Present Status of M3D

Josh Breslau
and the M3D group
PPPL

FDM3D Workshop
Princeton
March 19, 2007

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

Capsule History of M3D

- Original MH3D (W.P., early 1980s) was a serial Fortran code in a single source file solving resistive MHD using finite differences on a radial mesh with spectral treatment of θ and ϕ.

- Over more than a decade, gradual refinements and enhancements of the physical model (hybrid [W.P.] and two-fluid [L.S.] models) and numerical scheme (finite elements [H.S.]) were accreted onto this program, forming the Multilevel 3D Code (M3D). This was eventually parallelized using OpenMP.

- Around 1999, X.T. set out to create an MPI version of the code. Doing a complete rewrite, he created a C code distributed over many files within two layers of directories, using linear triangular finite elements on a domain decomposed both poloidally and toroidally to solve MHD only, using the PETSc software library to handle communications and linear solves. This was ParM3D.

- In order to retain much of the physics and flexibility of the original version, H.S. undertook to couple the two codes together, using ParM3D for mesh generation, I/O, and linear solvers with the original Fortran “m1.F” as the physics driver. Data would be passed between the C and Fortran parts of the new code using a new set of Fortran and C interface routines. Much of the now-unused part of ParM3D was left in the distribution in vestigial form. This is M3DP (still referred to as M3D).

- A CVS repository for the modern M3D was started in 2001. Changes made since then are archived in /p/m3d/README on the PPPL Unix cluster. Highlights include refinement of the two-fluid options; improvement and parallelization of the hot particle treatment; addition of 2^{nd}- and 3^{rd}-order element options; and addition of vacuum region/resistive wall capability. The current version number is 3.5.12.

Platforms

M3D has been ported to the following computers at NERSC, NCCS, Princeton, and ANL:

	OpenMP	MPI
IBM SP (Seaborg)	Y	Y
Opteron cluster (Jacquard)		Y
IBM Power 5 (Bassi)	?	Y
Cray X1E (Phoenix)		Y*
Cray XT3, XT4 (Jaguar)		Y
SGI Origin 2000 (Hecate)	Y	
SGI Altix (MHD)	Y	Y
BlueGene/L, Argonne		Y*

*****Not used for production runs.

Statistics

- Source code is divided into four directories (m3d, mhd, mesh, utility) with 34 subdirectories.

- There are approximately 264 C source files, 216 C header files, 33 Fortran source files, 16 Fortran header files, and 35 Makefiles.

- There are approximately 52,000 lines of C and 97,000 lines of Fortran source code.

- This includes a lot of code that is no longer executed (or, in many cases, compiled), but excludes standalone post-processing utilities and many trial routines that have not yet been committed to the repository.

- Libraries required include PETSc, parallel HDF5, and sometimes FFTW.

- Three standard input files (plus batch script), others optional; recently consolidated to a single Python script.

- Performance record: 240 Gflops on 10,240 XT3 cores (VN mode) during a 1D weak scaling test.

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

Extended MHD Equations

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}_i) = 0$$

$$\rho \left[\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} + (\mathbf{v}_i^* \cdot \nabla) \mathbf{v}_\perp \right] = -\nabla \mathbf{p} + \mathbf{J} \times \mathbf{B} + \mu \nabla^2 \mathbf{v}$$

$$\mathbf{E} + \mathbf{v} \times \mathbf{B} = \eta \mathbf{J} - \frac{\nabla_\parallel p_e}{ne}$$

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}$$

$$\mathbf{J} = \nabla \times \mathbf{B}$$

$$\frac{\partial p}{\partial t} + \mathbf{v} \cdot \nabla p = -\gamma p \nabla \cdot \mathbf{v} + \nabla \cdot n \chi_\perp \nabla \left(\frac{p}{\rho} \right) - \mathbf{v}_i^* \cdot \nabla p - \gamma p \nabla \cdot \mathbf{v}_i^* + \frac{\mathbf{J} \cdot \nabla p_e}{ne} + \gamma p_e \mathbf{J} \cdot \nabla \left(\frac{1}{ne} \right)$$

$$\frac{\partial p_e}{\partial t} + \mathbf{v} \cdot \nabla p_e = -\gamma p_e \nabla \cdot \mathbf{v} + \nabla \cdot n \chi_{\perp e} \nabla \left(\frac{p_e}{\rho} \right) + \frac{\mathbf{J}_\parallel \cdot \nabla p_e}{ne} - \gamma p_e \nabla \cdot \left(\mathbf{v}_e^* - \frac{\mathbf{J}_\parallel}{ne} \right)$$

where

$$\mathbf{v}_e^* \equiv -\frac{\mathbf{B} \times \nabla p_e}{neB^2}, \quad \mathbf{v}_i^* \equiv \mathbf{v}_e^* + \frac{\mathbf{J}_\perp}{ne},$$

$$\mathbf{v} \equiv \mathbf{v}_i - \mathbf{v}_i^* = \mathbf{v}_e - \mathbf{v}_e^* + \frac{\mathbf{J}_\parallel}{ne}$$

Artificial sound wave model for $\kappa_\parallel$:

$$\frac{\partial T}{\partial t} = s \frac{\mathbf{B} \cdot \nabla u}{\rho}$$

$$\frac{\partial u}{\partial t} = s \mathbf{B} \cdot \nabla T + \nu \nabla^2 u$$

M3D Scalar Variables

Field Variables

Write

$$\vec{B} = \nabla \psi \times \nabla \phi + \frac{1}{R} \nabla_\perp F + \left(R_0 + \tilde{I} \right) \nabla \phi$$

where

$$\nabla_\perp^2 F = -\frac{1}{R} \frac{\partial \tilde{I}}{\partial \phi}$$

so that

$$\vec{J} = \left(\nabla \tilde{I} - \frac{1}{R} \nabla_\perp F' \right) \times \nabla \phi + \frac{1}{R^2} \nabla_\perp \psi' - C \nabla \phi$$

where primes denote derivatives with respect to ϕ and

$$C \equiv -R J_\phi = \Delta^* \psi + \frac{1}{R} \frac{\partial F}{\partial z}$$

Velocity Variables

Write

$$\vec{V} = \frac{R^2}{R_0} \nabla U \times \nabla \phi + \nabla_\perp \chi + V_\phi \hat{\phi}$$

Others

ρ, $p_{(e,i)}$ or $T_{(e,i)}$

Note that

$$\nabla_\perp^2 \psi \equiv \frac{\partial^2 \psi}{\partial R^2} + \frac{\partial^2 \psi}{\partial z^2},$$

$$\Delta^* \psi \equiv \nabla_\perp^2 \psi - \frac{1}{R} \frac{\partial \psi}{\partial R} = \frac{\partial^2 \psi}{\partial R^2} - \frac{1}{R} \frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial z^2},$$

and

$$\Delta^\dagger \psi \equiv \nabla_\perp^2 \psi + \frac{1}{R} \frac{\partial \psi}{\partial R} = \frac{\partial^2 \psi}{\partial R^2} + \frac{1}{R} \frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial z^2}.$$

M3D Form of the Resistive MHD Equations

Define Poisson Bracket $\left[A,B\right]\equiv\nabla_{\perp}A\times\nabla_{\perp}B\cdot\hat{\phi}=\frac{\partial A}{\partial R}\frac{\partial B}{\partial z}-\frac{\partial A}{\partial z}\frac{\partial B}{\partial R}$

and $\left(A,B\right)\equiv\nabla_{\perp}A\cdot\nabla_{\perp}B=\frac{\partial A}{\partial R}\frac{\partial B}{\partial R}+\frac{\partial A}{\partial z}\frac{\partial B}{\partial z}$

Continuity:

$$\frac{\partial\rho}{\partial t}=-\nabla\cdot\left(\rho\vec{V}\right)=-\rho\left(\Delta^{\dagger}\chi+\frac{2}{R_0}\frac{\partial U}{\partial z}+\frac{1}{R}\frac{\partial V_{\phi}}{\partial\phi}\right)-\frac{R}{R_0}\left[\rho,U\right]-\left(\rho,\chi\right)-\frac{V_{\phi}}{R}\frac{\partial\rho}{\partial\phi}\qquad(1)$$

Operate on the momentum equation with $-R_0\hat{\phi}\cdot\nabla\times$ to get an equation for the evolution of $\Delta^{\dagger}U\equiv\nabla_{\perp}^{2}U+\frac{1}{R}\frac{\partial U}{\partial R}$ (called “`W`” in the code):

$$\begin{aligned}\frac{\partial}{\partial t}\Delta^{\dagger}U=&\frac{R}{R_0}\left[U,\Delta^{\dagger}U\right]-\left(\chi,\Delta^{\dagger}U\right)-\Delta^{\dagger}U\left(\Delta^{\dagger}\chi+\frac{2}{R_0}\frac{\partial U}{\partial z}\right)-\frac{V_{\phi}}{R}\frac{\partial}{\partial\phi}\Delta^{\dagger}U-\left(\frac{V_{\phi}}{R},\frac{\partial U}{\partial\phi}\right)\\&+2R_0\frac{V_{\phi}}{R}\frac{\partial}{\partial z}\frac{V_{\phi}}{R}+\frac{R_0}{R}\left[\frac{V_{\phi}}{R},\frac{\partial\chi}{\partial\phi}\right]+R_0\left\{\vec{B}\cdot\nabla\left(\frac{C}{R^2\rho}\right)+\vec{J}\cdot\nabla\left(\frac{1+\tilde{I}/R_0}{R^2\rho}\right)\right\}+\frac{2}{R^2\rho}\frac{\partial p}{\partial z}\\&+R\left[\frac{1}{R^2\rho},p\right]-R_0\nabla\phi\cdot\nabla\times\left(\frac{\mu\nabla^2\vec{V}}{\rho}\right)\end{aligned}\qquad(2a)$$

Evolution of the Compressible Velocity

From the definition of the velocity, it is clear that

$$\frac{\partial \chi}{\partial R} = \hat{R}\cdot\vec{V} - \frac{R}{R_0}\frac{\partial U}{\partial z} \qquad \text{and} \qquad \frac{\partial \chi}{\partial z} = \hat{z}\cdot\vec{V} + \frac{R}{R_0}\frac{\partial U}{\partial R}$$

so that, again using the momentum equation,

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{\partial \chi}{\partial R}\right) = &-\frac{R}{R_0}\frac{\partial}{\partial z}\left(\frac{\partial U}{\partial t}\right) - \vec{V}_\perp\cdot\nabla_\perp\left(\frac{\partial \chi}{\partial R} + \frac{R}{R_0}\frac{\partial U}{\partial z}\right) - \frac{V_\phi}{R_0}\frac{\partial U'}{\partial z} - \frac{V_\phi}{R}\frac{\partial \chi'}{\partial R} + \frac{V_\phi^2}{R} - \frac{1}{\rho}\frac{\partial p}{\partial R}\\ &+\frac{1}{R^2\rho}\left(R_0+\tilde{I}\right)\left[\frac{1}{R}\left(\frac{\partial F'}{\partial R} + \frac{\partial \psi'}{\partial z}\right) - \frac{\partial \tilde{I}}{\partial R}\right] + \frac{C}{R^2\rho}\left(\frac{\partial F}{\partial z} - \frac{\partial \psi}{\partial R}\right) + \frac{\mu}{\rho}\hat{R}\cdot\nabla^2\vec{V}\end{aligned} \qquad (2b)$$

and

$$\begin{aligned}\frac{\partial}{\partial t}\left(\frac{\partial \chi}{\partial z}\right) = &\frac{R}{R_0}\frac{\partial}{\partial R}\left(\frac{\partial U}{\partial t}\right) - \vec{V}_\perp\cdot\nabla_\perp\left(\frac{\partial \chi}{\partial z} - \frac{R}{R_0}\frac{\partial U}{\partial R}\right) + \frac{V_\phi}{R_0}\frac{\partial U'}{\partial R} - \frac{V_\phi}{R}\frac{\partial \chi'}{\partial z} - \frac{1}{\rho}\frac{\partial p}{\partial z}\\ &+\frac{1}{R^2\rho}\left(R_0+\tilde{I}\right)\left[\frac{1}{R}\left(\frac{\partial F'}{\partial z} - \frac{\partial \psi'}{\partial R}\right) - \frac{\partial \tilde{I}}{\partial z}\right] - \frac{C}{R^2\rho}\left(\frac{\partial F}{\partial R} + \frac{\partial \psi}{\partial z}\right) + \frac{\mu}{\rho}\hat{z}\cdot\nabla^2\vec{V}\end{aligned} \qquad (2c)$$

Evolution of the Toroidal Velocity

Dot the momentum equation with $\hat{\phi}$ to find

$$\frac{\partial V_\phi}{\partial t} = \frac{R}{R_0}\left[U, V_\phi\right] - \left(\chi, V_\phi\right) - \frac{V_\phi}{R}\left(V_\phi{}' + \frac{\partial \chi}{\partial R}\right) - \frac{V_\phi}{R_0}\frac{\partial U}{\partial z} - \frac{1}{R\rho}\frac{\partial p}{\partial \phi}$$

$$+ \frac{1}{R^2\rho}\left[\tilde{I}, \psi\right] + \frac{1}{R^2\rho}\left(\tilde{I}, F\right) + \frac{1}{R^3\rho}\frac{\partial}{\partial \phi}[\psi, F] - \frac{1}{2R^3\rho}\frac{\partial}{\partial \phi}\left(\left|\nabla_\perp \psi\right|^2 + \left|\nabla_\perp F\right|^2\right) \qquad (2d)$$

$$+ \frac{\mu}{\rho}\left[\nabla^2 V_\phi - \frac{V_\phi}{R^2} + \frac{2}{R^2}\frac{\partial}{\partial \phi}\left(\frac{R}{R_0}\frac{\partial U}{\partial z} + \frac{\partial \chi}{\partial R}\right)\right]$$

Electrostatic Potential

If $\vec{B} = \nabla \times \vec{A}$ and $\dfrac{\partial \vec{B}}{\partial t} = -\nabla \times \vec{E}$ then $\dfrac{\partial \vec{A}}{\partial t} = -\vec{E} + \nabla \Phi$ where, if we choose the gauge

$\nabla_\perp \cdot \vec{A} = 0,$ we find $\nabla_\perp^2 \Phi = \nabla_\perp \cdot \vec{E}.$

For the resistive MHD Ohm's law, that means

$$\nabla_\perp^2 \Phi = \frac{1}{R_0}\left(\tilde{I}, U\right) + \left(1 + \frac{\tilde{I}}{R_0}\right)\nabla_\perp^2 U - \frac{V_\phi}{R}\Delta^* \psi + \frac{R_0}{R^2}\frac{\partial \chi}{\partial z} + \left[\chi, \frac{\tilde{I}}{R}\right] - \left[F, \frac{V_\phi}{R}\right] - \frac{1}{R}\left(V_\phi, \psi\right)$$
$$+ \frac{\eta}{R^2}\left[\frac{1}{R}\left(\frac{\partial F'}{\partial z} - \frac{\partial \psi'}{\partial R}\right) - \frac{\partial \tilde{I}}{\partial z} + \frac{\partial C}{\partial \phi}\right] + \frac{1}{R}\left[\eta, \tilde{I}\right] - \frac{1}{R^2}\left[\eta, F'\right] + \frac{1}{R^2}\left(\eta, \psi'\right) \qquad (3)$$

Evolution of the Poloidal Field

The time derivative of ψ (called "a" in the code) is simply $R\hat{\phi}\cdot\dfrac{\partial \vec{A}}{\partial t}$,

$$\frac{\partial \psi}{\partial t}=\frac{R}{R_0}[U,\psi]+\frac{R}{R_0}(U,F)-(\chi,\psi)+[\chi,F]+\eta C+\frac{\partial \Phi}{\partial \phi}. \tag{4}$$

but for numerical stability, the quantity we generally choose to evolve is instead $C_a \equiv \Delta^* \psi$:

$$\begin{aligned}
\frac{\partial C_a}{\partial t} = {} & \frac{R}{R_0}\left\{[U,C_a]+\left[\Delta^\dagger U,\psi\right]+2\left[\frac{\partial U}{\partial R},\frac{\partial \psi}{\partial R}\right]+2\left[\frac{\partial U}{\partial z},\frac{\partial \psi}{\partial z}\right]\right\}+\frac{2}{R_0}\left[U,\frac{\partial \psi}{\partial R}\right]+\frac{2}{R_0 R}\frac{\partial U}{\partial z}\frac{\partial \psi}{\partial R} \\
& +\frac{R}{R_0}\left\{\left(U,\nabla_\perp^2 F\right)+\left(\Delta^\dagger U,F\right)+2\left(\frac{\partial U}{\partial R},\frac{\partial F}{\partial R}\right)+2\left(\frac{\partial U}{\partial z},\frac{\partial F}{\partial z}\right)\right\}+\frac{1}{R_0}\left(\frac{\partial F}{\partial R},U\right)-\frac{1}{R_0 R}\frac{\partial F}{\partial z}\frac{\partial U}{\partial z} \\
& -\left\{\left(\psi,\nabla_\perp^2\chi\right)+(C_a,\chi)+2\left(\frac{\partial \psi}{\partial R},\frac{\partial \chi}{\partial R}\right)+2\left(\frac{\partial \psi}{\partial z},\frac{\partial \chi}{\partial z}\right)\right\}+\frac{1}{R}\left(\frac{\partial \chi}{\partial R},\psi\right)+\frac{1}{R^2}\frac{\partial \psi}{\partial R}\frac{\partial \chi}{\partial R} \\
& +\left\{\left[\nabla_\perp^2\chi,F\right]+\left[\chi,\nabla_\perp^2 F\right]+2\left[\frac{\partial \chi}{\partial R},\frac{\partial F}{\partial R}\right]+2\left[\frac{\partial \chi}{\partial z},\frac{\partial F}{\partial z}\right]\right\}-\frac{1}{R}\left\{\left[\frac{\partial \chi}{\partial R},F\right]+\left[\chi,\frac{\partial F}{\partial R}\right]\right\} \\
& +\frac{\partial}{\partial \phi}\left(\nabla_\perp^2\Phi\right)-\frac{1}{R}\frac{\partial^2 \Phi}{\partial \phi\,\partial R}
\end{aligned} \tag{4$'$}$$

Evolution of the Toroidal Field

The magnetic field is completely specified by two scalar functions; the auxiliary variable F is related to the non-vacuum toroidal field $\tilde{I}/R$ by the elliptic equation given earlier. The evolution of $\tilde{I}$ can be found from the toroidal component of the field equation:

$$\begin{aligned}\frac{\partial \tilde{I}}{\partial t} = \frac{R}{R_0}\left[U,\tilde{I}\right] - \left(\chi,\tilde{I}\right) + R\left[\frac{V_\phi}{R},\psi\right] + R\left(\frac{V_\phi}{R},F\right) - \left(R_0+\tilde{I}\right)\Delta^*\chi - \frac{V_\phi}{R}\frac{\partial \tilde{I}}{\partial \phi} \\ +\eta\left[\Delta^*\tilde{I} - \frac{1}{R}\nabla_\perp^2 F' + \frac{2}{R^2}\left(\frac{\partial \psi'}{\partial z} + \frac{\partial F'}{\partial R}\right)\right] - \frac{1}{R}\left[\eta,\psi'\right] + \left(\eta,\tilde{I}\right) - \frac{1}{R}\left(\eta,F'\right)\end{aligned} \qquad (5)$$

The Energy Equation

The energy equation in the resistive MHD version M3D is normally solved in terms of the plasma pressure; simple substitution of the code variables into the pressure equation gives

$$\frac{\partial p}{\partial t} = \frac{R}{R_0}[U, p] - (\chi, p) - \frac{V_\phi}{R}\frac{\partial p}{\partial \phi} - \gamma p\left(\frac{2}{R_0}\frac{\partial U}{\partial z} + \Delta^\dagger \chi + \frac{1}{R}\frac{\partial V_\phi}{\partial \phi}\right) + \rho \nabla \cdot \left[\kappa_\perp \nabla\left(\frac{p}{\rho}\right)\right] \qquad (6)$$

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

The M3D Mesh

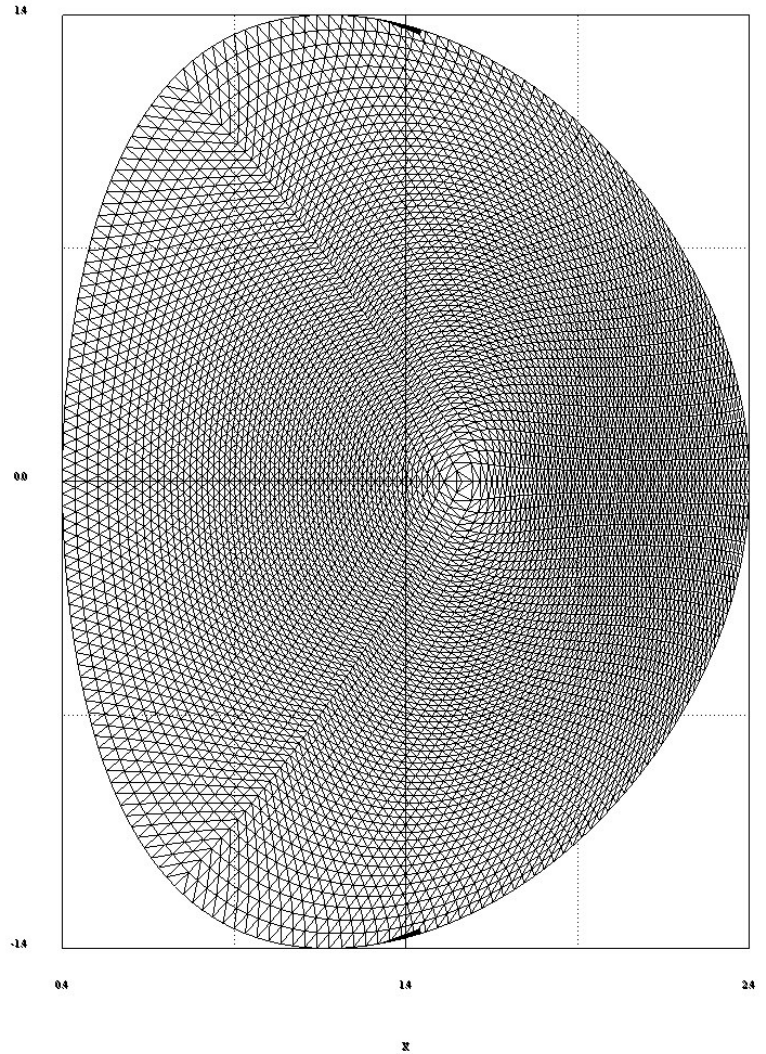
- Uses linear basis functions on **unstructured triangular finite element** mesh in each constant-ϕ plane.

- 3 parameters control mesh resolution: # of planes, # of radial grids, # of theta sections.

- Mesh has same topology in all planes. In the tokamak case, it has the same geometry in all planes as well.

- Mesh is aligned with equilibrium flux surfaces (from VMEC-generated input files) but does not follow field lines.

- Uses either 4th-order finite differences or pseudo-spectral derivatives between planes.



Packing the Mesh at a Flux Surface

In order to resolve fine structures at a particular surface, the option exists to concentrate zones of the M3D mesh about a given minor radius (1D packing).

Command line options:

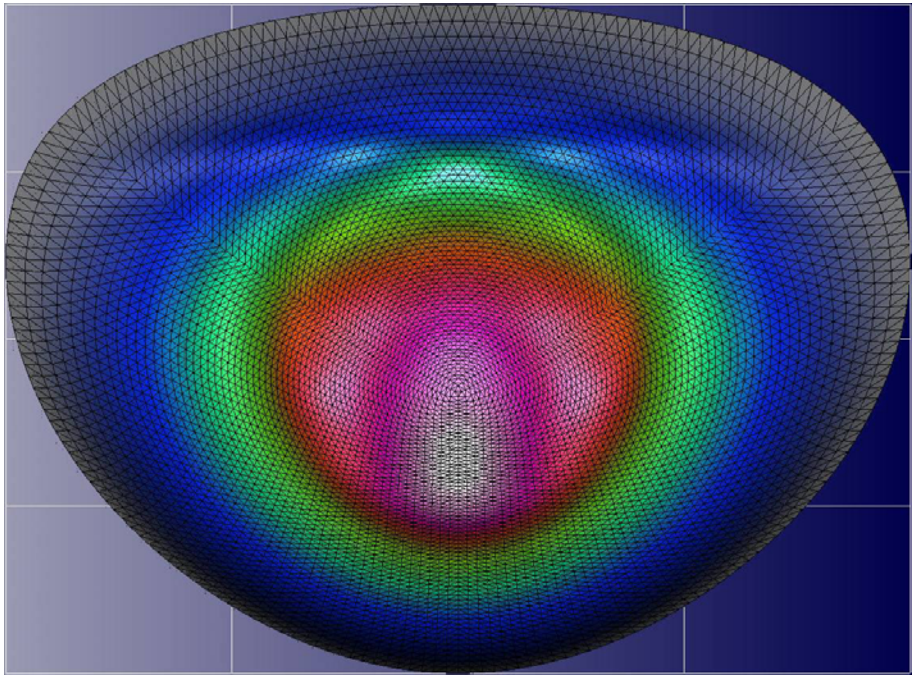
`-packingFactor` <*pf*> *Ratio of packed to unpacked mesh density.*

`-packingRadius` <x_0> *Relative position of packing surface (from 0 to 1).*

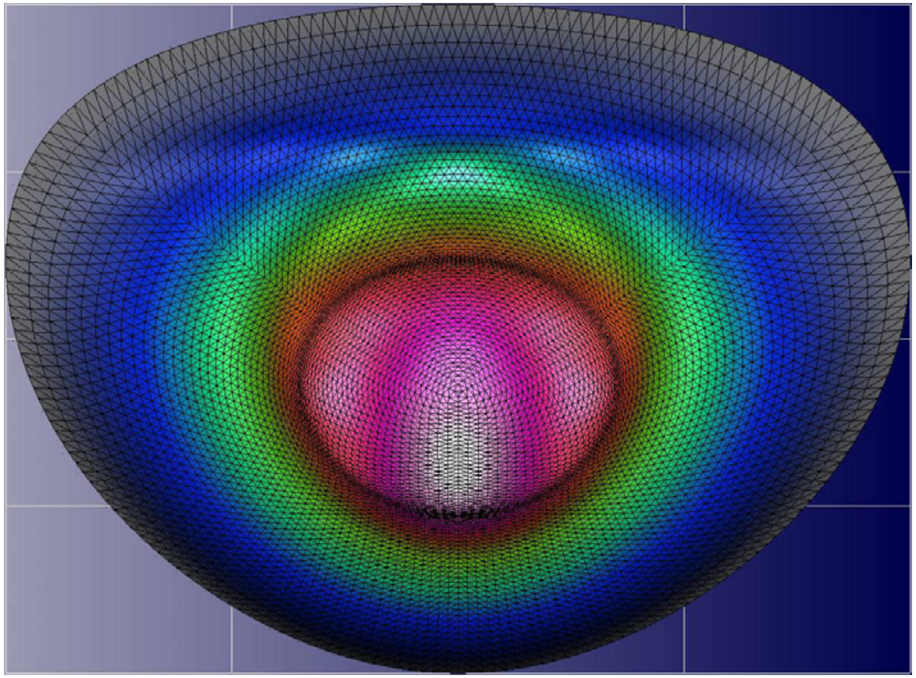
`-packingWidth` <*w*> *Relative width of peak packing area (on 0 to 1 scale).*

Example: pf=4.0; x_0=0.5; w=0.12:

Before packing

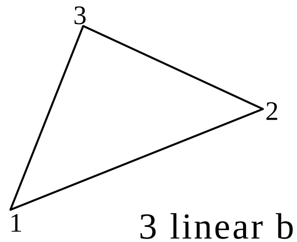


After packing



Linear Finite Elements

Linear basis functions on a triangle:



Side lengths $d\vec{r}_1 \equiv \vec{r}_2 - \vec{r}_3$, etc.

Area $\Delta = \frac{1}{2} d\vec{r}_1 \times d\vec{r}_2 \cdot \hat{\phi}$

3 linear basis functions $\lambda_\alpha(\vec{r}) = \frac{1}{4\Delta} \sum_{\beta \neq \alpha} \left(\vec{r} - \vec{r}_\beta\right) \times d\vec{r}_\alpha \cdot \hat{\phi}$

$\lambda_\alpha(\vec{r}_\alpha) = 1; \; \lambda_\alpha(\vec{r}_{\beta \neq \alpha}) = 0$

Galerkin method: integrate equations over each basis function to get “weak form” $\rightarrow$ linear algebraic equation.

$$f(R,z) = \sum_j f_j \lambda_j(R,z)$$

Mass matrix: $\iint \lambda_i f(R,z) d^2x = \sum_j f_j \iint \lambda_i \lambda_j d^2x \equiv \sum_j M_{i,j} f_j$

Stiffness matrix: $\iint \lambda_i \nabla_\perp^2 f\left(R,z\right) d^2x = \sum_j f_j \iint \lambda_i \nabla_\perp^2 \lambda_j d^2x = \sum_j f_j \left\{ \cancel{\iint \nabla_\perp \cdot \left(\lambda_i \nabla \lambda_j\right) d^2x} - \iint \nabla_\perp \lambda_i \cdot \nabla_\perp \lambda_j d^2x \right\} \equiv \sum_j S_{i,j} f_j$

"dRoverR" matrix: $\iint \frac{\lambda_i}{R} \frac{\partial}{\partial R} f(R,z) d^2x = \sum_j f_j \iint \frac{\lambda_i}{R} \frac{\partial \lambda_j}{\partial R} d^2x \equiv \sum_j R_{i,j} f_j$

Handy identity: $\iint_\Delta \lambda_1^\ell \lambda_2^m \lambda_3^n d^2x = 2\Delta \frac{\ell! m! n!}{\left(\ell + m + n + 2\right)!}$

Lumped mass matrix (diagonal): $\bar{\mathbf{M}}_{i,j} \equiv \delta_{i,j} \sum_j M_{i,j}$

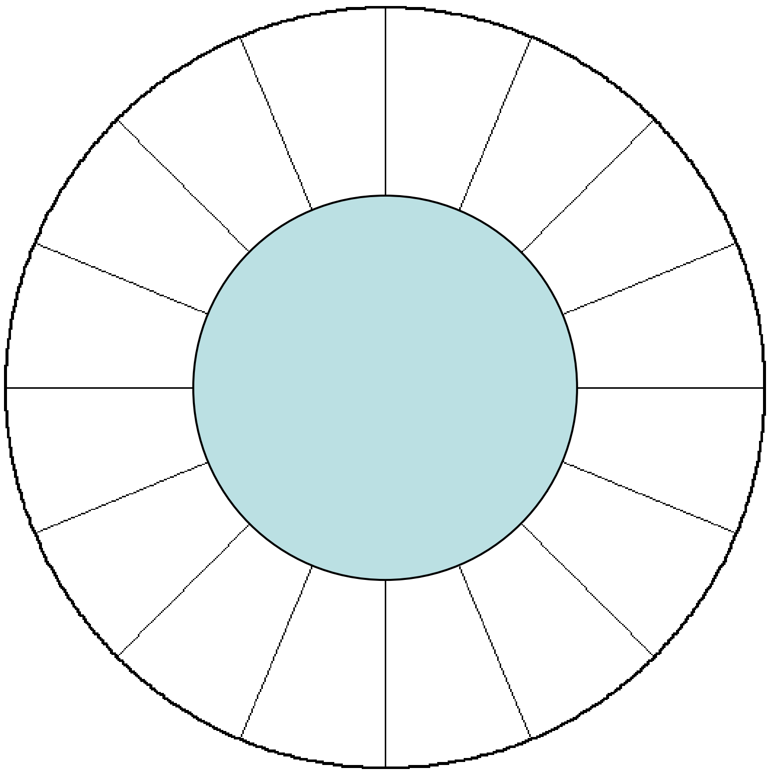
Boundary Conditions

- All calculations use a fixed boundary.
- Standard cases use perfectly conducting wall, with or without a “slot”.
- Slip or no-slip conditions may be imposed.
- Most of these are realized as Dirichlet b.c.s in linear solves. Exceptions: F, χ use Neumann.

Domain Decomposition

3 parameters control domain decomposition: # of toroidal PEs, # of radial PEs, # of theta PEs.

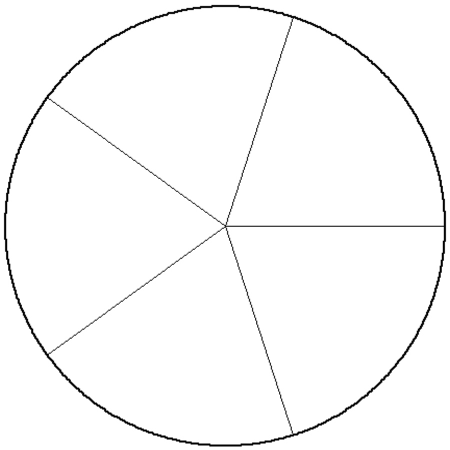
Toroidal
(overhead view)



B = 16

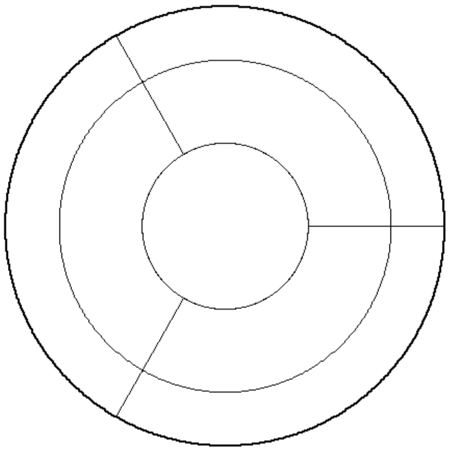
Linear solves are independent on each processor

Poloidal
(cross-section view)



D = 1
F = 5

or



D = 3
F = 3

Linear solves are parallel over processors

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

Time Discretization, Overview

- Equations (1-6) are advanced explicitly, except for parabolic and fast wave terms.
- Time discretization is typically 1st order, forward-in-time. 2nd-order predictor-corrector is also an option.
- Artifical sound term, if selected, is advanced in subcycles of the main time step.
- Code execution time is dominated by ~13 linear solves per time step, each of size *N*, where *N* is the number of vertices in a single plane.
  - Elliptic solves are more expensive than Helmholtz.
  - Neumann b.c.s are more expensive than Dirichlet.

Schematic of Equation Solve

Generic mixed hyperbolic/parabolic equation:

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = D\frac{\partial^2 f}{\partial x^2}$$

Galerkin F.E. method

1. Explicit solve: $f^* = f^n - (\delta t)u^n\left(\frac{\partial f}{\partial x}\right)^n$

2. Implicit solve:

$$\left[\frac{\partial^2}{\partial x^2} - \frac{1}{D\delta t}\right]\left(f^{n+1} - f_{source}\right) = -\frac{\left(f^* - f_{source}\right)}{D\delta t}$$

Order of Operations in Main Loop

1. Recompute dt based on CFL condition for shear Alfvén wave.
2. Adjust resistivity profile to track temperature.
3. Compute $I = \varepsilon + \tilde{I},\ B^2$
4. Advance particles if hybrid option is on.
5. Solve (2a) for vorticity $w = \Delta^{\dagger} U$; ideal terms explicitly, followed by implicit solve for viscous term and elliptic solve for *U*.
6. Simultaneously solve (5) for toroidal field, (2b-c) for $\nabla_{\perp}\chi$, and ideal part of (6) for pressure or temperature implicitly (in-plane) to step over fast wave time scale. Integrate to solve for χ. Many terms are still explicit; resistivity, viscosity and heat diffusion are still implicit, perpendicular to $\nabla\varphi$.
7. Apply perpendicular (or isothermal) heat conduction.
8. Advance (1) for density ρ.
9. Advance artificial sound wave.
10. Advance (2d) for toroidal velocity.
11. Solve elliptic equation (3) for electrostatic potential.
12. Solve (4) for ψ or Δ^*(4) for C_a followed by an elliptic solve for ψ.
13. Solve elliptic equation for *F*.
14. Diagnostics, output, checkpointing.

Artificial Sound Wave Substep

$$\frac{\partial T}{\partial t} = s\frac{\mathbf{B}\cdot\nabla u}{\rho}$$

$$\frac{\partial u}{\partial t} = s\mathbf{B}\cdot\nabla T + \nu\nabla^2 u$$

Repeat `napmax` times:

- Solve *T* equation with reduced time step `rdtdp` explicitly.
- Solve hyperbolic part of *u* equation explicitly.
- Solve parabolic part of *u* equation implicitly.
- Check stability.

Linear vs. Nonlinear

By default, the time advance is fully nonlinear. However an option exists to search for linear toroidal eigenmodes.

- Begin by adding a perturbation with toroidal mode #*n* to velocity variable *U* in equilibrium.
- With pseudospectral method, only three planes are needed to resolve the mode. (Use number of field periods = *n*).
- After each nonlinear advance of a variable, find the mode *n* component, add to the *n*=0 component from the original equilibrium to get advanced-time value.
- Fastest-growing mode will eventually dominate over others; growth rate determined from rate of change of total kinetic energy.
- Rescale perturbed quantities periodically to keep total kinetic energy below nonlinear level but above noise.

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

PETSc

- Portable, Extensible Toolkit for Scientific Computation.
  - MPI-based suite of data structures & routines for parallel solution of PDEs.
  - Maintained by PETSc group, Mathematics and Computer Science Division, Argonne National Lab.
  - Latest version is 2.3.2.

- The MPI version of M3D is highly dependent on PETSc.
  - Uses versions 2.1.6, 2.3.0.
  - Parallel data structures, ghost exchanges
    - Vectors (variable fields)
    - Matrices (linear operators)
  - Linear solves – great flexibility in solver choices
    - Asymmetric operators: GMRES
    - Symmetric operators: CG
    - Direct solves (SuperLU), Multigrid
  - Most of M3D computation occurs in PETSc solves, so we rely on PETSc optimization for performance, scalability.

HDF5

- Hierarchical Data Format
  - Widely adopted and supported portable binary format
  - Allows self-describing data organized in file-system-like hierarchies.
  - Random access

- M3D uses HDF5 as its primary output option.
  - A subset of the fields in the checkpoint (12 scalar, 1 vector) is written every several time steps, in single precision.
    - Mesh is described as a set of triangular prisms.
    - Data values are given at vertices.
  - Checkpoint files can also be converted between native binary and HDF5 for intersystem portability.
  - UCD (text) output is another option; the OpenMP version can also produce NCAR graphics.

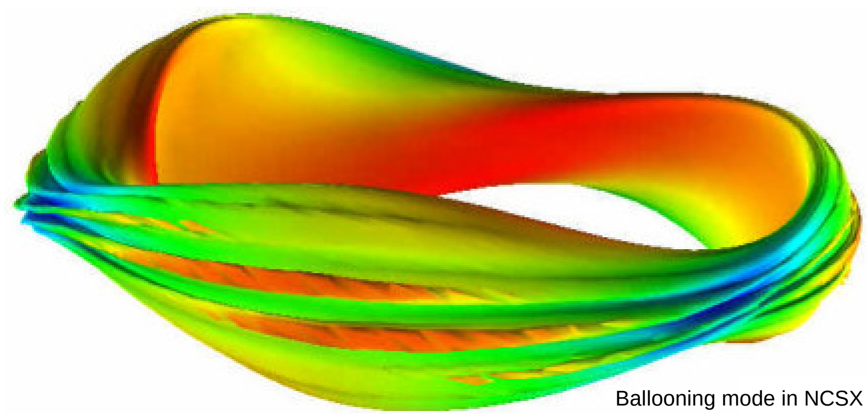
Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

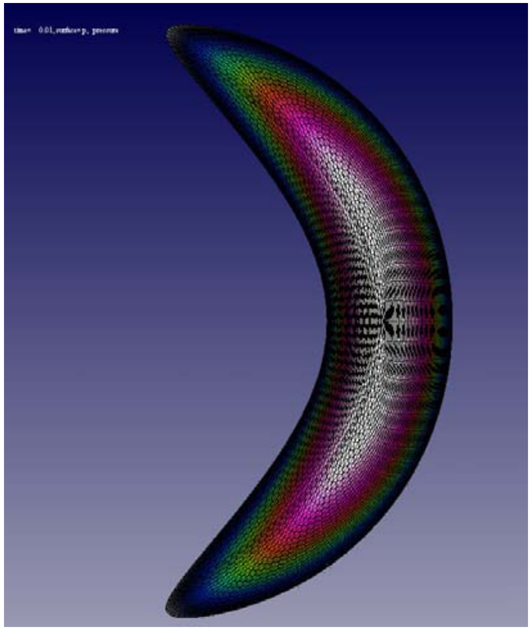
Stellarator

(H. Strauss)

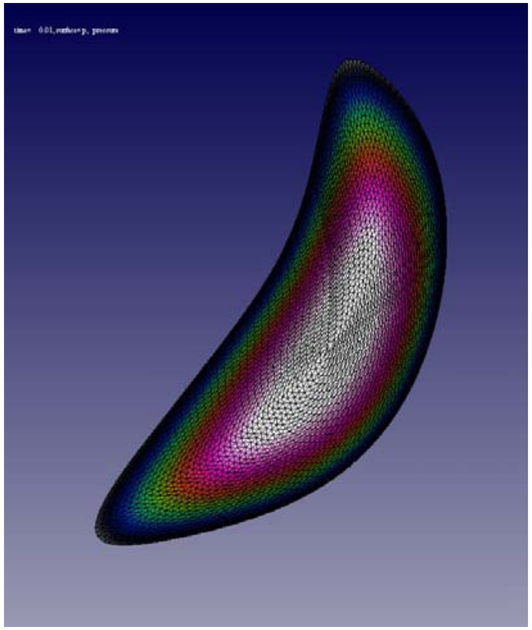
- By generating a mesh from a 3D equilibrium file, M3D can run stellarator cases.
- Planes can be made to span just one field period.
- Toroidal derivatives require extra terms for toroidal mesh variation, impacting speed and accuracy.



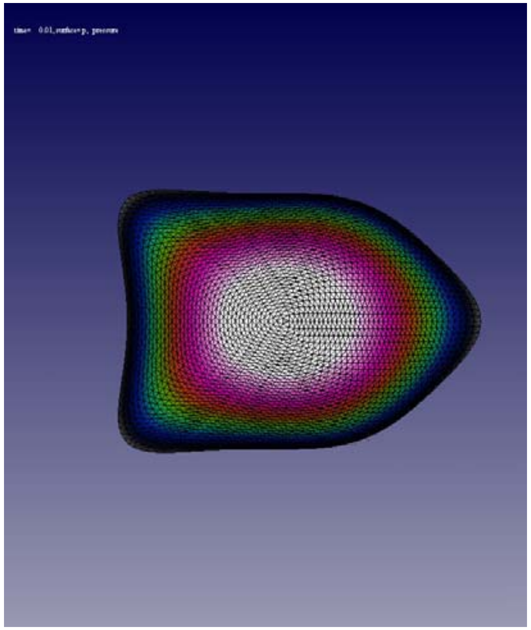
Ballooning mode in NCSX



$\varphi = 0$



$\varphi = \pi/6$



$\varphi = \pi/3$

Two Fluid

(L. Sugiyama)

- A hierarchy of extended MHD models exists in M3D.

- The simplest uses the drift ordering to approximate the ion gyroviscous stress tensor term in the momentum equation ($-\nabla\cdot\Pi_i^{gv}$) using the diamagnetic drift velocity:

$$\rho\left[\frac{\partial \mathbf{v}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{v}+\left(\mathbf{v}_i^*\cdot\nabla\right)\mathbf{v}_\perp\right]=-\nabla\left(\mathbf{p}_e+\mathbf{p}_i\right)+\mathbf{J}\times\mathbf{B}+\mu\nabla^2\mathbf{v}$$

$$\mathbf{v}_i^*\equiv\mathbf{v}_e^*+\frac{\mathbf{J}_\perp}{ne},\qquad \mathbf{v}_e^*\equiv-\frac{\mathbf{B}\times\nabla p_e}{neB^2},\qquad \mathbf{v}\equiv\mathbf{v}_i-\mathbf{v}_i^*=\mathbf{v}_e-\mathbf{v}_e^*+\frac{\mathbf{J}_{\|}}{ne}$$

- The Hall term can also be added to Ohm’s law, introducing the dispersive whistler wave, which is very difficult to stabilize.

Hybrid (Kinetic Hot Ions)

(G. Fu)

- Gyrokinetic particle push based on GTC group's formulation.
- Large ensemble of ions substepped through interpolated M3D ***B*** field.
- Hot ions couple back to fluid model through pressure tensor:

$$\rho \frac{d\mathbf{v}}{dt} + \rho\left(\mathbf{v}_i^* \cdot \nabla\right)\mathbf{v}_\perp = -\nabla P - \nabla \cdot \mathbf{P}_h + \mathbf{J} \times \mathbf{B} - \mathbf{bb} \cdot \nabla \cdot \Pi_i$$

where $\mathbf{P}_h = P_\perp \mathbf{I} + \left(P_\| - P_\perp\right)\mathbf{bb}$

based on moments taken over the particle distribution function

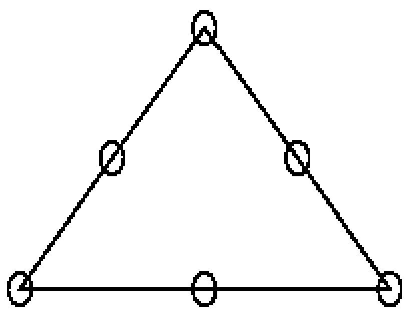
$$f = \sum_i \delta\left(\mathbf{R} - \mathbf{R}_i\right)\delta\left(v_\| - v_{\|,i}\right)\delta\left(\mu - \mu_i\right)$$

- MPI Parallelization follows domain decomposition of M3D mesh; particles can move between processors.
- Typical particle push time is comparable to fluid advance time.
- Fully kinetic ion model (with fluid electrons) also exists.

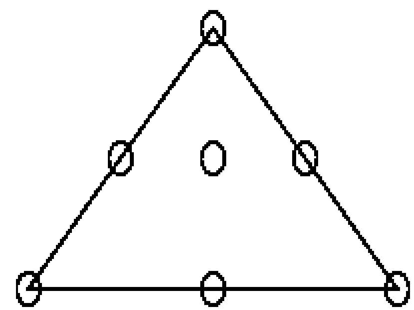
Higher-Order Elements

(H. Strauss, J. Chen)

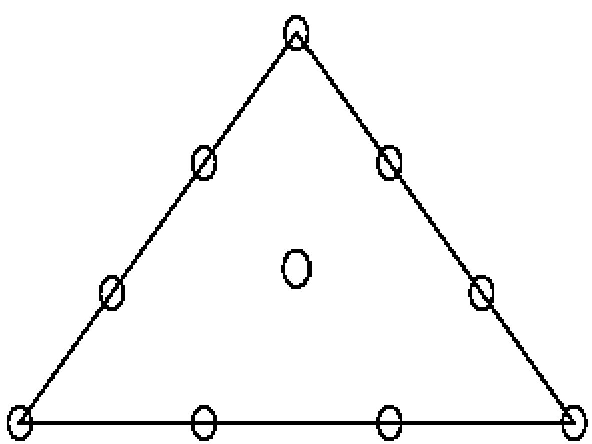
- 2^{nd} and 3^{rd}-order polynomial elements are available.
- Formed by adding nodes to existing mesh triangles.
- In "lumped" elements, nodes are placed at quadrature points of integral, resulting in a diagonal mass matrix for much faster evaluation, at a cost of more vertices.



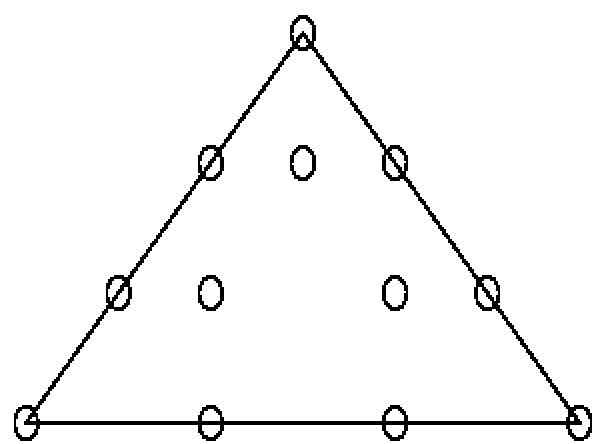
2^{nd}-order Lagrange



2^{nd}-order lumped



3^{rd}-order Lagrange

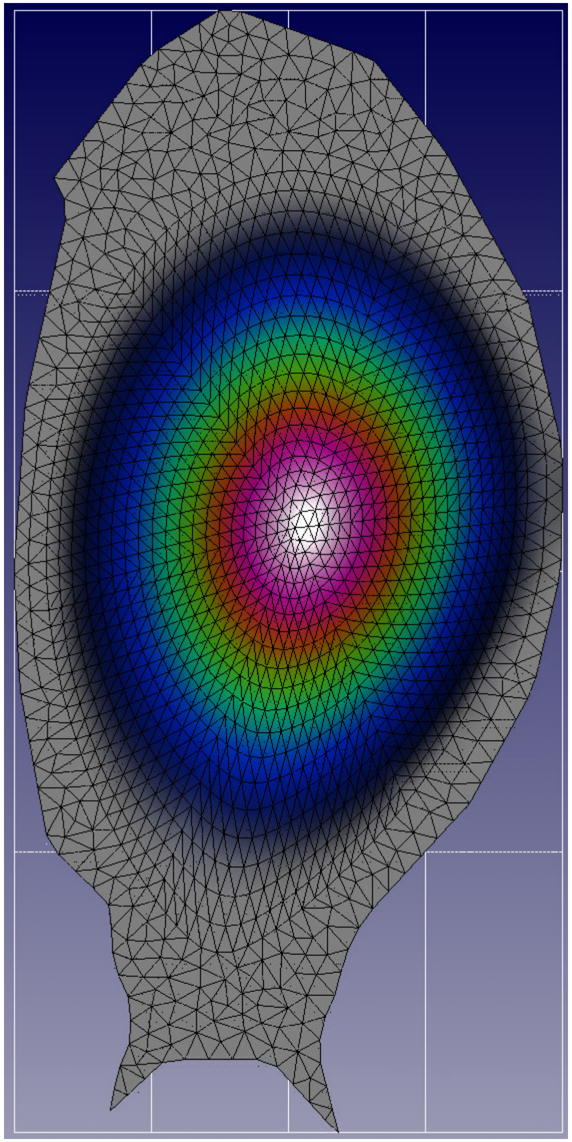


3^{rd}-order lumped

Resistive Wall

(H. Strauss, J. Breslau)

- OpenMP code uses external package to generate vacuum-region mesh extending M3D mesh out to wall.
- Mesh may exclude axis region (not shown) with internal boundary condition.
- MPI version can initialize from mesh+data file generated by OMP version.
- Vacuum region treated as low density, low temperature (high η) plasma.
- Boundary conditions on fields at wall are applied using Green's functions precomputed by GRIN for each toroidal mode based on boundary geometry.



VDE in ASDEX (early time).

Outline

- Overview
  - Version history
  - Platforms
  - Statistics
- Equations
  - Standard form
  - M3D form
- Spatial Discretization
  - Meshing
  - Linear elements
  - Domain decomposition
- Time advance
  - General form
  - Detailed scheme
  - Artificial sound wave
  - Linear operation
- Libraries
  - PETSc
  - HDF5
- Other Options
  - Stellarator
  - Two-fluid
  - Hybrid
  - Higher-order elements
  - Resistive wall
- Concluding thoughts

Concluding Thoughts

- The proliferation of new physics modules, options, and numerical techniques has made M3D very versatile and flexible but also very complex and challenging to maintain.

- A set of thorough standard tests for validation is badly needed.

- The code has been very productive on present machines, producing results few other MHD codes are capable of.

- But it could be a lot more efficient, and scaling up usefully to petascale runs remains a formidable hurdle.

    - Need more implicitness.
    - Need higher order elements.
    - Need efficient, scalable solvers.