Progress in Parallel Implicit Methods For Tokamak Edge Plasma Modeling

Michael McCourt[1,2],Lois Curfman McInnes[1]
Hong Zhang[1],Ben Dudson[3],Sean Farley[1,4]
Tom Rognlien[5], Maxim Umansky[5]

Argonne National Laboratory[1]
Cornell University[2]
University of York[3]
Illinois Institute of Technology[4]
Lawrence Livermore National Laboratory[5]

November 8, 2010

Work supported by the FACETS SciDAC project

FACETS: Framework Application for Core-Edge Transport Simulations

FACETS Overview

- PI: John Cary, `https://www.facetsproject.org`
- Coupling all three tokamak regions during simulation
- Collaboration with SciDAC TOPS Center on scalable solvers

FACETS

Office of Science
U.S. DEPARTMENT OF ENERGY

SciDAC
Scientific Discovery through Advanced Computing

TOPS
Towards Optimal Petascale Simulations
PI: David Keyes, Columbia Univ.
www.scidac.gov/math/TOPS.html

Overview

Challenges in edge-plasma simulations

- Strong nonlinearities and multiple time and spatial scales yield a poorly conditioned problem

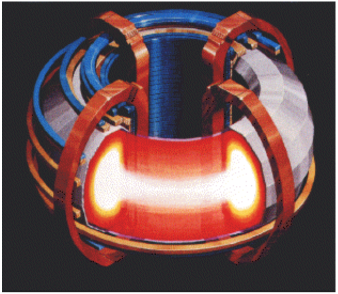
Our approach to edge region simulations

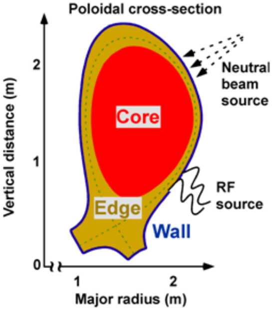
- Implicit time-stepping to handle stiffness
- Jacobian-free Newton-Krylov methods to solve the resulting nonlinear systems
- Precondition the Krylov solver using an approximate Jacobian

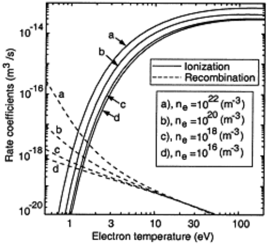
Goal

Improve the scalability of the solver by exploiting our understanding of the physics

Complexities Within The Edge Region







- Edge-plasma region is key for integrated modeling of fusion devices
- Edge-pedestal temperature has a large impact on fusion gain
- Plasma exhaust can damage walls
- Impurities from wall can dilute core fuel and radiate substantial energy
- Tritium retrieval key for safety

Implicit Solvers

When multiple time scales exist, an implicit discretization is often preferred. This requires solving a nonlinear system of the form

$$G(u_{k+1}) \equiv u_{k+1} - u_k - \Delta t F(u_{k+1}) = 0$$

at each time step k. These nonlinear systems can be solved with the Newton method, which in turn requires multiple linear solves (index n):

$$\begin{aligned} J(G)\left(u_{k+1}^n\right) \Delta u_{k+1}^n &= -G\left(u_{k+1}^n\right) \\ u_{k+1}^{n+1} &= u_{k+1}^n + \Delta u_{k+1}^n \end{aligned}$$

This is where the significant cost of implicit time-stepping arises.

Approximating matrix-vector products needed by Krylov Methods

Iterative linear solvers require matrix multiplication at each iteration. Finite difference Jacobian-vector products:

$$J(G)(u)v \approx \frac{1}{h}\left(G(u+hv) - G(u)\right)$$

Matrix-Free Newton-Krylov using PETSc

Portable, Extensible Toolkit for Scientific computing

- `http://www.mcs.anl.gov/petsc`

PETSc provides ...

- Efficient solvers for large, sparse nonlinear systems
- A fast, parallel algorithm for computing the Jacobian
- Access to external software for preconditioning options

Accelerating Newton-Krylov by preconditioning with the Jacobian

- Direct Solver: Invert Jacobian using MUMPS
  - Closest to true inverse; Limited scalability
- Additive Schwarz: ASM
  - Domain decomposition; Overlap is possible
- Algebraic Multigrid: AMG using hypre
  - Multilevel divide-and-conquer algorithm; Ideal for elliptic problems

Progress in UEDGE

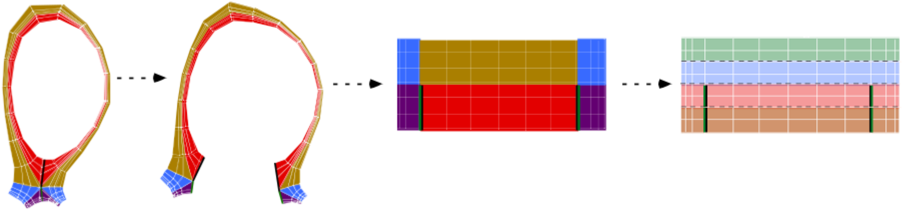
Key Aspects of UEDGE

- Developed at LLNL by Tom Rognlien et al.
- 2D fluid equations for plasma/neutrals
- Finite volumes in a non-orthogonal mesh
- Volumetric ionization, recombination, radiation loss
- Nonlinear solves via matrix-free Newton-Krylov

Challenges of UEDGE simulations

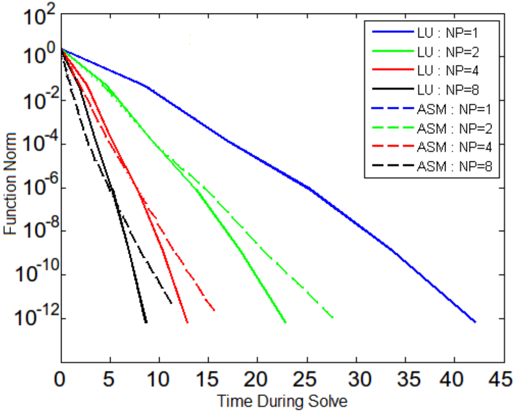
- Strong nonlinearities
- Large range of spatial and temporal scales
- Anisostropic plasma transport ($D_\perp/D_\| \sim (\rho/\lambda)^2$) coupled with isotropic neutral transport
- Scalable preconditioning

Domain Decomposition for Parallel UEDGE



- To store data across processors scalably, the domain needs to be decomposed so that each processor holds a disjoint partition.
- Then, the domain is transformed into a rectangle, which is then cut up into NP equally sized smaller rectangles.
- Experiments without neutral gas terms active have shown the anisotropy in non-neutral terms prefer a 1D decomposition.

Preliminary Scalability Experiments in UEDGE



Scalability

LU: 1/2 - 1.8
1/4 - 3.24
1/8 - 4.86

ASM: 1/2 - 1.5
1/4 - 2.7
1/8 - 3.78

Linear Iterations

LU: 2

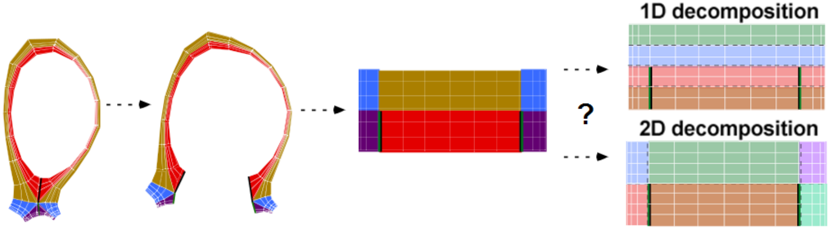
ASM - 1: 2
ASM - 2: 33
ASM - 4: 53
ASM - 8: 100

Test Case Specifications

- $\Delta t = 10^{-4}$, 128x64 mesh, in DIII-D single-null tokamak
- 5 Variables: H^{+}/e temperature, H^{+} density, H^{+} velocity, H density

Analyzing the Physics

Using ASM rather than LU to precondition is superior only when coupling between domains is minimized.

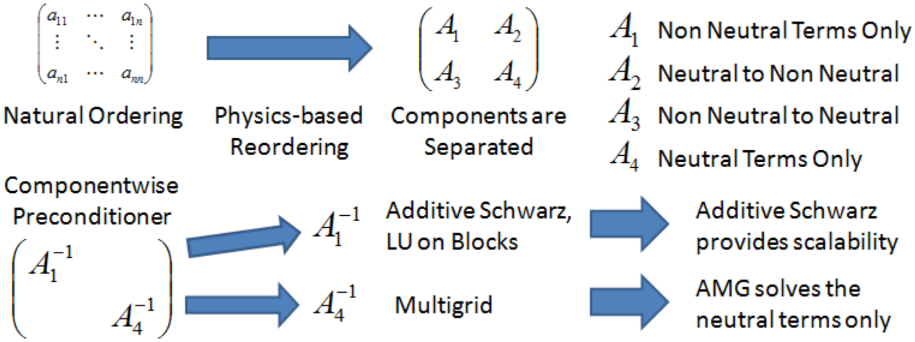


Neutral/Non-neutral contradictions

- Non-neutral terms have a strong anisotropy: 1D is preferred.
- The neutral terms would need to couple across 1D domains.

How can we reconcile these differences?

An Operator-Specific Preconditioner



$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}$$

Natural Ordering

Physics-based Reordering

$$\begin{pmatrix} A_1 & A_2 \\ A_3 & A_4 \end{pmatrix}$$

Components are Separated

A_1	Non Neutral Terms Only
A_2	Neutral to Non Neutral
A_3	Non Neutral to Neutral
A_4	Neutral Terms Only

Componentwise Preconditioner

$$\begin{pmatrix} A_1^{-1} & \\ & A_4^{-1} \end{pmatrix}$$

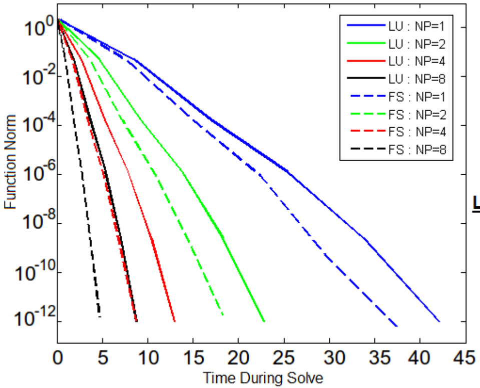
A_1^{-1} Additive Schwarz, LU on Blocks → Additive Schwarz provides scalability

A_4^{-1} Multigrid → AMG solves the neutral terms only

`PCFieldSplit`

- Synergy between physicists and PETSc developers has led to a componentwise preconditioning scheme.
- `PCFieldSplit` allows different preconditioners for different variables **without** code modification.

Improving Scalability in UEDGE



Scalability

LU: 1/2 - 1.8
1/4 - 3.24
1/8 - 4.86

FS: 1/2 - 2.0
1/4 - 4.2
1/8 - 7.98

ASM: 1/2 - 1.5
1/4 - 2.7
1/8 - 3.78

Linear Iterations

LU: 2

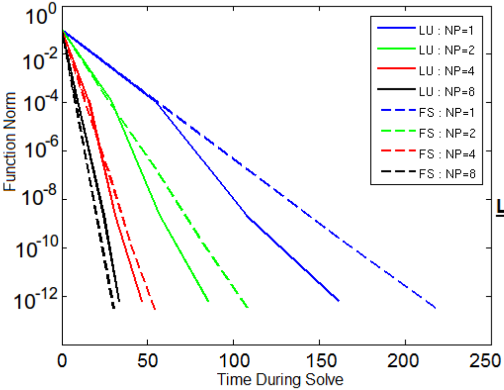
FS - 1: 5.8
FS - 2: 7.4
FS - 4: 8.2
FS - 8: 9.8

ASM - 1: 2
ASM - 2: 33
ASM - 4: 53
ASM - 8: 100

A Operator-Specific Preconditioner

- Neutral terms solved by AMG
- Non-neutral terms solved by ASM with LU on the blocks
- Coupling terms disregarded during preconditioning

Moving From 5 Variables to 16 Variables by Introducing Neon



Scalability

LU: 1/2 - 1.9
1/4 - 3.42
1/8 - 4.79

FS: 1/2 - 2.0
1/4 - 4.0
1/8 - 7.2

Linear Iterations

LU: 3

FS - 1: 14.25
FS - 2: 16.0
FS - 4: 16.0
FS - 8: 18.5

Test Case Specifications

- $\Delta t = 10^{-4}$, 64x32 mesh, in DIII-D single-null tokamak
- 16 total variables: $H^{+}/Ne^{+1}/.../Ne^{+10}$ ion densities, H^{+} velocity, H^{+}/e temperature, H/Ne neutral densities

Time-Dependent 3D Edge-Plasma Turbulence Via Implicit Stepping in BOUT++

Key Aspects of BOUT++

- BOUT originally developed at LLNL by Xueqiao Xu, Maxim Umansky
- BOUT++ is a C++ reformulation of BOUT: Ben Dudson (Univ of York), Umansky, Xu, and Sean Farley (IIT,ANL)
- Same general fluid equations as 2D UEDGE yield turbulence in 3D that drives the dominant radial plasma transport
- Tokamak geometry with separatrix, finite differences, 2D partitioning

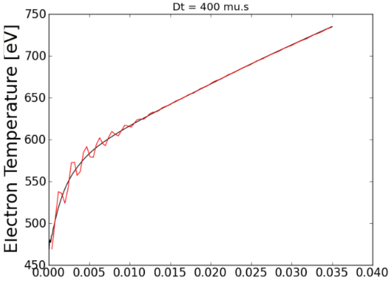
Progress in BOUT++

- Interface to implicit time-stepping via SUNDIALS (Carol Woodward, LLNL) implemented within PETSc
  - PETSc provides encapsulations for solvers and time stepping
- Extended design for flexibility and robustness
  - Enables runtime experimentation with preconditioners
  - Facilitates incorporation as a FACETS component

Core-Edge Coupling in FACETS

Initial experiments into core-edge coupling (A. Hakim et al) have involved an explicit time discretization between regions.

- Data is exchanged at the end of each step
    - Core to Edge: passes fluxes, Edge to Core: passes values



An initial discontinuity produces an instability for the large Δt needed in full discharge simulations.

- Research on implicit core-edge coupling is underway (J. Carlsson et al)

Summary

- `PCFieldSplit` shows great promise - exploiting knowledge of the component physics has improved scalability of the implicit solver within UEDGE.
- Implementing BOUT++ and UEDGE within PETSc has provided flexibility with solvers, preconditioners and time-stepping.

Future Work

- Experiment within UEDGE using new model decompositions, and retention of the inter-component coupling during preconditioning.
- Investigate alternative time-stepping and preconditioning schemes in BOUT++.
- Develop implicit simulations across the coupled core-edge domain.
  - Compare results to existing explicit simulations.
  - Study operator-specific preconditioning in the coupled system.
  - Study stability issues (joint with Don Estep et al. at CSU)

Appendix 1:Equations of Note

$$\frac{d}{dt} n_n + \nabla^T (n_n v_n) - K^i n_e n_n + K^r n_e n_i = 0$$