FACETS approach to Multiscale

Presenter: J.R. Cary[†*]
[†]Tech-X Corporation; [*]University of Colorado

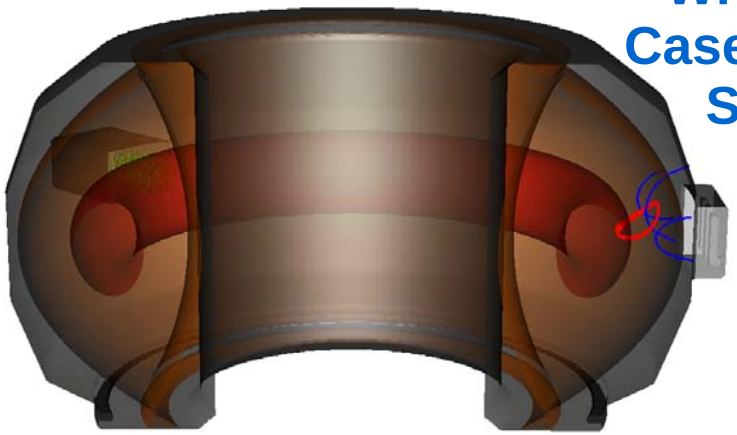
presented at

US-Japan Workshop on Multiscale

Click to edit Master subtitle style

November 21, 2008

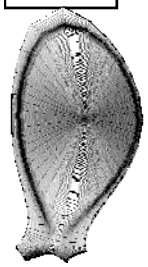
Project profile
Why core-edge coupling?
Why embedded couplings?
Case study in code integration
Summary of capabilities



IU
NYU
Lodestar

FACETS is a highly collaborative project with many contributors

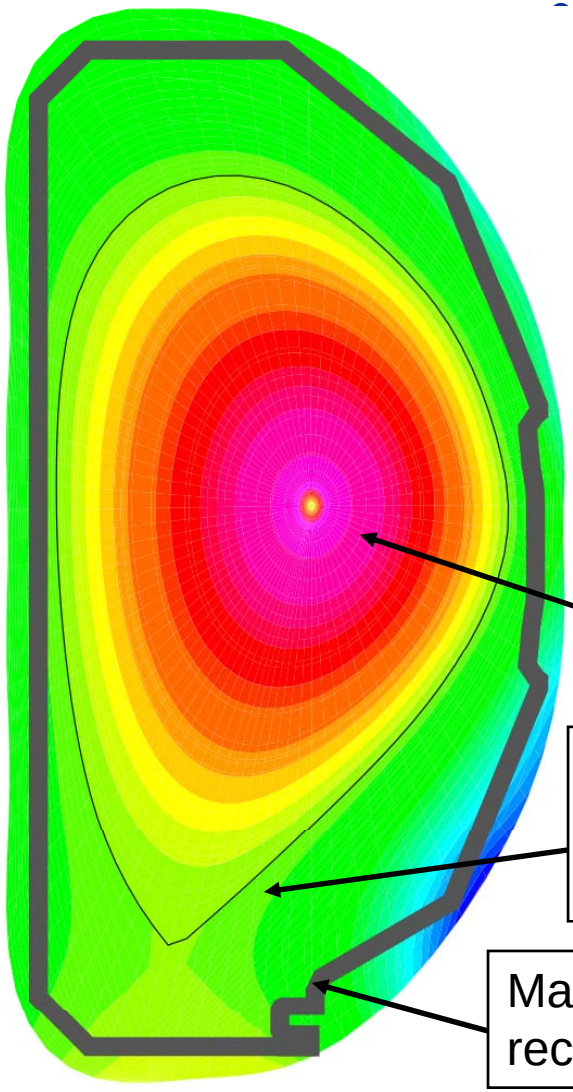
IU
NYU
Lodestar



- ANL (solvers): McInnes, Zhang, Balay
- ANL (coupling): Larson
- CSU (sensitivity research): Estep
- General Atomics (GYRO, turbulence understanding, long-range and/or refinement): Candy
- Lehigh (core modeling, SBIR subcontract): Pankin
- LLNL (edge physics): Rognlien, Cohen
- LLNL (interlanguage): Epperly
- ORNL (modeling, evangelism): Cobb
- ORNL (SSGKT): Fahey
- ParaTools (performance analysis): Maloney, Morris, Shende
- PPPL (core sources): McCune, Indireshkumar
- UCSD (wall): Pigarov
- Tech-X (framework, core): Cary, Hakim, Kruger, Pletzer, Vadlamani, Miah, Shasharina, Wang, Muzsala

FACETS goal: tight coupling framework for core-edge-wall

- Coupling on short time scales
- Inter-processor and in-memory communication
- Implicit coupling

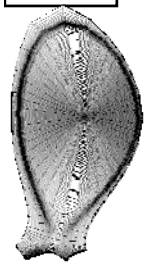


Hot central plasma: nearly completely ionized, magnetic lines lie on flux surfaces, 3D turbulence embedded in **1D** transport

Cooler edge plasma: atomic physics important, magnetic lines terminate on material surfaces, 3D turbulence embedded in **2D** transport

Material walls, embedded hydrogenic species, recycling

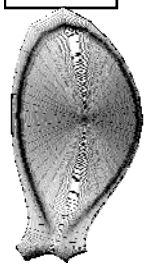
IU
NYU
Lodestar



Why do core-edge-wall coupling?

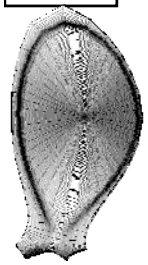
- Pedestal height strongly impacts fusion reactor performance
- Determine pedestal height for density and temperature profiles at core-edge interface self consistently
- L-H transition, ELMs, and transport barrier are dynamical processes requiring core-edge coupling for quantitative predictions (good individual computations still needed)
- Neutrals released by wall can penetrate core
- Coupled simulations vs monolithic codes exploits space and time scale disparities and makes use of proven techniques for incorporating important physics in each region

IU
NYU
Lodestar



FACETS: Embedded flux calculations for cross-surface transport

IU
NYU
Lodestar

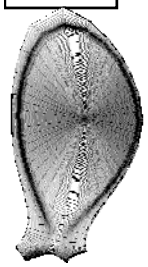


- **Small radial correlation lengths (at least in presence of flow shear)**
- **Time scale separation**
- **Fluxes depend on local quantities**
  - Densities and gradients
  - Temperatures and gradients
- **Multiple approaches to flux computations**
  - Reduced models (GLF23, TGLF, …)
  - Embedded turbulence computations (GYRO)

Why do reduced-model transport simulation?

- Would like to get to 1000 simulation second in one wall clock hour
- GYRO simulation: 1 ms on D3D takes 1 hour on 128 procs for well resolved.
- Off by 1e6 at the present time. Makes this a long-range research area (more parallelism, …)
- Can be used for refinement (see next talk on SSGKT), study of flattop

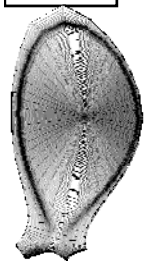
IU
NYU
Lodestar



FACETS: Sources may come from collocated modules

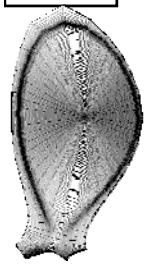
- **Neutral beams**
  - injected from wall
  - Slow down to become part of plasma
  - Source of particles, energy, momentum
- **Other sources**
  - RF
    - ICRF
    - LH
    - ECH
  - Neutrals from wall

IU
NYU
Lodestar



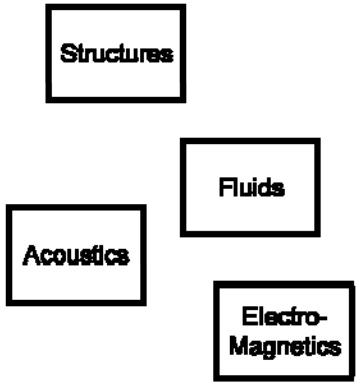
FACETS is moving towards the "deep interoperability" framework

IU
NYU
Lodestar



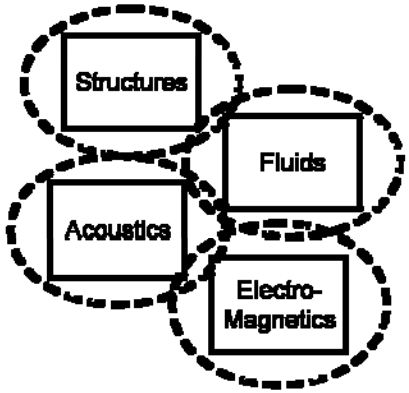
EVOLUTION OF APPLICATION PHYSICS SOFTWARE

Minimal Component Interoperability:



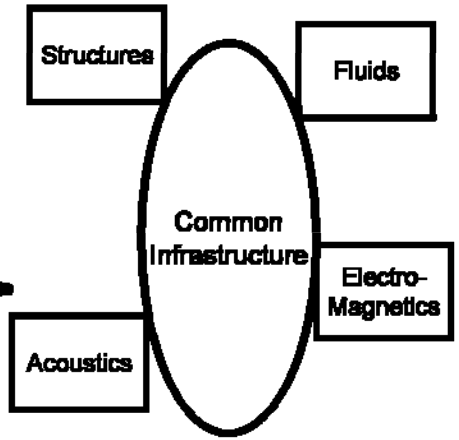
- Physics models are completely uncoupled.
- May exchange static datasets through flat files.

Shallow Component Interoperability:



- Physics models are loosely coupled.
- Data management and parallelism is independent in each module.
- Exchange common data events via wrappers (web services, etc.).

Deep Component Interoperability:



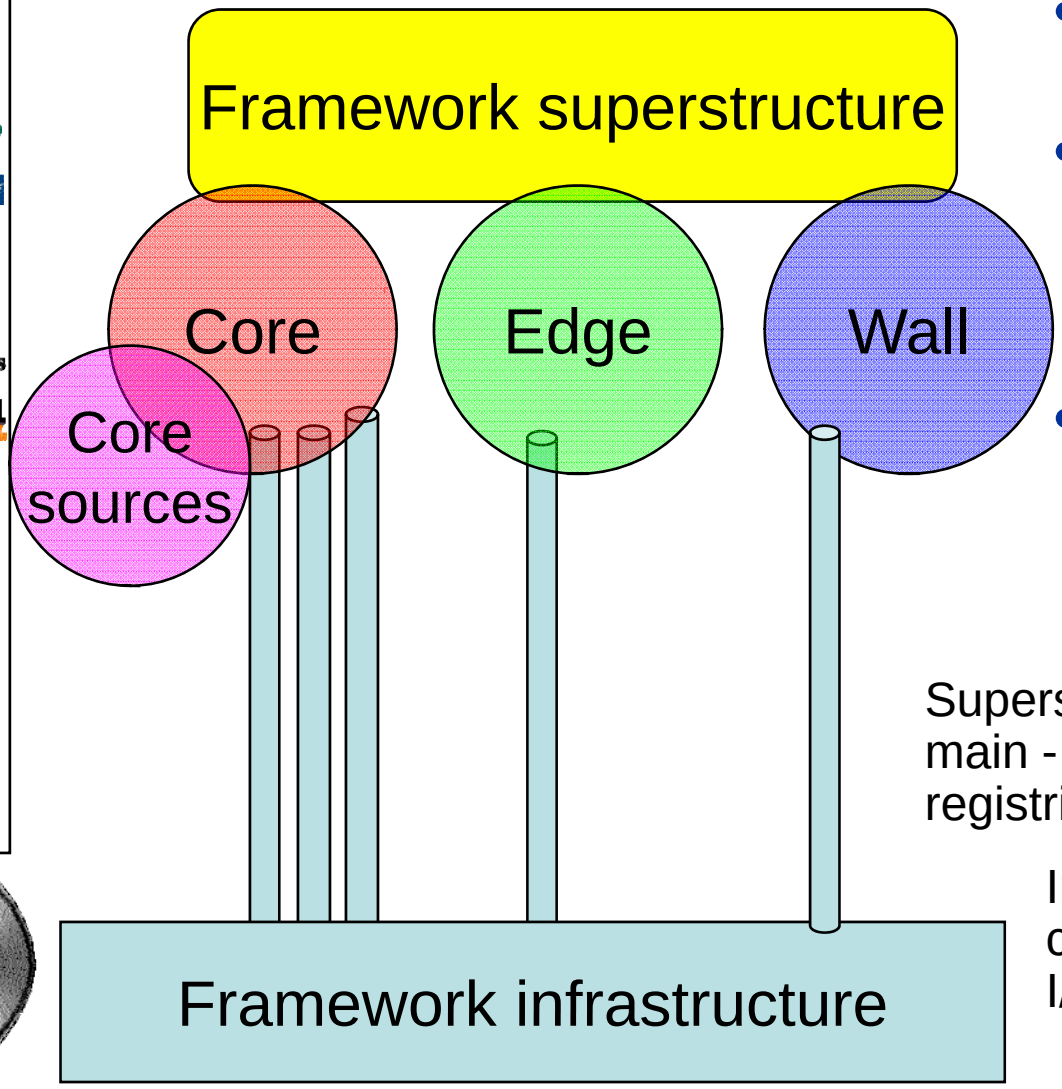
- Physics models are tightly coupled.
- Data exchange across shared service infrastructure.

FACETS

Time

from Shalf, ASCAC Boulder, Apr. 30, 2008

Reuse of legacy code requires a combination of shallow and deep

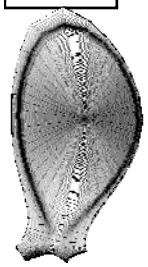


- **Simple, common interfaces**
- **Native components: extended interface, built on FACETS infrastructure**
- **Non native components: basic interfaces, importation of pre-written codes**

Superstructure: for pulling stuff into main - implementation and object registries, wrappers, …

Infrastructure: for building new components - grids, messengers, I/O, containers, …

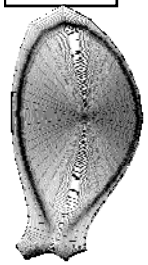
IU
NYU
Lodestar

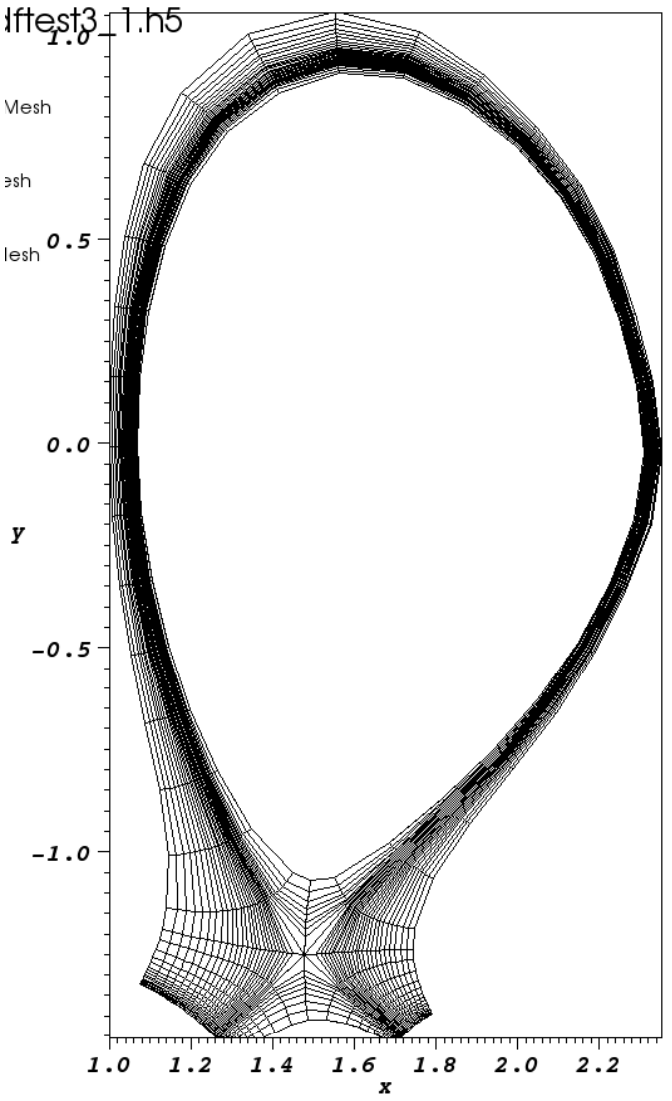


Case study for bringing in Edge component: UEDGE

- UEDGE is a fluid model for the plasma edge
- 2D, multiblock grid
- "Steerable"
  - BASIS
  - Python
- More used for analysis mode
  - Back out diffusivities
- Was parallel once

IU
NYU
Lodestar





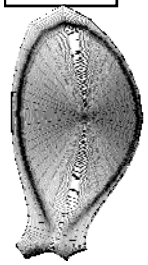
Bringing in legacy code required significant work

- Leadership class facilities (BG/P, XT4) do not allow shared libraries: reimplement scripted code in compiled
- No build system
- No rigorous testing system
- Interlanguage issues

- Took two years, roughly $1.5M

- (Ultimately) matching issues

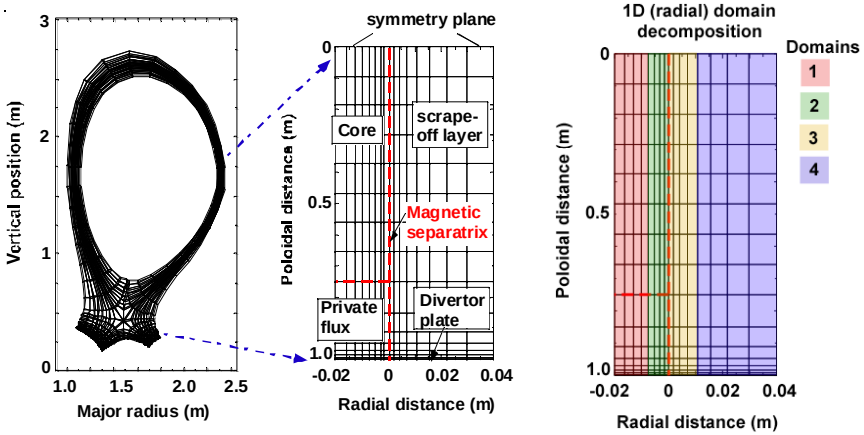
IU
NYU
Lodestar

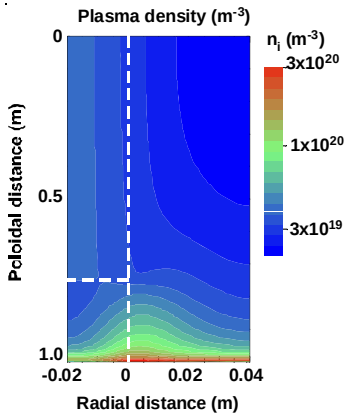


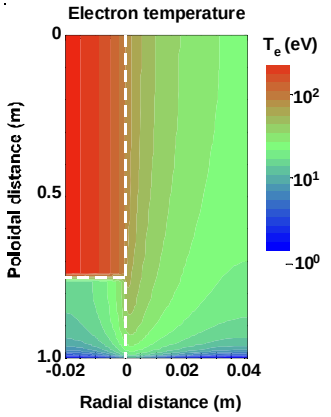
UEDGE Test Case defined for testing parallelism

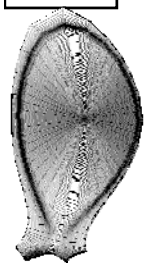
Full toroidal geometry is typically used, but initial parallel UEDGE tests with PETSc employ a simplified half-space slab

- **Outer midplane/ divertor regions are mapped to a slab with B_p/B_t = constant**
- **Same features such as closed and open B-field lines, private flux region, and divertor recycling are retained**
- **Decomposing radially is simplest because there are no internal special surfaces in that direction; 2D decomposition coding revived and being debugged**
- **Typical solutions show structure of plasma variables in edge region**
  - Ions recycle as neutral gas at the divertor (lower surface) to increase plasma density there
  - Electron heat loss at the divertor and ionization losses decreases temperature there









IU
NYU
Lodestar

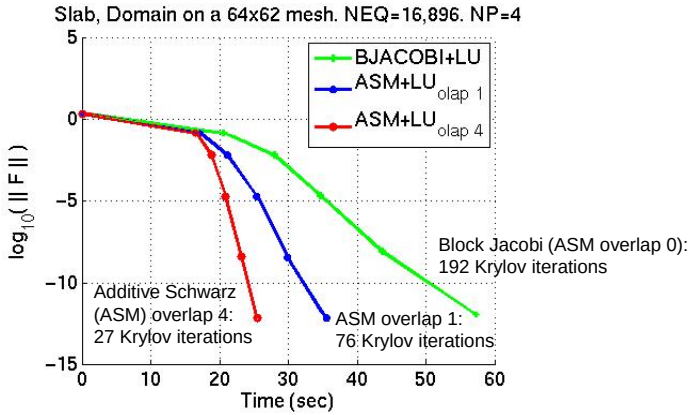
Office of Science SciDAC

UEDGE: achieved modest parallelism

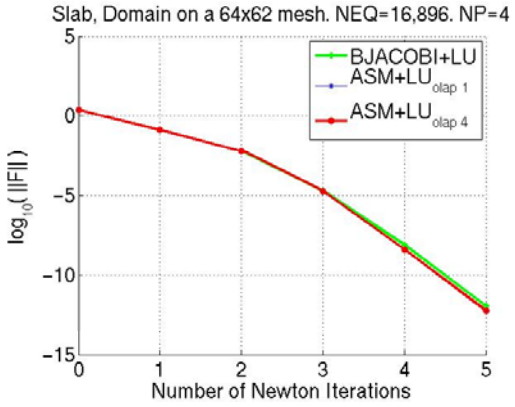
IU
NYU
Lodestar

Algorithmic Comparisons: Matrix-free Newton using restarted GMRES with various preconditioners

Function Norm vs. Time



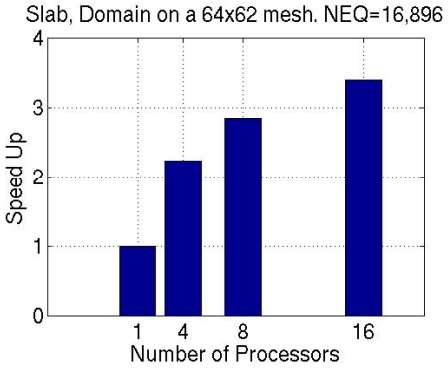
Function Norm vs. Nonlinear Iteration



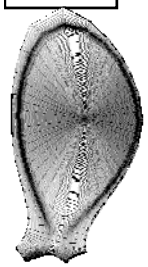
Computational Environment: Jazz @ ANL: Myrinet 2000 network, 2.4 GHz Pentium Xeon processors with 1-2 GB of RAM.

(64+2)x(62+2) mesh with 4 unknowns per mesh point: problem dimension 16,896

Initial Scalability (Total Simulation Time)



Work in progress: Exploring algorithmic options and scalability issues for various test cases

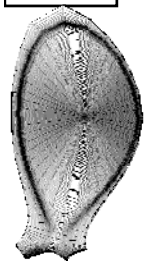


Interlanguage issue solved with Babel

- **Babel: interlanguage tool (Fortran, C, C++, Python)**
- **Interfaces defined by**

  https://www.facetsproject.org/facets/wiki/InterfacesAndNamingScheme
- **SIDL written by Epperly (LLNL)**
- **Corresponding python interface written by R. Cohen used for implementation**
- **Problems:**
  - Python generally not available on leadership-class machines (without A LOT of work)
  - Slower
- **Solution: build Babel from .v files to have direct calls to Fortran**

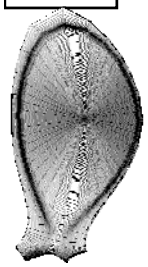
IU
NYU
Lodestar



Case study: core solver

- **No parallel core solver existed**
- **Potential speedups great**
  - Flux computations intensive, so can expect 1 proc per radial point
  - With embedded turbulence computations even more
- **New look allowed**
  - New solver

IU
NYU
Lodestar



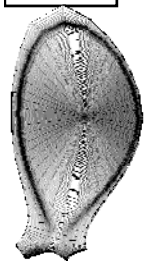
New fast, robust, and parallel core solver developed

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial r} = S(r,t)$$

Fluxes strongly nonlinear function of values, gradients

- **Taking large time steps is a well known challenge – convergence failure**
- **Observation: core solves converge well at lower resolution**
- **Implemented "nested iteration" (multigrid) to extend convergence radius**
  - Find solution at coarser resolution
  - Interpolate to get guess for finer resolution
- **Result: able to increase time step by 500x (ASTRA: 2.e-5s; FACETS >10ms) 7x perf.**
- **Next step: combine with block Broyden**

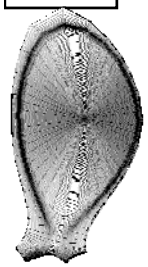
IU
NYU
Lodestar

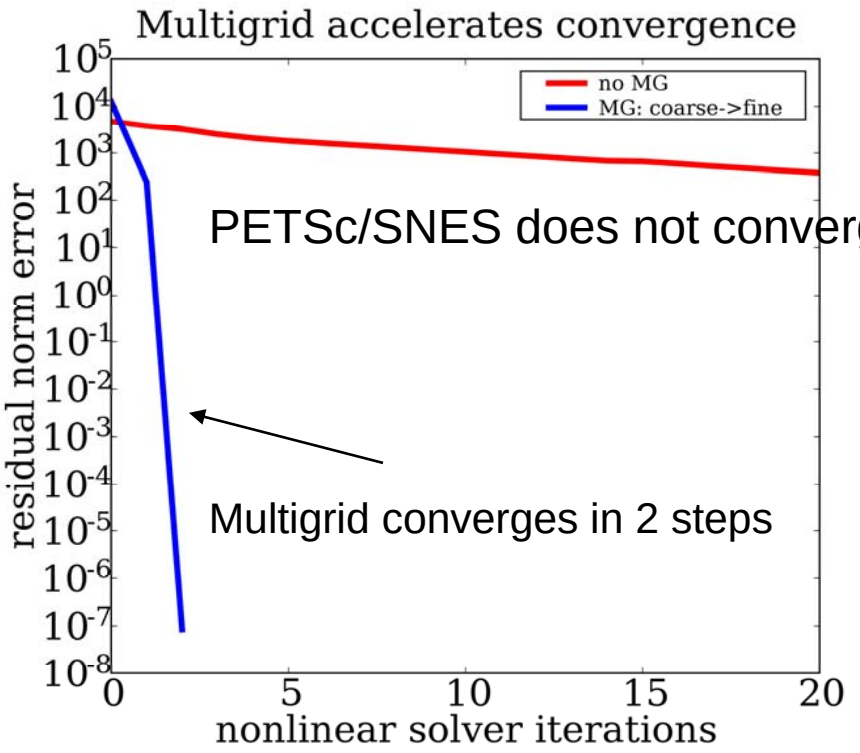


Core: multigrid & flexibility in time stepping

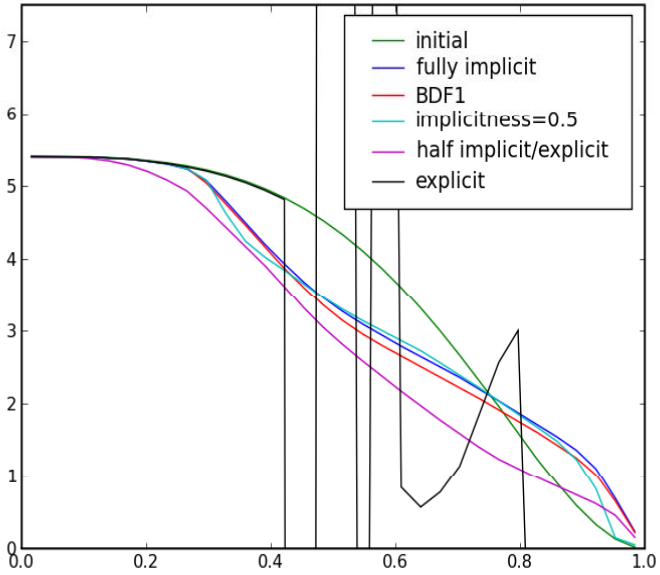
- **Effect of multigrid on stiff transport model equation**
- **Any diagonally implicit time steppers can assembled at run time with no change of source code required**

IU
NYU
Lodestar





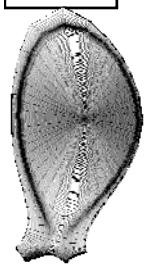
- Explicit is unstable
- half-implicit exhibits oscillations
- Backwards Euler & higher order
- schemes give similar results

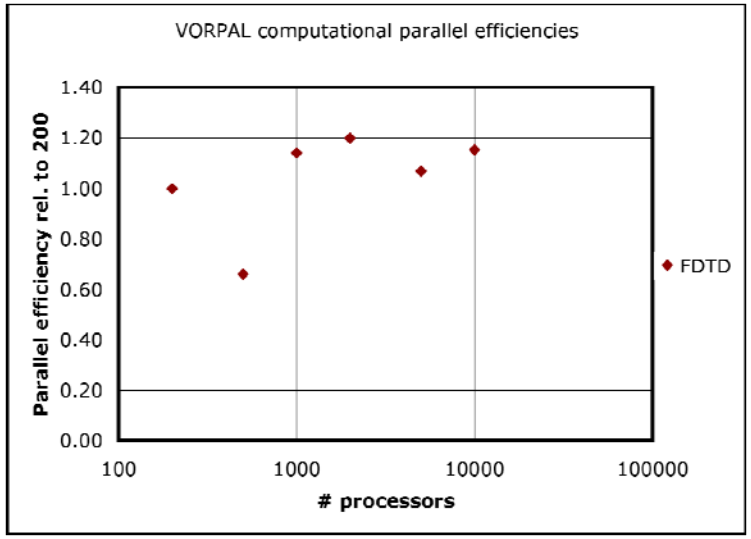


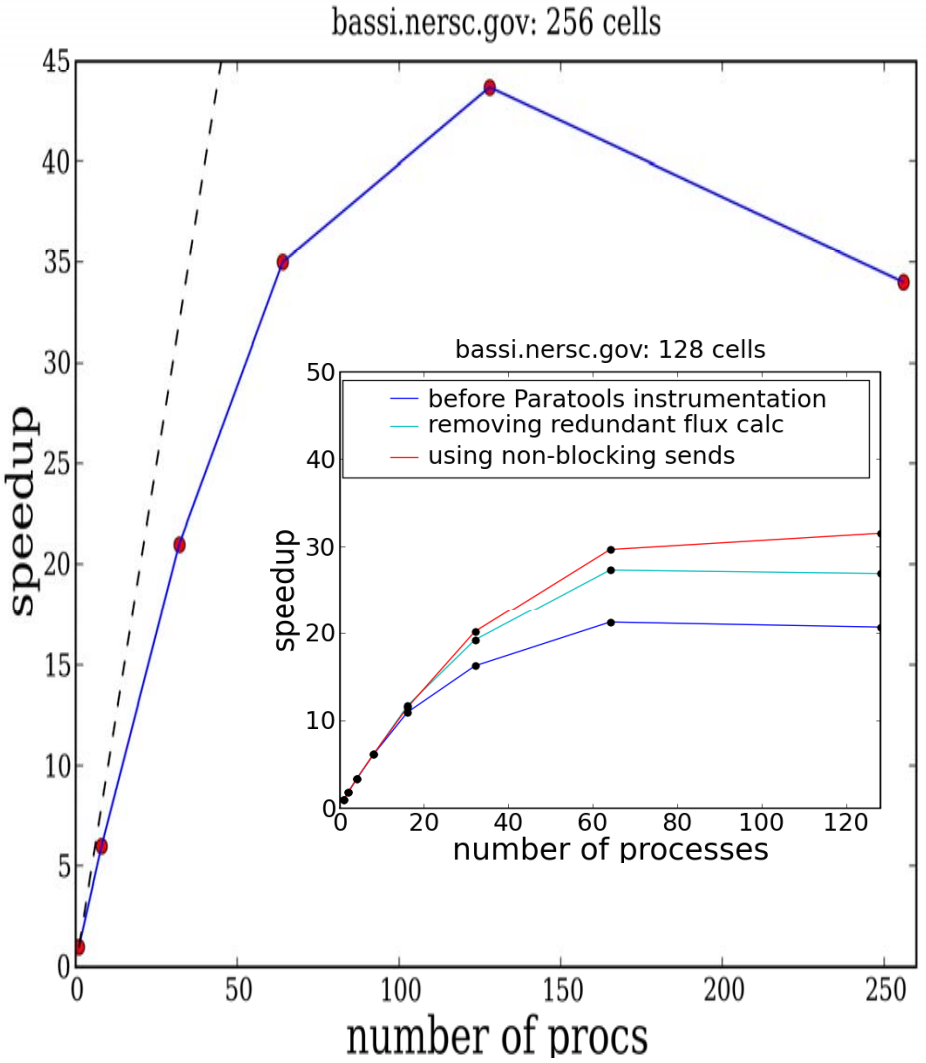
Parallel core solver scaled to 10s of procs, then improved

- Distributed flux calculations, matrix solves on rank 0
- Bassi: strong scaling, 128 cells
- Expected 1 cell per proc, since GLF23 so compute intensive
- Initially found 20x
- Improved to 30x (45x on 256 proc)

IU
NYU
Lodestar



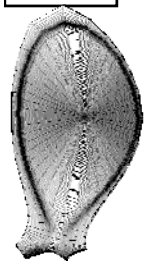




How parallel could core be?

- **Parallel GLF23 with load balancing should be able to effectively use 32*8 + 32*2 + 64 = 384 procs for 128 cells.**
- **TGLF should parallelize even better – 128 cells*20 procs/cell = 2500 procs, or 1200 for load balancing problems**
- **Embedded GYRO (cf SSGKT):**
  - 128-512 procs/GYRO
  - 8-32 radial values
  - 4 instances in the ensemble
    - 60000 procs
  - Possibly another 7x for Jacobian

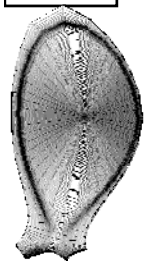
IU
NYU
Lodestar



Core: Verification efforts underway: comparison with ASTRA

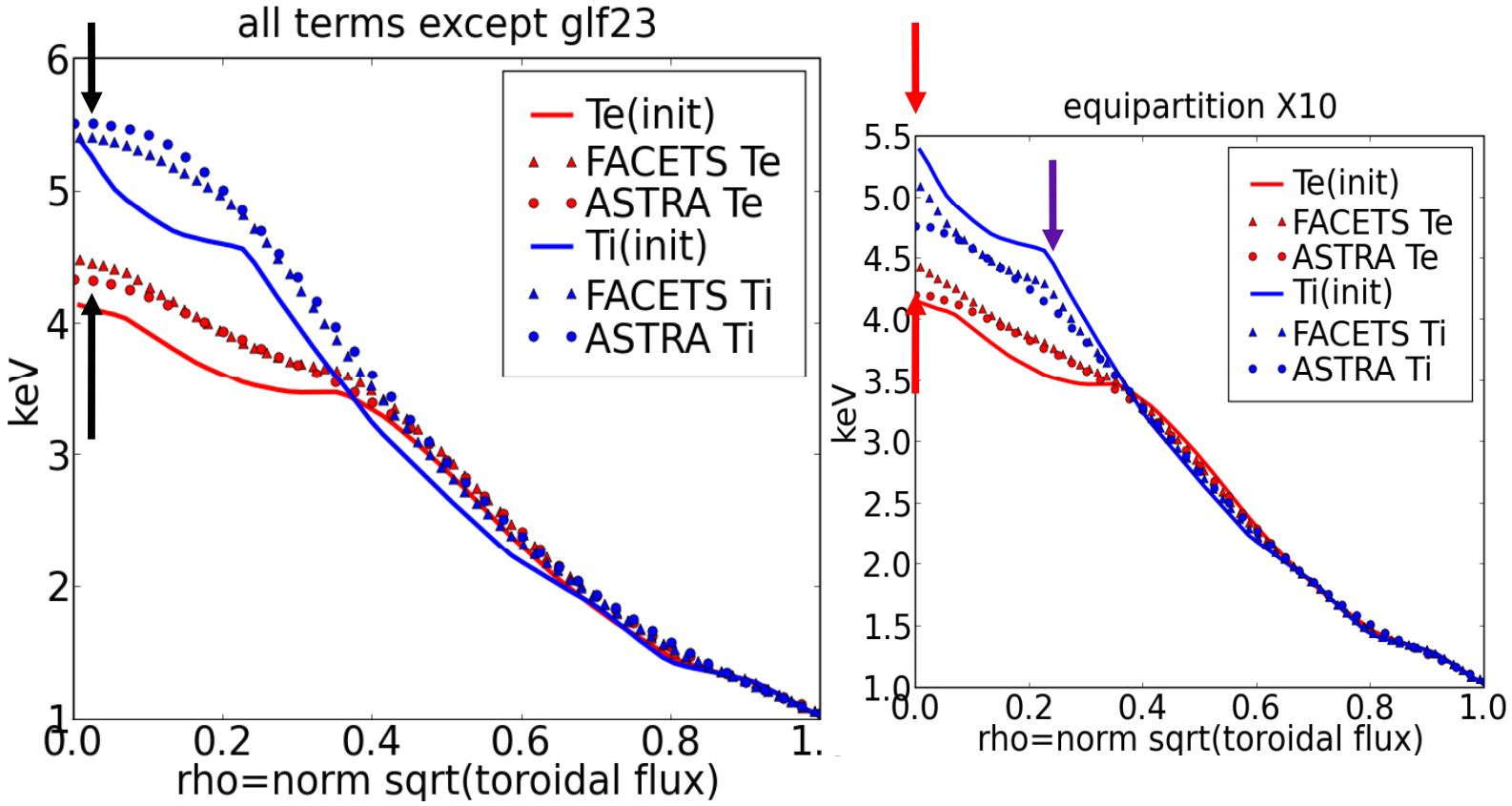
- **Systematic approach (Pletzer/Pankin/Kruger)**
  - Case 2: GLF23 only, no neoclassical, no heating
  - Case 3: Neoclassical only, no beams
  - Case 4: Beams only
  - Case 5: Beams and radiation
  - Case 6: Equipartion terms are on
  - Case 7: Ohmic heating is on
  - Case 8: Beams, radiation, neoclassical, and equipartion
  - Case 9: Case 8 + GLF23
  - Case 10: Equipartion terms times 10
  - Case 12: GLF23 only (case 2) without smoothing, unbounded chi
  - Case 13: The same as case 12, but only one small time step (dt=1.e-10)
  - Case 14: Case 9 without smoothing (10 msec)
  - Case 15: Case 9 without smoothing (1 msec)
- **Required additional implementations**
  - Equipartition
  - Neoclassical diffusivity (Chang-Hinton, 1986)
  - Implemented UFILE reader for
    - Initial profiles
    - Boundary conditions
    - Input powers
  - Metric coefficients from EFIT/ESC
- **Multiple issues resolved**
  - ASTRA smoothing
  - Flux coord normalization
  - Definition of minor radius [midplane versus sqrt(tor flux/ πB_T)]

IU
NYU
Lodestar

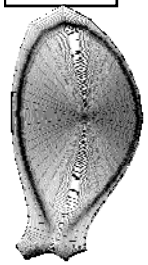


Core: Verification without GLF23 compares well (less well on axis)

- 2% disagreement for non-GLF23 near axis
- ASTRA appears to lack energy conservation near the axis
  - Sum not conserved
  - Shape changes



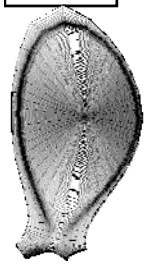
IU
NYU
Lodestar



Next step: embedded turbulence calculations

- For flattop, need only find steady state
- Acceptable to have 1 simulation hour to find.
- SSGKT (embedded SAPP) developing a steady-state, embedded turbulence transport calculation
- Issues to face
  - Turbulent noise in solver
  - Dynamic load balancing

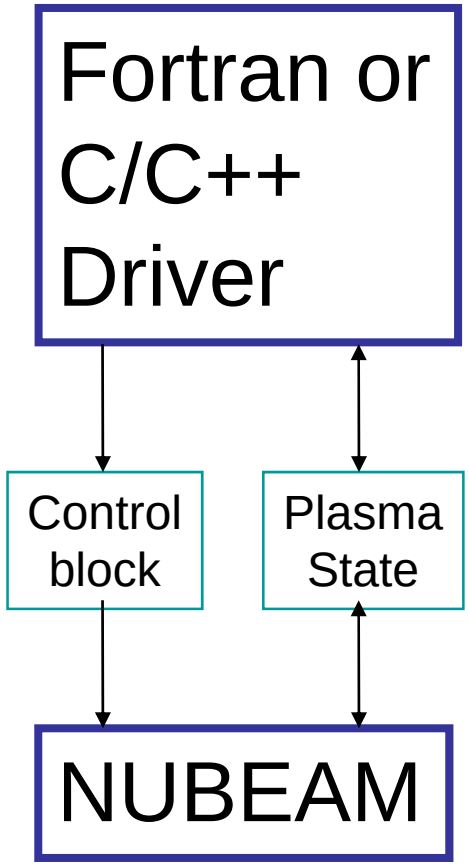
IU
NYU
Lodestar



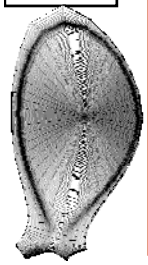
For collocated physics trying single-point communication

NUBEAM Driver Compatible with C/C++ (McCune, Indireshkumar, Pletzer)

- Fortran driver OK in TRANSP/PTRANSP.
- Control Block:
  - One for initialization
  - One for step advance
  - NUBEAM-specific numerical controls
  - Sequenced, static elements, directly instantiatable in C/C++
- Plasma State physics component interface:
  - Neutral beam geometry
  - Beam powers and voltages
  - Target plasma MHD equilibrium
  - Target plasma temperatures and densities
  - Fast ion density, heating and current drive profiles returned
  - C/C++ python-generated set/get interface using “opaque handle” (see previous slide) method for structure with dynamic elements (Tech-X & PPPL collab.)



IU
NYU
Lodestar



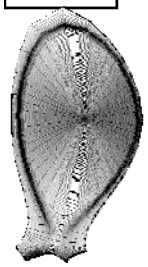
Initial core-edge coupled results obtained for DIII-D shot 118898.03400

IU
NYU
Lodestar

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

core-edge coupling point

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.

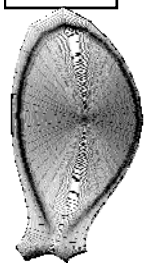


Ion temperature lineout and contours at t=0.01 seconds. A time-step of 10 microseconds was used.

UEDGE integration provides lessons in componentization

- Work towards detailed interface specifications
- Resolve philosophical differences (full inspection versus encapsulation)
- Removal of poorly maintained 3rd party tools can be time consuming
- Need automatic test suite for reliable modification
- Working towards uniform methodology will speed process (build, I/O, e.g. HDF5)
- Education of software providers on engineering aspects

IU
NYU
Lodestar



Current directions for coming year

- Framework: robust integration of wall
- Core sources: wrappers, parallelization, data access, …
- Edge: robust build as a component, new solvers, parallelization
- Wall: interface, solvers
- Coupling: implicit coupling, coupling analysis for stability and accuracy.
- Initial investigations of how to incorporate edge turbulence codes has begun

IU
NYU
Lodestar