Equilibrium Reconstruction in Stellarators: V3FIT

Stellarator Theory Teleconference
26 May 2005

James D. Hanson\(^1\), John Shields\(^1\), Steven P. Hirshman\(^2\), Stephen F. Knowlton\(^1\), Lang Lao\(^3\) and Edward A. Lazarus\(^2\)

\(^1\) Physics Department, Auburn University, Auburn, AL 36849.
\(^2\) Oak Ridge National Laboratory, Oak Ridge, TN 37831.
\(^3\) General Atomics, San Diego, CA 92186.
Equilibrium Reconstruction

• Use measured diagnostic signals to determine current and pressure profiles, and hence the MHD equilibrium configuration.

• Diagnostics
  – Magnetic Diagnostics: magnetic probes, flux loops, saddle coils, Rogowski coils etc.
  – Microwave Interferometry and Polarimetry
  – Thomson Scattering
  – Motional Stark Effect
Equilibrium Reconstruction

• A classic Inverse Problem
  – Forward problem, determine signals, given parameters.
    Known Function $S(p)$
  – We know (observe) the signals. What are the parameters?
    Determine Inverse Function $p(S)$
  – Use Maximum Likelihood / Least Squares.

• Axisymmetric: EFIT Code.
  – EFIT is \textit{tightly coupled} - the least squares iterations proceed \textit{at the same time} as the equilibrium iterations.

• Non-axisymmetric: V3FIT Code.
V3FIT Code Design Goals

- V3FIT will execute fast enough so that reconstructions can be done between shots.
  - Implies tightly coupled equilibrium and reconstruction iterations.
- V3FIT will be flexible, easy to understand, maintain, and modify.
  - V3FIT will be written in Fortran 95.
  - V3FIT must be written in a modular fashion, with clear and consistent data flow.
- The initial working version of V3FIT will use VMEC as the equilibrium solver, and it will have magnetic diagnostics as the primary signal.
- V3FIT will be designed with future enhancements in mind. Future enhancements include:
  - Adding new diagnostics and signals
  - Changing the equilibrium solver
V3FIT: Variables

- **X**: Equilibrium state. (Quantities that change with each equilibration step.)
  - In VMEC, $X = (R_{mn}, Z_{mn}, \lambda_{mn})$.
  - *No* implication that $X$ corresponds to a converged equilibrium.
- **f**: MHD forces. Very small in equilibrium.
- **p**: parameters. Input for Equilibrium Solver. $n_p$ of them.
  - *The parameters are what the Equilibrium Reconstruction will determine.*
  - Pressure profile (VMEC: parameterized as power series in s.)
  - Either iota or current profile
    - Iota profile (VMEC: Uses iota profile when $n_{cur} = 0$.)
    - Current profile (VMEC: Uses current profile when $n_{cur} = 1$.)
  - External currents (free-boundary)
  - Outer surface shape (fixed-boundary)
V3FIT: Variables (cont.)

- **D**: Diagnostics, data that comes from an experiment.
  - Magnetic diagnostics
  - Microwave Interferometry - Polarimetry
  - Motional Stark Effect, Thompson Scattering...

- **S**: Signals, either computed from a *model* \([S_M(p,X)]\) or *observed* (computed from diagnostics) \([S_O(p,D)]\). \(n_S\) of them.
  - To reconstruct an equilibrium we minimize the mismatch between observed and equilibrium signals.
  - Computing \(S_M(p,X)\) is the *forward* problem.
  - Often, the observed signal will be the diagnostic itself, \(S_O(p,D) = D\). This is the case for magnetic diagnostics.
V3FIT: The Inverse Problem

• **ASSUMPTION** - Gaussian Errors
  – The probability of observations $S_0$ is proportional to:

  $$f = \exp \left( -\frac{1}{2} (S_M(p,X) - S_O) \cdot C^{-1} \cdot (S_M(p,X) - S_O) \right)$$

  – For actual observations $S_O(p,D)$, we reinterpret the above probability distribution to be the probability of the parameters $p$.

• **Maximum likelihood, $\rightarrow$ least squares.**
  – The *most likely* value of the parameter vector $p$ will be where the Gaussian distribution function is a *maximum*.
  – Therefore, to find the most likely value of $p$, we minimize chi-squared:

  $$\chi^2 = (S_M(p,X) - S_O) \cdot C^{-1} \cdot (S_M(p,X) - S_O)$$
V3FIT: Two Level Optimizer

• Upper Level: Reconstruction - change $p$
  – Minimize chi-squared by varying $p$
  – $p$: estimate < 100 parameters
  – $S$: estimate about 200 signals
  – Computing the Jacobian is feasible
  – Try standard methods (Levenberg-Marquardt) for optimization.
  – Hope that upper and lower level proceed to convergence at the same time.

• Lower Level: Equilibration - change $X$
  – Equilibrium Solver (VMEC, or another solver)
  – Minimize $f$ by varying $X$ (for slowly varying $p$)
  – $X$: (100 modes)(100 Radial)(R,Z,λ) = 30,000
  – $f$: ~ same number as for $X$, = 30,000
  – Hessian $df/dX$ very large (~$10^9$ elements), time consuming to compute.
    => have to be clever (hence VMEC).
• Lots of possibilities for the "Compute Delta-p" step.
  – Jacobian estimation in here
  – Algorithm for Delta-P

• Will need to experiment with how much to iterate equilibrium solver in steps 2 and 5.

• Steps 4 and 5 may be combined: Add Delta-p slowly, over the course of many equilibrium iterations.
**V3FIT - VMEC Interface Schematic**

```
Rest of V3FIT

Module eq_interface

Subroutine runvmec

Rest of VMEC
```

- Interaction will occur ONLY through the `eq_interface` module and the `runvmec` subroutine.
- **V3FIT** needs to have access to the object files of the equilibrium solver.
Modules for Derived Types

- Derived types are implemented in modules (generally with names ending in T)
- The derived types reflect the structure of the problem.
- All have subroutines to construct, destroy, and assign derived types.
- Module bsc - Biot-Savart Coil
  - Derived type bsc_coil - a magnetic field coil
  - Derived type bsc_coilcoll - a collection of bsc_coil
  - Subroutines to compute $A$, $B$, gradients of $B$, magnetic flux through a coil, rotate and shift a coil
- Module diagnostic_T
  - Derived type diagnostic_desc - describe a diagnostic
  - Derived type diagnostic_data - carry the diagnostic values
- Module signal_T
  - Derived type signal_desc - describe a signal
  - Derived type signal_data - carry the signal values
  - Derived type signal_mrf - Magnetic Response Function
Modules for Derived Types

• Module eq - Equilibrium
  – Derived type eq_param_fix - Fixed parameters of the equilibrium solver
    • Logical switches
    • Array dimensions
    • Numbers of modes
    • Convergence, iteration parameters
  – Derived type eq_param_var - Variable parameters of the equilibrium solver
  – Derived type eq_state - Equilibrium state, the X variables
  – Derived type eq_aux_n - Variables computable from the state

• Module model - The model
  – Right now, just contains a pointer to an eq_state
  – Later may need to add on electron density, information about vacuum vessel (for eddy currents), etc.
Other Modules

- **Module eq_interface** - Interface with the equilibrium code (VMEC)
  - Subroutine `eq_init_file`
    - Read a namelist input file, and initialize the equilibrium code.
    - Construct an `eq_param_fix`, `eq_param_var`, and `eq_state`.
  - Subroutine `eq_init_structure`
    - Input an `eq_param_fix`, `eq_param_var`, and `eq_state`
    - Initialize the equilibrium code
    - (Not yet implemented)
  - Subroutine `eq_step`
    - Iterate an `eq_state` for some number of equilibrium iterations

- **Module signal_model** - Computations that need both a signal and a model
  - Subroutine `signal_model_compute`
    - Computes the solution to the forward problem, $S_M(p, X)$
Forward Problem - Magnetic Diagnostics

• Compute the magnetic flux through a diagnostic coil
  – Have to integrate over all the source currents.
  – Currents in field generating coils - effectively need mutual inductances
  – Currents in plasma - have to integrate over plasma volume
• Implemented in earlier codes V3RFUN and V3POST
• V3RFUN and V3POST being used to design magnetic diagnostic for NCSX
• Sections of code from V3POST reused in V3FIT
Forward Problem - Microwave Interferometry and Polarimetry

• Signal for Interferometry is phase change along path of microwave - proportional to the electron density
  \[ \Delta \varphi \propto \int n_e \, d\ell \]

• Signal for Polarimetry is the change in the angle of polarization of the microwave - proportional to the electron density times the parallel magnetic field.
  \[ \Delta \alpha \propto \int n_e \, \vec{B} \cdot d\vec{\ell} \]

• Coding for solution to forward problem is in progress.
• Will be implemented in existing structure of V3FIT
Status

• V3FIT structural coding is complete
• Modules for Derived Types coded, tested
• Interface with VMEC works.
• Implementation of microwave interferometry-polarimetry in progress
• Next TO DO item - coding of reconstruction algorithm
Extra slides follow
V3FIT: Variables (cont.)

• Why is there a distinction between Diagnostics (D) and observed Signals (S_O)?
  – Because for some diagnostics, we can't compute what we expect the diagnostic to be, using the MHD equilibrium information.
  – Example of when an observed signal is NOT the diagnostic:
    • Consider Te data along a chord through the plasma. The values of Te along the chord are the diagnostic.
    • We can't compute the Te measurements we expect from p and X, so can't consider the Te values as a signal.
    • However, we can use the Te data to constrain the flux surface geometry so that Te is a flux surface quantity.
    • If λ is a parameter along a chord through the plasma, then an appropriate signal would be λ_{outer}(λ_{inner}), suitably discretized.
    • We can calculate both an observed signal S_O(D) from the Te diagnostic, and an equilibrium signal S_E(p,X) from the equilibrium state.
Correlations between Observations

- The covariance operator $C$:
  - Diagonal elements are the variances of the signals: $C_{ii} = \sigma_i^2$
  - Off-diagonal elements contain information about the correlations between the signals. $C_{ij} = r_{ij} \sigma_i \sigma_j \quad i \neq j \quad -1 \leq r_{ij} \leq 1$
  - No correlations between the different signals - set the $r_{ij}$ to 0.
  - Note how the covariance operator takes care of the differing dimensions of the signals.
  - Given a probability density $f(x)$, the mean is: $\langle x \rangle = \int dx \, x \, f(x)$
  - The Covariance is (in component notation):
    $$C_{ij} = \int dx (x_i - \langle x_i \rangle)(x_j - \langle x_j \rangle) f(x)$$
  - The normalized Gaussian probability density:
    $$f(x) = \left( (2\pi)^N \det(C) \right)^{-1/2} \exp\left( -\frac{1}{2} (x - x_0) \cdot C^{-1} \cdot (x - x_0) \right)$$
    has mean $x_0$ and covariance $C$.
  - Even for a linear problem, the correlations can change the most likely parameter value.
Normalized Variables

- Normalize the parameters and the signals:
  \[
  a_k = \frac{p_k}{\tilde{p}_k} \quad y_{M_i}(a_k, X) = S_{M_i}(a_k \tilde{p}_k, X) / \sigma_i \\
  y_{O_i} = S_{O_i} / \sigma_i
  \]

- Error:
  \[
  e_i = y_{O_i} - y_{M_i}(a, X) \quad i = 1, n_s
  \]

- chi-squared:
  \[
  \chi^2 = \sum_{i,j} (y_{M_i}(a, X) - y_{O_i}) w_{ij} (y_{M_j}(a, X) - y_{O_j}) = e \cdot w \cdot e
  \]

- weights
  \[
  w_{ij} = (C^{-1})_{ij} \sigma_i \sigma_j
  \]

- Jacobian:
  \[
  J_{ik} = \frac{\partial y_{M_i}(a, X)}{\partial a_k} \quad i = 1, n_s; k = 1, n_p
  \]

- Gradient:
  \[
  \frac{\partial \chi^2}{\partial a_k} = \sum_{i,j} 2(y_{M_i}(a, X) - y_{O_i}) w_{ij} \left( \frac{\partial y_{M_j}(a, X)}{\partial a_k} \right)
  = -2e \cdot w \cdot J = -2J^T \cdot w \cdot e
  \]
Minimization

- Following *Numerical Recipes*, 2nd edition, section 15.5.
  \[ f(a + \delta a) \approx f(a) - \frac{1}{2} \delta f \cdot \delta a + \delta a \cdot \alpha \cdot \delta a \]
  \[ \beta = -\frac{1}{2} \nabla f = e \cdot w \cdot J = J^T \cdot w \cdot e \]
  \[ \beta_k = -\frac{1}{2} \frac{\partial f}{\partial a_k} = \sum_{i,j} \left( y_{Oi} - y_{Mi}(a, X) \right) w_{ij} \left( \frac{\partial y_{Mj}(a, X)}{\partial a_k} \right) \]
  \[ \alpha = +\frac{1}{2} \nabla \nabla f = J^T \cdot w \cdot J - e \cdot w \cdot \nabla J \]
  \[ a_{kl} = +\frac{1}{2} \frac{\partial^2 f}{\partial a_k \partial a_l} = \sum_{i,j} \left( \frac{\partial y_{Mi}(a, X)}{\partial a_k} \right) w_{ij} \left( \frac{\partial y_{Mj}(a, X)}{\partial a_k} \right) - \left( y_{Oi} - y_{Mi}(a, X) \right) w_{ij} \left( \frac{\partial^2 y_{Mj}(a, X)}{\partial a_k \partial a_l} \right) \]

- Following NR and everybody else, drop the second derivative term in alpha.
  \[ \alpha \approx J^T \cdot w \cdot J \]
Minimization (continued)

\[ f(a + \delta a) \approx f(a) - 2\beta \cdot \delta a + \delta a \cdot \alpha \cdot \delta a \]

- **Two Algorithms**
  1) Steepest Descent - go downhill fast. Implies
     \[ \delta a = \mu \beta \quad \mu > 0 \]
  2) Hit minimum when gradient is zero,
     \[ \nabla f(a + \delta a) = -2\beta + 2\alpha \cdot \delta a \]

     Set gradient to zero, solve:
     \[ \alpha \cdot \delta a = \beta \]

- **Levenberg-Marquardt**: Clever way to interpolate between these two algorithms.
  \[ \alpha'_{kl} = \alpha_{kl} (1 + \lambda \delta_{kl}) \quad \delta a = \alpha'^{-1} \beta \]
  
  - \( \lambda \) small, solving for gradient = 0.
  - \( \lambda \) large, matrix diagonally dominant, steepest descent-like step.
  - More signals than parameters, use SVD to get approximate inverse.

- **Levenberg-Marquardt step (in \( \lambda \rightarrow 0 \) limit)** with SVD minimizes
  \[ |\alpha \cdot \delta a - \beta|^2 = \left| (J^T \cdot w) \cdot J \cdot \delta a - (J^T \cdot w) \cdot e \right|^2 \]

  - \( n_p \) equations, \( n_p \) deviations \( \delta a \). Expect zero minimum.
Jacobian Calculation and Update

• Consider three types of Jacobian calculations.
  – Separately computable (I don't like this phrase. Got a better one?)
  – Finite Difference
  – Broyden Update

• Illustrative Example. Split the parameter vector into pieces.
  \[ \mathbf{a} = (\mathbf{a}_{ec}, \mathbf{a}_{cp}, \mathbf{a}_{pp}, \mathbf{a}_{other}) \]
  – \( \mathbf{a}_{ec} \) - all external currents (helical coils, VF coils, correction coils, etc.)
  – \( \mathbf{a}_{cp} \) - current profile parameters
  – \( \mathbf{a}_{pp} \) - pressure profile parameters
  – \( \mathbf{a}_{other} \) - other parameters
  – \( \hat{\mathbf{a}}_{ec} \) - unit vector in kth slot of external current portion of the \( \mathbf{a} \) vector.
  – Current, pressure profiles are function of the VMEC radial variable \( s \).
    \[ I(s) = \sum_{k=1}^{l(k)} p_l g_k(s) \quad P(s) = \sum_{k=1}^{m(k)} p_m h_k(s) \]
  – \( g, h \) are basis functions for profiles. \( l(k) \) and \( m(k) \) convert index values.
  – Assume Signals are magnetic diagnostics.
Jacobian - continued

• The magnetic diagnostic signals are linear in the currents.
• For external current parameters:

$$\frac{\partial y_{M_i}}{\partial a_{eck}} \approx S_{M_i}((\hat{a}_{eck}, 0, a_{pp}, a_{other}), X)/\sigma_i$$

– Doesn't depend on approach to equilibrium. Compute once, and don't update.
• For the current profile parameters:

$$\frac{\partial y_{M_i}}{\partial a_{cpk}} \approx S_{M_i}((0, \hat{a}_{cpk}, a_{pp}, a_{other}), X)/\sigma_i$$

– Depends on flux surface shape. Update as the equilibrium iteration proceeds.

• These two are examples of what I call "Separately Computable" Jacobian elements. We use physical knowledge about how the signals depend on the parameters.
Jacobian - Finite Difference

• Define the iteration operator $T$ for the equilibrium solver:
  – Takes a state $X_0$ and iterates it with the equilibrium solver $m$ times, using the normalized parameters $a$.
  – $T^m(a,X_0) = X_m$

• Jacobian:
  $$J \approx \frac{\partial y_M}{\partial a} = \frac{y_M(a + \delta a, T^m(a + \delta a, X)) - y_M(a, T^m(a, X))}{\delta a}$$
  – $m = 0$: fixed geometry. (Probably won't work for pressure profile parameters.)
  – $m \geq 1$: allows geometry to change.
  – $m \to$ Infinity: Using fully converged equilibria to compute Jacobian. Implemented with STELLOPT.

• How important is accuracy of signal computation?
Jacobian - Broyden Update

- Assume we started at parameter values $a^n$, with computed signals $y_M(a^n, X(a^n))$. A new set of parameters $a^{n+1}$ are computed, and equilibrium $X$'s iterated some more toward convergence, and new signal $y_M(a^{n+1}, X(a^{n+1}))$ computed. We would like to update the Jacobian $J^n$ we computed earlier, so that it will satisfy:

$$J^{n+1} \cdot (a^{n+1} - a^n) = y_M(a^{n+1}, X(a^{n+1})) - y_M(a^n, X(a^n))$$

- The new Jacobian gets the most recent change correct.
- Best way to do this is the Broyden update:

$$J^n_{B} + \frac{(\delta y_M - J^n \cdot \delta a)}{\delta a \cdot \delta a} \otimes \delta a$$

- Root find with Broyden update - similar to secant method in one dimension.
VMEC Flow Diagram

Program VMEC
Main Program

RUNVMEC
controls vmec run

READIN
read in namelist
read mgrid file

EVSOLVE
compute MHD forces
for snapshot R,Z,L values

eqstep control
allocate, initialize arrays

EVOLVE
time step R,Z,L
toward equilibrium

FIXARAY
initialize fixed array

RUNVMEC
controls vmec run

Program VMEC
Main Program