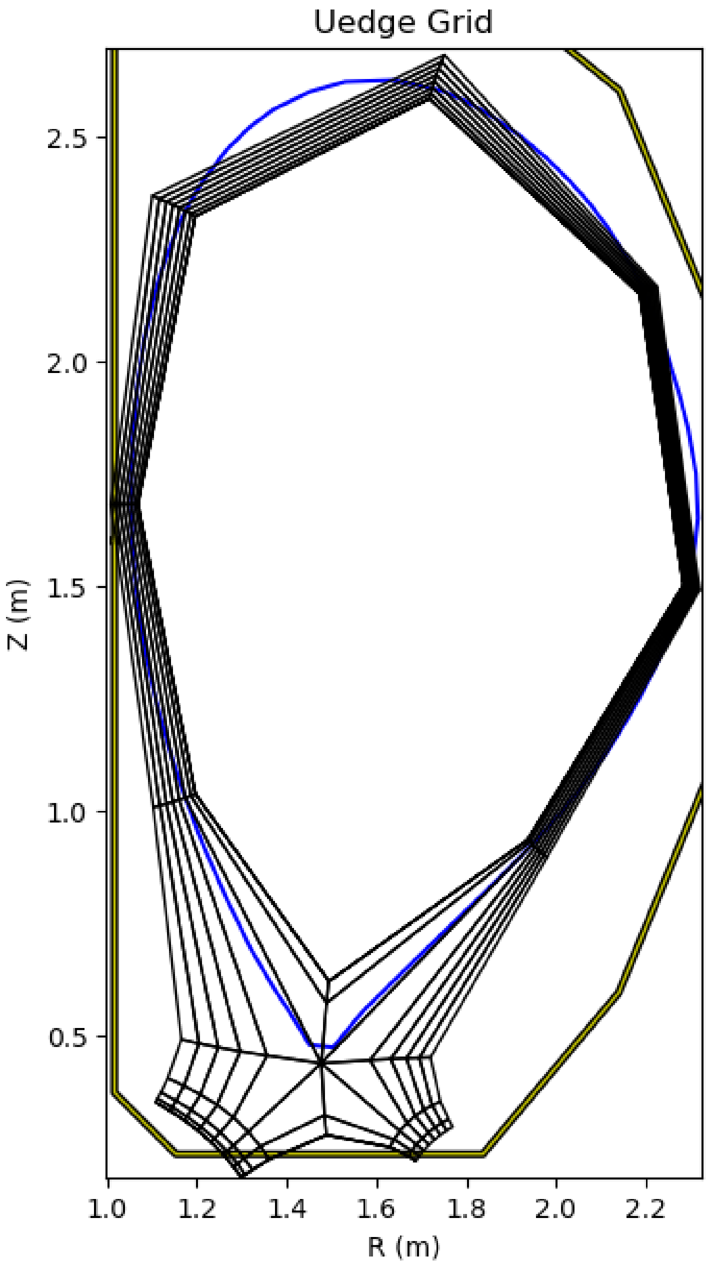
```
iter=   16 fnrm=     0.2164407857359585E-01 nfe=    350
iter=   17 fnrm=     0.1870635848416796E-01 nfe=    364
iter=   18 fnrm=     0.1542173728330657E-01 nfe=    375
iter=   19 fnrm=     0.6431384479753541E-02 nfe=    384
iter=   20 fnrm=     0.6303720286031002E-02 nfe=    401
 Updating Jacobian, npe =                      6
iter=   21 fnrm=     0.6167381370554297E-02 nfe=    413
iter=   22 fnrm=     0.6402013634952513E-02 nfe=    428
iter=   23 fnrm=     0.6261606570971315E-02 nfe=    440
iter=   24 fnrm=     0.6148186973454837E-02 nfe=    452
iter=   25 fnrm=     0.6429265295134812E-02 nfe=    467
 Updating Jacobian, npe =                      7
iter=   26 fnrm=     0.6317578573471724E-02 nfe=    479
iter=   27 fnrm=     0.6118824841501220E-02 nfe=    491
iter=   28 fnrm=     0.6488847380729581E-02 nfe=    506
iter=   29 fnrm=     0.6150420807580852E-02 nfe=    518
iter=   30 fnrm=     0.6586300364205886E-02 nfe=    530


nksol ---  iterm = 4.
           the maximum allowable number of nonlinear
           iterations has been reached.
Interpolants created; mype =                  -1
```

3 Plotting

3.0.1 Plot the Mesh

```
In [5]: from uedge.uedgeplots import *
        import warnings
        warnings.filterwarnings("ignore") # ignore warnings for this notebook
        # next line is only for this notebook, do not include this in your python files
        %matplotlib inline
        print(plotmesh.__doc__)
        plotmesh()
```

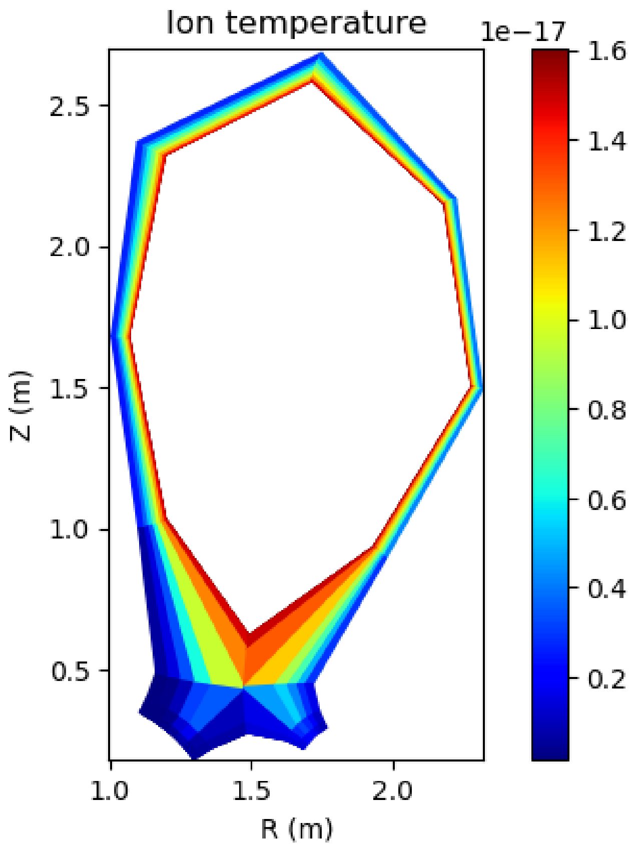
```
  plotmesh(ixmin=<int>,ixmax=<int>,iymin=<int>,iymax=<int>
           title=<string>,r_min=<val>,r_max=<val>,z_min=<val>,z_max=<val>,
           block=<True|False>)
     where ixmin, ixmax, iymin, and iymax are integer variables or
     expressions used to plot a portion of the grid. title is used as
     both the title and the figure name. Block default is True.

     The plot axis limits may be specified with r_rmin,r_max,z_min,z_max.
```



3.0.2 Plot a 2-D (mesh size) Quantity

```
In [6]: from uedge.uedgeplots import *
        import warnings
        warnings.filterwarnings("ignore") # ignore warnings for this notebook
        # next line is only for this notebook, do not include this in your python files
        %matplotlib inline
        print(plotmeshval.__doc__)
        print(bbb.tis.shape)
        plotmeshval(bbb.tis,title='Ion temperature')
```

```
   plotmeshval(val,ixmin=<int>,ixmax=<int>,iymin=<int>,iymax=<int>
         title=<string>,units=<string>,block=<True|False>)
     Display 2-D quantity using polyfill.
     where ixmin, ixmax, iymin, and iymax are integer variables or
     expressions used to plot a portion of the grid. title is used as
     both the title and the figure name. Units are displayed in the
     side colorbar. Block default is True.

     The plot axis limits may be specified with r_rmin,r_max,z_min,z_max.

(18, 10)
```

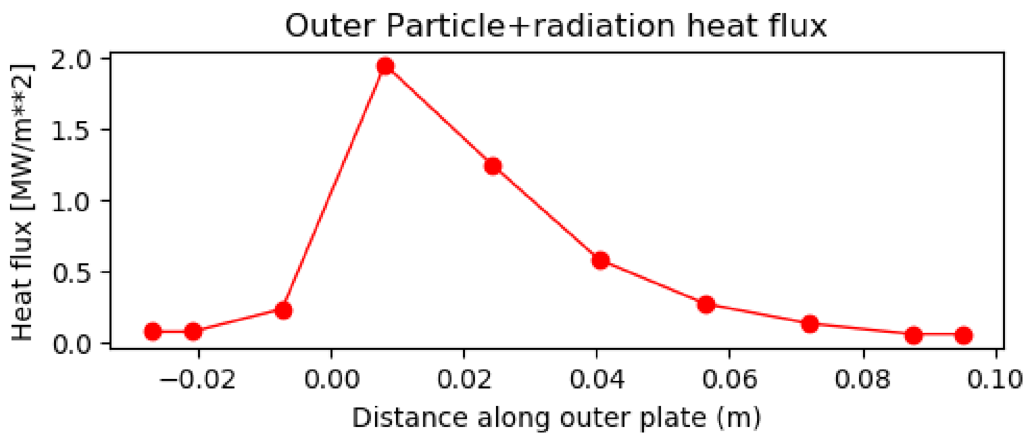


3.0.3 Plot a 1-D Profile

```
In [7]: from uedge.uedgeplots import *
        import warnings
```

```
warnings.filterwarnings("ignore") # ignore warnings for this notebook
# next line is only for this notebook, do not include this in your python files
%matplotlib inline
bbb.plateflux()
print(profile.__doc__)
#
#  Run in python these two traces will be on the same plot
#
profile(com.yyrb,1.e-6*(bbb.sdtrb+bbb.sdrrb),
        title="Outer Particle+radiation heat flux",
        xlabel="Distance along outer plate (m)",
        ylabel="Heat flux [MW/m**2]",
        figsize=(6,2),style='ro-'
    )
```

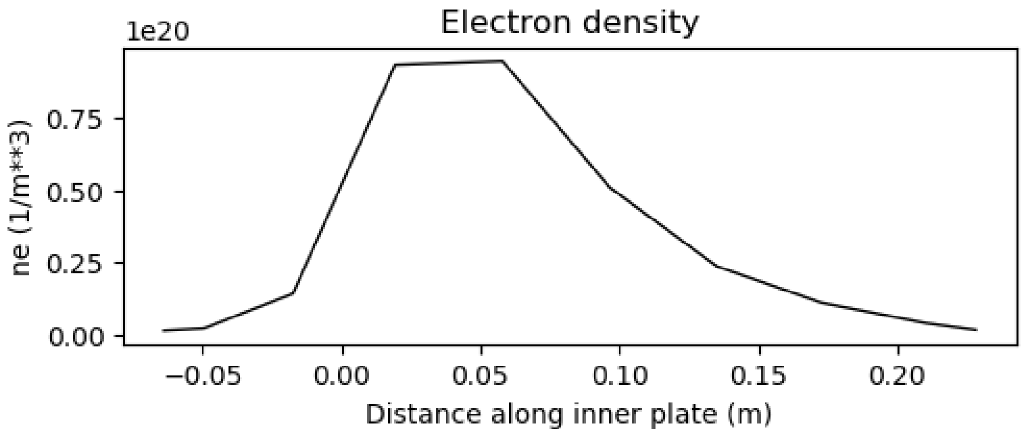
```
profile(xval,yval,title=<None>,style=<None>,linewidth=<None>,xlabel=<None>,ylabel=<None>)
   title is used as both the title and the figure name.
   Interactive is turned on so subsequent calls go to the same plot
   Style encoded color, line, and marker.  See matplotlib documention.
   examples: black solid line  - style='k-'
             red circle marks  - style='ro'
             green x marks and dotted line - style='gx--'
```



3.0.4 Misc Plots - uncomment and run cell

```
In [8]: #*******************************************************************
        #For details of generic plot options, see https://matplotlib.org/
        #*******************************************************************
        #
```

```
#------------------------------------------------------------------------
#First, be sure you have loaded uedgeplots within an ipython session:
#------------------------------------------------------------------------

from uedge.uedgeplots import *
import warnings
warnings.filterwarnings("ignore") # ignore warnings for this notebook
# next line is only for this notebook, do not include this in your python files
%matplotlib inline

#=======
#Plotting profiles on outer divertor plate:
#=======

#profile(com.yyrb,bbb.ne[com.nx,],title="Electron density", xlabel="Distance along outer

#profile(com.yyrb,bbb.ng[com.nx,],title="Hydrogen atom density", xlabel="Distance along

#profile(com.yyrb,bbb.te[com.nx,]/bbb.ev,title="Electron temperature", xlabel="Distance

#profile(com.yyrb,bbb.ti[com.nx,]/bbb.ev,title="Ion temperature", xlabel="Distance along

#profile(com.yyrb,bbb.feex[com.nx,],title="Electron thermal heat flux *area", xlabel="Di

#profile(com.yyrb,bbb.feix[com.nx,],title="Ion/atom thermal heat flux *area", xlabel="Di

# To plot the total heat flux on the outer divertor:
#bbb.plateflux()
#profile(com.yyrb,1.e-6*(bbb.sdtrb+bbb.sdrrb),title="Particle+radiation heat flux", xlab

#=======
#Plotting profiles on inner divertor plate:
#=======
profile(com.yylb,bbb.ne[0,],title="Electron density", xlabel="Distance along inner plate

#profile(com.yylb,bbb.ng[0,],title="Hydrogen atom density", xlabel="Distance along inner

#profile(com.yylb,bbb.te[0,]/bbb.ev,title="Electron temperature", xlabel="Distance along

#profile(com.yylb,bbb.ti[0,]/bbb.ev,title="Ion temperature", xlabel="Distance along inne

#profile(com.yylb,-bbb.feex[0,],title="Electron thermal heat flux *area", xlabel="Distan

#profile(com.yylb,-bbb.feix[0,],title="Ion/atom thermal heat flux *area", xlabel="Distan

# To plot the total heat flux on the inner divertor:
#bbb.plateflux()
#profile(com.yylb,1.e-6*(bbb.sdtlb+bbb.sdrlb),title="Particle+radiation heat flux", xlab
#=======
#Plotting midplane profiles:
#=======

#profile(com.yylb,bbb.ne[bbb.ixmp,],title="Electron density at midplane", xlabel="Distan

#profile(com.yylb,bbb.ng[bbb.ixmp,],title="Hydrogen atom density at midplane", xlabel="D

#profile(com.yylb,bbb.te[bbb.ixmp,]/bbb.ev,title="Electron temperature at midplane", xla

#profile(com.yylb,bbb.ti[bbb.ixmp,]/bbb.ev,title="Ion temperature at midplane", xlabel="
```



```
In [ ]:
```