Stepped Pressure Equilibrium Code: Flowchart (last updated Dec 2015)

http://w3.pppl.gov/~shudson/Spec/spec.html [S.R.Hudson, et al. Phys. Plasmas 19,112502 (2012)]

main program : xspech

Initialize

1. call readin ; reads `ext.spec` ;
2. call al00aa ; allocates, initializes, ...
3. call gf00aa(P) ; 'packs' geometrical freedom; $\mathbf{x} \equiv (\text{iRbc}_{j,l}, \text{iZbs}_{j,l}, \text{iRbs}_{j,l}, \text{iZbc}_{j,l})^T/\Psi_{j,l}$ for $j = 1, \texttt{mn}$; $l = 1, \texttt{Mvol} - 1$
4. call vo00aa ; $V_l \equiv \int_{\mathcal{V}_l} dv$
5. if Ladiabatic=0, adiabatic$[l] \equiv P_l = p_l V_l^\gamma$

Compute Equilibrium

if `Ngeometricaldof` > 0, solve for $\mathbf{x}$:

1. if Lminimize $= 1$, call pc00aa($\mathbf{x}$)
2. if Lfindzero > 0, find $\mathbf{x}$ s.t. $\mathbf{F_x}[\mathbf{x}] = 0$, if(Igeometry=1 or Igeometry=2), $\mathbf{F_x}[\mathbf{x}] \equiv ([[p + B^2/2]]_{j,l}\, w_j)^T$. if Igeometry=3, $\mathbf{F_x}[\mathbf{x}] \equiv ([[p + B^2/2]]_{j,l}\, w_j, I_{j,l}\, v_j)^T$, where $\mathbf{I} \equiv$ {spectral constraints}. call jk03aa($\mathbf{x}$)

Diagnostics / Output Files

1. `Lcomputederivatives=F` call fc02aa($\mathbf{x}, \mathbf{F_x}$) computes $\mathbf{a}_l[\mathbf{x}; \{\boldsymbol{\psi}_l, K_l, \mu_l, \iota_\pm\}]$.
2. if(LHevalues, LHevectors, Lperturbed, or Lcheck=5), call he01aa
3. call ra00aa(W) ; write $\mathbf{a}_l$ to `.ext.sp.A`
4. call writin ; write `ext.sp.end`, etc.
5. do $l = 1$, `Mvol` ! begin parallel
   if Lcheck=1, call jo00aa(l); $|\nabla \times \mathbf{B}_l - \mu_l \mathbf{B}_l|$;
   call sc00aa(l); B_s, B_θ, B_ζ;
   call pp00aa(l); constructs Poincaré plot;
   `enddo` ! end parallel

jk03aa($\mathbf{x}$)

1. `Lcomputederivatives=F` call fc02aa($\mathbf{x}, \mathbf{F_x}$)
2. if $|\mathbf{F_x}| <$ forcetol, return
3. iterate on $\delta\mathbf{x} = -(\nabla_\mathbf{x} F_\mathbf{x})^{-1} \cdot \mathbf{F_x}$ to find $\mathbf{F_x}(\mathbf{x}) = 0$.
4. if Lfindzero $= 1$, `Lcomputederivatives=F` uses `C05NDF`(fc02aa; $\mathbf{x}$; c05xtol,c05factor) function values only
5. if Lfindzero $= 2$, `Lcomputederivatives=T` allocate `hessian`$\equiv \nabla_\mathbf{x} \mathbf{F_x}$ uses `C05PDF`(fc02aa; $\mathbf{x}$; c05xtol,c05factor) user supplied derivative deallocate `hessian`

readin

1. read input namelists from `ext.spec`
2. normalize toroidal flux, $\psi_{t,l} \to \psi_{t,l}/\psi_{t,\text{Nvol}}$.
3. $\sum_1^{\text{mn}} f_j \equiv \sum_0^{0} \sum_0^{\text{Ntor}} f_{m,n} + \sum_1^{\text{Mpol}} \sum_{-\text{Ntor}}^{\text{Ntor}} f_{m,n}$
4. if Lfreeboundary=0, `Mvol`=Nvol, if Lfreeboundary=1, `Mvol`=Nvol+1.
5. set geometrical regularization factor, e.g. for Igeometry= 3, if $m_j = 0$, $\Psi_{j,l} \equiv \psi_{t,l}^{1/2}$, if $m_j \neq 0$, $\Psi_{j,l} \equiv \psi_{t,l}^{m_j/2}$, for $l = 1$, Nvol.
6. if Linitialize=0, read $\texttt{iRbc}_{j,l}$, $\texttt{iZbs}_{j,l}$, ... if Linitialize=1, interpolate: e.g. $\texttt{iRbc}_{j,l} = \texttt{Rbc}_{j,0} + (\texttt{Rbc}_{j,\text{Nvol}} - \texttt{Rbc}_{j,0})\Psi_{j,l}$

al00aa

1. `Ngeometricaldof`$\approx$ (`Mvol`-1)mn ; (depends on `Igeometry` & `Istellsym`).
2. $\Delta\psi_{t,l} = (\psi_{t,l} - \psi_{t,l-1})\Phi_{edge}/2\pi$
   $\Delta\psi_{p,l} = (\psi_{p,l} - \psi_{p,l-1})\Phi_{edge}/2\pi$

   `do` $l = 1$, `Mvol`
3. if(Igeometry=2 or Igeometry=3) & l=1 `Lcoordinatesingularity=T`
4. if $l \leq$ Nvol, `Lplasmaregion=T`, if $l >$ Nvol, `Lvacuumregion=T`.
5. if `Lplasmaregion`{ $\mathbf{a}_l \equiv (A_{\theta,e,j,p}, A_{\zeta,e,j,p}, A_{\theta,o,j,p}, A_{\zeta,o,j,p})^T$, $\boldsymbol{\psi}_l \equiv (\Delta\psi_{t,l}, \Delta\psi_{p,l})^T$. } if `Lvacuumregion`{ $\mathbf{a}_l \equiv (\Phi_{e,j,p}, \Phi_{o,j,p})^T$, $\boldsymbol{\psi}_l \equiv (I_{tor}, G_{pol})^T$. }
6. if `Lplasmaregion`{ if `Lcoordinatesingularity`, $\bar{s} = (s+1)/2$, $\varphi_j \equiv \bar{s}^{m_j/2}$.
   $$A_\theta = \sum_{j,p} \begin{matrix} A_{\theta,e,j,p}\varphi_j(s)T_p(s)\cos(m_j\theta - n_j\zeta) \\ A_{\theta,o,j,p}\varphi_j(s)T_p(s)\sin(m_j\theta - n_j\zeta) \end{matrix}$$
   $$A_\zeta = \sum_{j,p} \begin{matrix} A_{\zeta,e,j,p}\varphi_j(s)T_p(s)\cos(m_j\theta - n_j\zeta) \\ A_{\zeta,o,j,p}\varphi_j(s)T_p(s)\sin(m_j\theta - n_j\zeta) \end{matrix}$$
   }.
7. if `Lvacuumregion`,
   $$\Phi = \sum_{j,p} \begin{matrix} \Phi_{e,j,p}T_p(s)\cos(m_j\theta - n_j\zeta) \\ \Phi_{eo,j,p}T_p(s)\sin(m_j\theta - n_j\zeta) \end{matrix}.$$
   where $T_p(s) \equiv$ Chebyshev polynomial

   `enddo`
8. if Linitgues=2, call ra00aa(R) ; reads $\mathbf{a}_{l=1,\texttt{Mvol}}$ from `.AtAzmn` .
9. if LBeltrami=1,3,5,7, `LBsequad=T`
   if LBeltrami=2,3,6,7, `LBnewton=T`
   if LBeltrami=4,5,6,7, `LBlinear=T`

fc02aa($\mathbf{x}, \mathbf{F_x}$)

do $l = 1$, `Mvol` ! begin parallel

1. if(Igeometry=2, Igeometry=3) & $l = 1$, `Lcoordinatesingularity=T`
2. if $l \leq$ Nvol, `Lplasmaregion=T` if $l >$ Nvol, `Lvacuumregion=T`
3. allocate 'Beltrami matrices', $\mathcal{A}[\mathbf{x}], \mathcal{B}[\mathbf{x}], \mathcal{C}[\mathbf{x}], \mathcal{D}[\mathbf{x}], \mathcal{E}[\mathbf{x}], \mathcal{F}[\mathbf{x}]$.
4. call ma00ab(A, l) allocate `TTee(1:6,1:L,1:L,1:mn,1:mn)`, ... call ma00aa
   $$\texttt{TTee}_{1,l,p,i,j} \equiv \iiint \varphi_i T_l\, \varphi_j T_p\, e^{i\alpha_i} \frac{g_{\mu\nu}}{\sqrt{g}} e^{i\alpha_j}\, ds\, d\theta\, d\zeta$$
   where $\alpha_i \equiv m_i\theta - n_i\zeta$.
5. if `Lplasmaregion`, call ma01ag if `Lvacuumregion`, call va00aa compute $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}, \mathcal{E}, \mathcal{F}$.
   $$W_l = \int_{\mathcal{V}_l} \left(\frac{p}{\gamma - 1} + \frac{B^2}{2}\right) dv = \frac{1}{2}\mathbf{a}_l^T.\mathcal{A}.\mathbf{a}_l + \boldsymbol{\psi}_l^T.\mathcal{B}.\mathbf{a}_l + \boldsymbol{\psi}_l^T.\mathcal{C}.\boldsymbol{\psi}_l,$$
   $$K_l = \int_{\mathcal{V}} \mathbf{A} \cdot \mathbf{B}\, dv = \frac{1}{2}\mathbf{a}_l^T.\mathcal{D}.\mathbf{a}_l + \boldsymbol{\psi}_l^T.\mathcal{E}.\mathbf{a}_l + \boldsymbol{\psi}_l^T.\mathcal{F}.\boldsymbol{\psi}_l.$$
6. call ma02aa(l) returns $\mathbf{a}_l[\mathbf{x}; \{\boldsymbol{\psi}_l, K_l, \mu_l, \iota_\pm\}]$.
7. call vo00aa ; $V_l \equiv \int_{\mathcal{V}_l} dv$
8. `do` $i = 0, 1$; on inner/outer interface; call bb00aa ; returns $[[p + B^2/2]]$, I, `enddo`
9. if `Lcomputederivatives=T`, `do i=0,1 ; do j=1,mn ;` call ma00aa compute $\partial_x\mathcal{A}$, $\partial_x\mathcal{B}$, $\partial_x\mathcal{C}$, $\partial_x\mathcal{B}$, $\partial_x\mathcal{E}$, $\partial_x\mathcal{F}$. call ma01ag or va00aa $\partial_x \mathbf{F} \equiv \mathcal{M}^{-1} \cdot (\partial_x \mathbf{b} - \partial_x\mathcal{M} \cdot \mathbf{x})$ call tr00ab ; $\partial_x \mu_l|_\iota$ call vo00aa ; $\partial_x V_l$ call bb00aa ; $\partial_x B^2$ `enddo ; enddo`
10. call ma00ab(D, l) ; deallocate `TTee`, etc.

enddo ! end parallel

1. call bc00aa(l) ; broadcast ;
2. construct $\mathbf{F_x}[\mathbf{x}]$
3. if `Lcomputederivatives=T`, construct $\nabla_\mathbf{x} \mathbf{F_x} \equiv \left.\dfrac{\partial F_{x,i}}{\partial x_j}\right|_{\{\boldsymbol{\psi}, K_l, \mu_l, \iota_\pm\}}$

ma02aa(l)

1. if `Lplasmaregion` and `LBsequad`,
2. if `Lplasmaregion` and `LBnewton`, must provide initial guess for $(\mu_l, \mathbf{a}_l)^T$
   i. only for Lconstraint= 2
   $$F_\mathbf{a}\begin{pmatrix}\mu_l \\ \mathbf{a}_l\end{pmatrix} \equiv W_l - \frac{\mu_l}{2}(K_l - \text{helicity}[l])$$
   iterate on
   $$\delta\begin{pmatrix}\mu_l \\ \mathbf{a}_l\end{pmatrix} = -(\nabla^2_{\mu_l,\mathbf{a}_l} F_\mathbf{a})^{-1} \cdot \nabla_{\mu_l,\mathbf{a}_l} F_\mathbf{a}$$
   to find $\nabla_{\mu_l,\mathbf{a}_l} F_\mathbf{a} = 0$.
   uses `C05PBF`(df00aa; $(\mu_l, \mathbf{a}_l)^T$; mupftol),
3. if `Lplasmaregion` and `LBlinear`, must provide $(\mu_l, \Delta\psi_{p,l})^T$
   i. if Lconstraint= 0, call mp00ac($l, \mu_l, \Delta\psi_{p,l}$)
   ii. if Lconstraint= 1, iterate on $(\mu_l, \Delta\psi_{p,l})^T$ to find
   $$\mathbf{f}\begin{pmatrix}\mu_l \\ \Delta\psi_{p,l}\end{pmatrix} = \begin{pmatrix}\iota_{inn} - \text{oita}[l-1] \\ \iota_{out} - \text{iota}[l]\end{pmatrix} = \mathbf{0}$$
   uses `C05PBF`(mp00ac; $(\mu_l, \Delta\psi_{p,l})^T$; mupftol)
   iii. if Lconstraint= 2, not yet supported, try LBeltrami= 2.
4. if `Lvacuumregion`,

mp00ac($l, \mu_l, \Delta\psi_{p,l}$)

1. given $(\mu_l, \Delta\psi_{p,l})^T$, solve for $\mathbf{a}_l$, $(\mathcal{A}_l + \mu_l\mathcal{D}_l) \cdot \mathbf{a}_l = (\mathcal{B}_l + \mu_l\mathcal{E}_l)$
2. if Lconstraint= 1, compute interface transform, call tr00ab ; $\theta_s = \theta + \lambda(\theta, \zeta)$

df00aa(`iflag`$, l, \mu_l, \mathbf{a}_l$)

1. if `iflag`=1, compute first derivatives, $\dfrac{\partial F_\mathbf{a}}{\partial \mu_l}$ and $\dfrac{\partial F_\mathbf{a}}{\partial \mathbf{a}_l}$.
2. if `iflag`=2, compute second derivatives, $\dfrac{\partial^2 F_\mathbf{a}}{\partial\mu_l\partial\mu_l}$, $\dfrac{\partial^2 F_\mathbf{a}}{\partial\mathbf{a}_l\partial\mu_l}$ and $\dfrac{\partial^2 F_\mathbf{a}}{\partial\mathbf{a}_l\partial\mathbf{a}_l}$.

he01aa

1. `Lcomputederivatives=T` allocate `hessian`$\equiv \nabla_\mathbf{x}\mathbf{F_x}$
2. call fc02aa
3. if Lcheck= 5, compare $\nabla_\mathbf{x}\mathbf{F_x}$ with finite-difference estimate
4. if(LHevalues,LHevectors), compute eigenvalues & eigenvectors of $\nabla_\mathbf{x}\mathbf{F_x}$
5. if Lperturbed, compute linear displacement, $\delta\mathbf{x} = -(\nabla_\mathbf{x}\mathbf{F_x})^{-1} \cdot \nabla_b \mathbf{F_x} \cdot \delta b$;
6. deallocate `hessian`

`typefont indicates internal variable or NAG routine`; red indicates input variable; green indicates input initial guess; blue indicates subroutine;