Extended MHD plasma simulation for fusion

Linda Sugiyama
MIT

Fusion SciDAC CEMM + CPES

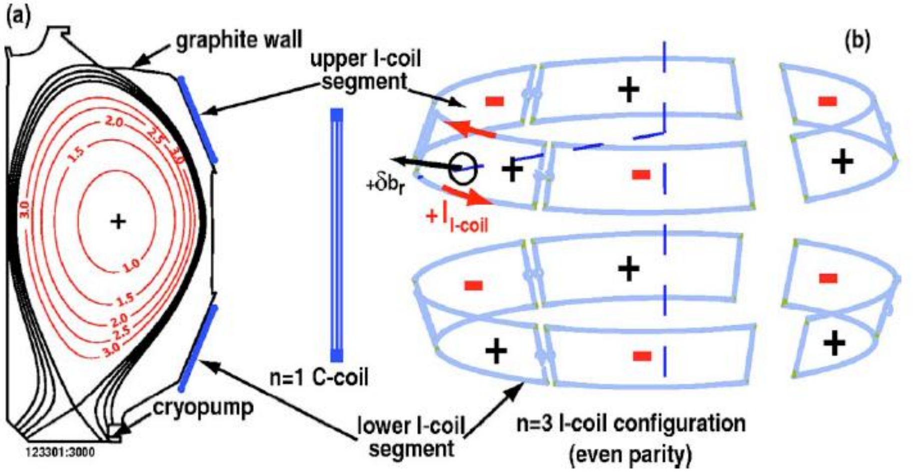
CScADS Petascale Workshop
26-30 July 2009

M3D code

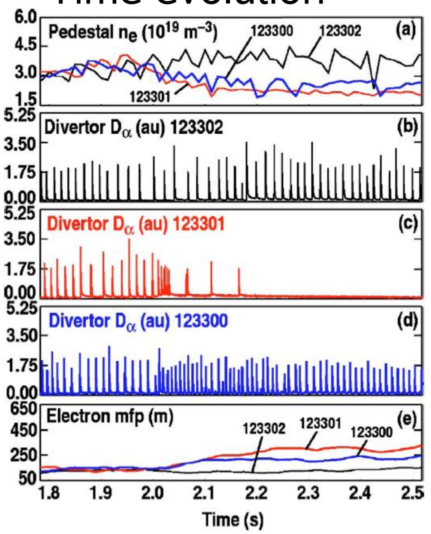
- Magnetically confined, toroidal plasmas for fusion research
  - Nonlinear time evolution of plasma (n,T,**v**) and magnetic field (**B**,**J**) in real configurations
- Original code aimed at macroscopic scale, coherent instabilities
- Physics evolving
  - Recent extension to 'vacuum' surrounding the plasma with freely moving plasma boundary → chaotic magnetic fields and plasma!
- Computational capabilities improving
  - Simulations are approaching turbulence resolutions (toroidal Fourier harmonics n > 20-40, relative perturbation size few x 10^{-4}).
  - Petascale → full turbulence?
- Nature of code changing: How to make the transition effectively?

Close connection to experiments

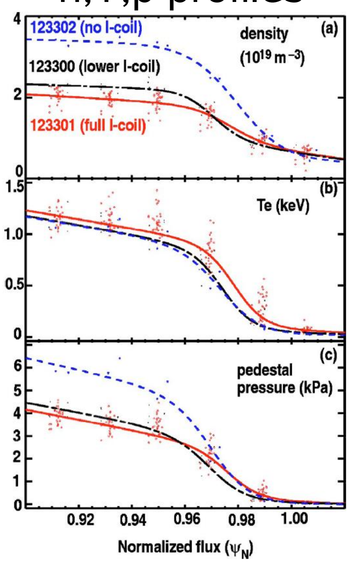
- Add perturbations (eg, field coils)
- Sophisticated diagnostics, but many things remain difficult to measure; large error bars
- Example: edge instabilities (ELM)

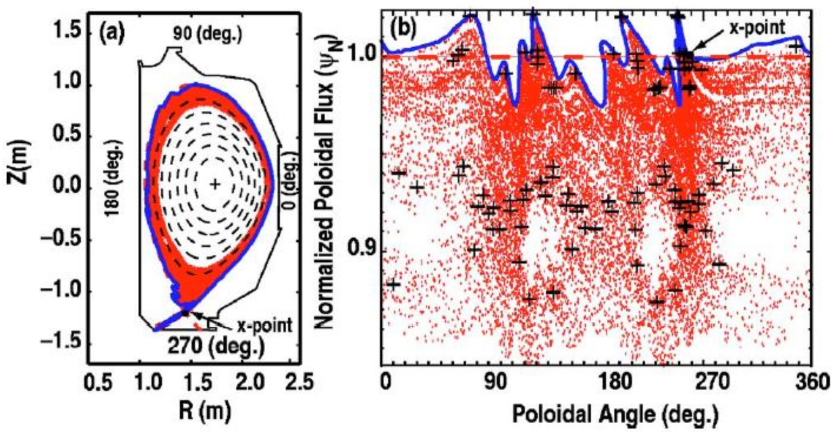


Time evolution



n,T,p profiles

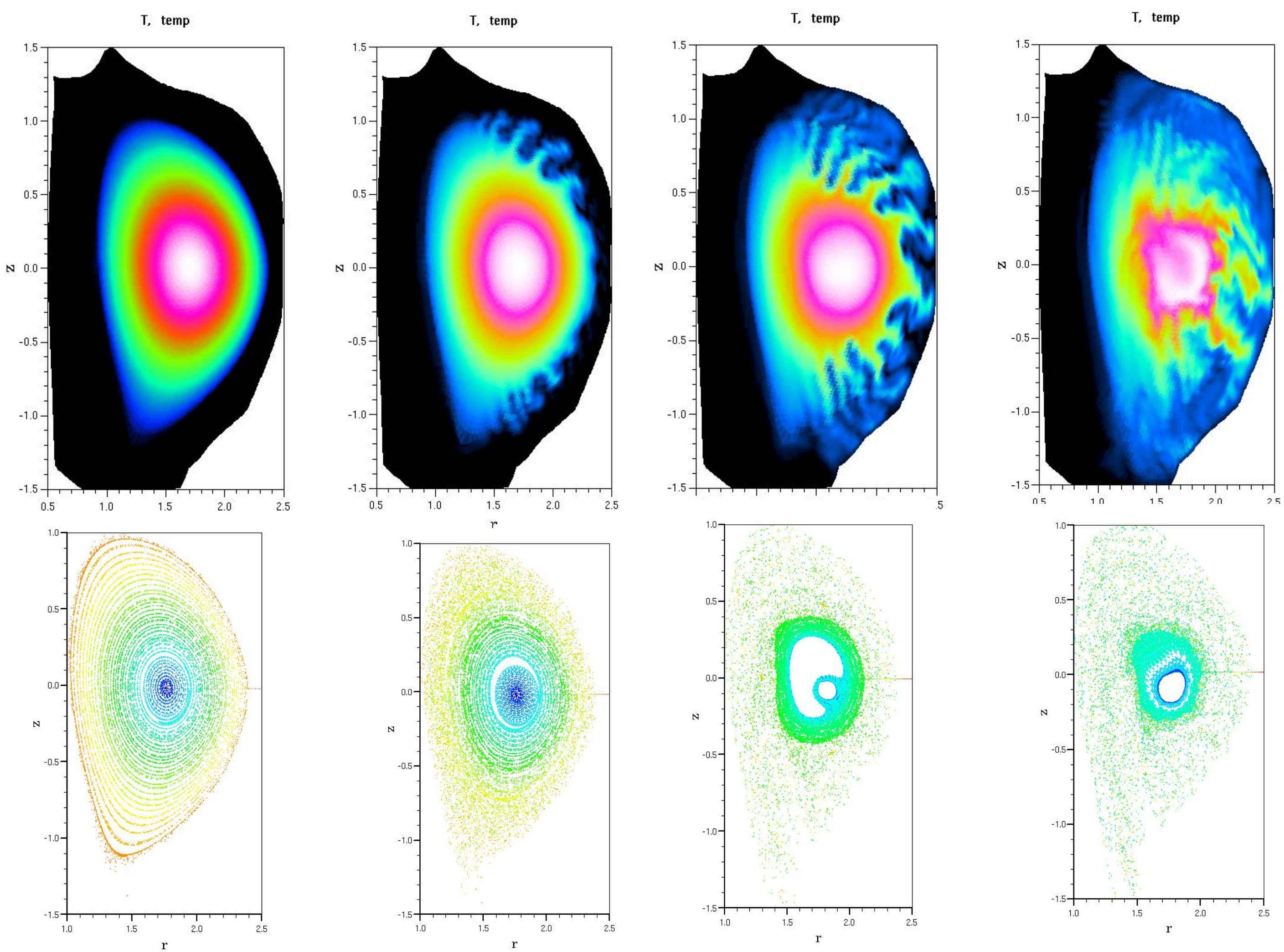




Apply small B field near edge to stabilize ELMs – chaotic

MHD simulation shows that edge instabilities (ELMs) couple coherent plasma structures to stochastic magnetic fields.

- Few 100's cpus on Cray XT4 at NERSC



Plasma temperature in cross section of torus

Magnetic field lines plot traced in +ϕ-direction

$t=66.9\tau_A$ 89.3 133.4 178.3

L. Sugiyama 4/23/09
DoE Fusion SciDAC

Numerics

- Time advance solved by partially implicit method (each variable)
  - Fourier in toroidal direction
  - Solve in each 2D poloidal plane by finite difference
    $\partial A/\partial t = (A - A^{old})/\Delta t = RHS + \nabla \cdot \eta \nabla A$
    $$A = A^{explicit} + \Delta t\, (\nabla \cdot \eta \nabla A)^*$$
    $$A^{explicit} = A^{old} + RHS + \text{explicit part of } \nabla^2$$
- 2D finite elements (volumes)
  - Unstructured grid, packed
- Fortran and C
  - PETSc MPI library
  - Fortran code contains physics; can use with OpenMP operators
- Solvers optimized for 'nice' problem – MPP scaling to few 1000 cpus

Initial conditions, 'steady state,' input/output

- Ideal MHD equilibrium (fast time scale) for toroidal plasma
  - Well-known solution: Grad-Shafranov equation for magnetic flux ψ, given profiles of pressure $p(\psi)$ and toroidal field or current
  - Single/few processor equilibrium calculations (Open MP); MPP time evolution also needs spatial load balancing
  - Combine different solutions: Experimental reconstruction + code-smoothed
- Plasma rotation or non-MHD effects – new equilibrium needed (solution may be unknown)
- Non-axisymmetric perturbations: no good equilibria
  - Experimental analysis ignores non-axisymmetry!
  - Can only measure magnetic field in vacuum – need code for plasma
- Diagnostics, coupling to other large codes doing different physics often goes through the 2D equilibrium; need to recalculate regularly
- Output, manipulate, visualize 3D info

Chaos and turbulence

- Hamiltonian chaos in B leads to intrinsically stochastic plasma. How to handle computationally?
- Toroidal magnetic field is a 2 degree of freedom Hamiltonian system.
  - In plasma interior, mostly nice, nested flux surfaces
  - At edge of fusion plasma, surfaces have a 'X-point' and split into two asymptotic limits when perturbed. Surfaces intersect $\longrightarrow$chaos!
  - Hamiltonian system has infinities (plasma can't, but tries)
- Existing code set up for 'nice' interior case
  - Grids, control of spatial resolution need improvement
- Consistent physics: add $\delta T = T_{\parallel} - T_{\perp}$, relative size few $\times 10^{-4}$
- Preserve global connections: magnetic field has both local and global properties – accurate $\nabla \cdot B=0$ and bc's

Summary

- MPP versus few-processor calculations
  - Input and output use few processors compared to main calculation
  - Extract information regularly from run to (1) monitor (2) store time-slice and time-dependent info
- How to handle full 3D info – input, output, analysis
- Visualization – interactive: post-processing marginal now, need MPP
- Scale to larger runs
  - Large scale runs change character – smooth to turbulence
  - More physics: MHD $T_{\parallel}$ and $T_{\perp}$ instead of scalar T, couple to other large scale codes (eg, particle)
  - Efficiency – Courant time step limit at small grid spacing
- How to debug MPP? No more print statements!