Basic Structure of Orbit

R.B. White, Jan 2007

The guiding center code Orbit, using guiding center equations derived in The theory of toroidally confined plasmas, Imperial College Press 2001, has been improved to be no longer restricted to Boozer coordinates. It can use equal arc, PEST, or any other straight field line coordinates ψ_p, θ, ζ which along with the parallel velocity and energy completely specify the particle position and velocity. The guiding center equations depend only on the magnitude of the B field, not on components. The code uses a fourth order Runge Cutta integration routine. It is divided into a main program Orbit.F, which is essentially a heavily commented name list and a set of switches for choosing the type of run, the diagnostics, the data storage, and output. Subroutines are included in initial.f, deposit.f, collisions.f, perturb.f, step.f, record.f, and orbplot.f. The output data is stored as orbit.out, and plot files all of the form file.plt. The Makefile, which will accept compilers pgf90, pathf90, and lf95, deletes all these files so that there are no misleading data lying about from a previous run. So output must be renamed or stored after running if you wish to keep it. **orbit.out contains particle data including velocity, mass, charge, gyro freq, gyro radius, all equilibrium data and all run parameters, including reasons for run abortion.**

- **Read Equilibrium**
- **Load particles, add perturbations and ripple if desired**
- **Advance particles and collect data**
- **Output data**

To obtain a copy take all files from /u/ftp/pub/white/Orbit
To submit a job modify the script batch, modify Orbit.F to choose the run desired,
type make, and type qsub -q sque batch

I. EQUILIBRIUM

The equilibrium is created using mex2eqs, written by Alex Pletzer and maintained by Doug Mccune. Mex2eqs reads either TRANSP or EFIT data, and produces the file map01.cdf, which is read by eqs.f. The program eqs.f reads map01.cdf and produces the spline data needed for Orbit. It can also without map01.cdf produce analytic Shafranov

shifted tokamak equilibria. The equilibrium data does not include field ripple. If ripple is desired it must be fit using analytic expressions available in eqs.f. Included is an NSTX equilibrium file both in equal arc coordinates map01ea.cdf and in Boozer coordinates map01bz.cdf, along with the spline data produced by eqs.f, spdata.ea and spdata.bz.

II. MAIN, ORBIT.F

The main program allows choice of type and length of run, particle distribution, field perturbations, and scattering operator, The selection is made through setting switches in the main.

There are four types of runs available, chosen by **nplot** but if other diagnostics are required the best way to proceed is to start a new value of nplot, nplot = 6, and add switches to choose dep6 to deposit particles, rcrd6 to collect the data you want, and plot6 to produce the plot files.

nplot = 1 gives single particle orbit,
nplot = 2 local diffusion and loss data,
nplot = 3 Poincare plot of field,
nplot = 4 distribution modification due to modes
nplot = 5 bootstrap current

particle parameters

nprt = 1000 : number of particles
zprt = 1.D0 : charge in proton units
prot = 1.D0 : mass in proton units
ekev = 20.D0 : energy in KeV

iseed = 0 : seed for random number initiation.
Used for distributions and for scattering operator.
ntor = 6 : simulation run time, in on-axis toroidal transits
bkg = 3.557 : field strength in kGs- the field can be scaled
npert = 1 : field perturbation choice, 0 = none
pamp = 1 : potential amplitude, Kev, not used if npert = 0
ndist = 4 : 1=shell distrib, 2=sampledep, read from TRANSP data,
3=poincare, 4=alphas
polo = .5*pw ! initial flux surface for nplot =1, 2, pw = last flux surface
p1 = .6*pw ! minimum surface for Poincare
p2 = .95*pw ! maximum surface for Poincare
pchi = .6D0 ! initial pitch for single particle

dele = 5.D-8 : energy conservation per time step. Any code modification or inclusion of perturbations should be checked that energy is conserved. The time step is adjusted to keep the energy conservation dE/E less than dele. Note that all particles have their own clocks, they are not at the same time. If time synchronization is desired you must use *dele* > 1 which forces all time steps to be dt0, set in initial.f. The default value is 200 steps per toroidal transit time, and this is sufficient unless you insert perturbations with large n values. One should check a single particle run with dele = 5e-8 to see what time step gives acceptable energy conservation, then set dt0 accordingly. If time dependent perturbations are used energy is no longer conserved, so dele larger than one is necessary. Energy conservation should be controlled using zero frequency.

if(nplot.eq.4) dele = 5. dele larger than 1 gives fixed time step, needed for instantaneous distribution determination.

Collisions

ncol = 0 ! no collisions, ncol=1, energy dependence only,

ncol = 2 full profiles

col = 1.D0/(50*tran) : pitch angle scattering frequency

drag = 0.D0/(200*tran) : slowing down frequency

For ncol = 2 see scatr in collisions.f and functions denb, deni, tempi, tempe

massb = 2 ! background plasma mass

chgb = 1 ! background plasma charge

imp = 0 ! imp=1 impurity species, 0=none

massi = 5 ! impurity mass

chgi = 5 ! impurity charge

eion = 1.1 ! ion energy in kev for plots

A. Single particle orbit, nplot=1

Setting nplot=1 automatically limits the run to include a single particle, launched at surface polo with pitch = pchi, and calls the diagnostic routine rcrd1. Data is recorded at time intervals of dt1, set in initial.f with default of .01*tran for nplot=1. Data recorded is time, energy change, time step, $\psi_p, \zeta, \theta, \lambda$, B, q. This can easily be augmented or changed. See rcrd1 in record.f. At the end of the run this data is written to files traj1.plt and traj2.plt for postprocessing.

B. Local diffusion and particle loss, nplot=2

For nplot=2 monoenergetic particles are initially uniformly distributed on a single flux surface. The routine rcrd2 does a running sum of the mean square displacement from this surface vs time, the mean pitch distribution and the mean energy change. These can easily be changed or augmented. Data is recorded at intervals of nskip steps, set in set1, initial.f, with default to produce 200 plot points in the course of the run. At the end of the run the routine plot2 in orbplot.f writes the plot data set diffusion.plt. The lost particle data is stored in lost.plt, so distributions of lost particles can be plotted.

C. Poincare plot, nplot=3

For a Poincare plot the particles are automatically set to have very low energy and pitch of 1, so that they simply follow field lines. They are deposited uniformly with ψ_p between p1 and p2, set in the main. Data is collected in rcrd3 in record.f every time a particle crosses $\zeta = 0$. At the end of the run plot data is written by plot3 in orbplot.f in the file poincare.plt

D. Modification of distribution by perturbations, nplot=4

To find the modification of a particle distribution by a mode such as a TAE mode, excellent statistics can be generated by running the initial distribution for some time in the presence of the mode and then collecting distribution data every time step for the remainder of the run. This produces time average data for the modified distribution function. The default is that the data is collected for some last fraction of the run time, see the call to rcrd4 in orbit.F.

E. Local bootstrap current, nplot=5

This is a δf calculation of local bootstrap current. See chapter 8 of my book. The current is normalized to the theoretical value for small collision frequency in a circular equilibrium.

III. SET UP, INITIAL.F

These routines calculate initial constants etc for run. Normally nothing has to be changed here

IV. LOAD PARTICLES, DEPOSIT.F

Chosen with ndist, The distributions available in deposit.f are a monoenergetic distribution distributed evenly on one flux surface polo, given by shelldep, a distribution evenly distributed between two bounding surfaces p1 and p2, poindep, the possibility of reading a particle distribution directly from TRANSP, sampledep, and a monoenergetic alpha particle distribution with a choice of radial profiles.

Any distribution in space, energy, or pitch can be constructed using a simple Monte Carlo procedure.

For example to make a distribution of particles in ψ of the form $dn/d\psi = f(\psi)$ simply write

$$\frac{dn}{d\psi} = \frac{dn}{dx}\frac{dx}{d\psi} = f(\psi) \tag{1}$$

and choose $x = ranx()$ giving a uniform distribution of particles in x, ie $dn/dx = constant$. Integrate to find $x = \int f(\psi)d\psi$. Invert this to find $\psi(x)$, numerically if necessary. Then this routine will deposit particles with the required distribution. Integration constants are chosen to adjust the range of the deposition.

```
   do 10 k = 1,nprt
5        continue
         x = ranx()
         ptry = psi(x)
         prob = f(ptry) ! must be normalized to be less than one
         dum = ranx()
         if(prob.lt.dum) go to 5
         psi(k) = ptry
10       continue
```

Similary, distributions can be constructed that are functions of all variables energy, pitch, and position, Since the deposition is done only once before the run begins this is not expensive.

In functions.f there is a function maxwell(tdum), which produces one random value of energy from a Maxwellian distrubution of 'temperature' tdum in KeV. Thus a routine setting particle energy through en(k) = maxwell(tdum) will give a Maxwellian distribution in energy. Attention, if an energy distribution is used the time step must accomodate the fastest particles.

V. TRANSP PRODUCED BEAM DISTRIBUTIONS

To use beam particle distributions existing in the device ndist = 2 will read beam particle distributions produced by TRANSP. The subroutine is SAMPLEDEP. A sample beam distribution is supplied,$fbm_dist.dat$. It is an ASCI file with simple format. There is also a script file getfh4orbit.scr to procuce the beam data from TRANSP.

VI. PERTURBATION, PERTURB.F

For orbit the perturbations must be of the form $\delta B = \nabla \times \alpha(\psi, \theta, \zeta)B$, normally using harmonics of the form $\alpha = sin(n\zeta - m\theta - \omega t)$. This form produces exactly the component perpendicular to the equilibrium flux surface, responsible for magnetic islands, and is automatically divergence free. It is a good representation for all low beta MHD perturbations. In perturb.f there are the routines called every time step to add the field perturbation to the field, as well as means to construct analytic perturbations of MHD or resistive type. See spln in perturb.f. The routine readptrb will read a perturbation amplitude from an external data set ptrb.dat. A sample 2-harmonic TAE mode is supplied with the data set ptrb.dat, which is a simple asci file. to see the perturbations use the data harmonics.plt, and the supermongo file harmonics.p

VII. COLLISIONS, COLLISIONS.F

ncol = 1 gives a simple pitch angle scattering operator

col = 1.D0/(50*tran)

gives the collision frequency as 1/50 of a transit time. Slowing down is given in the same manner.

drag = 1.D0/(200*tran)

using ncol = 2 a much more elaborate collision operator is given, using the Rosenbluth psi function, and profiles for density and temperature, which must be supplied. These routines are all in collisions.f

VIII. DIAGNOSTICS, RECORD.F

rcrd0 - records initial particle data

rcrd1 - records single particle data at intervals of dt1

rcrd2 - records the mean square displacement from the flux surface as well as the mean pitch and energy change.

rcrd3 - records data for a Poincare plot at crossings of $\zeta = 0$
rcrd4 - performs a time average of the distribution,
starting at a specified time late in the run.
rcrd5 - Bootstrap current data

These routines can be easily modified. To add a new diagnostic, simply copy one of these, call it rcrd6, make the necessary modifications, and introduce nplot=6 in the same manner.

IX. OUTPUT, ORBPLOT.F

plot1 - writes single particle data at intervals of dt1 in traj1.plt and traj2.plt
plot2 - writes the mean square displacement from the flux surface as well as the mean pitch and energy change in diffusion.plt
plot3 - writes poincare.plt, the Poincare data
plot4 - writes the time average of the distribution, starting at a specified time into the run in distave.plt.
plot5 - writes the Bootstrap current vs time, boot.plt.

These routines can be easily modified. To add a new diagnostic, simply copy one of these, call it plot6, and introduce nplot=6.

X. POSTPROCESSING WITH SUPERMONGO

Super mongo was written by Robert Lupton at Princeton. It is a simple plotting routine that produces postscript files and is written in Pascal. There are copies of routines to handle all of the file.plt output files produced by Orbit available with the code. Those enamored of IDL can read the same output files and use IDL to obtain plots.

An advantage of Super Mongo is that it supports Latex, so that plots are easily produced for publication. All the supermongo files are of the type file.p. Super Mongo is supported at PPPL on the unix devices. Simply type sm, after which one types **input file.p**, and the plot appears. The pound sign at the beginning of a line makes that line inoperative. Thus the first line

device postencap :SY@: :OF@: new.ps

turns the output into the postscript file new.ps, but if preceded by a pound sign the output only goes to the screen for checking. The normal procedure is to play with the plot until it is acceptable, then make this first line operative, and rename new.ps as desired.

supermongo files available are

diff.p for diffusion $< d\psi_p^2 >$ vs time

dist.p	initial, final or mean particle distributions in space
lost.p	particle loss vs time
harmonics.p	harmonics of perturbation
poin.p	Poincare plot
trajxz.p	particle trajectory data in poloidal section
trajtop.p	particle trajectory data top view
trajt.p	particle trajectory data vs time
distave.p	mean distribution
equilibrium.p	equilibrium shape
profiles.p	equilibrium profiles
histogram.p	binned particle distributions in ψ_p, θ, etc
scatr.p	profiles for advanced collision operator
boot.p	bootstrap current

These files can be easily modified to plot other features.