EFIT++ Equilibrium Reconstruction

Lynton Appel§
Michele Romanelli§
David Muir§
John Storrs§
Clive Michael§
Maarten de Bock§
Nicolas Mercadier – Ecole Centrale Paris
Martin David - Ecole Centrale Paris
Frederic Imbeaux – CEA
Gabriele Manduchi – Padua, Italy.
Jakub Urban – IPP Prague, Czech.
Guiseppe Calabro, Edmondo Giovanozzi -Frascati, Italy

§ EURATOM/CCFE Fusion Association, Culham Science Centre, Oxon, UK

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

Background

- **EFIT++** is a rewrite of Lang Lao's original **EFIT**[1] code
  - 1996 Porting of **EFIT** to CCFE from GA.
  - 2005 Machine independent **EFIT**; interface to **ITM**.
  - 2006 New C++ driver.
  - 2007 Development of documentation, test suite.
  - 2008 Development of module to compute Boozer coordinates.
  - 2009 Generalised MSE constraint, netCDF-4 file output

[1]*Lao, L L et al, Nuclear Fusion* ***25*** *(1985) 1611.*

Grad-Shafranov equation

EFIT++ solves the equation of equilibrium force balance in a tokamak in the presence of finite toroidal rotation , *ω*, and zero poloidal rotation:

$$\mathbf{j}\times\mathbf{B} = \nabla p + \rho_m \frac{d\mathbf{v}}{dt}$$

$$\approx \nabla p - \rho_m \omega^2 R\mathbf{e}_R$$

Resulting Grad-Shafranov equation is[1]

[1]*Thyaragaraja, A and McClements, K.G., Phys of Plasmas **13** 026502 (2006)*

$$\Delta^* \psi_p = -\mu_o R J(R,Z)$$

where

$$J_\varphi(R,Z) = \frac{1}{\mu_o R} ff'(\psi_p) + Rp'(\psi_p, R)$$

in which

$$p(\psi_p, R) = p_o(\psi_p)\exp\left[\left(\frac{\omega(\psi_p)}{c_s(\psi_p)}\right)^2 \frac{R^2}{2}\right]$$

$$f(\psi_p) = RB_\varphi$$

and $c_s = \sqrt{T(\psi_p)/m}$ is the ion sound speed.

Grad-Shafranov equation

EFIT++ requires J_φ to be a linear function of the flux functions. This is possible in the limit $\omega R/c_s<1$

$$p \approx p_1(\psi_p) + \left(\frac{R^2 - R_T^2}{R_T^2}\right) p_w(\psi_p)$$

where

$$p_w(\psi_p) = \frac{R_T^2\, p_o(\psi_p)}{2}\left(\frac{\omega(\psi_p)}{c_s(\psi_p)}\right)^2$$

and

$$p_1(\psi_p) = p_o(\psi_p) - p_w(\psi_p)$$

yielding

$$J_\varphi(R,Z) = \frac{1}{\mu_o R}\, ff'(\psi_p) + Rp'(\psi_p) + \left(\frac{R^2 - R_T^2}{R_T^2}\right) Rp_w'(\psi_p)$$

EFIT++ Flux Functions

Users select either polynomial flux-functions

$$g(\tilde{\psi}) = \sum_{i=1}^{i=n} \alpha_i \tilde{\psi}^{i-1} - H\tilde{\psi}^{n} \sum_{i=1}^{i=n} \alpha_i$$

$$\tilde{\psi} = (\psi_p - \psi_p^{axis})/(\psi_p^{LCFS} - \psi_p^{axis})$$

(with H=0 or H=1), or a tension spline[1] representation:

$$g(\tilde{\psi}) = \frac{g_i''}{T^2}\left[\frac{\sinh\left(T\left(\tilde{\psi}_{i+1} - \tilde{\psi}\right)\right)}{\sinh\left(Th_i\right)} - \frac{1}{h_i}\left(\tilde{\psi}_{i+1} - \tilde{\psi}\right)\right] + \frac{g_i}{h_i}\left(\tilde{\psi}_{i+1} - \tilde{\psi}\right) +$$

$$h_i = \tilde{\psi}_{i+1} - \tilde{\psi}_i$$

$$\frac{g_{i+1}''}{T^2}\left[\frac{\sinh\left(T\left(\tilde{\psi} - \tilde{\psi}_i\right)\right)}{\sinh\left(Th_i\right)} - \frac{1}{h_i}\left(\tilde{\psi} - \tilde{\psi}_i\right)\right] + \frac{g_{i+1}}{h_i}\left(\tilde{\psi} - \tilde{\psi}_i\right)$$

- *Continuity in g' enforced at each internal knot.*
- *Users can prescribes g or g" at any knot location.*

*[1]Cline, A,,K, ACM **17** (1974) 218*

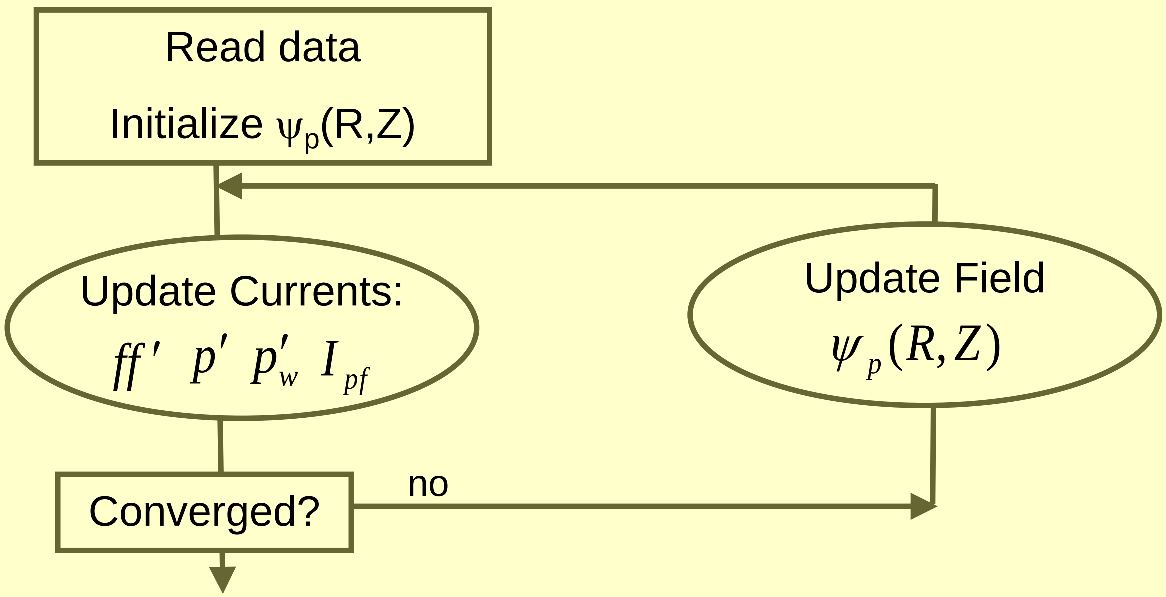
EFIT++ algorithm

EFIT++ needs to solve a 2-D nonlinear elliptic equation of the form

$$\Delta^* \psi_p = F(ff'(\psi_p), p'(\psi_p), p'_w(\psi_p), R, \{I_{pf}\})$$

Algorithm uses a Picard's iteration scheme. Each iteration solves two separate problems:

- Given $\psi_p(R,Z)$ obtain $ff', p', p'_w, R, \{I_{pf}\}$.
- Given $ff', p', p'_w, R, \{I_{pf}\})$ obtain $\psi_p(R,Z)$



EFIT++ Picard Iteration

1) Given $\psi_p(R,Z)$ obtain $ff', p', p'_w, R, \{I_{pf}\})$.

➢ *Obtain chi-squared fit to spline coefficients and poloidal currents*

$$\chi^2 = \left|\sum_{i=1}^{N} \chi_i^2\right|_{Min}$$

$$\chi_i^2 = \left(\frac{M_i - P_i}{\sigma_i}\right)^2$$

M_i is measured value of constraint *i input by user*

ε_{abs} *(absolute error)* and ε_{rel} *(relative error) input by user*

W_i *(weight) input by user*

Form linearised constraint equations, eg. for magnetic detectors:

$$\sigma_i = \frac{\left|\varepsilon_{abs}, \varepsilon_{rel} M_i\right|_{MAX}}{W_i}$$

Currents and Flux function coefficients

$$P_i(r_i) = \sum_j G(r_i; r_{pf(j)}) I_{pf(j)}^{(m+1)} + \int_{plasma} G(r_i; R, Z) J_\phi(R, \psi_p^{(m)}, \alpha_{1\ldots p}^{(m+1)}, \beta_{1\ldots q}^{(m+1)}, \gamma_{1\ldots r}^{(m+1)}) dRdZ$$

EFIT++ Picard Iteration

2) Given $ff', p', p'_w, R, \{I_{pf}\})$ obtain $\psi_p(R,Z)$.

- *Field solution by solving linearised Grad-Shafranov equation*

$$\Delta^* \psi_p = -\mu_o R J_\varphi$$

using Finite Difference Grid with boundary-integral to represent external fields.

Configurable constraints

- Flux and saddle loops.
- Magnetic pick-up coils.
- Pf circuits.
- q_0
- Plasma current
- Diamagnetic flux and Bphi.
- Total pressure
- Rotational pressure
- B-poloidal
- Position of Separatrix
- Iso-flux surfaces
- Faraday Rotation
- MSE
- p' proportional to ff'
- Relational constraints: p', ff', p_{rot}, fcoils.

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

CCFE CULHAM CENTRE FOR FUSION ENERGY

MSE

MSE principle:

Measure polarisation angle Stark lines parallel (p) or perpendicular (s) to (local) electric field

- $E = E_s + v_{beam} \times B$

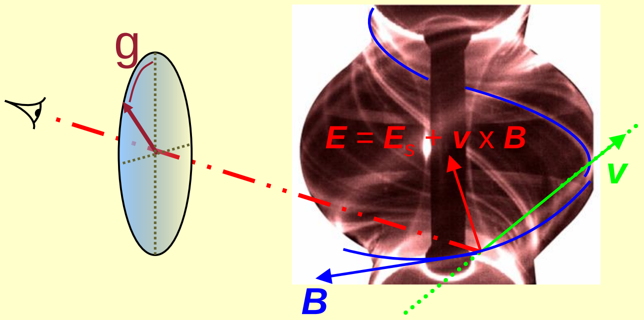
where

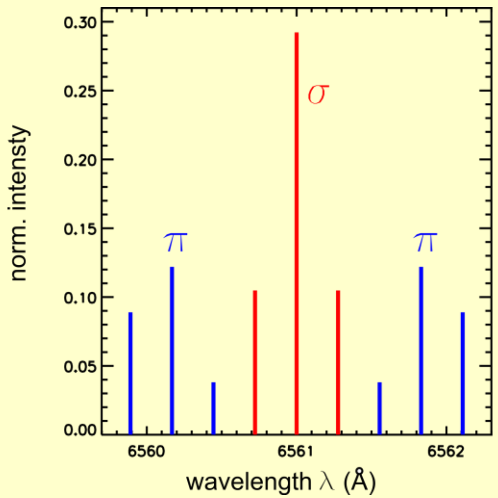
$$E_s = -v_{plasma} \times B + \frac{1}{qn} \nabla p + \frac{m}{q} v_{plasma} \cdot \nabla v_{plasma}$$

- if v_{beam} and E_s are known, B can be determined:

eg for an inifinitesimal collection volume,

$$\tan \gamma = \frac{A_o B_z + A_1 B_R + A_2 B_\phi}{A_3 B_z + A_4 B_R + A_5 B_\phi}$$





- MSE on MAST operates routinely intershot with
  - 35 channels close to mid-plane.
  - Spatial resolution: 2.5cm
  - Time-resolution 0.5ms.
  - RMS noise $\Delta\gamma \leq 0.5^o$.
- Reliable MSE-controlled EFIT++ reconstructions.

MSE constraint

- Generalised MSE constraint requires full description of Neutral beam and MSE optics. Entire system is implemented with 21 object classes.

MSE hardware

- MseChannels
- MseChannel
- OpticalFibreBundle
- OpticalFibreBundles
- OpticalLens
- OpticalLenses
- OpticalFibre
- OpticalFilter
- OpticalFilters

Neutral Beam systems

- NeutralBeam
- NeutralBeamlet
- NeutralInjectorBox
- NeutralInjectorBoxes

Time-slice structures

- TimeSliceMseChannel
- TimeSliceMseChannels
- TimeSliceOpticalFibreBundle

Code-specific data

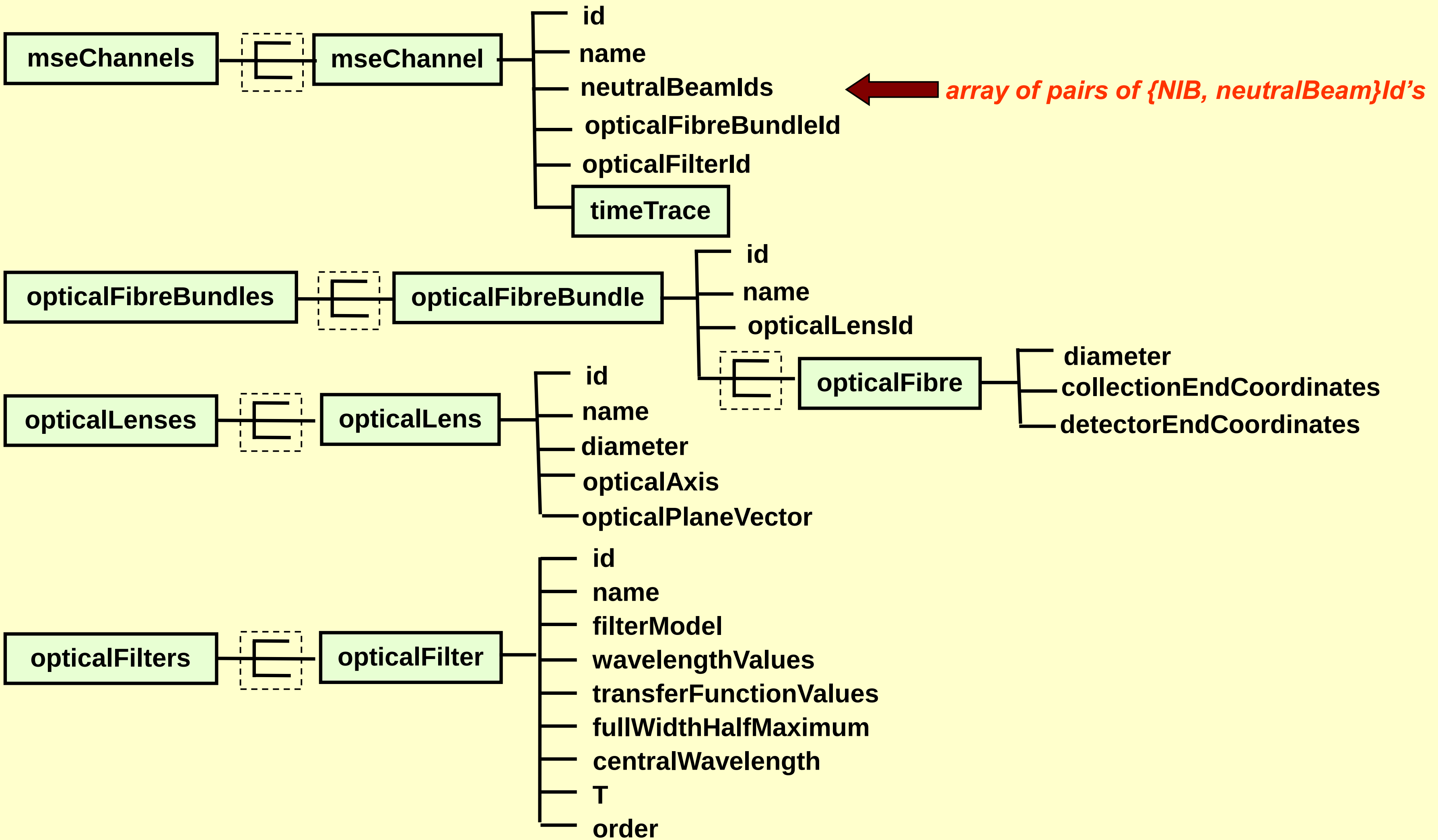
- DataSelection
- EfitMseChannel
- EfitMseChannels
- EfitNeutralInjectorBoxes

Time-dependent signal:f(t)

- TimeTrace

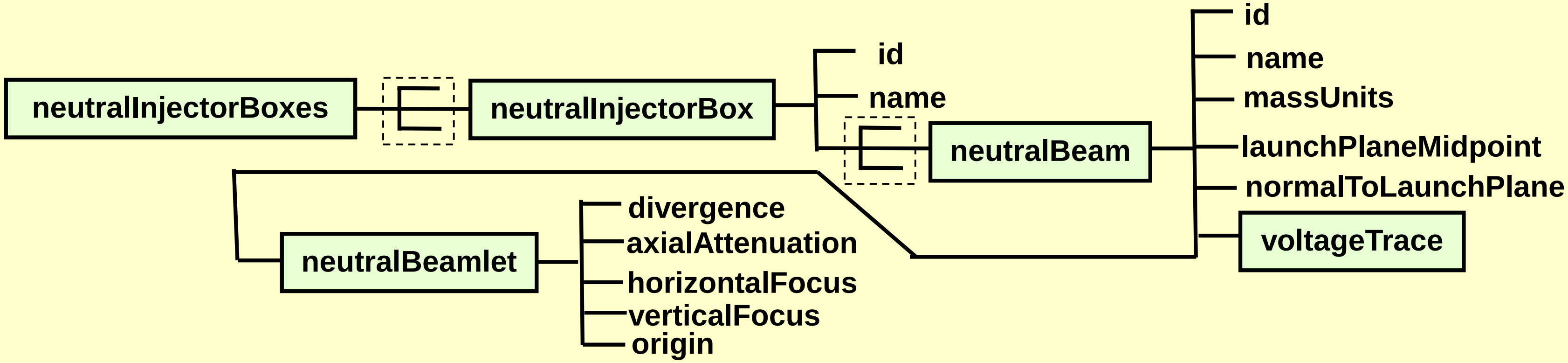
MSE constraint

- Tokamak data structures MSE-system

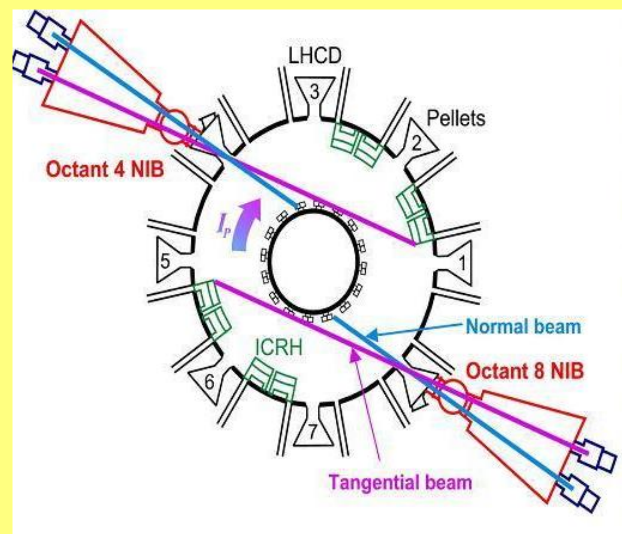


MSE constraint

- Tokamak data structures representing neutral beams



Example system, JET with 2 Neutral Injector Boxes (NIBS), each with 8 Positive Ion Neutral Injectors (PINIS), each PINI being composed of many beamlets.

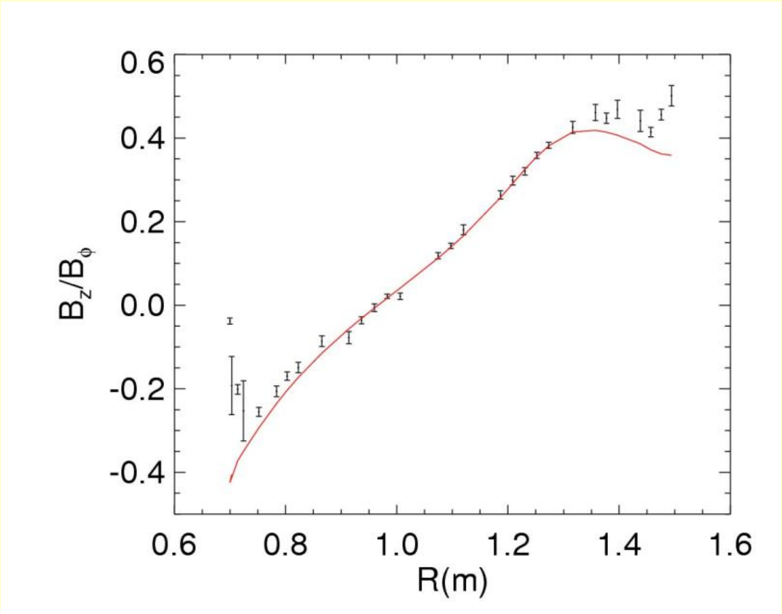


Inconsistencies in MSE data

•Fit to MSE data typically best near magnetic axis on outboard size

•Poorest fits for MSE channels

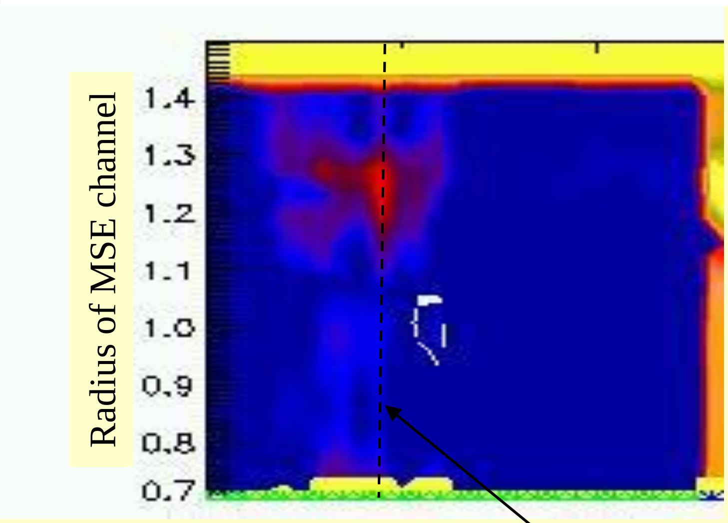
- ➢inboard of magnetic axis.
- ➢near outer edge of plasma.



•Analysis of constraint matrix (S) reveals cause of inconsistency close to outer edge.

•Construct SS^{-1} (S^{-1} is the SVD psuedo-inverse)

•Large off-diagonal values of SS^{-1} indicate co-variance between **outer** MSE channels and **inboard** magnetic signals.

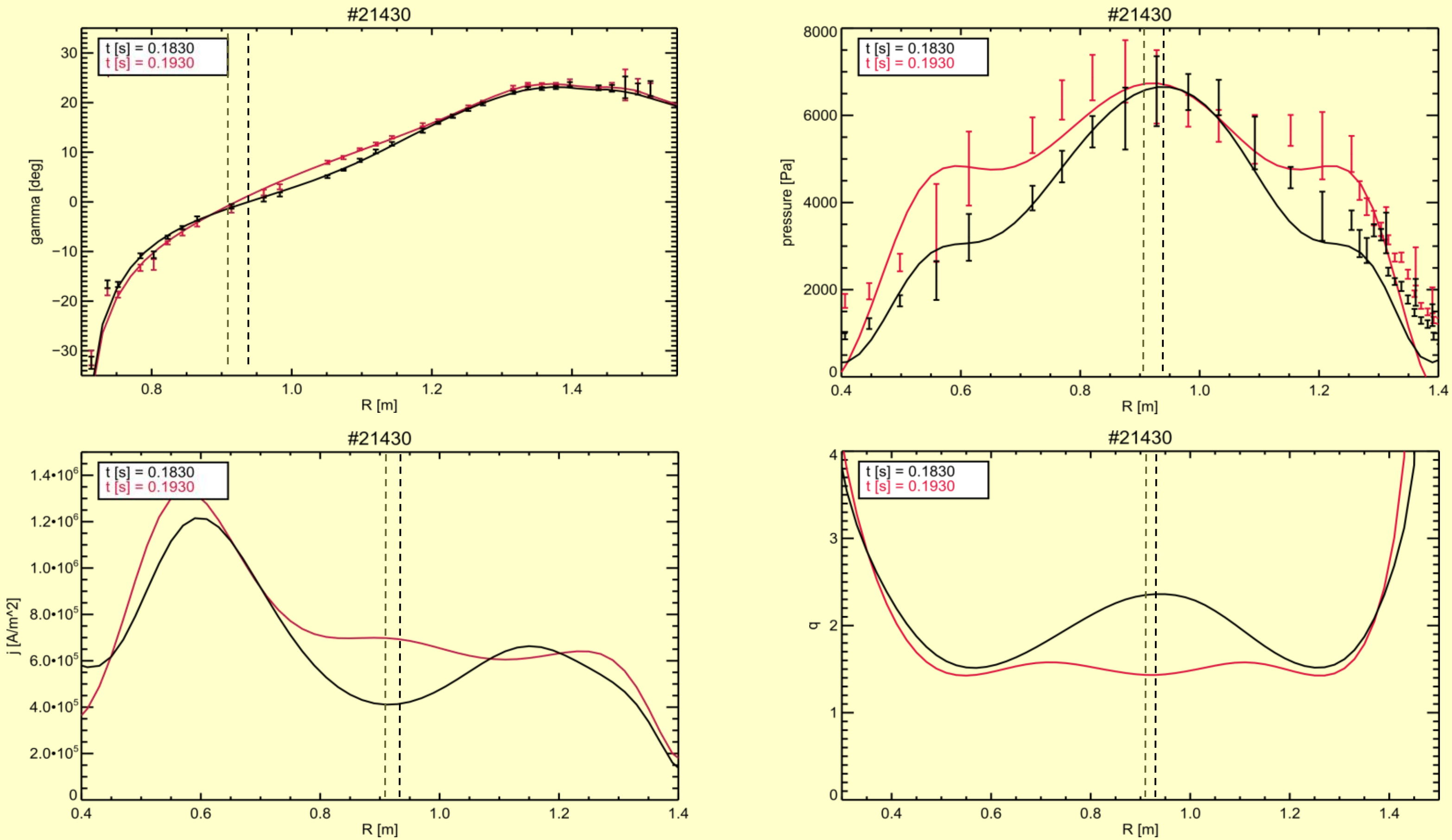


Centre of inboard magnetics

MSE on MAST

Example MAST reconstruction constrained with 35 MSE channels, pressure measurements, and magnetics.

- *Evidence of hollow current and reversed q-profiles.*



dashed lines indicate magnetic axis

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

Flux coordinates

- **EFIT++** solves Grad Shafranov equation in cylindrical (R,φ,Z) coordinates whereas **HAGIS** code uses flux-coordinates.
- Originally, flux-coordinates were computed by HELENA.
  - Particles could not be followed outside the separatrix.
- **EFIT++** now extended:
  - Flux coordinates computed directly from the unbounded EFIT++ equilubrium.
  - Flux coordinates continuous over both interior and exterior domains.

CCFE CULHAM CENTRE FOR FUSION ENERGY

Flux coordinates: implementation

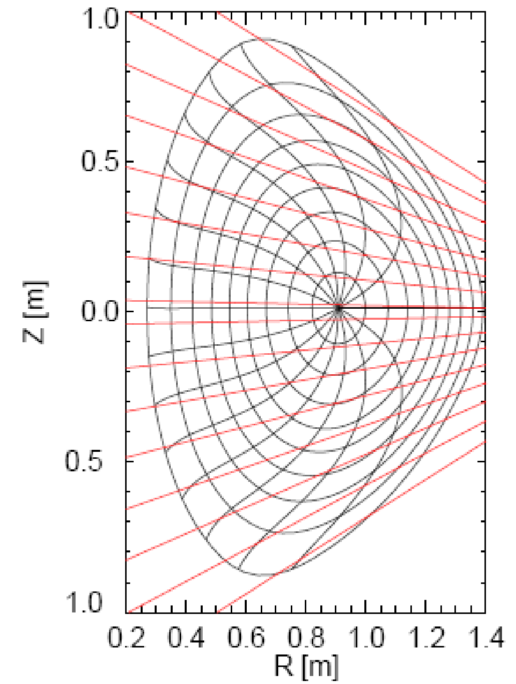
- HAGIS coordinates are

$$B = \delta(\psi_p, \theta)\nabla\psi_p + I(\psi_p)\nabla\theta + g(\psi_p)\nabla\zeta$$

$$B = \nabla\zeta \times \nabla\psi_p + q\nabla\psi_p \times \nabla\theta$$

$$J = \frac{I + gq}{B^2} \qquad q = \frac{d\zeta}{d\theta} = \frac{B.\nabla\zeta}{B.\nabla\theta}$$

- Interior plasma EFIT++ region uses *HAGIS* coordinates.
- But singularity problem at separatrix due to x-point!



Contours of constant ψ and θ of MAST equilibrium (red lines indicate SXR sight lines)

Flux coordinates: implementation

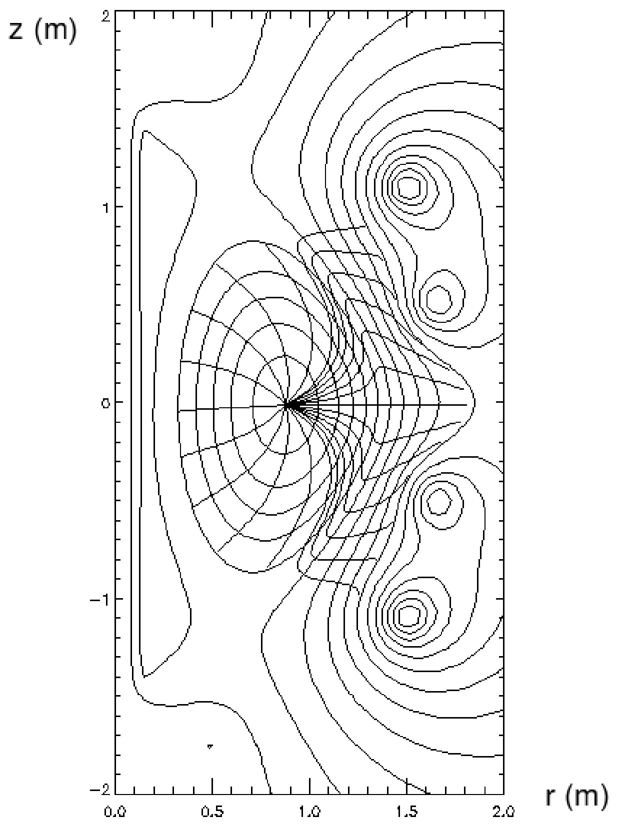
Singularity avoided in **EFIT++**

- For $0 < \overline{\psi}_p < 0.95$ construct straightfield line coordinates.
- For $\overline{\psi}_p > 0.95$ construct Boozer coordinates on open flux surface between $z_{min} < z < z_{max}$
- In exterior region, $I(\psi_p)$ and $q(\psi_p)$ are arbitrary functions.
  - ➢ Ensure that $I(\psi_p)$ and $q(\psi_p)$ match at boundary; also define

$$I = I_{edge} + \left.\frac{\partial I}{\partial \psi_p}\right|_{edge} \psi_N \left(1 - e^{-\frac{|\psi_p - \psi_{edge}|}{\psi_N}}\right)$$

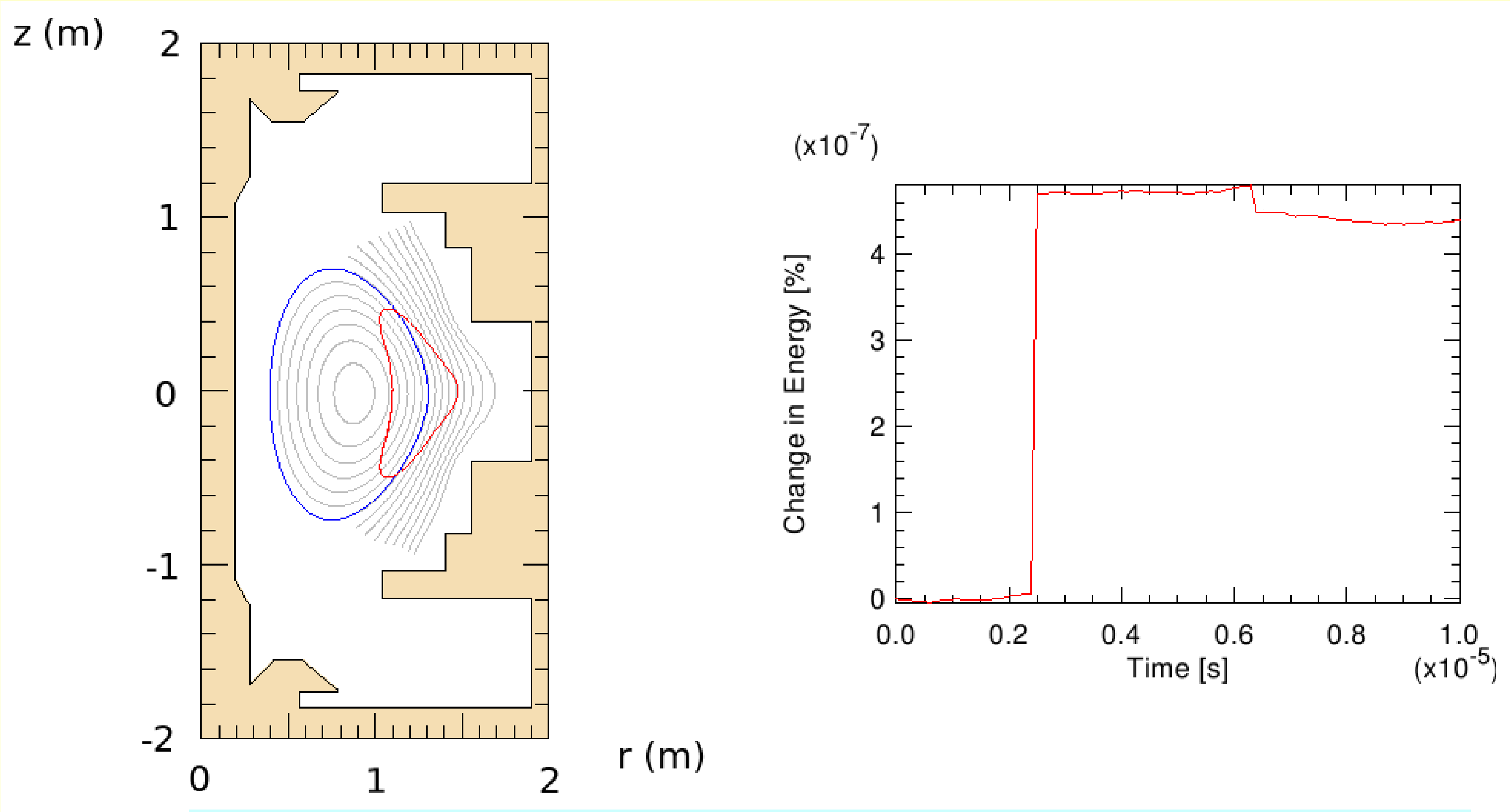
and maximal variation of θ on a surface ψ_p:

$$\Delta\theta = \Delta\theta_{edge} + \left.\frac{\partial \Delta\theta}{\partial \psi_p}\right|_{edge} \psi_N \left(1 - e^{-\frac{|\psi_p - \psi_{edge}|}{\psi_N}}\right)$$



Hagis simulation

- **EFIT++**-generated flux surfaces interfaced to **HAGIS** code.
  - ➢Now possible to tracking energetic particle motion across the separatrix..



Hagis simulation of a 200keV Deuteron travelling across the separatrix

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

Induced Currents

- Induced currents are included with a separate programme

  INDUCTION

  - Solves induction equation on all passive structures.
  - Plasma current contribution modelled as a distributed curent source scaled to the measured plasma current signal.
  - Option exists to run **INDUCTION** iteratively using **EFIT++-** generated plasma current distribution.

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

Treatment of errors

- EFIT++ obtains 'optimum' solution of currents and plasma function coefficients by solving linearised constraint equation

$$\underline{\underline{A}}\,\underline{X} = \underline{B}$$

- In case where all flux functions are polynomials, solution is obtained using SVD of A,

$$\underline{\underline{A}} = \underline{\underline{U}} \begin{vmatrix} w_1 & & \\ & \ddots & \\ & & w_n \end{vmatrix} \underline{\underline{V}}^T$$

  - Least squares solution and covariances are

$$\underline{X} = \underline{\underline{V}} \begin{vmatrix} 1/w_1 & & \\ & \ddots & \\ & & 1/w_n \end{vmatrix} \underline{\underline{U}}^T \quad ; \quad \sigma_{jk} = Cov(X_j, X_k) = \sum_{i=1}^{n} \frac{V_{jk} V_{ki}}{w_i^2}$$

  - Errors of derived parameters, eg $q = f(\underline{X})$ given by

$$\sigma_q^2 = \sum_i \sum_j \frac{\partial f(\underline{X}_o)}{\partial X_i} \frac{\partial f(\underline{X}_o)}{\partial X_j} \sigma_{ij}$$

where $\underline{X}_o$ denotes optimal solution)

I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

Running EFIT++

Four ways to run **EFIT++:**

1. Invoke **EFIT++** executable directly *(for the expert only).*
2. Execute **EFIT++** using efit++ shell command:

   efit++ [-h] –[d] [–e<executable>] [-pN] [-f<hostFile>] [-g] –[i<plasmaCutoffCurrent] [-o<dir>] [shotBegin] [shotEnd]

3. Execute **EFIT++** using IDL controller **efit4idl**
   - A range of pre-configured run configurations.
   - Group-settings of run parameters.
   - Run-time
   - Output plots generated.
4. Execute **EFIT++** using **MC3**.
   - Integrated analysis packagage
   - Sophisticated visualisation and run-time control.

- Parallelisation implemented using MPI

XML input

- User-interface mirrors underlying OO code. Two generic types of data
  - ***Tokamak data***: data specific to tokamak device.
  - ***Code-specific data***: data specific to EFIT++.
- Data input
  - Using IDAM from JET/COMPASS/MAST/FTU data repositories including via MDS+ database.
  - XML data files.
- Input XML can be split into multiple files as desired by user.
- Most code parameters configurable as time-dependent *or* time-independent, eg:

```
<numericalControls
        interpolationMethod=3
        timeMargin=0.2
         times =“0.1 0.2 0.3 0.4” />
         <pp ndeg ="2 3 6 7" edge="1" func="0" />
</numericalControls> </Top>
```

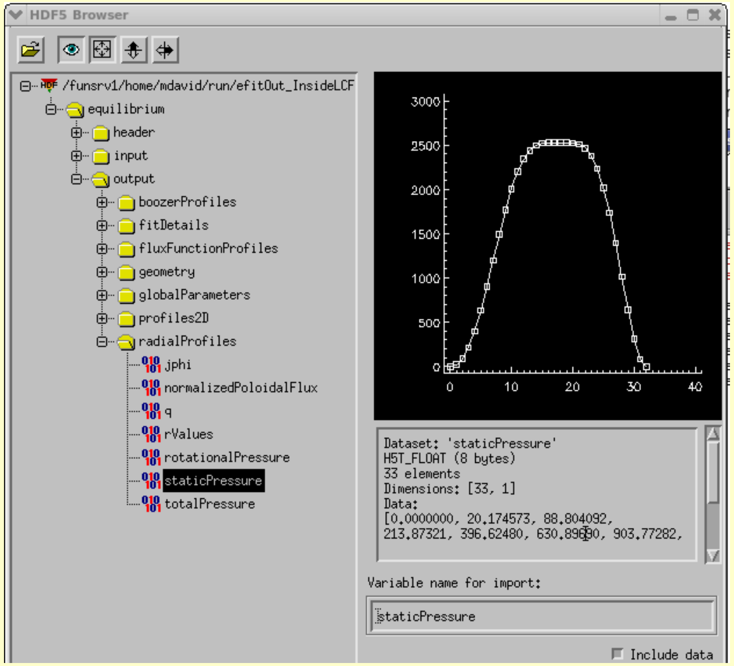
XML input

- Example tokamakData.xml file.

```
<Top>
   <include file="tokamakDataSource.xml" />
   <include file="debug.xml" />
   <include file="outputOptions.xml" />
   <include file="times.xml" />
   <include file="numericalControls.xml" />
   <include file="relationalffprime.xml" />
   <include file="relationalpprime.xml" />
   <include file="relationalrotationalpprime.xml" />
   <include file="grid.xml" />
   <include file="current.xml" />
   <include file="efitOptions_submse.xml" />
   <include file="boundary.xml" />
   <include file="pressure.xml" />
   <include file="cx.xml" />
   <include file="efitOptions_subpf.xml" />
   <include file="efitOptions_submp.xml" />
   <include file="efitOptions_subfl.xml" />
</Top>
```

EFIT++ Output

- Output written to **HDF5** and/or **netCDF4** data files.
  - Interpreters for both formats widely available eg in Mathematica, Matlab, IDL…
  - **EFIT++**-specific data visualizers available within **IDL4EFIT** and **MC3.**



I. EFIT++ code

II. MSE

III. Flux Coordinates

IV. Induced Currents

V. Errors

VI. Running EFIT++

VII. Documentation

EFIT++ documentation

- Extensive *user's* documentation and *programmer's* documentation
  https://mastweb.fusion.org.uk/svndocs/efit++/index.html
- EFIT++ code maintained in 'open-access' SVN repository
- Extensive benchmark tests run prior to each code release.
  - Specific tests for JET,MAST,FTU, COMPASS.
- All previous releases maintained online.
  https://mastweb.fusion.org.uk/svnroot/efit++/development/
- Current information and views in EFIT WIKI
  http://fusweb1/culham.CCFE.org.uk/fusionwiki/index.php/EFIT

……………………access to EFIT++

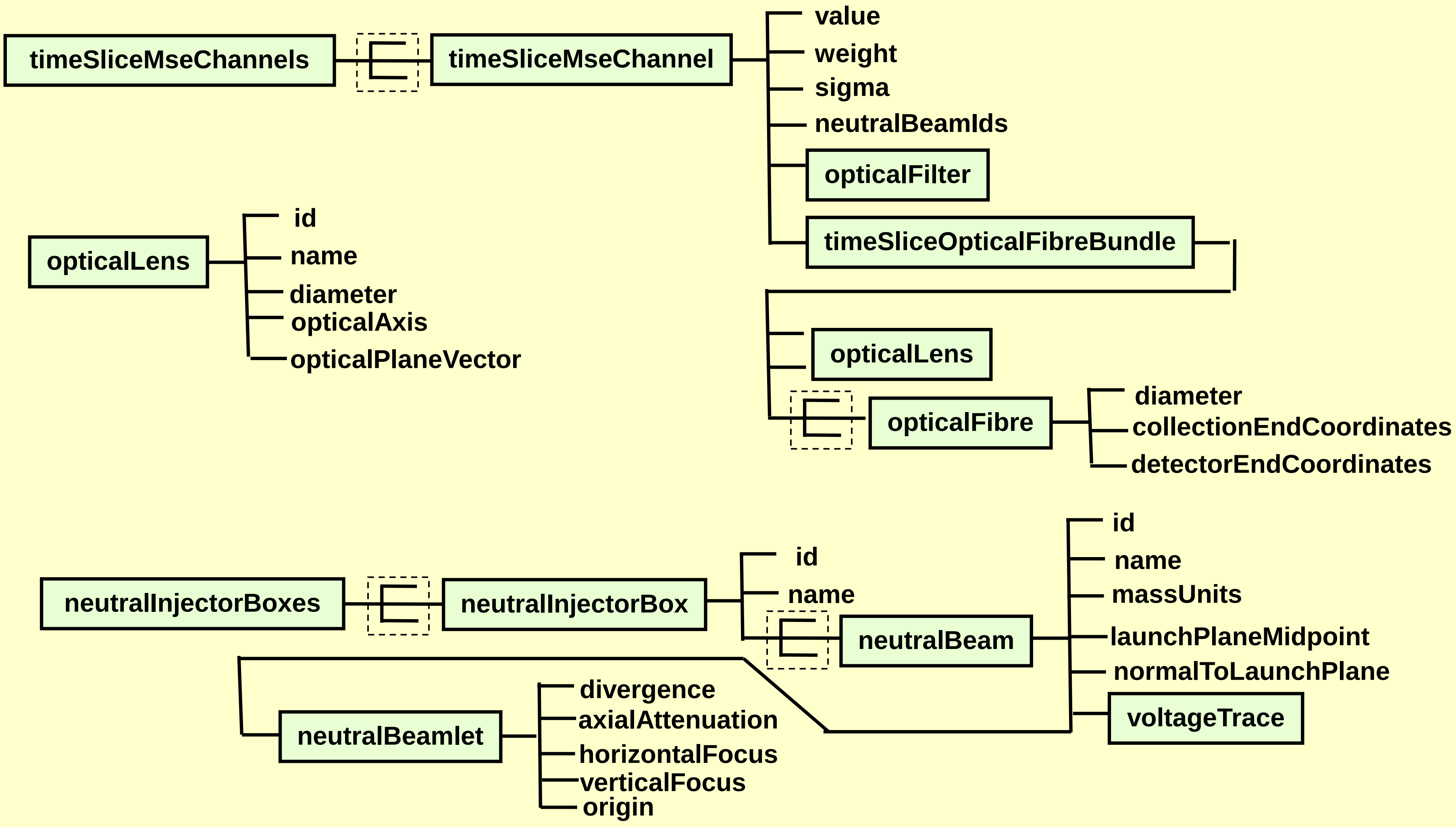
- **EFIT++ svn repository**, and **WEB documentation** located at https://mastweb.fusion.org.uk/svndocs/
- **Read access** available to anyone with a **JETNET** domain or **FUSION** domain account.

Conclusions

- **EFIT++** computes equilibrium reconstruction to provide
  - Routine reconstructions.
  - Tailored reconstructions.
  - Operation as a module of the **ITM**
  - Flexible XML-based input removes 'requirement' to introduce short-term *hacks.*
- Code implementation is machine independent
  - **EFIT++** in use on **MAST, JET, COMPASS and FTU tokamaks.**
- **Integral test-suite** run routinely.
- Comprehensive **web-based documentation**.
- International project team.

MSE constraint

- Tokamak data structure representation of one timeSlice



MSE constraint

- Code-specific data structures