How to use d3d-dt programs

A.Wingen 6/28/13

Contents:

1. Parameter file
2. Outputfile structure
3. Programs
3.1 dtplot
3.2 dtfix
3.3 dtman
3.4 dtfoot_mpi
3.5 dtlaminar_mpi

1. Parameter files

All programs need a diiidsup.in file, a g-file and the following parameter file:

```
# 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
```

Their filenames always start with an underscore '_' and end with '.dat'. Example: '_plot.dat'
The first line is just a comment.
In the second line the shot and the time have to be specified.
The third line is optional and sets the path to the g-file. The path needs to end with a forward slash (/).

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 data list.
The name is followed by a tab and the parameter value:

- free_Parameter
  it is not commonly defined and is used in different ways by the different programs. See the respective program for details.
- itt
  Number of toroidal iterations to perform by the program
- psimin, psimax, thmin, thmax
  Range of initial conditions in *psi* and *theta*. The latter has to be between *0* and *2*pi*. For CreatePoints=2 case: *r = t* and *theta = phi* is used
- N
  Number of initial conditions
- phistart
  toroidal angle in degrees *(0°..360°)* for the Poincaré section
- MapDirection: +1 or -1 or 0
  iteration in positive/negative phi direction or both
- PlasmaResponse: 0=no, 1=yes
  use vacuum or plasma response from M3D-C1 (Nate Ferraros's MHD code)
- Field: -1=M3D-C1 off, 0=Eq, 1=I-coil, 2=both
  which part of M3D-C1 output is used, -1 is the default to not include any M3D-C1
- target
  specifies the target plate to choose the initial conditions on. Only necessary if CreatePoints = 2 is chosen. For details see d3dfoot.
- _CreatePoins
  Choose method to get initial conditions:
  0 = Points are taken from a regular grid specified by min, max and N (sqrt(N) points in each direction respectively).
  1 = random generator chooses N initial conditions in grid range
  2 = initial conditions are chosen on target plate from a regular grid
  3 = Points are taken from a regular grid specified by psimin, psimax, thmin, thmax
  4 = random generator chooses N initial conditions in psi and theta
- _useFcoil, useCcoil, useIcoil
  specifies, whether to use these coils (=1) or not (=0)
- _ParticleDirection, ParticleCharge
  The code includes particle-drift effects. For field lines only, use ParticleDirection=0. ParticleCharge has no effects on field lines
- _Ekin, lambda
  kinetic energie of particles and radial part of energy (in percent, only estimate) **no effect on field lines**
- _useFilament
  includes current filaments. 0 = none, >0 = all; see filament section for more detail
- _pi, 2*pi are not read by any program or necessary at all, they are just helpfull to set e.g. thmax by copy and paste

2. 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 precission 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.

3. Programs

3.1 dtplot

```
// Program calculates Poincaré Plot for D3D
// Fortran Subroutines for Perturbation are used
//
// Input: 1: Parameterfile 2: praefix(optional)
// Output: Poincaré field line data file
// log-file
```

The program needs the name of the parameter file as an input and gives the opportunity to use some arbitrary string (called praefix here) to be added to the output filename. It gives back an ASCII file with the data and a log file, which includes various informations like startup parameters, progress reports during runtime and error messages.
Example:
> dtplot _plot.dat tryit
Output:
plot_tryit.dat and log_dtplot_tryit.dat (log-file)
Exceptions:
free_Parameter = Ntheta,
so Npsi = N/Ntheta
rmin, rmax <--> psimin, psimax for createPoints = 0 or 1 <--> 3 or 4

3.2 dtfix

```
// Program searches for periodic fixed points
// Fortran Subroutines for Perturbation are used
//
// Input: 1: Parameterfile 2: period of fixed point
// 3: praefix (optional)
// Output: fixed points
// log-file
```

Similar input as dtplot. But the second input is now the period of the fixed point, that is searched for.
Example:
> dtfix fix.dat 1 x_point
Output:
fix_1_all.dat and log_dtfix_1_all.dat (log-file)
If the period is set to 1, the program assumes that the main x-point is the target, so it stops after finding one fixed point. If period >1 it runs though all initial conditions and returns all fixed points it finds (Usually the same points over and over again -> pick the ones you need manually)
Exceptions:
free_Parameter, itt, target and CreatePoints are **not** used in this program

3.3 dtman

```
// Program calculates unstable manifold of a hyp. fixed point for D3D
// For stable manifold use reverse Integration (see MapDirection)
// For left or right hand-sided set sign of 'verschieb' in Parameterfile to
// - or + respectively
// Fortran Subroutines for Perturbation are used
//
// Input: 1: Parameterfile 2: File with fixed points
// 3:praefix(optional)
// fixed points have to be in toroidal coordinates, not cylindrical!!!
// Output: one file for each of the fixed points specified in the input-
// file, giving the respective manifold
// log-file
```

dtman reads the output file of dtfix and calculates a manifold for each of the fixed points specified in the dtfix output file.
Example:
> dtman _fix.dat dtfix_1_ x_point.dat a_manifold

Output:
dtman_unstr1_0_ a_manifold.dat and log_dtman_unstr_ a_manifold.dat (log-file)

The output filename is set by the following rules:
unst = unstable manifold, st = stable
r = right-hand sided, l = left
1 = period of fixed point
0 = first outputfile. All output files are numbered according to the number of fixed points in the input file
Exeptions:
itt, rmin, rmax, thmin,thmax, N, target and CreatePoints are **not** used in this program.
free_Parameter(verschieb)
is a suggestion of a necessary shift out of the position of the fixed point. **Typical value: 1e-4.** Its sign determines if the right-hand sided, positive, or left-hand sided, negative, manifold is calculated.
MapDirection
determines if the stable or unstable manifold is calculated.

3.4 dtfoot _mpi

```
// Program calculates connection length and penetration depth for D3D
// Fortran Subroutines for Perturbation are used, uses open_mpi and open_mp
//
// Input: 1: Parameterfile 2: praefix(optional)
// Output: 2d footprint data for colored contour plot
// log-files
```

Since the data is for a 3d plot, it has to be prepared for plotting afterwards (e.g. with matlab)
The data is written in columns, although it would be matrices. The first column is varied first.
Program uses open_mpi for parallelization.
Example:
> mpirun –n xx dtfoot_mpi _inner.dat scan
Output:
foot_in_scan.dat and log-files
log_dtfoot_in_scan_Master.dat, log_dtfoot_in_scan_Node0.dat .. log_dtfoot_in_scan_Nodexx.dat
The Master log file documents the node communication while each Node log-file documents the activities of each working node.
Exceptions:
phistart, MapDirection and CreatePoints are **not** used in this program
free_Parameter(Np) = number of *phi* grid points
rmin, rmax = tmin, tmax: grid boundarys in length parameter *t*
thmin, thmax = phimin, phimax: grid boundarys in toroidal angle *phi*
N = Nt: number of *t* grid points
Targets: (positions in m)
vertical (0): P1=(1.0161,-1.22884) <--> *t* = 0 P2=(1.0161,-1.034873) <--> *t* = -1 length=19.3967cm
inner (1): P1=(1.0161,-1.22884) <--> *t* = 0 P2=(1.15285,-1.3664) <--> *t* = 1 length=19.3967cm
outer (2): P1=(1.15285,-1.3664) <--> *t* = 0 P2=(1.372,-1.3664) <--> *t* = 1 length= 21.915cm
shelf (3): P1=(1.372,-1.25) <--> *t* = 0 P2=(1.59115,-1.25) <--> *t* = 1 length= 21.915cm
If target 1 (inner) is chosen, negative t values are allowed, which automatically are related to positions on target 0 (vertical wall). Note that target 0 and 1 have the same length which guarantees correct scaling of *t*. Note further that target 2 and 3 scale differently to 0 and 1.

3.5 dtlaminar_mpi

```
// Program calculates connection length and penetration depth for D3D
// inside the plasma volume
// Fortran Subroutines for Perturbation are used, uses open_mpi and open_mp
//
// Input: 1: Parameterfile 2: praefix(optional)
// Output: 2d connection length data for colored contour plot
// log-files
```

similar to dtfoot, but program works in the machine coordinate system *(R,Z)* or in (psi,theta)

Exceptions:

CreatePoints is **not** used in this program
free_Parameter(NZ) = number of *Z* grid points
psimin, psimax = Rmin, Rmax: grid boundarys in *R*
thmin, thmax = Zmin, Zmax: grid boundarys in *Z*
N = NR: number of *R* grid points

or

free_Parameter(Nth) = number of *theta* grid points
N = Np: number of *psi* grid points