How to use MAFOT

A.Wingen 6/3/15

Contents:

0. Setup (one time only)

Check the Readme.txt in the MAFOT folder. Add to your .bash_profile:

```
PATH=$PATH:/home/wingen/bin
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/home/wingen/lib/64:/home/wingen/lib/64/blitz/lib
```

or similar, as well as the LD_lib paths to your openMPI, M3D-C1 and HDF5 libs

1. Tools in MAFOT

plot - **returns the Poincaré poloidal cross-section**
For a given toroidal angle this tool traces field-lines (particle drift orbits) and simulates the Poincaré plot.

fix - **returns periodic fixed points (hyperbolical and elliptical)**
It uses a 2D Newton method to search for periodic fixed points. Note that the Newton method requires an initial guess, given by a rectangular search grid, but the algorithm can leave the grid area. For periodicity = 1 the search stops after it finds the first point.

man - **returns the stable and unstable manifolds related to a hyperbolic fixed point**
It needs the fixed point found by "fix". It uses a step size controlled algorithm to trace the manifold until it intersects three times with the vessel wall. Note that if the manifold is too long for your purposes, just truncate the file.

laminar - **returns the connection length and penetration depth in a poloidal cross-section**
It calculates the data on a rectangular grid, suitable for 2D contour plots. Note that if the field line does not intersect with the wall after the given limit of toroidal iterations, the integration of that field line stops and the length and minimum penetration up to this point is returned.

foot - **returns the connection length and penetration depth footprint on the divertor target**
same as laminar, but on the vessel wall. As coordinates it uses a length parameter along the wall (t, y-axis) and the toroidal angle (phi, x-axis). The plotting script d3dplot.py automatically converts t back into a meaningful length.
Tip: since one rarely knows the proper limits in t, make a very low-resolution run along the entire t-range, plot it with the physical = 0 flag (see options in d3dplot.py), locate the footprint and thereby the proper t-limits. Then do the full resolution run.

2. Required files

MAFOT runs are controlled by a control file. Their filenames always start with an underscore '_' and end with '.dat'. Example: '_plot.dat'. The file is generated by the GUI.

MAFOT always needs a g-file. If the code exits directly after launching, check the shot, time and path to the g-file. If the problem persist, check the header line in the g-file, sometimes (rarely) is not standard. This can cause problems. The line should look like this:

```
  EFITD    03/13/2012    #148712  4101ms           3 129 129
```

MAFOT further needs the following files in its current working directory, depending on the *Field* you want to run:

Field:	Files:
g-file Vacuum	**diiidsup.in** (or itersup.in or nstxsup.in): containing the perturbation-coil currents. There are examples for every machine in the MAFOT/example folder.
M3D-C1	**m3dc1sup.in**: a list of the following structure `path/to/files/C1.h5  <perturbation scaling factor>` You can list up to 10 C1.h5 files, each in a new line. MAFOT will combine the perturbations from all using the provided scaling factor
VMEC	**wout.nc** and **xpand.dat**: the first is the VMEC wout-file, the second is the output of *xpand*.
SIESTA	**siesta.dat**: the file with the B-field on a 3D rectangular grid, that Siesta optionally returns. Note that SIESTA only works inside the Siesta last closed flux surface so far.

3. GUI

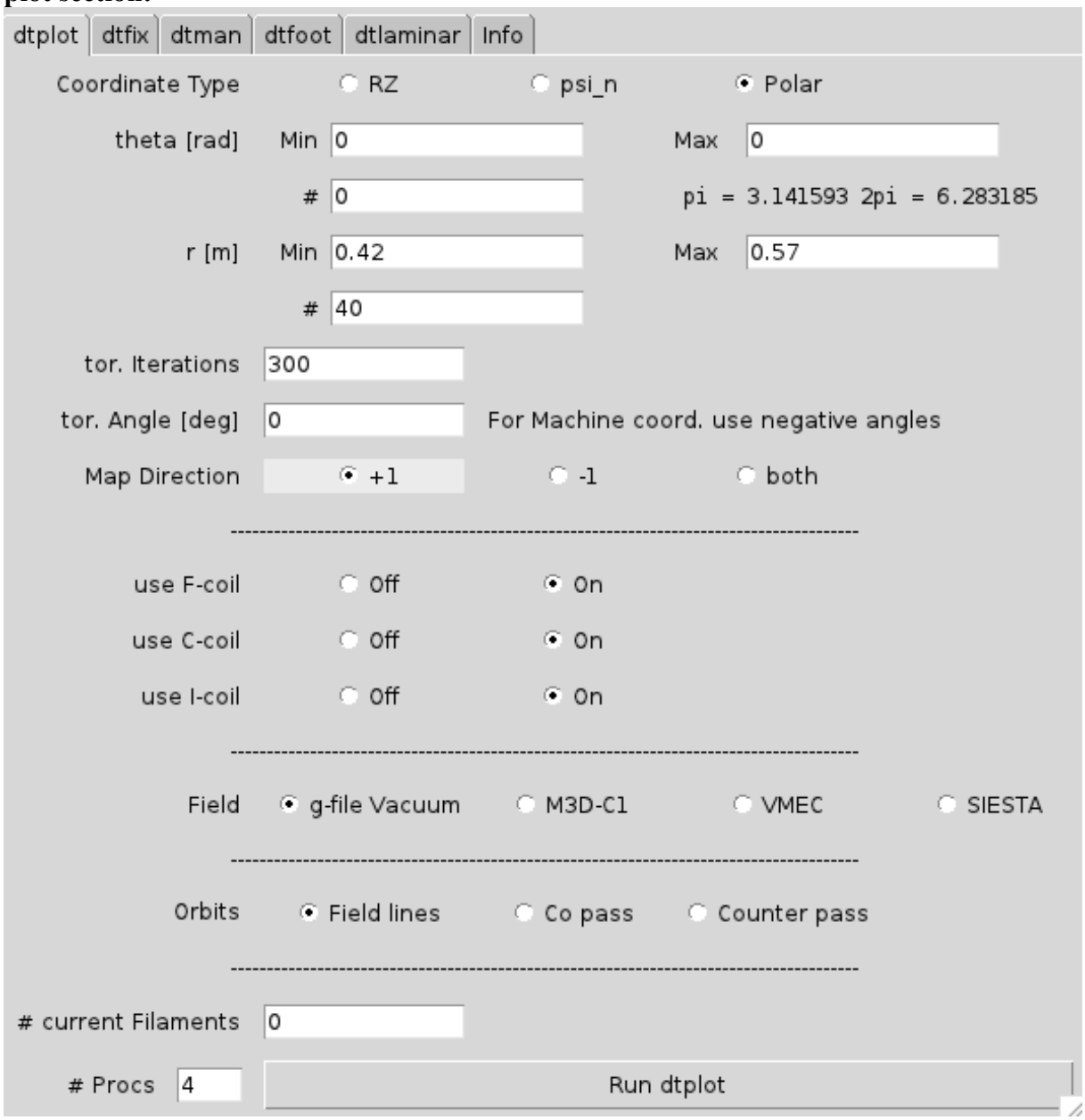
Launch the GUI by calling (the path to my bin might be different):

via Python	or	via binary (only where available)
python /home/wingen/bin/mafot_gui.py		/home/wingen/bin/mafot_gui

Common section:

Select a machine.
Enter a shot, time (must match the standard g-file name) and path for the g-file
File Tag is an arbitrary string, appended to all output files
Working Dir selects where to run MAFOT. The GUI reads already available control files to provide the last settings you used. To prevent that behavior, uncheck the *reload settings* box before changing *Working Dir*.

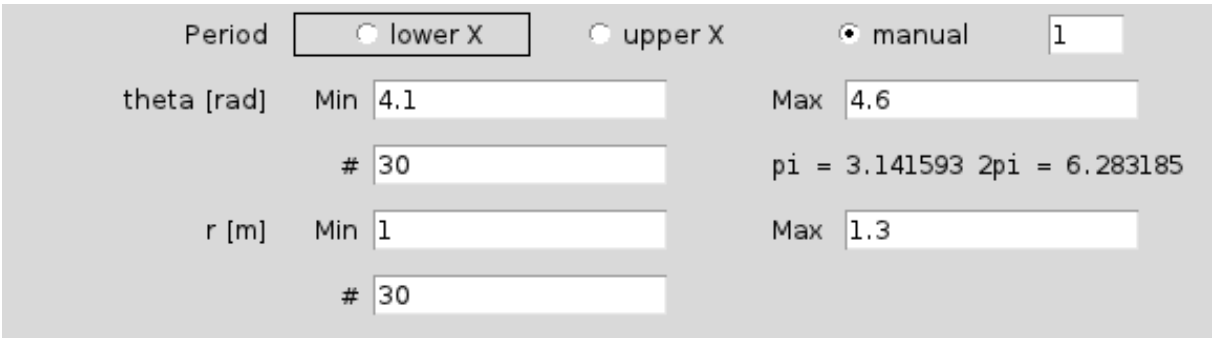
plot section:



Select your coordinate system: (R,Z), (theta, psi_n), or (theta, r); theta is the geometric poloidal angle
Select the ranges and number of grid points (#), values of pi and 2*pi are provided for copy/paste.
tor. Iterations: number of toroidal turns
tor Angle: mathematically oriented (ccw) toroidal angle phi; use negative for DIII-D machine angles
Map Direction: beginning from initial condition, integrate field line (orbit) in this phi direction
use coils: turn on/off perturbations
Field: where to get the B-field from
Orbits: field lines or guiding center drift particles; particles can move in either direction along their orbit. *Co pass*: with B_phi, *Counter pass*: against B-phi direction Note, that switching *Map Direction* flips *Co* and *Counter*.
 # current Filaments: number of arbitrary line currents to add as an additional perturbation. Requires additional files!

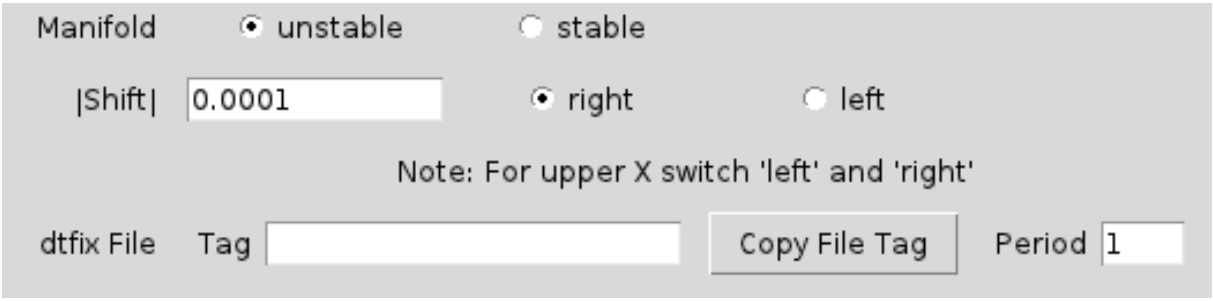
Procs: if tool is parallel, then you can give the number of openMPI processes here
Run Button: generates control file and launches MAFOT

fix section:



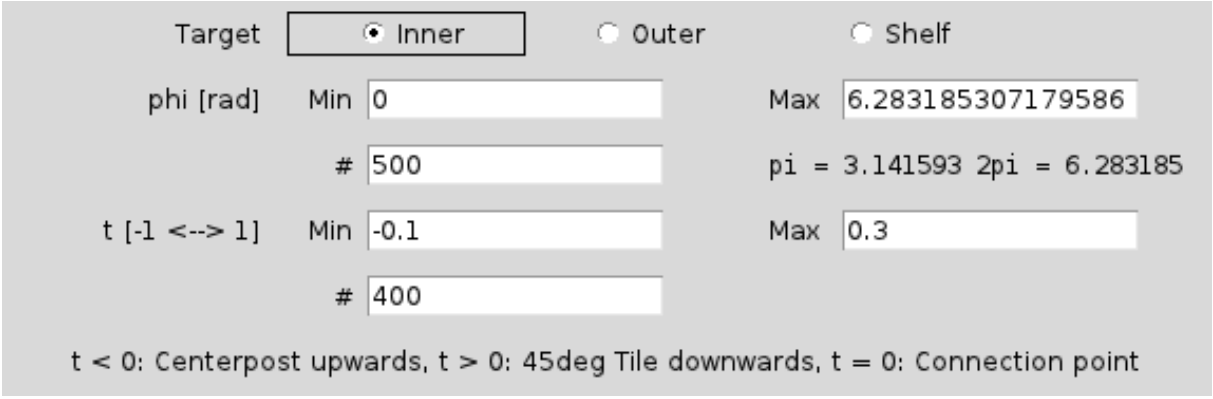
Select periodicity and search grid. Defaults for lower and upper X-point are provided. The rest is same as plot section.

man section:



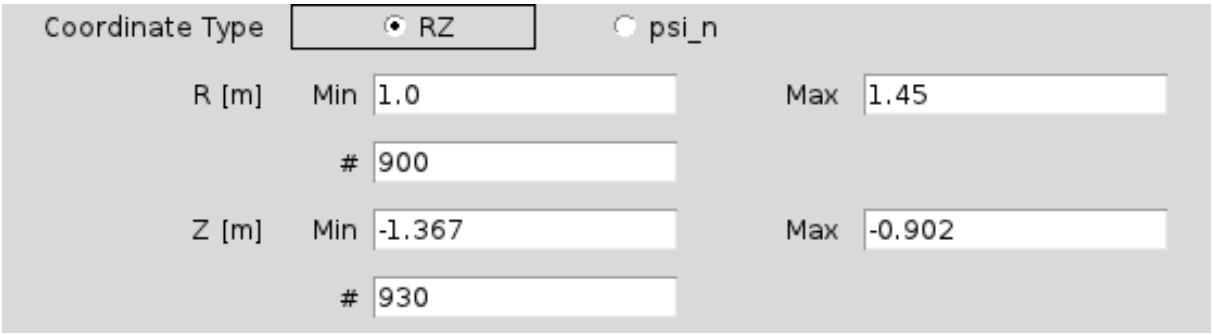
Select the type of manifold.
Shift: a guess of how much to move away from X-point. Default usually works fine. Smaller *Shift* makes manifold more accurate, but computation takes longer.
right/left: direction of shift; this selects which branch of the manifold you get. For X-point, one branch goes straight to the wall, while the other goes around the plasma; you want the latter.
Tag: man needs the result of fix. Give the *File Tag* you used for fix. The *copy* button fills in the tag that is currently in the common section.
Period: fixed point periodicity
The rest is same as plot section.

foot section:



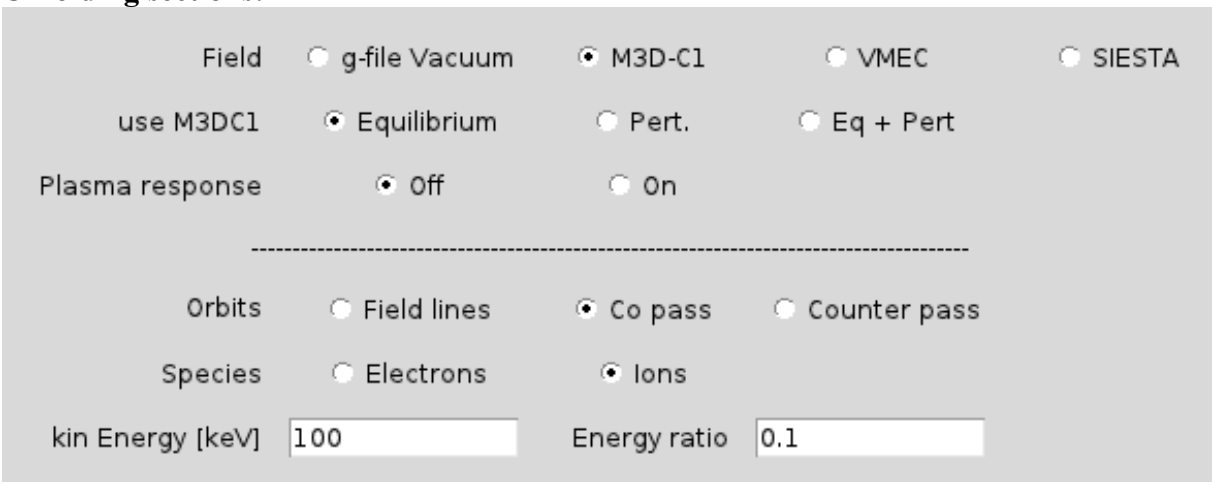
Select the target, meaningful defaults are provided
Select the grid range. Remember that t is a dimensionless length parameter along the wall. Use the tip given in Sec. 1. The text gives some idea of where t is at the wall.
The rest is same as plot section.

laminar section:



Select your coordinate system: (R,Z) or (theta, psi_n) and grid range.
The rest is same as plot section.

Unfolding sections:



Sometimes new options unfold and become available, depending on your selections. E.g.:
use M3D-C1: *Equilibrium* uses only the axisymmetric part in C1.h5, *Pert.* uses the g-file equilibrium + the perturbation from C1.h5, *Eq + Pert* uses equilibrium and perturbation from C1.h5 (Typically *Eq* is very similar to g-file, but note, that it is a recalculation of the g-file, done by M3D-C1)
Plasma response: turn on/off effects of plasma response. Note, that the off case should give the same result than g-file Vacuum with coils on.
If you *use M3D-C1*: *Pert.* or *Eq+Pert*, the *coils* will be set to off, regardless of your selection in the coils section

Orbits: if you use particles, then specify
Species, the *kinetic Energy* of particles and the *Energy ratio* of how much energy goes into translation vs. gyration.

4. Control file

All tools need a control file, generated by the GUI. Here is an example:

```
# Parameterfile for DIII-D Programs
# Shot: 129194 Time: 3000ms
# Path: /u/wingen/d3d/gfiles/
free_Parameter= 500
itt= 100
psimin= 0.8
psimax= 1.1
thmin= 0
thmax= 0
N= 31
phistart(deg)= 0
```

```
MapDirection= 1
PlasmaResponse(0=no,1=yes)=     1
Field(-1=M3D-C1_off,0=Eq,1=I-coil,2=both)=      -1
target(0=cp,1=inner,2=outer,3=shelf)=  1
createPoints(3=set,4=random,2=target)= 0
useFcoil(0=no,1=yes)= 1
useCcoil(0=no,1=yes)= 1
useIcoil(0=no,1=yes)= 1
ParticleDirection(1=co-pass,-1=count-pass,0=field-lines)= 0
ParticleCharge(-1=electrons,>=1=ions)= 1
Ekin[keV]= 100
lambda= 0.1
useFilament(0=no)= 0
pi= 3.141592653589793
2*pi= 6.283185307179586
```

You can at any point skip the GUI and edit the control file by hand. Then you have to know what you are doing! The name of any parameter can be arbitrary as long as no spaces are used. The name has no influence on the way the data is read. The data is read by its position in the file.
After changing the file, you can launch the code by hand via calls like
#> dtfix _fix.dat 1 arbitrary_tag
or
#> mpirun –np 4 dtlaminar_mpi _lam.dat arbitary_tag

5. Outputfile structure

All output files have a header which includes all relevant parameters for the respective run. Example:

```
# dtplot
#-------------------------------------------------
### Parameterfile: _plot.dat
#-------------------------------------------------
### Switches:
# Target (0=vertical, 1=45°, 2=horizontal, 3=shelf): 1
# Create Points (0=grid, 1=random, 2=target): 0
#-------------------------------------------------
### Global Parameters:
# Steps till Output (ilt): 360
# Step size (dpinit): 1
# Boundary Rmin: 1.016
# Boundary Rmax: 2.4
# Boundary Zmin: -1.367
# Boundary Zmax: 1.36
# Magnetic Axis: R0: 1.757
# Magnetic Axis: Z0: -0.004
#-------------------------------------------------
### additional Parameters:
# Max. Iterations: 300
# Points: 100
# rmin: 0.46
# rmax: 0.54
# thmin: 0
# thmax: 0
# phistart: 225
# MapDirection: 1
#-------------------------------------------------
### Data:
# theta[rad] r[m] phi[deg] psi
#
2.783539745069958 0.5935466234728225 585 0.7916423537837542
4.551706379276331 0.8405327650042975 945 0.790854672604256
```

```
2.144874365875646 0.7848006675379068 1305 0.7938579085980841
4.139298920881824 0.813733863778083 1665 0.7958004296252812
```

All comment lines start with '#', the default comment character of gnuplot. The data in the columns is double precision and the columns are separated by tabs. The kind of data is named in the last header line. In this example the first column is the poloidal angle in rad, the second is the minor radius with respect to the magnetic axis, the third is the toroidal angle in degrees and the last is the normalized flux.

All tools return log files, which store the screen output as well as give error messages. After your run terminates normally you can safely delete the log files. For openMPI codes, the *Master* log file protocols the communication and provides progress information; just look at its tail.

6. Xpand

xpand - **Calculates the B-field outside of the VMEC last closed flux surface**
It runs separate from all other MAFOT tools and does not need a control file or a g-file. It only needs a VMEC *wout*- and a *points*-file, which contains the list of points where you want to know the B-field

In the MAFOT/python folder there is a Python script that creates the points-file suitable for xpand and necessary for Fields = VMEC. But you can always make one yourself.
A points file looks like:

```
# 128x128x48 points 786432
1.0    0.0    -1.5
...
```

The first column is R[m], the second is phi[rad] and the third is Z[m]. R varies first, then Z, then phi. The first header line is just FYI and has here NRxNZxNphi and the total number of points (You need/are only allowed this one line of comments). xpand is parallel, so it always needs to be launched by mpirun.

To run xpand, look at the shell help by:
#> mpirun -n 4 xpand_mpi -h

The output is like:

```
# Xpand results from VMEC file wout.nc
# 16384 points
# R[m]         phi[rad]      Z[m]          BR[T]         Bphi[T]       BZ[T]
1      0      -1.5   0.2000096130793928  -3.329098836152724  -0.01426245230691l6
...
```

7. Python tools

Besides the GUI there is:

d3dplot.py - Can plot the laminar and foot MAFOT outputs.
use_xpand.py - Provides a points file for xpand, loads the output, plots the output a and can "heal" bad points

How to run:
#> python d3dplot.py -h show the shell help. There are many options. Just try it out.

use_xpand requires a python or ipython session. You have to import it, initialize the class and go from there. See the comments in the file.