IMP4

December 17, 2020

Contents

1 IMP4

The IMP4 web site is here. [1]

2 Scientific Rationale and Main Objectives

Provide turbulence, neoclassical, and linear instability codes as well as simple transport modules to the rest of the ITM. Also, some standards keeping. For more detail see above.

3 Documentation

Some of this exists here but most is on the main page. However, all relevant links are here:

- running the IMP4 benchmark [2]
- IMP4 codes in ITM workflows [3]
- an example of how to get CPO information including parameters into your code is the ETAIGB project [4] on Gforge [5]
- **for ETS/IMP3 users:** test workflows for IMP4 actors [6]
- serving transport quantities to the rest of ITM (5)
- the running exponential average (6)
- how to load the `coretransp` CPO (7)
- Neoclassical code comparisons
  - bootstrap current (8)
  - Ds and Chis (9)
- auxiliary IMP4 actors (like Ds and Vs, flux handling, etc) (10)

4 Private IMP4 pages

None exist (they may do in the future... for now I want IMP4 stuff open to the rest of the ITM).

(For access to the private IMP4 pages [7], an IMP4 password would have been needed)

5 Turbulent Flux Quantities in Transport Models

5.1 Overview

In conventional transport modelling, all quantities appearing in the equations are 1-D, in some radial coordinate (poloidal flux, normalised radius, etc). In general any monotonic radial coordinate is acceptable. In the TF-ITM, the toroidal flux radius is standard. All we need from the radial coordinate is the transformation to

[1] http://www.rzg.mpg.de/~bds/cyclone/
[2] http://home.rzg.mpg.de/~bds/cyclone/benchdoc.pdf
[3] http://home.rzg.mpg.de/~bds/cyclone/workflows.pdf
[4] https://gforge6.eufus.eu/svn/etaigb
[5] https://gforge.efda-itm.eu
[6] http://home.rzg.mpg.de/~bds/cyclone/profcheck.html
[7] https://www.efda-itm.eu/IMP4/html/index.html

get to V, the volume enclosed by the flux surface, which is fundamental to the governing equations, which are conservation laws.

What we have to do is to take a measured result, which is a time-averaged fluctuation-based transport flux and turn it into 1-D quantities suitable to modelling. This is done using the flux surface average, explained in **conventions.** The transport equations themselves constitute a mean field approximation to the 3-D conservation laws. For the fundamentals encountered in transport modelling see R Hazeltine and J Meiss, *Plasma Confinement* (Addison-Wesley, 1992) chapter 8. For the special properties of transport driven by small-scale pressure driven ExB microturbulence see B Scott, "The character of transport caused by ExB drift turbulence," *Phys Plasmas* **10** (2003) 963-976.

For ambipolarity we follow the rules for dynamical alignment, which follows the physics of how electron fluctuations determine the ExB velocity fluctuations, which then advect all species. Magnetic flutter nonlinearities act independently of this, but in our modelling they are used solely for heat fluxes since the averaged particle transport due to magnetic flutter and the current cancels, leaving the parallel ion velocity which we neglect for this purpose. The reference for dynamical alignment is B Scott, "Dynamical alignment in three species tokamak edge turbulence," *Phys Plasmas* **12** (2005) 082305.

Note: there are now auxiliary actors provided for this purpose: IMP4DV, which does the D/V conversion and enforces ambipolarity assuming absence of impurities, and IMP4imp, which subsequently enforces ambipolarity for the set of main ion and impurity species. The IMP4DV actor should be invoked directly after the transport model actor in the workflow chain, if the model produces only fluxes or if the coefficients have to be modified with the flux given. Ambipolarity is done using IMP4imp if the `coreimpurity` CPO is used in the workflow. These auxiliary actors are described on the auxiliary actors page. (10)

5.2 Particle Flux as an Example

The mean field equation governing particle balance is the transport equation for electrons,

$$\frac{\partial}{\partial t}\langle n\rangle + \langle \vec{\nabla}\cdot \tilde{n}\tilde{\vec{v}}_E\rangle = S$$

in which the tilde symbol over the n and v denotes fluctuating quantities and we neglect all transport processes except ExB eddy diffusion. The ExB velocity is given by

$$\vec{v}_E = \frac{c}{B^2}\vec{B}\times\vec{\nabla}\phi$$

where ϕ is the electrostatic potential.

The angle brackets denote the flux surface average, and we will use the property that the flux surface average of a divergence of a vector is the volume derivative of the flux surface average of a contravariant volume component of the vector, in this case

$$\langle \vec{\nabla}\cdot\vec{\Gamma}\rangle = \frac{\partial}{\partial V}\langle \Gamma^V\rangle$$

where Γ is the particle flux whose flux-surface averaged volume component is

$$\langle \Gamma^V\rangle = \langle \tilde{n}\tilde{v}_E^V\rangle$$

This is converted to expression in terms of the radial coordinate ρ using the fact that both V and ρ are flux quantities whose gradients are parallel to each other. We have

$$\frac{\partial}{\partial V} = \frac{1}{V'_\rho}\frac{\partial}{\partial \rho} \qquad \Gamma^\rho = \frac{1}{V'_\rho}\Gamma^V \qquad V'_\rho = \frac{\partial V}{\partial \rho} \qquad g^{VV} = (V'_\rho)^2 g^{\rho\rho}$$

so we can write the transport equation as

$$\frac{\partial n}{\partial t} + \frac{1}{V'_\rho}\frac{\partial}{\partial \rho}V'_\rho\langle\Gamma^\rho\rangle = S,$$

where we have replaced $\langle n \rangle$ with n following the assumptions of the 1-D version of mean field transport theory.

With all quantities now expressed in terms of flux quantities, we are free to characterise the transport flux $\langle \Gamma^\rho \rangle$ in an arbitrary way, so long as only flux quantities appear. The flux expansion within the flux surface as well as expansion or contraction of surfaces of constant ρ

is treated using the metric coefficient $g^{\rho\rho}$ which is dimensionless. This way we can characterise transport in terms of an effective diffusivity and an effective frictional slip velocity which are given in SI units. By convention both of these are done solely via $g^{\rho\rho}$ for convenience, also reflecting that the effective velocity is actually marking off-diagonal diffusive elements. Our convention for this follows the ETS code and is given by

$$\langle \Gamma^\rho \rangle = \langle g^{\rho\rho} \rangle \left(n V_{\text{eff}} - D_{\text{eff}} \frac{\partial n}{\partial \rho} \right)$$

So despite the special spatial distribution of any particular transport process (ie, the underlying instability or nonlinear free energy access), the flux-surface averaged flux itself and its expression in terms of diffusion and frictional slip are identical characterisations.

5.3 Metric Coefficients

Transport modellers want the Ds and Vs as physical quantities in SI units. In general the fluxes are (magnetic) flux surface averaged quantities, which implies the existence of metric elements in the conversion. In our case we need $\langle g^{\rho\rho} \rangle$ where ρ is the toroidal flux radius in meters, so the metric elements are dimensionless. In the `equilibrium CPO` , this is `gm3` under `equilibrium%profiles_1d` in the structure.

Note this is different from the ASTRA code which casts the Vs as proper velocities, i.e., with one factor of grad-rho given by $\langle \sqrt{g^{\rho\rho}} \rangle$ which is `gm7` under `equilibrium%profiles_1d` in the structure. The units are the same and the informational content is the same, but this difference has to be taken into account in any transport modelling and benchmarking.

5.4 Heat Fluxes

The heat flux is treated in a similar way, with transport equation

$$\frac{3}{2} \frac{\partial p_e}{\partial t} + \frac{1}{V'_\rho} \frac{\partial}{\partial \rho} V'_\rho \langle q_e^\rho \rangle = Q_e + \Sigma_{\text{ions}} T_{ei},$$

for electrons, with T_{ei} giving the species transfer and Q_e the source. For ExB transport the heat flux has a advective (also called convective) and a conductive piece given by

$$q_E = q_{E\text{cond}} + (3/2) T \Gamma_E$$

which appears with a $3/2$ due to the Poynting cancellation. For magnetic flutter transport the advective piece appears with the usual factor,

$$q_m = q_{m\text{cond}} + (5/2) T \Gamma_m$$

Here the forms are given for each species and E and m refer to the ExB eddy and magnetic flutter channels, respectively. For reasons given below we are neglecting the magnetic flutter piece Γ_m for the time being, and then the flutter piece merely adds to the heat diffusivity.

The forms of these due to the fluctuations are then

$$\langle q^\rho \rangle = (3/2) \langle \widetilde{p} \widetilde{v}_E^\rho \rangle + \langle \widetilde{q}_\parallel \widetilde{b}^\rho \rangle$$

which breaks into advective and conductive pieces according to linearisation of the pressure fluctuations

$$\langle q^\rho_{\rm cond}\rangle = (3/2)n\langle \widetilde{T}\widetilde{v}^\rho_E\rangle + \langle \widetilde{q}_\parallel \widetilde{b}^\rho\rangle \qquad \langle q^\rho_{\rm adv}\rangle = (3/2)T\Gamma = (3/2)T\langle \widetilde{n}\widetilde{v}^\rho_E\rangle$$

hence the density fluctuation piece is accounted for by the particle flux. Neglect of the magnetic flutter advective piece (and particle flux) is the same as neglect of the $\widetilde{u}_\parallel \widetilde{b}^\rho$ nonlinearity (in the delivery of the results, not in the turbulence computations themselves).

The total conductive flux is then represented by

$$\langle q^\rho_{\rm cond}\rangle = \langle g^{\rho\rho}\rangle \left(nTY_{\rm eff} - n\chi_{\rm eff}\frac{\partial T}{\partial \rho}\right)$$

with χ and Y giving the heat diffusion and frictional slip pieces for each species, respectively (these are in diff_eff and vconv_eff in the CPO for each quantity).

Operationally, the turbulence module communicates the diff_eff and vconv_eff due to each transport channel for each species to the transport solver, and the metric coefficients are used by both modules. The two modules can be on arbitrarily different grids, which communicate through standard interpolation. This despite the fact that transport at the micro-level is angle dependent (in general, it can be 3-D in the time average if the sources are 3-D). The effective transport is 1-D so long as parallel sound transit within the flux surface remains fast compared to the local transport time. This breaks down anyway in the edge, so the fact that the volume is a problematic coordinate and the flux surface average is a problematic operation on open field lines doesn't enter.

5.5 Ds and Vs from Turbulence Codes to Transport Solvers

To serve the results from turbulence codes to transport solvers, we have to turn the fluxes (results) into diffusivities and effective velocities (coefficients, Ds and Vs for short), which represent more information than is at hand. Transport solvers must work with Ds and Vs because they use implicit schemes. The matrix must be diagonally dominant; hence one cannot simply use the Vs. Fluxes which are zero and/or negative should be given with positive diffusivities for the solvers to work. We need a set of rules to provide this.

Considering the particle and heat transport fluxes for a given species, we convert the gradient in to a logarithmic derivative and express the flux in terms of a specific flux, which has units of velocity,

$$F = \tfrac{1}{n}\langle g^{\rho\rho}\rangle^{-1}\langle \Gamma^\rho\rangle = V_{\rm eff} - D_{\rm eff}\frac{\partial \log n}{\partial \rho}$$
$$G = \tfrac{1}{nT}\langle g^{\rho\rho}\rangle^{-1}\langle q^\rho_{\rm cond}\rangle = Y_{\rm eff} - \chi_{\rm eff}\frac{\partial \log T}{\partial \rho}$$

wherein the conductive part of the heat flux (without the $3\Gamma/2$) enters.

The choice of what to do with the Ds and Vs is somewhat arbitrary. The needs of implicit transport solvers is for a positive D regardless of the value or sign of either flux. We decide this by putting a limit on the effective Prandtl number or its inverse: the larger specific flux is taken to be entirely diffusive, with the effective velocity set to zero. Furthermore, to address cases with very small or negative gradients, we use proxy variables for the scale lengths to calculate the provisional diffusivities before using the Prandtl number limitation to turn these into actual diffusivities. Finally, the rest of the flux is asigned to the effective velocity, so that the D and V formula reflects the actual specific flux.

The Prandtl number limitation is expressed as follows. If the smaller specific flux is within a factor of 5 of the larger, then both are purely diffusive and the effective velocities are both zero. If not, then the D ratio is set to 5, with the result that the smaller D, having been corrected, is accompanied by the corresponding V, which is now nonzero. The specific flux with the larger D will be returned with a V which is zero.

The rationale is that the turbulent mixing by the ExB velocity affects all processes, but that linear forcing can shift the average phase shift of the fluctuations such that the effective flux can be small or negative. The simplest example is adiabatic electrons, for which the ion heat flux is robust but the particle flux is zero. In most situations the specific heat flux will be the larger, and hence the familiar situation is that of a D and V for the particle flux but a D (the chi) only for the conductive heat flux.

The full algorithm starting with the specific fluxes appears as

$$L_n^{-1} = \max\left(\tfrac{1}{R}\,,\ \left|\tfrac{\partial \log n}{\partial \rho}\right|\right) \qquad L_T^{-1} = \max\left(\tfrac{1}{R}\,,\ \left|\tfrac{\partial \log T}{\partial \rho}\right|\right)$$
$$D' = |F|\, L_n \qquad \chi' = |G|\, L_T$$
$$D = \max\left(D'\,,\ \tfrac{1}{5}\chi'\right) \qquad \chi = \max\left(\chi'\,,\ \tfrac{1}{5}D'\right)$$
$$V = \left(F + D\tfrac{\partial \log n}{\partial \rho}\right) \qquad Y = \left(G + \chi\tfrac{\partial \log T}{\partial \rho}\right)$$

and all four elements are set. Note that the channels are done in parallel except for the Prandtl correction, in which the Max's are taken sequentially. For the provisional diffusivities, absolute values are used to ensure positive values which are needed by transport solvers.

Note how in the end the actual gradients are used. If the gradients are moderate then their actual values are used, and if the Prandtl correction is not invoked, then both channels are diagonal. In any case the full relation is used to get the effective velocities (V and Y) so having set the rules to handle the arbitrariness of the diffusivities (D and chi) to guarantee reasonable diagonal dominance in a transport solver, the D's and V's agree with the fluxes themselves.

If there are more than two specific fluxes per species to consider, then we treat each scale length separately as above and use N-way maxima in the Prandtl correction for the N channels.

5.6 Ambipolarity

There remains the issue of ambipolarity of the D and V for particle flux. For a pure singly charged plasma the ion and electron Ds and Vs should be equal. Even if the turbulence model is gyrokinetic or gyrofluid, in which case the gyrocenter charge density is not zero but is equal to the generalised vorticity (polarisation), the quantities given to a transport solver should follow the rules for a fluid representation. However, transport modelling usually applies ambipolarity rules to the electrons after computing the ions, while the action of turbulence is actually the other way around: Dynamical alignment refers to the process by which (1) electron parallel dynamics controls the electrostatic fluctuations, then (2) the resulting ExB velocity advects all species equally. So we correct the particle fluxes by assuming the electrons determine the D according to the above procedure and then (1) the fluctuations in the flux-inducing part of the spectrum for the logarithmic densities are the same, and (2) the D's are the same. Then the V's are solved for again, by taking

$$D_z = D_e = D \qquad V_z = V_e + D\tfrac{\partial \log b_z}{\partial \rho} \qquad b_z = n_z/n_e$$

This is better than the transport modelling convention but will give them the same information in a different way, and they will compute ambipolar particle fluxes (radial transport of charge is zero).

5.7 Statistical Character

Turbulence has a statistical character, so convergence to a mean is not monotonic and when within one std dev of the mean there is no further convergence. The diffusivity for ExB turbulence is comparable to

$$D_E = \langle(\widetilde{v}_E)^2\rangle / \langle(\varpi)^2\rangle^{1/2} \qquad \varpi_E = \tfrac{c}{B}\nabla_\perp^2\widetilde{\phi}$$

where ϖ_E is the ExB vorticity fluctuation, and these angle brackets denote the ensemble average. To get an ensemble average over a statistical quantity in practice, one must do some sort of finite-time running averaging.

For transport modelling, the transport coefficients derived from a turbulence code should always be given in terms of running exponential averages. (6)

A HOWTO for loading the `coretransp` CPO is given here. (7)

---

last update: 2015-03-27 by bds

6 Running Exponential Average

6.1 Overview

In conventional transport modelling, turbulent fluxes are modelled in terms of processes which are diffusive in the local relaxation sense, with the average flux given by a diffusion coefficient and an effective pinch velocity. The equations are of dominantly parabolic character, which means in practice that an iterate will move monotonically towards the solution in parameter space.

This is not the case for turbulence. Convergence is statistical, which is something different than a diffusive relaxation. If turbulence is stationary, it is meant only that the mean of a distribution of iterates is stationary, not the iterates themselves. The standard deviation can be significant, of order unity compared to the mean, of any distribution of iterates.

This makes for a noisy signal if the output of a turbulence code is used for transport coefficients in a workflow. A sound way to overcome the attendant problems is to use a moving average. Even an average over a moving window can be as noisy as the original signal, however. What works better is a weighted average over recent past values. A method to get this is called a **running exponential average,** which is essentially the same thing as a convolution integral over an exponential memory decay times the past signal. It turns out to be very easy to obtain this without saving past values.

The original reference for the following is S W Roberts, "Control Chart Tests Based on Geometric Moving Averages," *Technometrics* **1** (1959) 239-250, cited by all the good WWW resources, including the Wikipedia page on Moving Averages and the NIST Statistical Handbook online.

6.2 Definition

Consider a process $p(\vec{u})$ which is a functional of dependent variables $\vec{u}$. Measure p at discrete time intervals t_n, with values $p_n = p(t_n)$

and interval length $\tau = t_n - t_{n-1}$. The moving exponential average $A_n = A(p_n)$ on the n-th interval is defined as

$$A_n = \epsilon p_n + (1-\epsilon) A_{n-1} \qquad \text{with} \qquad \epsilon = \alpha\tau$$

in which the small parameter ϵ is given in terms of the interval τ and an inverse time constant α.

In the first instance p is measured there is no A so the first value of A is simply set to p since it can be assumed that the initial state for p has persisted for infinite previous time up to the initial time point.

6.3 Differential Equation

The equivalent differential equation is found by forming the relevant finite difference,

$$A_n - A_{n-1} = \epsilon(p_n - A_{n-1})$$

which we can also cast as

$$(1-\epsilon)(A_n - A_{n-1}) = \epsilon(p_n - A_n)$$

Taking the limit $\tau \to 0$ is the same as taking $\epsilon \to 0$ so both of these expressions become equivalent to

$$\frac{\partial A}{\partial t} = \alpha(p - A)$$

whose solution is given below.

6.4 Equivalence to Past-Time Convolution Integral

The solution of the above differential equation is given by the method of undetermined coefficients,

$$\frac{\partial A}{\partial t} + \alpha A = \alpha p$$
$$e^{-\alpha t}\frac{\partial}{\partial t}\left(e^{\alpha t} A\right) = \alpha p$$
$$\frac{\partial}{\partial t}\left(e^{\alpha t} A\right) = \alpha p\, e^{\alpha t}$$

We may integrate this over all past time, to find

$$A(t) = \int_{-\infty}^{t} \alpha\, dt'\, p(t')\, e^{-\alpha(t-t')}$$

This is a convolution integral over the kernel $e^{-\alpha(t-t')}$ and the signal $p(t')$. The time constant α^{-1} is just the memory decay time, while if p is constant then the integral yields unity times p. This is the same as the normalisation with the $(1-\epsilon)$ factor in the average formula above, which is needed since the interval is of finite size.

Hence the running exponential average is operationally the same as a memory decay integral over past time. The elegant feature is the need to keep only the current value of A, as it already contains all that is needed of the past time evolution of p.

6.5 notes

Some properties of the running exponential average and how to choose its main time-memory parameter:

- The $(1-\epsilon)$ factor is needed for normalisation
- if $p =$ constant then $A = p$ for all t
  - the integral with $\alpha\, dt'$ yields unity
  - the ϵ and $(1-\epsilon)$ factors add to unity
  - therefore set the first value of A to the first value of p
- in choosing the memory decay time $\alpha^{-1}\ldots$
  - one should have $\alpha\tau_{\text{cor}} \ll 1$
  - best results are for $\alpha\tau_{\text{sat}} \sim 1$
  - some trial/error required; edge turbulence likes $\alpha^{-1} = 200 L_{\perp}/c_s$

In these expressions τ_{cor} and τ_{sat} are the correlation and saturation times of the turbulence, respectively.

---

last update: 2012-03-19 by bscott

7 How to Load the Coretransp CPO

7.1 Where to Put the Ds and Vs additional to the Fluxes

The CPO form for each transport channel contains `diff_eff`, `vconv_eff`, and `flux` as the three pieces of relevance to us.

As the simplest example we take the electron temperature channel (conductive heat flux). The entire section is `coretransp%te_transp` which is of type `transcoefel`. The contents are then given by the relevant section of `euitm_schemas` which is

```
type type_transcoefel
  real(DP),pointer  :: diff_eff(:) => null()
  real(DP),pointer  :: vconv_eff(:) => null()
  real(DP),pointer  :: flux(:) => null()
  type (type_offdiagel) :: off_diagonal
  integer  :: flag=-999999999
endtype
```

So for these the first three elements are all dimensioned to `nrho` which is the number of radial points on the turbulence code's grid. It has been decided (Cyprus 2011 Code Camp) that the IMP4 codes should not try to interpolate onto another CPO grid, because the transport modelling should be left with the flexibility (ie, take the core portion from a global code and the edge layer from an edge code which might have different physics). The off-diagonal elements are left blank by turbulence codes.

The next is the electron density channel (main particle flux). The entire section is `coretransp%ne_transp` which is of type `ne_transp`. The contents are then given by the relevant section of `euitm_schemas` which is

```
type type_ne_transp
  real(DP),pointer  :: diff_eff(:,:) => null()
  real(DP),pointer  :: vconv_eff(:,:) => null()
  real(DP),pointer  :: flux(:) => null()
  type (type_offdiagel) :: off_diagonal
  integer  :: flag=-999999999
endtype
```

which is just like the `transcoefel` except the first two elements have an extra index (the last one). This is dimensioned `1:3` and tells the transport solver which coefficient appears with convective heat fluxes: the multiplier is 0, 3/2, or 5/2, and the relevant D and V go into position 1, 2, or 3, respectively. It is possible to have two channels with different multipliers. The ExB transport has multiplier 3/2 due to the Poynting cancellation, but the parallel velocity component of magnetic flutter has multiplier 5/2. So the conductive D and V coefficients are added together into the `te_transp` channel, and the particle D and V coefficients from the ExB and flutter pieces are put into positions 2 and 3, respectively. Note that although the part of magnetic flutter transport arising from parallel current and radial magnetic fluctuations vanishes in the average, the ion parallel velocity is left over and can contribute in unusual situations.

The ion species are done just like the electrons, but the types are `transcoefion` and `ni_transp` with an extra second index in all variables, for the ion species index. For a pure plasma (e and i only), the dimension of the second index is `nion = 1` and the density D and V now have a third index which is the one for the 0, 3/2, or 5/2 multipliers. All other considerations are as for the electrons.

Examples for how these should be loaded appear in `bscott/public/HDF5_files/global/` and `bscott/public/HDF5_fil` for global and fluxtube codes, respectively, with the filename being `itmfluxes.f90`

---

last update: 2012-03-19 by bscott

8 Neoclassical Bootstrap Current Comparison

8.1 The Models

NeoWes is a simple neoclassical module intended for circular models, using a set of formulae from Wesson's textbook *Tokamaks*. NeOS is Olivier Sauter's package using a set of fitting coefficients to Fokker-Planck results. NeoArt is the multi-species neoclassical code from Arthur Peeters. Here we do a simple comparison.

In a model using simple profiles and the equilibrium code BDSEQ (circular boundary surface, shifted-circle flux surfaces) we compare the run of the bootstrap current and conductivity in the two models as a function of ρ which is defined as the normalised toroidal flux radius `rho_tor_norm` in the CPO. An electron-proton plasma with equal temperatures and densities is assumed with

$$T = T_0 \exp(-8\rho^2/3) \qquad n = n_0 \exp(-8\rho^2/7) \qquad J_\varphi = -\frac{2B_0}{\mu_0 q_0}(1-\rho^2)\exp(-\rho^2)$$

with parameters

$$B_0 = -5.3\,\mathrm{T} \qquad R_0 = 6.2\,\mathrm{m} \qquad a = 2\,\mathrm{m}$$
$$T_0 = 10\,\mathrm{keV} \qquad n_0 = 8.7\times10^{19}\,\mathrm{m}^{-3} \qquad q_0 = -1.5$$

Of course, $Z_{\mathrm{eff}} = 1$. Note that in the CPOs J_φ is R_0 times `jphi` (ie, it is the covariant component). The sign conventions for J_φ and B_0 follow ITER (out of the poloidal plane, on the right of the symmetry axis, from the point of view of an observer in the toroidal midplane outside the tokamak).

The necessary CPO input to BDSEQ is

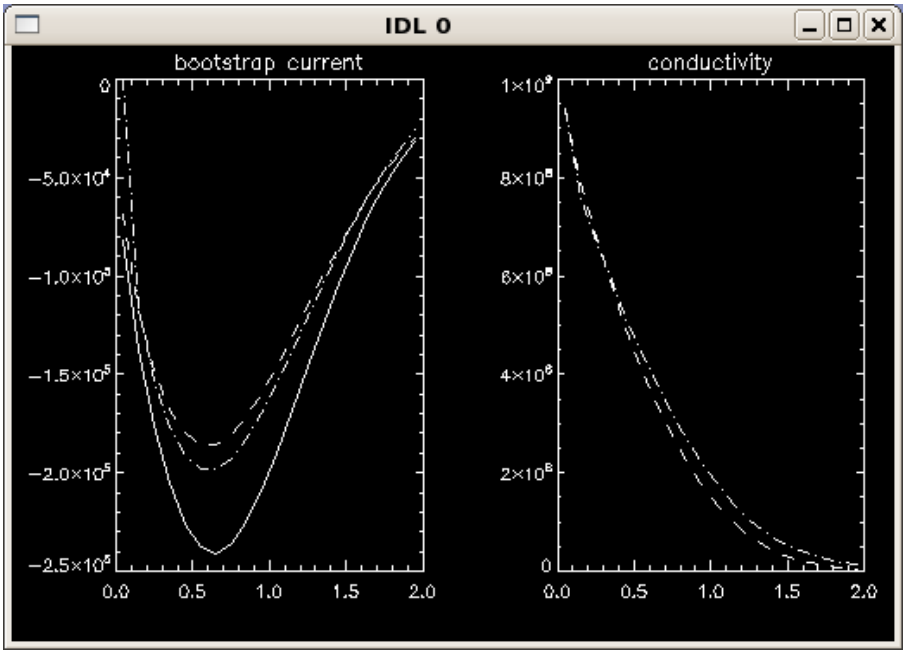
```
equilibrium%global_param%toroid_field%b0
equilibrium%global_param%toroid_field%r0
equilibrium%eqgeometry%a_minor
equilibrium%profiles_1d%rho_tor
equilibrium%profiles_1d%pressure
equilibrium%profiles_1d%jphi
```

The first three are B_0, R_0, a, respectively, and `rho_tor` is $\rho\, a$ on the `coreprof` grid. The pressure and current are found from

jphi $= R_0 J_\varphi$ **pressure** $= 2nk_BT$

also on the `coreprof` grid.

The following graph gives the results of the comparison. The solid lines are from NeoArt, the dashed ones from NeoWes, and the dot-dashed ones from NeOS. In the context of such a circular model the results are in good agreement. On the conductivity, we don't have the one from NeoArt yet so the other two are shown.



last update: 2012-03-28 by bscott

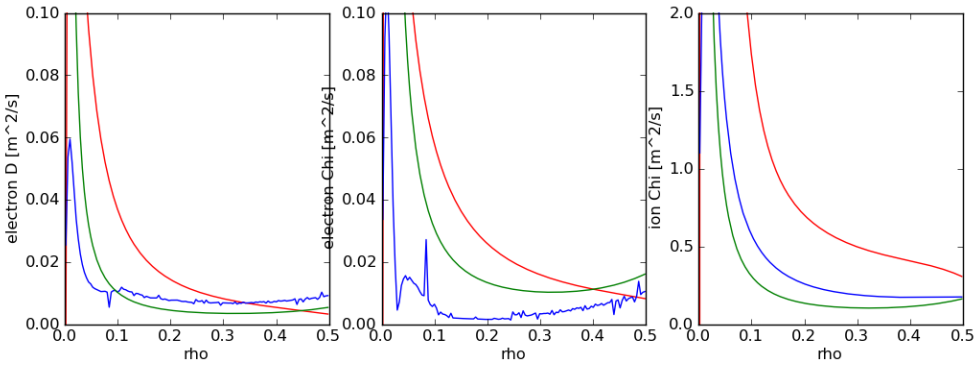
9 Neoclassical Diffusivity Comparison

9.1 The Models

NeoWes is a simple neoclassical module intended for circular models, using a set of formulae from Wesson's textbook *Tokamaks*. NClass is Wayne Houlberg's code, well known as a standard tool. NeoArt is the multi-species neoclassical code from Arthur Peeters. Here we do a simple comparison of the electron and ion diffusivities using the IMP4 Shot 1 Run 1 base case (pure electron-deuterium plasma).

The equilibrium is re-done with BDSEQ, a simple model of circular boundary surface, shifted-circle flux surfaces, since some newer equilibrium CPO elements are needed. Hence only the profiles and a, R_0, and B_0 are used from the database.

We compare the run of the particle and heat diffusivities with toroidal flux radius. The density and temperature profiles are shown, and q goes from about -1.5 to -3.3. The UALPython actor was used to do the plots, which are shown for each neoclassical code. There appears to still be a problem with this implementation of NClass.



last update: 2012-03-28 by bscott

10 Auxiliary IMP4 Actors

10.1 Where to get them

These are located on Gforge in kepleractors/tags/4.10b/imp4/ and kepleractors/trunk/4.10b/imp4/ where the number refers to UAL version and will update in step with the UAL. Most up to date for developers: the versions in /pfs/home/bds/public/4.10b.10_actors which are ready for import (latest: 16h00 Thu 26 Mar 2015). For interpolation purposes rho tor norm must be set and filled.

10.2 IMP4DV

Purpose: accept coretransp fluxes and fill diff_eff and vconv_eff fields

- inputs: equilibrium, coreprof, coretransp
- output: coretransp
- no parameters

10.3 IMP4imp

Purpose: fill the impurity fields, diff_eff and vconv_eff for nz_transp (dynamical alignment rules, use D for electrons and V to set ambipolarity)

- inputs: coreprof, coreimpur, coretransp
- output: coretransp
- no parameters

last update: 2015-03-26 by bds

11 Meetings

11.1 2011/03/7-18 Code Camp in CEA/Cadarache

Slides from the IMP4 contributions to the code camp are here [8]

[8] http://www.rzg.mpg.de/~bds/cyclone/CC_IMP4.pdf

11.2 2010/09/13-17 ITM General Meeting in Lisbon

11.2.1 Posters

- *The IMP4 wrapper for running IMP4 codes in UAL framework* (pdf [9]), by D. Reiser and A. Nielsen

---

last update: 2011-02-10 by konz

---

last update: 2011-04-15 by bscott

---

last update: 2011-03-12 by bscott

[9] https://www.efda-itm.eu/ITM/imports/imp4/public/meetings/20100913-17_Lisbon/Poster_ITM_Lisbon_2010.pdf