Use CORSICA to Generate MHD Equilibrium and Other Tools

Guoqiang Li [1,2]

[1]Institute of Plasma Physics, Chinese Academy of Sciences, Hefei, China.

[2]Lawrence Livermore National Laboratory, Livermore, CA 94550, USA

BOUT++ workshop

Sep. 3 2013, Livermore

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Security, LLC, Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, and is supported by the China Natural Science Foundation under Contract No. 10975161, the National Magnetic Confinement Fusion Science Program of China under Contracts No. 2013GB111000.

LLNL-PRES-642908

What is CORSICA

- CORSICA is an integrated modeling tool for equilibrium, MHD stability, transport/heating/CD
- Developed at LLNL
- Corsica runs on GA clusters and hopper on nersc
- http://wormhole.ucllnl.org/caltrans/: download and documentation
- This lecture will only talk about equilibrium part of CORSICA

What is BASIS

- BASIS is an interactive script language, like python, IDL, Matlab …
- CORSICA is wrapped by BASIS https://wci.llnl.gov/codes/basis/
- Expressions are similar to FORTRAN
- Declare variables
  - Variables must be declared before using
  - integer, real, double, character, logical
- ‘list’ is very useful to get help information on names
- Has vector operations like F90
- Build-in plot functions and math functions

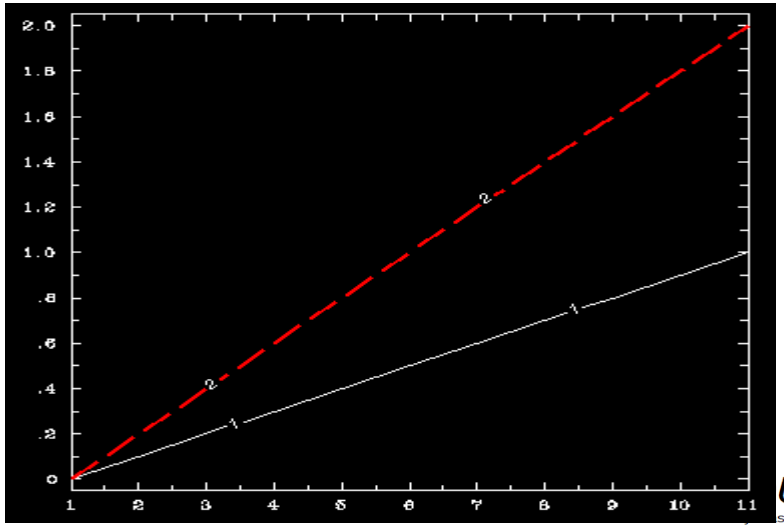
Before exercise

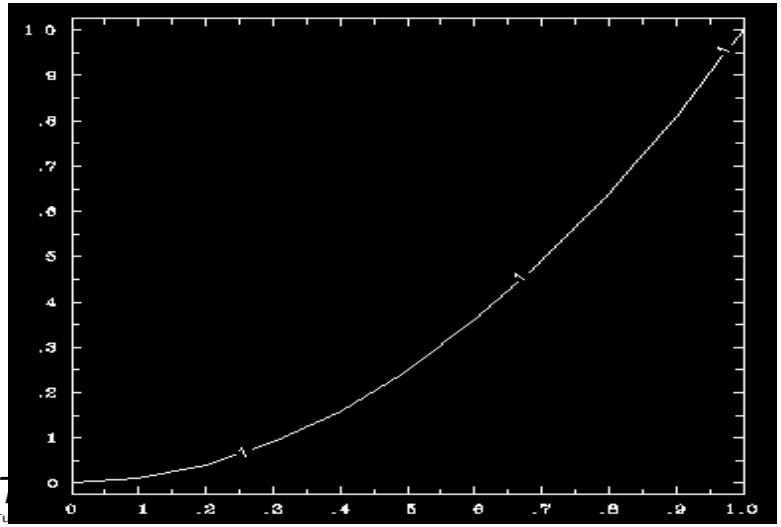
```
$ cd
$ cp –r /global/homes/l/ligq/PUBLIC/corsica-ex ./
$ cp corsica-ex/Makefile BOUT_Workshop_2013/BOUT-2.0/tools/tokamak_grids/gato/
$ source corsica-ex/set-env
```

The ~/corsica-ex/README file has all the exercise command lines

Exercise for BASIS

```
$ caltrans
> real xx = (0:1.:11)     # declare xx and set vaules
> xx
> list xx
> real yy=xx**2           # vector operation
> yy
> plot xx
> plot xx*2., color=red, thick=5, style=dashed  # overlay the xx**2
> nf                      # New framework
> plot yy,xx              # plot yy vs xx
> quit
```



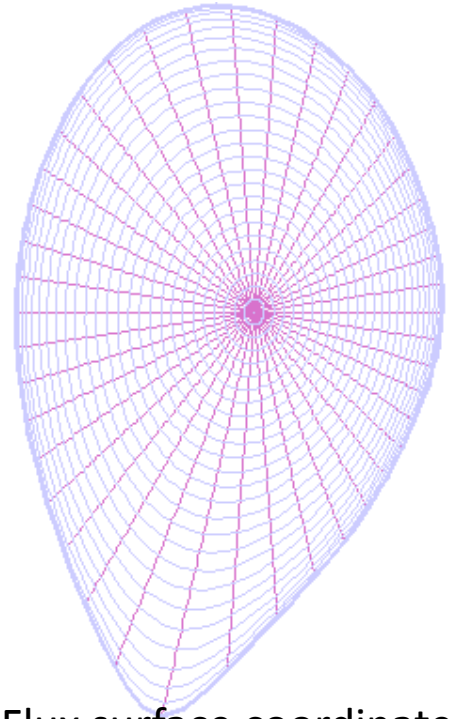


Equilibrium solver in CORSICA

- Grad-Shafranov equation solver, ψ poloidal flux

$$\Delta^* \psi = -\mu_0 R J_\phi \ , \quad J_\phi = RP'(\psi) + \frac{\mu_0 FF'(\psi)}{4\pi^2 R}$$

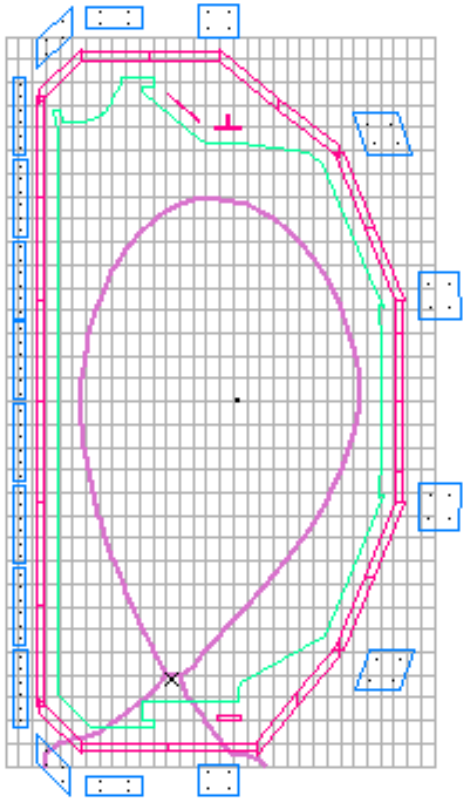
- CORSICA has both direct and inverse solver
  - **Inverse solver**: (ψ, θ) coordinate, solve for R, Z
    - prescribed-boundary: the Grad-Shafranov eq'n is solved inside a region specified by two arrays (input) and the R,Z points around the boundary



Flux surface coordinate

Equilibrium solver in CORSICA (Cont.)

- **Direct solver**: (R, z) coordinate, solve for ψ
  - free boundary: the separatrix (or limiter plasma/vacuum boundary) is found as part of the solution (i.e., output); but there are input parameters to exert substantial control over the boundary.
  - Use pprime and Fprime to evaluate Jphi on R Z grid
  - run inverse solver to get J, then interpolate to R, Z
  - Solve on infinite domain with coils

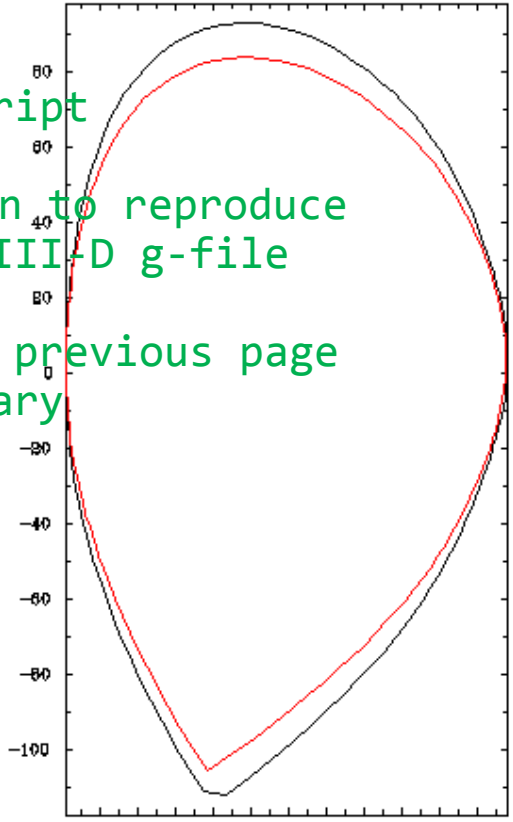


(R,z) coordinate

Change the plasma shape

- Do this work with CORSICA direct solver
- First read a g-file equilibrium, then change the shape
  - g-file is a community standard equilibrium file format, originaly from EFIT

```
$ cd corsica-ex
$ caltrans
> read d3.bas          # read in the predefined d3 sript
> d3("g098128.02500",0) # read in the g-file
                        # d3 is an predefined function to reproduce
                            direct equilibrium from DIII-D g-file
> win on
> layout(0,0)          # generate a plot like one on previous page
> plot zls,rls,scale=equal    # plot actual boundary
> zfbd = zfbd*0.9      # modify requested z boundary
> run                  # execute direct solver
> plot zls,rls,color=red
> weqdsk("g")          # write out the new g-file
```



BOUT++
Boundary Plasma Turbulence Code

Create a circular plasma from "dead start"

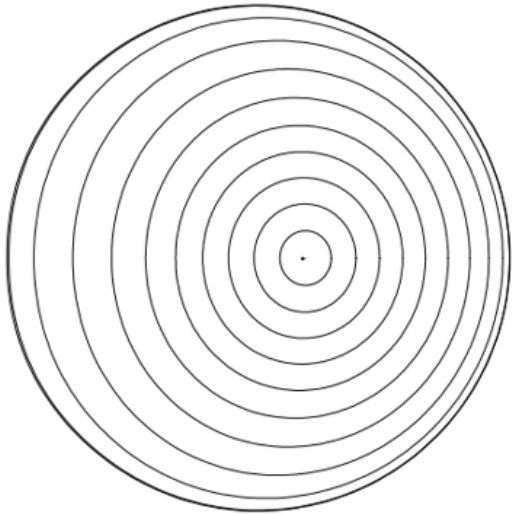
- "Dead start": create an equilibrium from a small set of parameters given in a text file.

```
$ caltrans
> read tokamak.bas          # read in predefined script
> ds("circ.inp")            # dead start with circ.inp file
> start_inv                 # convert to inverse equilibrium
> saveq("circ_inv.sav")     # save the inverse equilibrium to a file
```

```
"circ"

Plasma...
      1.00 MA       plasma current
      2.00 m        major radius
      0.50 m        minor radius
      0.00 m        Zaxis
      1.00          95% elongation
      0.00          95% triangularity
      0.00 m        Dsep (DN)
      0.80          poloidal beta
      0.90          li
      0.00 Wb       External flux linkage

Toroidal field...
2.00 T @ R = 2.00 m
```



Change the profiles

- Two profiles(P, [FF', q, Jpar, Jt]) are required to determine an equilibrium
- teq_inv(inv_k, inv_p): # command to run inverse solver, inv_k: select which two 1d arrays, inv_p select which scalar sets scale

Table 1: `teq_inv` constrained profile options via `inv_k`.

`inv_k`	*constrains...*
0	$p \mapsto$ `psave` and $q \mapsto$ `qsave`
1	$p \mapsto$ `psave` and $FF' \mapsto$ `frsrf*fpsrf`
2	$p \mapsto$ `psave` and $\langle \mathbf{J} \cdot \mathbf{B} \rangle / \langle F/R^2 \rangle \mapsto$ `jtsave`
3	$p \mapsto$ `psave` and $\langle \mathbf{J} \cdot \mathbf{B} \rangle / \langle B^2 \rangle \mapsto$ `jparsave`

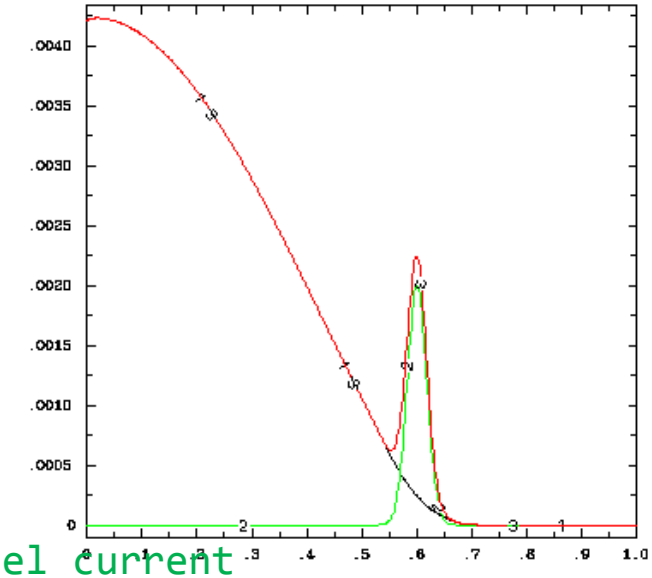
Table 2: `teq_inv` constraints $\Delta\psi_p$, F_{edge} or I_p via `inv_p`

`inv_p`	`inv_k=0,3`	`inv_k=1,2`
		$F^2_{edge} = F^2_{wall}$
< 0	$\Delta\psi_p$	$\Delta\psi_p$ and profiles scaled
$= 0$	F_{edge}	profiles not scaled
> 0	I_p	I_p and profiles scaled

BOUT++
Boundary Plasma Turbulence Code

Exercise of changing profiles

```
$ caltrans cbm18_dens8_inv.sav  # start CORSICA and restore an equilibrium
> win on
> plot psave, psibar            # plot pressure vs psi
> psave = psave * 1.2
> teq_inv(0,0)                  # excute inverse solver
> plot psave, psibar, color=red
> nf
> plot jparsave, psibar
> real jedge = 0.002*exp(-((psibar-0.6)/0.025)**2)
                                # define an edge current
> plot jedge, psibar, color=green
> jparsave = jparsave + jedge   # change the parallel current
> teq_inv(3,0)
> plot jparsave, psibar, color=red
> shotName = “10000”
> shotTime = 1.0
> weqdsk(“t”)                   # Write the equilibrium to “t-file” (dskgato file)
```



Prepare grid-file for BOUT++

- Grid-file is an input file for BOUT++
- It has the grid information and equilibrium information (configuration, profiles)

```
$ cd ~/BOUT_Workshop_2013/BOUT-2.0/tools/pdb2idl
$ ./configure
$ make
$ cd ~/BOUT_Workshop_2013/BOUT-2.0/tools/tokamak_grids/gato/
$ make
$ cd ~/corsica_ex/
$ gato2pdb t10000.01000_inv_teq   #convert dskgato (t-file) to pdb file
$ idl
IDL> pdb2bout, 't10000.01000_inv_teq.pdb', output='t10000.01000.nc'
; This IDL routine is to read in pdb file and generate grid file for
BOUT
```

```
IDL> pdb2bout, 't10000.01000_inv_teq.pdb', output='t10000.01000.nc'
Loading input file...
% Compiled module: FILE_LIST.
% Compiled module: FILE_READ.
% Compiled module: FILE_CLOSE.
**Last poloidal point duplicates the first. Removing...
Calculating poloidal field using SVD...
Difference in Bpxy computed two different ways:     0.019302715
Maximum percentage difference:        36.634303
Using given Bp, Br and Bz values
Calculating toroidal field from input f
Difference in Bt computed two different ways:   9.5367432e-07
Maximum percentage difference:    2.3522648e-05
Using Bt from input file
***Maximum mu0p is       31093.6
% Compiled module: GET_YESNO.
Is this pressure (not mu0*pressure)?y
% Compiled module: GET_INTEGER.
Grid number at the plasma boundary:90
PSI normalised range: 0 to 1.44444
Number of radial grid points  : 131
Number of poloidal grid points: 128
Edge Q: 2.89919
Range of parallel current density [A/m^2]:       -780864.00       367337.19
% Compiled module: GET_FLOAT.
Inner Psi boundary:0.4
Outer Psi boundary:1.2
Psi range      0.400000      1.20000
Number of radial grid points:           73
Of which inside the plasma:           55
Is this range ok?y
====== SETTING PLASMA PROFILES =======
Some plasma parameters given in input file
Use given parameters?n
Generating plasma profiles:
  1. Flat temperature profile
  2. Flat density profile
  3. Te proportional to density
Profile option:2
Setting flat density profile
Density [10^20 m^-3]:0.5
Maximum temperature (eV):      1940.92
Is this ok?y
Increase radial resolution?y
Number of radial points:132
% Compiled module: XINTERP.
% Compiled module: INTERPOL.
======== GENERATING ORTHOGONAL COORDINATES FOR BOUT =========
Number of poloidal grid points:64
Enter x index of equal hthe [0, 131] :80
% Compiled module: GEN_ORTHOG2.
% Compiled module: FFT_INTERP.
% Compiled module: REAL_PART.
Lines done:64 of 64
Interpolating Rxy
Interpolating Zxy
```

```
Is this ok?y
Interpolating values onto new grid
Jpar
Bxy
Bpxy
Btxy
% Compiled module: BOUT_OUTPUT.
% Compiled module: IN_LIST.
Density 0.50000000 -> 0.50000000
Maximum pressure [Pa]:        28545.376
Add vacuum region?n
% Compiled module: DERIV.
Equilibrium correction options:
  0  No correction
  1  RBt using force balance
  2  hthe and RBt using force balance and q (FAILS)
  3  hthe and RBt using force balance and jpar
Enter option:0
Calculating poloidal arc length dthe
% Compiled module: FFT_INTEGRATE.
% Program caused arithmetic error: Floating divide by 0
% Program caused arithmetic error: Floating underflow
Maximum difference in hthe:     0.0028385068
Maximum percentage difference:       0.17026461
Use new hthe?n
Checking parallel current
****Equilibrium has -ve toroidal field
Maximum difference in jpar0:        27398.298
Maximum percentage difference:        125803.70
Use new Jpar?n
Performing integrals
q is negative. Reversing values from equilibrium
Use new qsafe?n
% Compiled module: FFT_DERIV.
****Minimum pressure is very small:      0.39290048
****Setting minimum pressure to 1e-4 of maximum
% Compiled module: NCDF_EXISTS.
% Loaded DLM: NCDF.
% Compiled module: FILE_WRITE.
% Compiled module: REVERSE_INDS.
DONE
```