Wendelstein 7-X

Max-Planck-Institut für Plasmaphysik

Spectral Condensation in VMEC

- the latest episode in code archeology -

Meeting of the SPECtaculars

J. Schilling, M. Cianciosa, M. Landreman, F. Hindenlang, O. Maj, J. Geiger

2021-05-04

Outline

- **Introduction**
- **Literature Review: Spectral Condensation in VMEC (brief !)**
- **VMEC algorithm overview**
- **Details…**

Introduction

Max-Planck-Institut für Plasmaphysik

- **VMEC is a spectral MHD equilibrium code**
- **assumes nested flux surfaces → radial coordinate == encl. toroidal flux**
- **on surfaces: angle-like coordinates** θ (poloidal), ζ (toroidal) [2π - periodic]
- **on each surface: 2-dim. Fourier expansion of R, Z,** λ
  →Inverse coordinate representation $X(s, \theta, \zeta)$ with $X \in \{R, Z, \lambda\}$
- **Toroidal coordinate** ζ **== cylindrical angle** ϕ
- **Poloidal coordinate** θ **has tangential degree of freedom**
  - can be exploited to make most efficient use of available truncated Fourier coefficients
  - λ is used to represent straight magnetic field lines

Goals: $\boxed{\textbf{unique poloidal angle \& „economical" Fourier spectrum of } R, Z, \lambda}$

Literature Review: Spectral Condensation in VMEC

Wendelstein 7-X

IPP Max-Planck-Institut für Plasmaphysik

1. Hirshman & Whitson (1983) „Steepest Descent Method …“
   - Basic idea of VMEC, variational method, MHD forces, first results, …
2. Hirshman & Weitzner (1985) „A convergent spectral representation …“
   - „quasi-polar“ angle constraint, conditions on individual Fourier coefficients, …
3. Hirshman & Meier (1985) „Optimized Fourier representations …“
   - „spectral condensation“, <M>, DESCUR/SCRUNCH, energy principle, „VMEC without MHD“, …
4. Hirshman & Hogan (1986) „ORMEC: A Three-Dimensional MHD … Code“
   - Consolidation of algorithm, some numerical details, constrained-current method, m=1 constraint
5. Hirshman & Lee (1986) „MOMCON: A Spectral Code for …. 3D MHD Equilibira“
   - Integration of „spectral condensation forces“ with MHD forces, relatively close to source code, …
6. Hirshman, Schwenn & Nührenberg (1990) „Improved Radial Differencing …“
   - Interaction between full- and half-radial mesh, even-m/odd-m decomposition
7. Hirshman & Betancourt (1991) „Preconditioned Descent Algorithm …“
   - 1D-preconditioner (radial), some „inspiration“ for time-step algorithm, …
8. Hirshman & Breslau (1998) „Explicitly spectrally optimized Fourier series …“
   - Quasi-polar representation vs. „full“ spectral condensation, in code: _HBANGLE, …

VMEC algorithm overview

- **Initialization, read input file, initial guess, …**
- **Multi-grid iterations: # of surfaces increasing, force tolerance decreasing**
  - Iterative „time evolution“ of Fourier coefficients
    == viscous motion along force vector (~ „conjugate-gradient“ optimizer)
    - Inverse Fourier transform from R, Z, Lambda Fourier coefficients into real space
    - Evaluate forces in real space: need magnetic field components, iota, free-boundary contribution (NESTOR), constraint forces, preconditioner, …
    - Fourier-transform real-space forces to Fourier coefficients of forces
    - Check force residual in Fourier space
- **Final diagnostics, write output file(s)**

VMEC boundary: user input

Max-Planck-Institut für Plasmaphysik

$$R(\theta,\zeta) = \sum_{m,n} \left[Rbc(n,m)\cos(m\theta - n\zeta) \boxed{+ Rbs(n,m)\sin(m\theta - n\zeta)}\right]$$
$$Z(\theta,\zeta) = \sum_{m,n} \left[Zbs(n,m)\sin(m\theta - n\zeta) \boxed{+ Zbc(n,m)\cos(m\theta - n\zeta)}\right]$$

if **non-stellarator-symmetric**, check if Δ = 0 where $\Delta \equiv \dfrac{Rbs(0,1) - Zbc(0,1)}{|Rbc(0,1)| + |Zbs(0,1)|}$

→ If not, re-scale coefficients as follows:

$$Rbs(n,m) \leftarrow Rbs(n,m)\cos(m\Delta) - Rbc(n,m)\sin(m\Delta)$$
$$Rbc(n,m) \leftarrow Rbs(n,m)\sin(m\Delta) + Rbc(n,m)\cos(m\Delta)$$
$$Zbs(n,m) \leftarrow Zbs(n,m)\cos(m\Delta) - Zbc(n,m)\sin(m\Delta)$$
$$Zbc(n,m) \leftarrow Zbs(n,m)\sin(m\Delta) + Zbc(n,m)\cos(m\Delta)$$

Resulting operation: **shift of poloidal angle**

$$R(\theta,\zeta) = \sum_{m,n} \left[Rbc(n,m)\cos(m(\theta \boxed{+\Delta}) - n\zeta) + Rbs(n,m)\sin(m(\theta \boxed{+\Delta}) - n\zeta)\right]$$
$$Z(\theta,\zeta) = \sum_{m,n} \left[Zbs(n,m)\sin(m(\theta \boxed{+\Delta}) - n\zeta) + Zbc(n,m)\cos(m(\theta \boxed{+\Delta}) - n\zeta)\right]$$

Conversion to „internal“ Fourier series representation

(analogously for Z)

$$R(\theta,\zeta) = \sum_{m=0,\, n=-ntor}^{mpol-1,\, ntor} \left[Rbc(n,m)\cos(m\theta - n\zeta) + Rbs(n,m)\sin(m\theta - n\zeta) \right]$$

$$R(\theta,\zeta) = \sum_{m=0,\, n=-ntor}^{mpol-1,\, ntor} \{ \quad Rbc(n,m)\left[\cos(m\theta)\cos(n\zeta) + \sin(m\theta)\sin(n\zeta)\right]$$
$$+ Rbs(n,m)\left[\sin(m\theta)\cos(n\zeta) - \cos(m\theta)\sin(n\zeta)\right]\}$$

$$\begin{aligned} R^{cc}_{m,|n|} &\Leftarrow \quad Rbc(n,m) \\ R^{ss}_{m,|n|} &\Leftarrow \quad Rbc(n,m) \\ R^{sc}_{m,|n|} &\Leftarrow \quad Rbs(n,m) \\ R^{cs}_{m,|n|} &\Leftarrow -Rbs(n,m) \end{aligned}$$

for n=-ntor, ..., ntor

Split into four coefficients → only n>=0 required!

$$R(\theta,\zeta) = \sum_{m=0\, \boxed{n=0}}^{mpol-1,\, ntor} \Big[\quad R^{cc}_{mn}\cos(m\theta)\cos(n\zeta) + R^{ss}_{mn}\sin(m\theta)\sin(n\zeta)$$
$$+ R^{sc}_{mn}\sin(m\theta)\cos(n\zeta) + R^{cs}_{mn}\cos(m\theta)\sin(n\zeta) \Big]$$

Coefficient mapping: user input to VMEC-internal arrays

Max-Planck-Institut für Plasmaphysik

	R: stellarator-symmetric Z: non-stellarator-symmetric	Z: stellarator-symmetric R: non-stellarator-symmetric
2D and 3D (ntor >= 0) Tokamak and Stellarator	$X^{cc}_{mn} = \begin{cases} X^{\cos}_{m0} & : n = 0 \\ X^{\cos}_{mn} + X^{\cos}_{m,-n} & : n > 0 \end{cases}$	$X^{sc}_{mn} = \begin{cases} 0 & : n = 0, m = 0 \\ X^{\sin}_{m0} & : n = 0, m > 0 \\ X^{\sin}_{mn} + X^{\sin}_{m,-n} & : n > 0, m > 0 \end{cases}$
only 3D (ntor > 0) only for Stellarator	$X^{ss}_{mn} = \begin{cases} 0 & : n = 0 \\ X^{\cos}_{mn} - X^{\cos}_{m,-n} & : n > 0, m > 0 \end{cases}$	$X^{cs}_{mn} = \begin{cases} 0 & : n = 0 \\ -X^{\sin}_{mn} + X^{\sin}_{m,-n} & : n > 0 \end{cases}$

used within VMEC

from user input

X ϵ {R, Z}

2D: ntor = 0 (Tokamak)

3D: ntor > 0 (Stellarator)

Check for change of sign-of-Jacobian; flip θ otherwise

$$\left.\begin{aligned} r_{\text{test}} &= \sum_{n=0}^{N} R_{1n}^{cc} \\ z_{\text{test}} &= \sum_{n=0}^{N} Z_{1n}^{sc} \end{aligned}\right\}$$

if sgn(r_{test}) != sgn(z_{test}), need to flip θ

Current working hypothesis: (thanks to J. Geiger!)
r_{test}, z_{test} are related to the leading terms of d(R,Z)/dθ at (θ, ζ)=(π/2, 0) for R and at (θ, ζ)=(0, 0) for Z.
If the leading derivatives have the same sign, the path is probably going counter-clockwise, with different signs it is likely going clockwise.

This is how θ is flipped in the coefficient arrays (X=R,Z, *=c,s):

$$X_{mn}^{c*} \leftarrow X_{mn}^{c*} \begin{cases} -1 & : m \text{ odd} \\ +1 & : m \text{ even} \end{cases}$$

$$X_{mn}^{s*} \leftarrow X_{mn}^{s*} \begin{cases} +1 & : m \text{ odd} \\ -1 & : m \text{ even} \end{cases}$$

for all m=1, …, (mpol-1)

m=1 constraint on geometry of flux surfaces

Max-Planck-Institut für Plasmaphysik

for all n=0, …, ntor

„proper“ names of these quantities

$$R^{+}_{\mathrm{sym},n} = R^{ss}_{1n} \leftarrow \frac{1}{2}\left(R^{ss}_{1n} + Z^{cs}_{1n}\right)$$

$$R^{-}_{\mathrm{sym},n} = Z^{cs}_{1n} \leftarrow \frac{1}{2}\left(R^{ss}_{1n} - Z^{cs}_{1n}\right)$$

Only if 3D

… and this is just where they are stored within the code

$$R^{+}_{\mathrm{asym},n} = R^{sc}_{1n} \leftarrow \frac{1}{2}\left(R^{sc}_{1n} + Z^{cc}_{1n}\right)$$

$$R^{-}_{\mathrm{asym},n} = Z^{cc}_{1n} \leftarrow \frac{1}{2}\left(R^{sc}_{1n} - Z^{cc}_{1n}\right)$$

Only if non-stellarator-symmetric

Forward transform (as shown above):

→ done only on initial guess; then maintained during iterations

Reverse operation (see next slides):

→ convert_sym, convert_asym in totzsp()

mscale(m), nscale(n): just some normalization factors ...

$$m_{\text{scale}} = \begin{cases} 1 & : m = 0 \\ \sqrt{2} & : m > 0 \end{cases} \Rightarrow \oint \cos(m\theta)\cos(m'\theta)\mathrm{d}\theta = \frac{1}{2}\delta_{m,m'}$$

$$n_{\text{scale}} = \begin{cases} 1 & : n = 0 \\ \sqrt{2} & : n > 0 \end{cases} \Rightarrow \oint \cos(n\zeta)\cos(n'\zeta)\mathrm{d}\zeta = \frac{1}{2}\delta_{n,n'}$$

included in Fourier basis functions (small excerpt):

e.g.

$$\texttt{cosmu}(i,m) = \cos\left(2\pi\frac{i-1}{n_\theta}m\right)m_{\text{scale}}$$

$$\texttt{sinmu}(i,m) = \sin\left(2\pi\frac{i-1}{n_\theta}m\right)m_{\text{scale}}$$

poloidal Fourier basis

$$\texttt{cosnv}(k,n) = \cos\left(2\pi\frac{k-1}{n_\zeta}n\right)n_{\text{scale}}$$

$$\texttt{sinnv}(k,n) = \sin\left(2\pi\frac{k-1}{n_\zeta}n\right)n_{\text{scale}}$$

toroidal Fourier basis

These normalization factors need to be factored out when reading user input for geometry!

Spectral Width from Fourier coefficients

Wendelstein 7-X

IPP Max-Planck-Institut für Plasmaphysik

$$\mathrm{xmpq}(m,1) = m(m-1)$$

actual term used to „create“ constraint forces

Need to dig deeper!

$$\left.\begin{aligned}\mathrm{xmpq}(m,2) &= m^{\mathrm{pexp}} \\ \mathrm{xmpq}(m,3) &= m^{\mathrm{pexp}+1}\end{aligned}\right\}$$

p=4, q=1 and with Q=p+q we identify this as the case Q_2=5 in Hirshman & Meier (1985)

$$\mathrm{pexp} = 4 \text{ (fixed)}$$

$f(m) = m^p(m^q - M)$

[maybe p=1,q=1, M=1 in f(m) near Eqn. (5) in Hirshman, Meier (1985)]

But this is only used as a screen diagnostic in VMEC !?!?!?

(threed1-file from VMEC run for W7-X)

S	<RADIAL FORCE>	TOROIDAL FLUX	IOTA	<JSUPU>	<JSUPV>	d(VOL)/d(PHI)	d(PRES)/d(PHI)	<M>
0.00E+00	-6.45E-03	-0.0000E+00	8.6631E-01	-3.619E+05	8.330E+01	1.602E+01	5.834E+04	1.000
2.00E-02	-3.07E-03	-3.4208E-02	8.6694E-01	-3.508E+05	1.588E+02	1.600E+01	5.620E+04	1.016
4.00E-02	3.05E-04	-6.8416E-02	8.6762E-01	-3.397E+05	2.345E+02	1.599E+01	5.406E+04	1.038
6.00E-02	3.68E-04	-1.0262E-01	8.6831E-01	-3.270E+05	2.981E+02	1.597E+01	5.204E+04	1.055
8.00E-02	2.93E-04	-1.3683E-01	8.6905E-01	-3.148E+05	3.572E+02	1.596E+01	5.013E+04	1.073
1.00E-01	1.66E-04	-1.7104E-01	8.6999E-01	-3.034E+05	4.146E+02	1.595E+01	4.832E+04	1.091
1.20E-01	8.59E-05	-2.0525E-01	8.7116E-01	-2.925E+05	4.717E+02	1.593E+01	4.661E+04	1.109
1.40E-01	2.90E-05	-2.3946E-01	8.7253E-01	-2.822E+05	5.294E+02	1.592E+01	4.499E+04	1.128
1.60E-01	9.49E-09	-2.7366E-01	8.7408E-01	-2.725E+05	5.884E+02	1.591E+01	4.345E+04	1.146
1.80E-01	-2.08E-05	-3.0787E-01	8.7579E-01	-2.633E+05	6.489E+02	1.590E+01	4.199E+04	1.165
2.00E-01	-3.05E-05	-3.4208E-01	8.7762E-01	-2.545E+05	7.114E+02	1.588E+01	4.060E+04	1.184
2.20E-01	-3.81E-05	-3.7629E-01	8.7956E-01	-2.461E+05	7.760E+02	1.587E+01	3.928E+04	1.202
2.40E-01	-4.07E-05	-4.1050E-01	8.8157E-01	-2.382E+05	8.428E+02	1.586E+01	3.803E+04	1.221
2.60E-01	-4.25E-05	-4.4470E-01	8.8366E-01	-2.306E+05	9.119E+02	1.585E+01	3.683E+04	1.239
2.80E-01	-4.16E-05	-4.7891E-01	8.8579E-01	-2.233E+05	9.832E+02	1.584E+01	3.569E+04	1.257
3.00E-01	-4.03E-05	-5.1312E-01	8.8797E-01	-2.164E+05	1.057E+03	1.583E+01	3.460E+04	1.276

$$M(p,q) \equiv \frac{\sum_{m,n} m^{p+q}(R_{mn}^2 + Z_{mn}^2)}{\sum_{m,n} m^{p}(R_{mn}^2 + Z_{mn}^2)}$$

include m = 1, ..., (mpol-1)

n = 0 , ..., ntor

Extra-/Interpolation of initial guess for geometry

- **Initial guess of coefficients for R,Z in volume from interpolation**
  - **m=0** - modes interpolated between axis and boundary

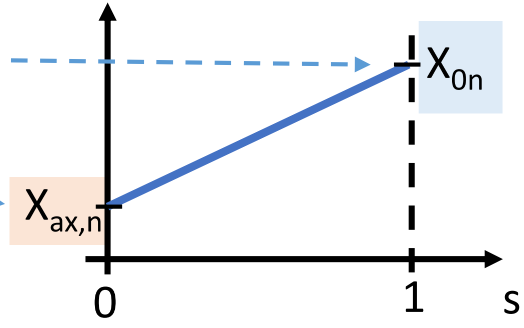
$$R^{cc}_{0n}(s_j) \leftarrow \frac{1}{m_{\text{scale}} \cdot n_{\text{scale}}} \left(s_j R^{cc}_{0n} + (1-s_j) R^{cc}_{\text{ax},n}\right)$$

$$Z^{cs}_{0n}(s_j) \leftarrow \frac{1}{m_{\text{scale}} \cdot n_{\text{scale}}} \left(s_j Z^{cs}_{0n} - (1-s_j) Z^{cs}_{\text{ax},n}\right)$$

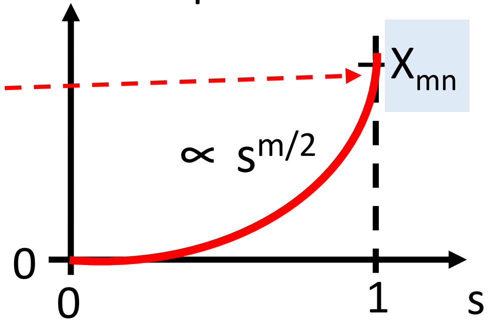
$$R^{cs}_{0n}(s_j) \leftarrow \frac{1}{m_{\text{scale}} \cdot n_{\text{scale}}} \left(s_j R^{cs}_{0n} - (1-s_j) R^{cs}_{\text{ax},n}\right)$$

$$Z^{cc}_{0n}(s_j) \leftarrow \frac{1}{m_{\text{scale}} \cdot n_{\text{scale}}} \left(s_j Z^{cc}_{0n} + (1-s_j) Z^{cc}_{\text{ax},n}\right)$$

Linear interpolation in s for **m=0**



polynomial interpolation in s for **m>0**



  - **m>0** – modes extrapolated from boundary towards axis

$$X^{**}_{mn}(s_j) \leftarrow \frac{1}{m_{\text{scale}} \cdot n_{\text{scale}}} s_j^{m/2} X^{**}_{mn}$$

… and axis

user-input boundary… Rbc, Zbs, … raxis_cc, zaxis_cs, …

Inverse-Fourier transform of geometry to real space (1/2)

- **initial guess for the geometry of all flux surfaces is now available**
  →first iteration: transform geometry coefficients back to real space and eval MHD forces etc.
- **first need to undo m=1 constraint:**

convert_sym()

$$\boxed{\begin{aligned} R^{ss}_{1n} &\leftarrow R^{+}_{\mathrm{sym},n} + R^{-}_{\mathrm{sym},n} \\ Z^{cs}_{1n} &\leftarrow R^{+}_{\mathrm{sym},n} - R^{-}_{\mathrm{sym},n} \end{aligned}} \begin{aligned} &\equiv R^{ss}_{1n} + Z^{cs}_{1n} \\ &\equiv R^{ss}_{1n} - Z^{cs}_{1n} \end{aligned}$$

convert_asym()

$$\boxed{\begin{aligned} R^{sc}_{1n} &\leftarrow R^{+}_{\mathrm{asym},n} + R^{-}_{\mathrm{asym},n} \\ Z^{cc}_{1n} &\leftarrow R^{+}_{\mathrm{asym},n} - R^{-}_{\mathrm{asym},n} \end{aligned}} \begin{aligned} &\equiv R^{sc}_{1n} + Z^{cc}_{1n} \\ &\equiv \underbrace{R^{sc}_{1n} - Z^{cc}_{1n}}_{} \end{aligned}$$

this is where the coefficients are stored in the code

Inverse-Fourier transform of geometry to real space (2/2)

Max-Planck-Institut für Plasmaphysik

X_{mn} are now available in „physical“ form (m=1 - constraint not active)

→ can do inverse Fourier transform to real space now!

<u>but</u>: handle even / odd-m individually (needed for precise radial derivatives)

$$X = X_e + \sqrt{\Phi} X_o,$$

for X ϵ {R, Z}

$$\texttt{scalxc}(s_j, m) = \begin{cases} 1 & : m \text{ even} \\ 1/\sqrt{s_j} & : s_j > 0, m \text{ odd} \\ 1/\sqrt{\texttt{ns}-1} & : s_j = 0, m \text{ odd} \end{cases}$$

$$X_e = \sum_{\text{even } m;\, n} X_{mn}(\Phi) \exp[i(m\theta - n\zeta)],$$

for $j = 1, \ldots, \text{ns},$

$m = 0, \ldots, (\texttt{mpol} - 1)$

$$X_o = \Phi^{-1/2} \sum_{\text{odd } m;\, n} X_{mn}(\Phi) \exp[i(m\theta - n\zeta)]$$

Hirshman, Schwenn & Nührenberg (1990) „Improved Radial Differencing …“

Spectral-Constraint „source terms“: Rcon, Zcon

$$R^{\mathrm{con}}_{e,o} = \sum_{m,n} m(m-1) \left[R^{\cos}_{mn} \cos(m\theta - n\zeta) + \ldots\right]$$

additional terms for non-stellarator-symmetric geometry

$$Z^{\mathrm{con}}_{e,o} = \sum_{m,n} \underbrace{m(m-1)}_{xmpq(\mathrm{m},1)} \left[Z^{\sin}_{mn} \sin(m\theta - n\zeta) + \ldots\right]$$

Next assemble Rcon, Zcon from even/odd-m contributions:

$$R^{\mathrm{con}} = R^{\mathrm{con}}_e + \sqrt{s} R^{\mathrm{con}}_o$$

$$Z^{\mathrm{con}} = Z^{\mathrm{con}}_e + \sqrt{s} Z^{\mathrm{con}}_o$$

side note:

Compare R^{con}, Z^{con} to X and Y in Hirshman, Meier (1985):

$$I(\theta,\phi) = X(\theta,\phi)\, x_\theta + Y(\theta,\phi)\, y_\theta, \quad \text{(5a)}$$

$$X(\theta,\phi) = \sum_{m=1,n} f(m)\, x_{mn} \cos(m\theta - n\phi), \quad \text{(5b)}$$

$$Y(\theta,\phi) = \sum_{m=1,n} f(m)\, y_{mn} \sin(m\theta - n\phi), \quad \text{(5c)}$$

$$f(m) = m^p (m^q - M)$$

for p=1, q=1, M=1, f(m) = *xmpq*(m,1)

Initial Rcon0, Zcon0: extrapolation into volume

$$R_0^{\mathrm{con}} = s_j R^{\mathrm{con},(1)}$$

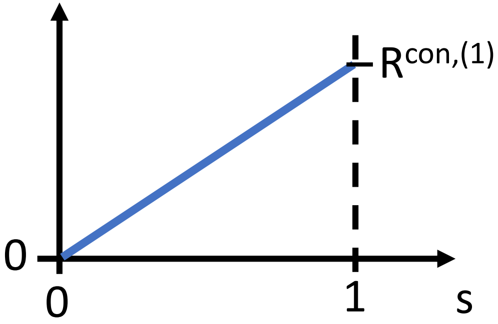
$$Z_0^{\mathrm{con}} = s_j Z^{\mathrm{con},(1)}$$

R^{con}, Z^{con} (previous slide) from first iteration for current # of surfaces

Inter-/extrapolation towards axis

Fixed for current multi-grid iteration

Linear interpolation in s



then used for constraint force:

$$R_{\mathrm{eff}}^{\mathrm{con}} = \boxed{R^{\mathrm{con},(i)} - R_0^{\mathrm{con}}}$$

$$Z_{\mathrm{eff}}^{\mathrm{con}} = \boxed{Z^{\mathrm{con},(i)} - Z_0^{\mathrm{con}}}$$

cf. MOMCON article:

To avoid the inconvenience associated with such preprocessing, MOMCON has been written to minimize the following linearized constraint: ?

$$\langle M \rangle_{\mathrm{L}} = \sum_{m=2;n} m^{p+1} \Big[\big(\boxed{R_{mn} - R^0_{mn}}\big)^2 + \big(\boxed{Z_{mn} - Z^0_{mn}}\big)^2 \Big]. \qquad (5)$$

Here, (R^0_{mn}, Z^0_{mn}) are suitable guesses for the Fourier coefficients, which are usually chosen equal to the initial profiles.

--- ongoing work ---

- Cliffhanger -

funct3d: MHD forces forward model (1)

Max-Planck-Institut für Plasmaphysik

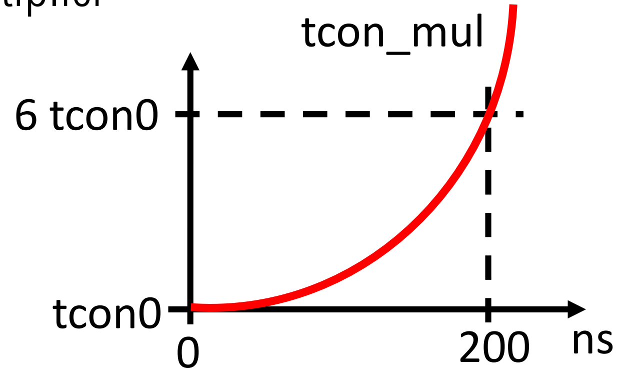
- **compute Jacobian => Z_s, R_s, R, R_θ, Z_θ, τ**
  - make use of properly regularized radial derivatives
  - check if Jacobian changes sign => restart iteration (irst==2)
- **now call bcovar(λ_θ, $-\lambda_\zeta$)**
  - compute metric elements on half-mesh
  - compute plasma volume and dV/ds
  - compute „first half“ of B^θ, B^ζ from $\boldsymbol{\lambda_\theta}$, $\boldsymbol{\lambda_\zeta}$ by adding even, odd contributions and interp to half-mesh
  - add magn. fluxes to B^θ, B^ζ to make them whole (also adjust iota profile for constrained-current)
  - use metric elements to obtain $B_\theta = \mathbf{g_{\theta\theta}} B^\theta + \mathbf{g_{\theta\zeta}} B^\zeta$, $B_\zeta = \mathbf{g_{\theta\zeta}} B^\theta + \mathbf{g_{\zeta\zeta}} B^\zeta$ (covariant B components)
  - now can compute $|B|^2/2 = (B^\theta B_\theta + B^\zeta B_\zeta)/2$ (→ vol.-int. to get W_B) and thermal/“kinetic“ $W_\mathrm{P} = \int_0^1 p(s) \frac{\partial V}{\partial s} \mathrm{d}s$
  - put B_θ, B_ζ onto full-mesh
    → these are the **λ**-forces, and since **λ** is on the full-mesh, need forces there as well
  - calc. avg. force balance → calc_fbal() → yields grad(p) - <**j** x **B**>

funct3d: MHD forces forward model (2)

- **bcovar() cont‘d**
  - 1d-preconditioner, computation of tcon profile: constraint-force multiplier
  - tcon0 given by user (have you ever wondered about the name…?)

$$\mathtt{tcon\_mul} = \frac{1}{16}\mathtt{tcon0}\left(1 + \frac{\mathtt{ns}}{60} + \frac{\mathtt{ns}^2}{200 \cdot 120}\right)$$

parabola in ns:



tcon0 is value of tcon_mul for ns → 0 (hence tcon<u>0</u>)

from preconditioner…

$$t^{con}(s_j) = \min\left(\frac{|a_d^R(s_j,1)|}{a^R_{\mathrm{norm}}}, \frac{|a_d^Z(s_j,1)|}{a^Z_{\mathrm{norm}}}\right) \cdot \mathtt{tcon\_mul} \cdot \left(\frac{32}{\mathtt{ns}-1}\right)^2$$

for j=2, …, ns-1

with

$$\begin{cases} a^R_{\mathrm{norm}} = \int_0^{2\pi}\int_0^{2\pi}\left(\frac{\partial R}{\partial\theta}\right)^2 \mathrm{d}\theta\mathrm{d}\zeta \\ a^Z_{\mathrm{norm}} = \int_0^{2\pi}\int_0^{2\pi}\left(\frac{\partial Z}{\partial\theta}\right)^2 \mathrm{d}\theta\mathrm{d}\zeta \end{cases}$$

t^{con} likely is α(s) from MOMCON paper, Eqn. (4)

funct3d: MHD forces forward model (3)

- **free-boundary contribution from NESTOR**
  - for each call to NESTOR, multiply R^{con}_0, Z^{con}_0 by 0.9 → gradually turn off
  - geometry is assembled from xc*scalxc (temp. re-use of gc array); indep. inv-DFT in NESTOR
  - „vacuum force" is then given by:

$$r_{\mathrm{bsq}} = \left[\left(\frac{|\mathbf{B}_{\mathrm{vac}}|^2}{2} + p_{\mathtt{ns}} \right) - \frac{|\mathbf{B}_{\mathrm{plasma}}|^2}{2} \right] \cdot R \cdot \frac{1}{\mathtt{ns} - 1}$$

?

vacuum pressure (from NESTOR)

pressure at LCFS (if present…)

total plasma pressure at LCFS (from VMEC)

funct3d: MHD forces forward model (4)

- **compute constraint force**

$$
\begin{aligned}
\texttt{extra1} \leftarrow & \left(R^{\mathrm{con},(i)} - R_0^{\mathrm{con}}\right) \frac{\partial R}{\partial \theta} + \left(Z^{\mathrm{con},(i)} - Z_0^{\mathrm{con}}\right) \frac{\partial Z}{\partial \theta} \\
= & \left(\sum_{m,n} m(m-1) \left[R_{mn}^{\cos,(i)} - R_{mn}^{\cos,(0)} s_j \right] \cos(m\theta - n\zeta) \right) \frac{\partial R}{\partial \theta} \\
& + \left(\sum_{m,n} m(m-1) \left[Z_{mn}^{\sin,(i)} - Z_{mn}^{\sin,(0)} s_j \right] \sin(m\theta - n\zeta) \right) \frac{\partial Z}{\partial \theta}
\end{aligned}
$$

funct3d: MHD forces forward model (5)

- **de-alias by Fourier bandpass: only retain m=1, …, M-1 (alias() routine)**
  - first step: DFT of constraint force

$$g_{mn}^{cs}(s_j) = t^{\mathrm{con}}(s_j) \int_0^{2\pi}\int_0^{2\pi} \mathtt{extra1} \cdot \frac{\cos(m\theta_i)}{n_\theta n_\zeta} \cdot \sin(n\zeta_k)\,\mathrm{d}\theta_i \mathrm{d}\zeta_k$$

$$g_{mn}^{sc}(s_j) = t^{\mathrm{con}}(s_j) \int_0^{2\pi}\int_0^{2\pi} \mathtt{extra1} \cdot \frac{\sin(m\theta_i)}{n_\theta n_\zeta} \cdot \cos(n\zeta_k)\,\mathrm{d}\theta_i \mathrm{d}\zeta_k$$

  - second step: inv-DFT of constraint force; only retain m=1, …, M-1

factor for constraint

$$g^{\mathrm{con}}(\theta_i, \zeta_k, s_j) = \sum_{m=1,n=0}^{M-1,N} \left[g_{mn}^{cs}(s_j)\cos(m\theta_i)\sin(n\zeta_k) + g_{mn}^{sc}(s_j)\sin(m\theta_i)\cos(n\zeta_k)\right] \cdot \mathtt{faccon}(m)$$

where $\mathtt{faccon}(m) = \begin{cases} 0 & m = 0 \text{ or } m_{\mathrm{pol}} - 1 \\ \frac{1}{4}\frac{1}{(m+1)^2 m^2} & \text{else} \end{cases}$ (not used anyhow… ?)

funct3d: MHD forces forward model (6)

- **assembly of forces: subroutine forces()**

$$\boxed{\begin{aligned} F^R_{mn} &= A^R_{mn} - \frac{\partial B^R_{mn}}{\partial \theta} + \frac{\partial C^R_{mn}}{\partial \zeta} \\ F^Z_{mn} &= A^Z_{mn} - \frac{\partial B^Z_{mn}}{\partial \theta} + \frac{\partial C^Z_{mn}}{\partial \zeta} \\ F^\lambda_{mn} &= - \frac{\partial B^\lambda_{mn}}{\partial \theta} + \frac{\partial C^\lambda_{mn}}{\partial \zeta} \end{aligned}}$$

three types of force contributions:

A → constant

B → poloidal derivative of something

C → toroidal derivative of something

Fourier-transform of this ends up in gc vector

[based on comment in tomnsp]

funct3d: MHD forces forward model (7)

- **constraint force: contributions to F^R, F^Z in real-space**

$$A^R_{mn} \leftarrow A^{R,\mathrm{MHD}}_{mn} + m(m-1) \cdot g^{\mathrm{con}} \cdot \frac{\partial R}{\partial \theta}$$

$$A^Z_{mn} \leftarrow A^{Z,\mathrm{MHD}}_{mn} + m(m-1) \cdot g^{\mathrm{con}} \cdot \frac{\partial Z}{\partial \theta}$$

$$B^R_{mn} \leftarrow B^{R,\mathrm{MHD}}_{mn} + \left(R^{\mathrm{con},(i)} - R^{\mathrm{con}}_0\right) \cdot g^{\mathrm{con}}$$

$$B^Z_{mn} \leftarrow B^{Z,\mathrm{MHD}}_{mn} + \left(Z^{\mathrm{con},(i)} - Z^{\mathrm{con}}_0\right) \cdot g^{\mathrm{con}}$$

funct3d: MHD forces forward model (8)

- **now scale gc ← gc * scalxc**
- **Fourier-transform forces: F^R, F^Z, F^λ → [gcr, gcz, gcl] == gc vector**
- **call residue()**
  - use the opportunity to scale (m=1)-components of forces (constrain_m1() routine)

$$\mathrm{gc}^R_{1n} \leftarrow \frac{1}{\sqrt{2}}\left(\mathrm{gc}^R_{1n} + \mathrm{gc}^Z_{1n}\right)$$

$$\mathrm{gc}^Z_{1n} \leftarrow \begin{cases} 0 & \texttt{fsqz} < 10^{-6} \text{ or } \texttt{iter2} < 2 \\ \frac{1}{\sqrt{2}}\left(\mathrm{gc}^R_{1n} - \mathrm{gc}^Z_{1n}\right) & \text{else} \end{cases}$$

- for lthreed, apply to gc^{Rss}, gc^{Zcs}
- for lasym, apply to gc^{Rsc}, gc^{Zcc}

  - call getfsq() to compute force residuals: summed over whole volume, all Fourier coefficients

$$g^R_\mathrm{norm} = g_\mathrm{norm} \sum_{j,m,n} \left(\mathrm{gc}^R_{mn}(s_j)\right)^2$$

$$g^Z_\mathrm{norm} = g_\mathrm{norm} \sum_{j,m,n} \left(\mathrm{gc}^Z_{mn}(s_j)\right)^2$$

with $g_\mathrm{norm} = \frac{1}{4}\texttt{fnorm}$

see preconditioner part in bcovar() …

funct3d: MHD forces forward model (9)

- **in residue()**
  - call scale_m1 to apply 1d-preconditioner (?)

$$\mathrm{gc}^R_{1n} \leftarrow \mathrm{gc}^R_{1n} \cdot \frac{a^R_d + b^R_d}{a^R_d + b^R_d + a^Z_d + b^Z_d}$$

$$\mathrm{gc}^Z_{1n} \leftarrow \mathrm{gc}^Z_{1n} \cdot \frac{a^Z_d + b^Z_d}{a^R_d + b^R_d + a^Z_d + b^Z_d}$$

output from precondn() …

- for lthreed, apply to gc^{Rss}, gc^{Zcs}
- for lasym, apply to gc^{Rsc}, gc^{Zcc}

  - scalfor() gets called; apply 1d-preconditioner, solve tri-diagonal system
  - getfsq() again; this time call for preconditioned forces → preconditioned force residuals fsq*1

… funct3d() done here …