Advancing parabolic operators in thermodynamic MHD models: Explicit super time-stepping versus implicit schemes with Krylov solvers

Ronald M. Caplan, Zoran Mikic,
Jon A. Linker, and Roberto Lionello

Predictive Science Inc.

Slides available at:
www.predsci.com/~caplanr/astro16

Outline

- The problem
- Implicit schemes with Krylov solvers
- Explicit Super Time-stepping schemes
- The MAS thermodynamic MHD model
- Real-world test case and HPC setup
- Performance and scaling results
- Outlook

The Problem

- Thermodynamic MHD models have multiple time scales leading to vastly different explicit time-step stability requirements
- In order to make coronal simulations *tractable*, we need to exceed such explicit limits
- Focus on parabolic terms
  - Implicit methods (need iterative solvers)
  - Explicit methods with unconditional/extended stablility
  - Reformulation of the model (e.g. thermal waves)
- Here, we compare a *super time-stepping* method to an implicit method

$$\Delta t_{\text{flow}} \sim \Delta x / v$$
$$\Delta t_{\text{wave}} \sim \Delta x / v_f$$
$$v_f \gg v$$
$$\Delta t_{\text{para}} \sim \Delta x^2 / \alpha$$
$$\alpha \in \{\kappa, \eta, \nu\}$$

NOTE! When exceeding explicit time-step limits, one must be very careful about accuracy. Using too large of a time step can result in large errors!

Method Comparison

Implicit Scheme with Krylov Solver

Backward-Euler solved with
Preconditioned Conjugate Gradient (BE+PCG)

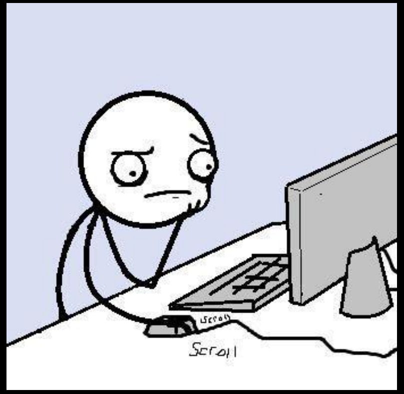
VS

Explicit Super Time-stepping Scheme

2^{nd}-order Runge-Kutta Legendre scheme (RKL2)
[Meyer et al, 2014]

Comparison Criteria

- Basic validation of accuracy and stability
- Ease/difficulty of formulation and implementation
- Features and limitations
- Overall performance
  - Number of iterations needed for each large time-step
  - Computational cost per iteration
- Parallel performance
  - Parallel communication needed per iteration
  - How well does the method scale?



Implicit scheme with Krylov solvers: Backward Euler (BE)

- Backward Euler (BE): simplest L-stable method

$$\frac{\partial u}{\partial t} = F(u, \mathbf{r}) \quad \Longrightarrow \quad \frac{u^{n+1} - u^{n}}{\Delta t} = \mathbf{M}\, u^{n+1}$$

- Applying BE to PDE yields a system of equations to solve

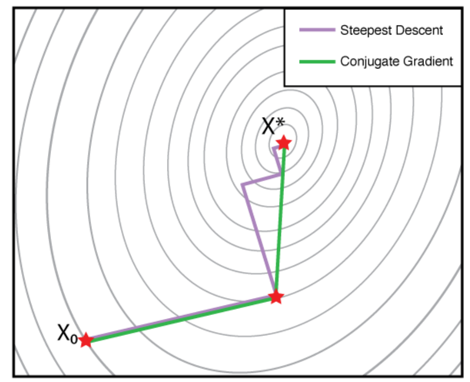
$$(1 - \Delta t\, \mathbf{M})\, u^{n+1} = u^{n} \quad \Longrightarrow \quad \mathbf{A}\, x = y$$

- To avoid the need for nonlinear solvers, we linearize any nonlinear terms with lagged diffusivity, e.g.

$$\nabla \cdot \left[\kappa(T^{n+1})\, \nabla\, T^{n+1}\right] \quad \Longrightarrow \quad \nabla \cdot \left[\kappa(T^{n})\, \nabla\, T^{n+1}\right]$$

Implicit scheme with Krylov solvers: PCG

- For operators yielding large sparse matrices, Krylov subspace methods are popular
- For linear symmetric (and nearly-symmetric) matrices: Conjugate Gradient
- Preconditioned Conjugate Gradient:
  Apply an approximate inverse of the matrix to "precondition" the problem so it will converge more quickly
- Need to balance cost of preconditioner with its reduction in iterations



Implicit scheme with Krylov solvers: Preconditioners

- We use two preconditioning options:
  - **PC1: Point-Jacobi / Diagonal scaling (DIAG)**
    Cheap, but not very effective…
    Communication free
  - **PC2: Non-overlapping domain decomposition with zero-fill incomplete LU factorization (NDD+ILU0)**
    Expensive, but much more effective!
    Communication free

Drawback: $N_{\mathrm{processors}} \rightarrow N_{\mathrm{grid}}$ ⇒ $\mathrm{PC2} \rightarrow \mathrm{PC1}$

Implicit scheme with Krylov solvers: PCG

PC1

$$\texttt{do j} = 1:\mathrm{N} \quad P_{jj} = A_{jj} \quad \texttt{enddo}$$

$$\texttt{do i} = 1:\mathrm{N} \quad z_i = P_{ii}\, r_i \quad \texttt{enddo}$$

PC2

```
LU = A
do i = 2 : N
    do k = 1 : i − 1          (LU_ik ≠ 0)
        LU_ik = LU_ik / LU_kk
        do j = k + 1 : N      (LU_ij ≠ 0)
            LU_ij = LU_ij − LU_ik LU_kj
        enddo
    enddo
enddo
P = LU
```

```
do i = 1 : N
    z*_i = r_i
    do j = 1 : i          (LU_ij ≠ 0)
        z*_i = z*_i − LU_ij z*_j
    enddo
enddo
do i = N : 1
    z_i = z*_i
    do j = i + 1 : N      (LU_ij ≠ 0)
        z_i = z_i − LU_ij z_j
    enddo
    z_i = z_i / LU_ii
enddo
```

Point-2-Point comm+sync

Global comm+sync

$$x_0 = u^n \qquad z_0 = \mathbf{P}^{-1} r_0$$

$$r_0 = b - \mathbf{A}\, x_0 \qquad p_0 = z_0$$

$$\boxed{\mathbf{P} \approx \mathbf{A}} \qquad r_r = r_0 \cdot z_0$$

do k = 0 : $\mathrm{k}_{\max}$

$$y_k = \boxed{\mathbf{A}\, p_k}$$

$$\alpha_k = r_r / \boxed{(p_k \cdot y_k)}$$

$$x_{k+1} = x_k + \alpha_k\, p_k$$

$$r_{k+1} = r_k - \alpha_k\, y_k$$

$$z_{k+1} = \boxed{\mathbf{P}^{-1}\, r_{k+1}}$$

$$r_{\text{old}} = r_r$$

$$r_r = \boxed{r_{k+1} \cdot z_{k+1}}$$

Check r_r for convergence

$$\beta_k = r_r / r_{\text{old}}$$

$$p_{k+1} = \beta_k\, p_k + z_k$$

enddo

$$u^{n+1} = x_{k+1}$$

Implicit scheme with Krylov solvers: BE+PCG

- ✓ Robust, proven method
- ✓ Can be very efficient
- ✓ Pre-made libraries available for many applications
- ✓ L-stability allows the use of very large time-steps

- Requires a quality preconditioner to be efficient, which can be very difficult to formulate, implement, vectorize, and scale. The PCs can also suffer "break down"
- Requires linear operator
- Choosing the convergence tolerance is not trivial
- Global communication for dot products limits scaling (global sync)
- Dot product can suffer from floating point errors requiring quad precision

Explicit Super Time-stepping

New!

- Relatively new methods (1996-2015), relatively uncommon
- Unconditionally stable, but *explicit*!
- Main idea: Runge-Kutta method with stages added for more stability, rather than for more accuracy
- Two main flavors are RKC (Chebyshev-based) and RKL (Legendre-based) and can be recursive or factored
  [See [O'Sullivan, 2015] (yesterday's talk) for high-order factored RKC (FRKC) methods]
- STS (RKC) is currently being used in several MHD codes with success (*FLASH, PLUTO, Lare3D*) and is planned for inclusion in others
- We use the 2nd-order recursive RKL2 from [Meyer et al, 2014] because it can have better stability properties for non-uniform and non-linear diffusion coefficients

Explicit Super Time-stepping: RKL (Derivation [Meyer et al, 2014])

PDE: $\frac{\partial u}{\partial t} = \mathbf{M}\, u(t)$

Solution expansion: $u(t) = e^{t\mathbf{M}}\, u(0) \approx \left(1 + t\,\mathbf{M} + \frac{1}{2}(t\,\mathbf{M})^2 + \ldots\right) u(0)$

Discretized form: $u^{n+1} = e^{z}\, u^{n} \approx \left(1 + z + \frac{z^2}{2} + \ldots\right) u^{n}, \quad z = \Delta t\,\mathbf{M} \quad \Delta t \ll 1$

Multi-step explicit scheme: $u^{n+1} = R(\Delta t\,\mathbf{M})\, u^{n}$

For accuracy, need: $R(z) = 1 + z + z^2/2 + O(z^3)$

For stability, need: $|R(\Delta t\,\lambda)| \leq 1, \;\; \forall \lambda \in \mathbf{M}$

Example: First-order explicit Euler method $(\lambda \in \mathcal{R})$

$R(z) = 1 + z$

$u^{n+1} = (1 + \Delta t\,\mathbf{M})\, u^{n}$

$\frac{u^{n+1} - u^{n}}{\Delta t} = \mathbf{M}\, u^{n}$

$|1 + \Delta t_{\text{Euler}}\,\lambda| \leq 1$

$\Delta t_{\text{Euler}} \leq \frac{2}{|\lambda|_{\max}}$

1D HEAT EQ: $\Delta t_{\text{Euler}} \leq \frac{\Delta x^2}{2}$

Legendre polynomial $P_s(x)$

$P_j(x) = (1/j)\,[(2j-1)\,x\,P_{j-1}(x) - (j-1)\,P_{j-2}(x)]$

$|P_s(x)| \leq 1, \;\; x \in [-1, 1]$

RKL: $R(z) = a_s + b_s\,P_s(1 + w_1\,z)$

<u>Accuracy</u>

$O(\Delta t)$

$a_s = 1 - b_s$

$b_s = 1$

$w_1 = \frac{2}{s^2 + s}$

$O(\Delta t^2)$

$a_s = 1 - b_s$

$b_s = \frac{s^2 + s - 2}{2\,s\,(s+1)}$

$w_1 = \frac{4}{s^2 + s - 2}$

<u>Stability</u>

$-1 \leq 1 + w_1\,\Delta t\,\lambda \leq 1$

Select s, get max dt:

$\Delta t \leq \frac{\Delta t_{\text{Euler}}}{w_1}$

Select dt, get min s:

$w_1 \leq \frac{\Delta t_{\text{Euler}}}{\Delta t}$

Recursion relation leads to easy implementation

Explicit Super Time-stepping: RKL2 [Meyers et al, 2014]

$$y_0 = u^n$$
$$F_0 = \mathbf{M}\, y_0$$
$$y_1 = y_0 + \tilde{\mu}_1 \,\Delta t\, F_0$$

Point-2-Point comm+sync

```
do j = 2 : s
```

$$y_j = \mu_j\, y_{j-1} + \nu_j\, y_{j-2} + (1-\mu_j-\nu_j)\, y_0 + \tilde{\mu}_j \,\Delta t\, \boxed{\mathbf{M}\, y_{j-1}} + \gamma_j \,\Delta t\, F_0$$

```
enddo
```

$$u^{n+1} = y_s$$

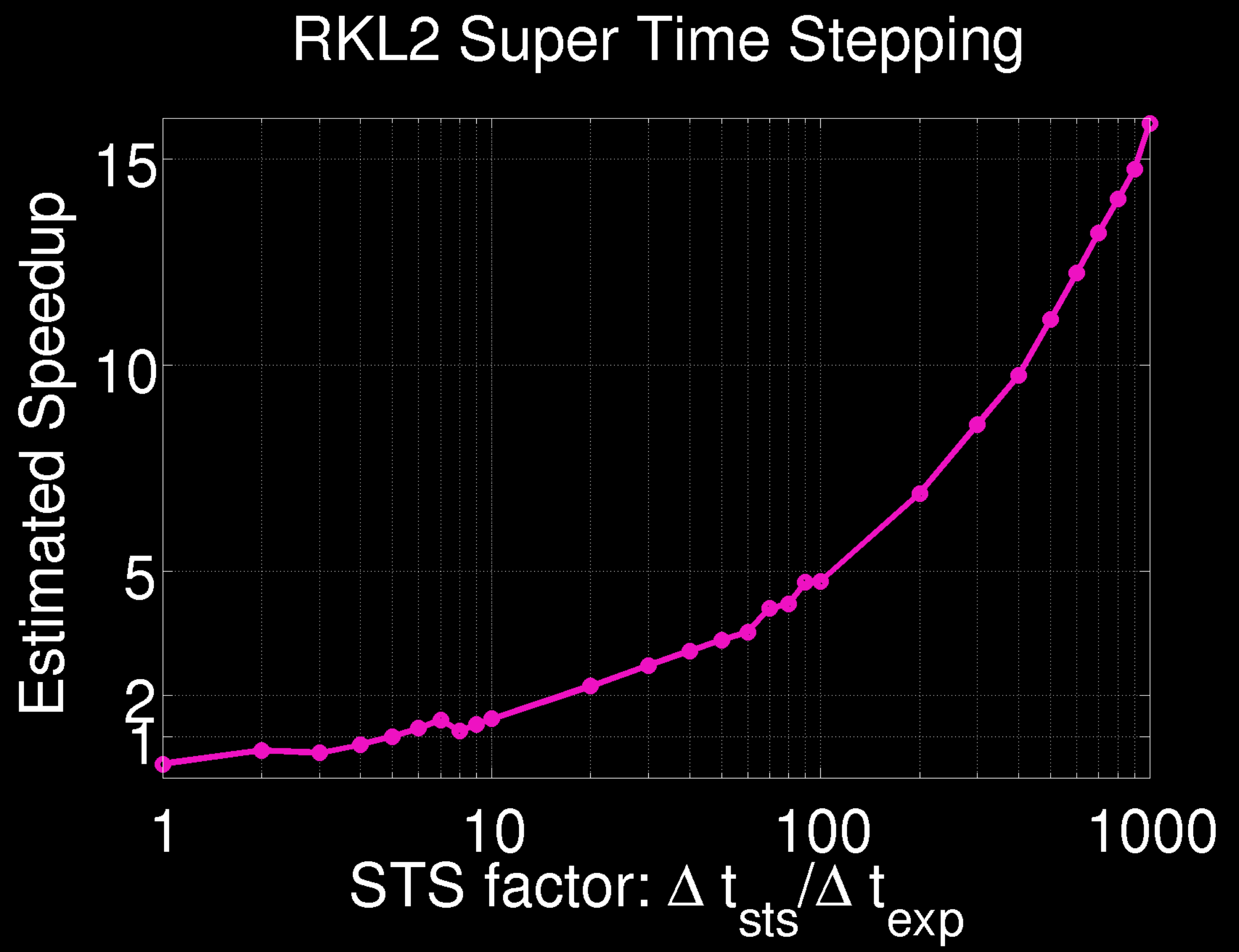
$$b_0 = b_1 = b_2 = \frac{1}{3} \qquad b_j = \frac{j^2+j-2}{2\,j\,(j+1)}$$

$$\tilde{\mu}_1 = \frac{4}{3\,(s^2+s-2)} \qquad \tilde{\mu}_j = \frac{4\,(2\,j-1)}{j\,(s^2+s-2)}\,\frac{b_j}{b_{j-1}}$$

$$\mu_j = \frac{2\,j-1}{j}\,\frac{b_j}{b_{j-1}} \qquad \nu_j = -\frac{j-1}{j}\,\frac{b_j}{b_{j-2}}$$

$$\gamma_j = (b_j - 1)\,\tilde{\mu}_j$$

$$s \ge \left\lceil \frac{1}{2}\left(\sqrt{9 + 16\,\frac{\Delta t}{\Delta t_{\text{Euler}}}} - 1\right)\right\rceil \qquad \operatorname{mod}(s,2) \ne 0$$



Predictive Science Inc.

Explicit Super Time-stepping: RKL2

- ✓ Very easy to implement (ideal for directive parallelization e.g. OpenMP/OpenACC)
- ✓ No global synchronization points
- ✓ Can include nonlinearities and/or flux limiting

- o New method – low circulation
- o Number of sub-cycles can be large depending on the problem and grid
- o Amplification factor not monotonically decreasing for increasing wave modes – *may lead to undesirable results when using large time-steps and/or short simulation times*

Thermodynamic MHD model: MAS

MAS
Magnetohydrodynamic
Algorithm
Outside a Sphere

- Established MHD Solar coronal/heliospheric model with over 10 years of development
- Written in FORTRAN 90, parallelized with MPI-2
- Has been used extensively in solar physics research described in numerous publications
- Available for use at NASA's Community Coordinated Modeling Center
http://ccmc.gsfc.nasa.gov

Thermodynamic MHD model: MAS [Governing Equations]

$$\frac{\partial \mathbf{A}}{\partial t} = \mathbf{v} \times (\nabla \times \mathbf{A})$$
$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho\, \mathbf{v})$$
$$\frac{\partial T}{\partial t} = -\nabla \cdot (T\mathbf{v}) - (\gamma - 2)\,(T\,\nabla \cdot \mathbf{v})$$

$$\frac{\partial \mathbf{v}}{\partial t} = -\mathbf{v} \cdot \nabla\, \mathbf{v} + \frac{1}{\rho}\left[\frac{1}{c}\mathbf{J} \times \mathbf{B} - \nabla p + \rho\, \mathbf{g}\right]$$

$$\nabla \cdot \mathbf{B} = 0$$

$\mathbf{B} = \nabla \times \mathbf{A}$ $\quad p = 2\,k\,T\,\rho/m_p$ $\quad v_A^2 = |\mathbf{B}|^2/(4\,\pi\,\rho)$

$\mathbf{J} = \frac{c}{4\,\pi}\nabla \times \mathbf{B}$ $\quad \mathbf{g} = -g_0\,R_\odot^2\,\hat{\mathbf{r}}/r^2$ $\quad \gamma = 5/3$ $\quad \hat{\mathbf{b}} = \mathbf{B}/|\mathbf{B}|$

Thermodynamic MHD model: MAS [Governing Equations]

$$\frac{\partial \mathbf{A}}{\partial t} = \mathbf{v}\times(\nabla\times\mathbf{A}) - \boxed{\frac{c^2\,\eta}{4\,\pi}\nabla\times\nabla\times\mathbf{A}}$$

RESISTIVITY

$$\frac{\partial \rho}{\partial t} = -\nabla\cdot(\rho\,\mathbf{v})$$

$$\frac{\partial T}{\partial t} = -\nabla\cdot(T\mathbf{v}) - (\gamma-2)\,(T\,\nabla\cdot\mathbf{v}) + \frac{\gamma-1}{2\,k}\frac{m_p}{\rho}\left[\boxed{-\nabla\cdot(\mathbf{q}_1+\mathbf{q}_2)} - \frac{\rho^2}{m_p^2}\boxed{Q(T)} + \boxed{H}\right]$$

THERMAL CONDUCTION

$$\mathbf{q}_1 = -f(r)\,\beta_{\text{Tcut}}(T)\,\kappa_0\,T^{5/2}\,\hat{\mathbf{b}}\hat{\mathbf{b}}\cdot\nabla T$$

[Spitzer et al, 1953]

$$\mathbf{q}_2 = (1-f(r))\,\frac{k}{(\gamma-1)}\,\frac{\rho}{m_p}\,T\,\mathbf{v}\,\hat{\mathbf{b}}\hat{\mathbf{b}}$$

[Hollweg, 1976]

RADIATIVE COOLING

$$Q(T) = \frac{10^{-22}}{\beta_{\text{Tcut}}(T)}\left[\begin{array}{l} 0.4\exp[-30\,(\log_{10}(T)-4.60)^2] \\ +4.0\exp[-20\,(\log_{10}(T)-4.90)^2] \\ +4.5\exp[-16\,(\log_{10}(T)-5.35)^2] \\ +2.0\exp[-04\,(\log_{10}(T)-6.10)^2] \end{array}\right]$$

[Athay, 1986]

HEATING

$$H = 0.044\,\frac{|B|\,(|B_{\theta\phi}|/|B|)^2}{|B_r|+0.25}\exp\left[\frac{-(r-1)}{0.3}\right] + 0.000435\exp\left[-(r-1)\right] + 0.13\exp\left[\frac{-(r-1)}{0.03}\right] + 0.15\,(1-\hat{\mathbf{b}}_{\mathbf{r}}^2)^2\;|\mathbf{B}_{R_\odot}|\;\exp\left[\frac{-(r-1)}{0.03}\right]$$

ALFVÉN WAVES

$$\frac{\partial \epsilon_+}{\partial t} = -\nabla\cdot\left(\epsilon_+\left[\mathbf{v}+v_A\,\hat{\mathbf{b}}\right]\right) - \frac{1}{2}\epsilon_+\,\nabla\cdot\mathbf{v}$$

$$\frac{\partial \epsilon_-}{\partial t} = -\nabla\cdot\left(\epsilon_-\left[\mathbf{v}-v_A\,\hat{\mathbf{b}}\right]\right) - \frac{1}{2}\epsilon_-\,\nabla\cdot\mathbf{v}$$

$$\frac{\partial \mathbf{v}}{\partial t} = -\mathbf{v}\cdot\nabla\mathbf{v} + \frac{1}{\rho}\left[\frac{1}{c}\mathbf{J}\times\mathbf{B} - \nabla p - \boxed{\nabla\left(\frac{\epsilon_+ + \epsilon_-}{2}\right)} + \rho\,\mathbf{g}\right] + \boxed{\frac{1}{\rho}\nabla\cdot(\nu\rho\nabla\mathbf{v})} + \boxed{\frac{1}{\rho}\nabla\cdot\left(S\,\rho\,\nabla\frac{\partial\mathbf{v}}{\partial t}\right)}$$

VISCOSITY

SEMI-IMPLICIT OPERATOR

$$\nabla\cdot\mathbf{B} = 0$$

$\mathbf{B} = \nabla\times\mathbf{A}$ $\quad p = 2\,k\,T\,\rho/m_p$ $\quad v_A^2 = |\mathbf{B}|^2/(4\,\pi\,\rho)$

$\mathbf{J} = \dfrac{c}{4\,\pi}\nabla\times\mathbf{B}$ $\quad \mathbf{g} = -g_0\,R_\odot^2\,\hat{\mathbf{r}}/r^2$ $\quad \gamma = 5/3$ $\quad \hat{\mathbf{b}} = \mathbf{B}/|\mathbf{B}|$

$$\beta_{\text{Tcut}}(T) = \begin{cases} (T/T_{\text{cut}})^{-5/2} & T < T_{\text{cut}} \\ 1, & T \geq T_{\text{cut}} \end{cases}$$

[Lionello et al, 2009]

$T_{\text{cut}} = 5\times10^5\,K$ $\quad f(r) = 0.5 - 0.5\tanh\left[2\,(r-10\,R_\odot)/R_\odot\right]$

$S = (\Delta t^2\,\tilde{k}^2)^{-1}\left(C_w^2/(1-C_f)^2 - 1\right)$ $\quad v_c^2 = \gamma\,p/\rho$

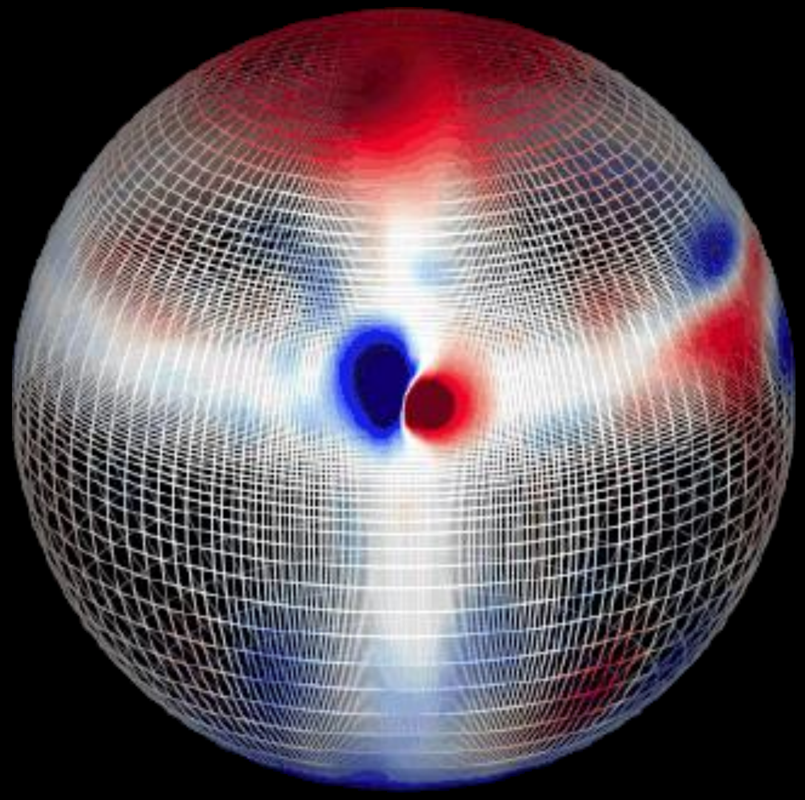
[Lionello et al, 1999]

$C_f = \Delta t\,\tilde{k}\cdot\mathbf{v}$ $\quad \tilde{k}^2 = 4\left(\Delta r^{-2} + (r\,\Delta\theta)^{-2} + (r\,\Delta\phi\,\sin\theta)^{-2}\right)$

$C_w^2 = 0.25\,\Delta t^2\,\tilde{k}^2\,(v_c^2+v_A^2)$

Thermodynamic MHD model: MAS [Numerical Methods]

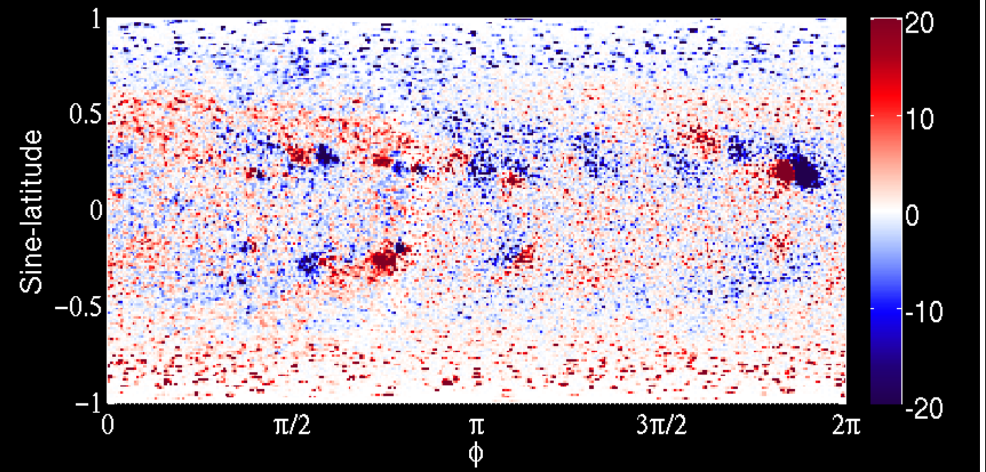
- Finite difference on logically rectangular non-uniform spherical grid
- Integrates $\mathbf{A}$ on staggered grid to ensure $\nabla \cdot \mathbf{B} = 0$
- Advective terms: upwinding
- Wave terms: Predictor-corrector
- Semi-implicit term solved using BE+PCG for both predictor and corrector
- Parabolic terms are operator split and use second-order central differences
- Matrix operators stored in modified DIA format, PC2 preconditioner stored in CSR format



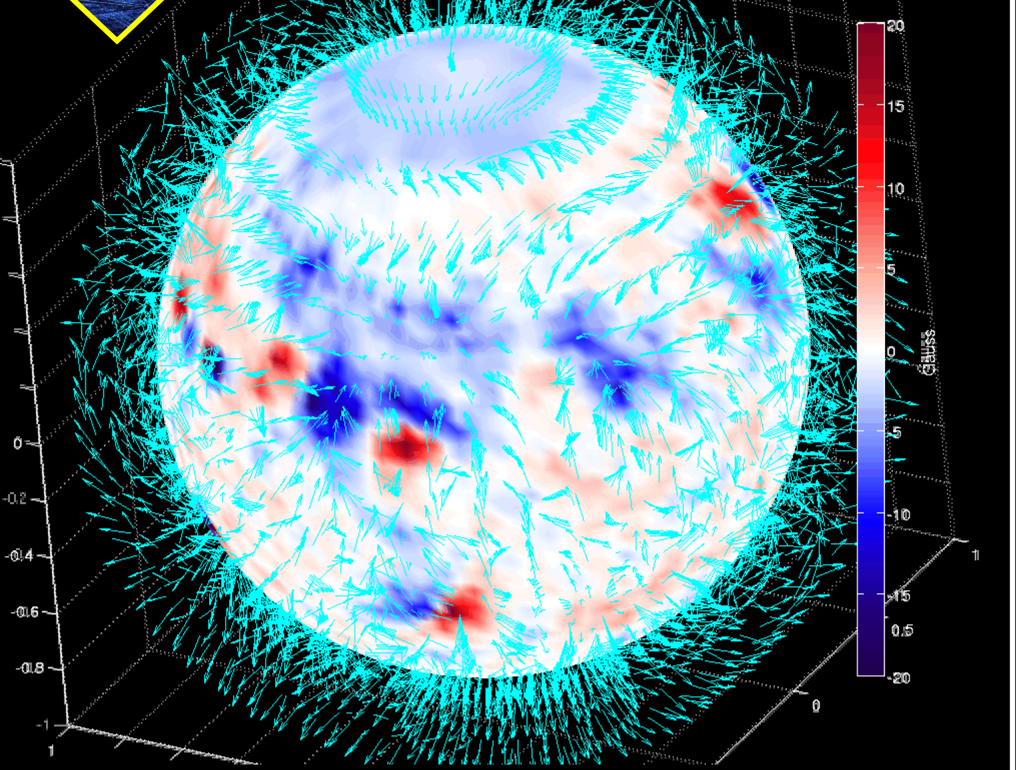
$(\Delta r, \Delta\theta, \Delta\phi)$

Real-world Test Case: Synchronic Coronal Relaxation

Air Force Data Assimilative Photospheric Flux Transport (ADAPT) [Arge et al. 2009]

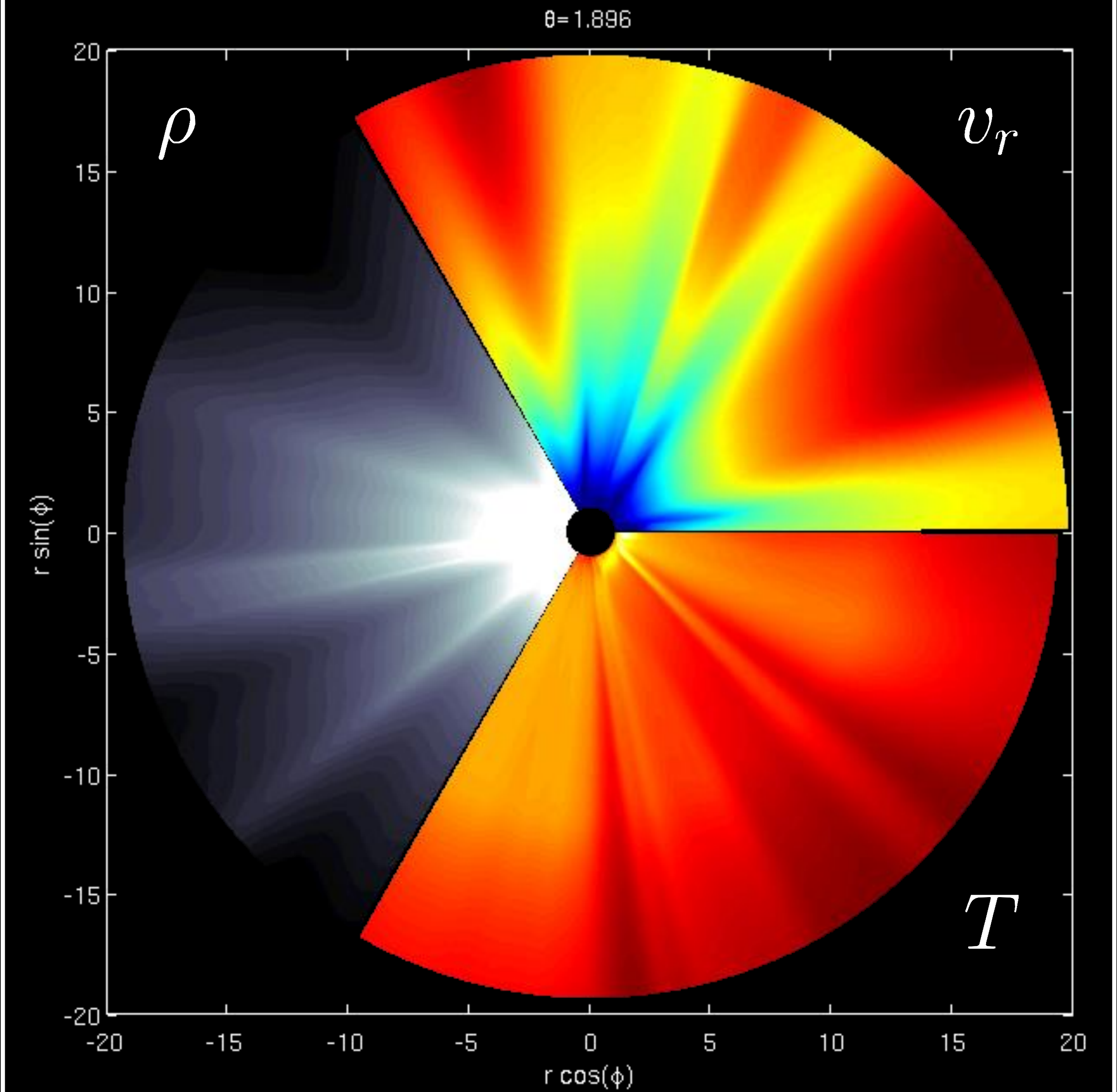


Process and interpolate onto grid



Solve potential field to get initial conditions of **A** and **B**

Insert solar wind approximation for initial plasma values



Integrate to a relaxed state. Relaxation time ~ 48 sim-hours

Validation runs

We integrate the relaxation for ~8 sim-hr

Timing runs

We integrate for 6 sim-min starting with the solution after the ~8 sim-hr relaxation

The Grid

181x251x602

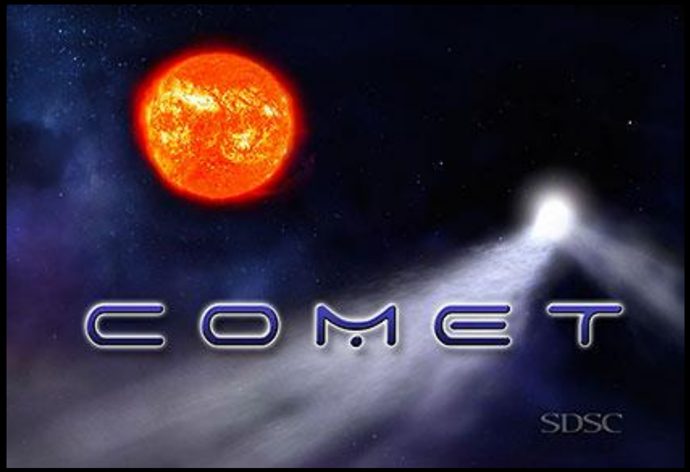
~27 million points

$$\min\left[\Delta r\right] \approx 340\,\mathrm{km}$$

$$\max\left[\Delta r\right] \approx 590,000\,\mathrm{km}$$

$$\max\left|1 - \frac{\Delta r_{i+1}}{\Delta r_i}\right| < 6\%$$

Real-world Test Case: HPC Environment



SDSC
SAN DIEGO SUPERCOMPUTER CENTER

Allocation provided by:

Processor Model	Intel Xeon E5-2660v3 (Haswell)
Clock Speed	2.5 GHz
Cores per node	24
Max Flop/s per node	$\approx$ 1 TFlop/s
DDR4 DRAM per node	128 GB
Total # of nodes	1944

Operating system: CentOS

Compiler: Intel 2015.2.164

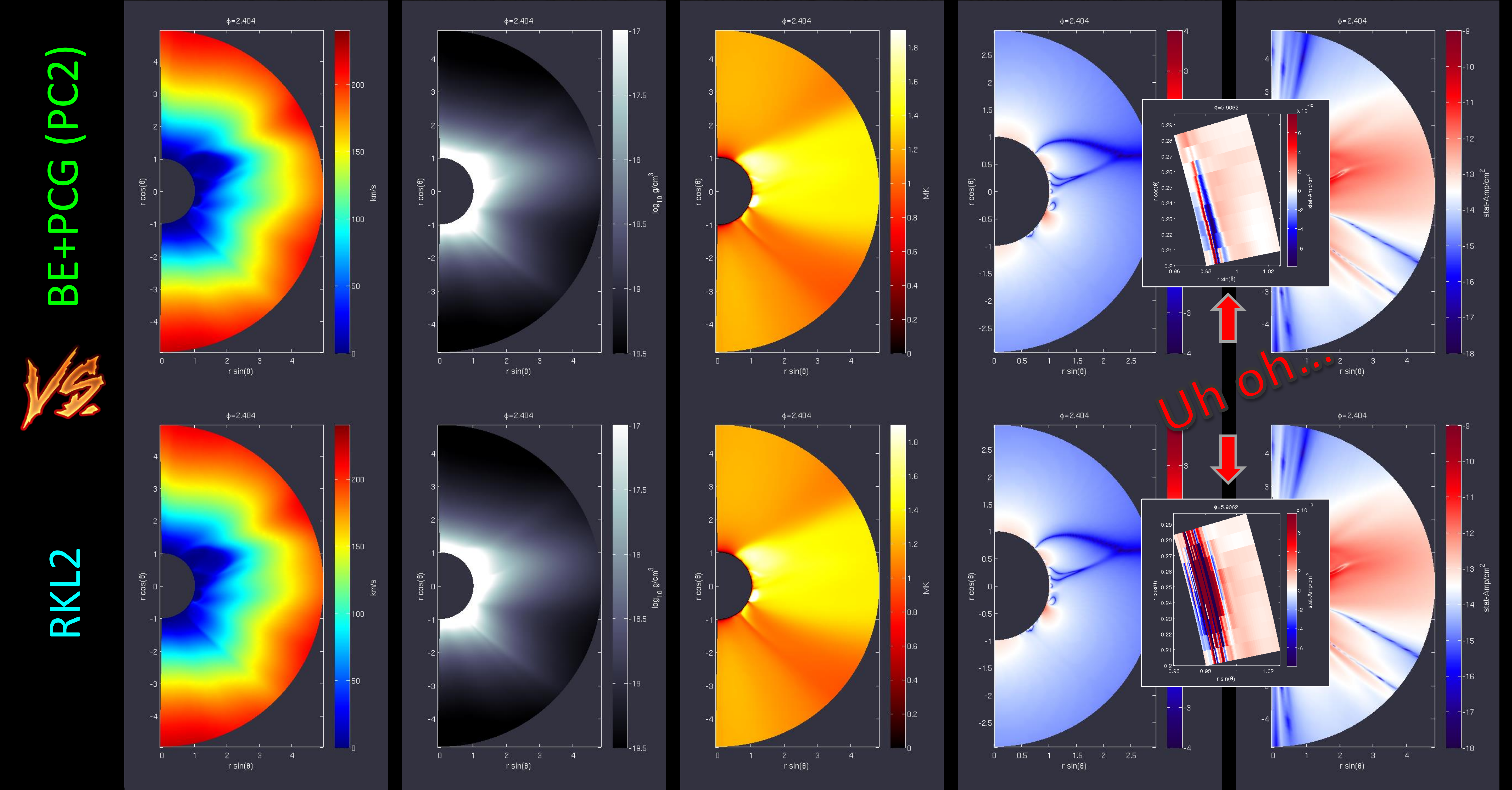
MPI library: MVAPICH for infiniband v2.1

slurm workload manager

Max # cores per run: 1728

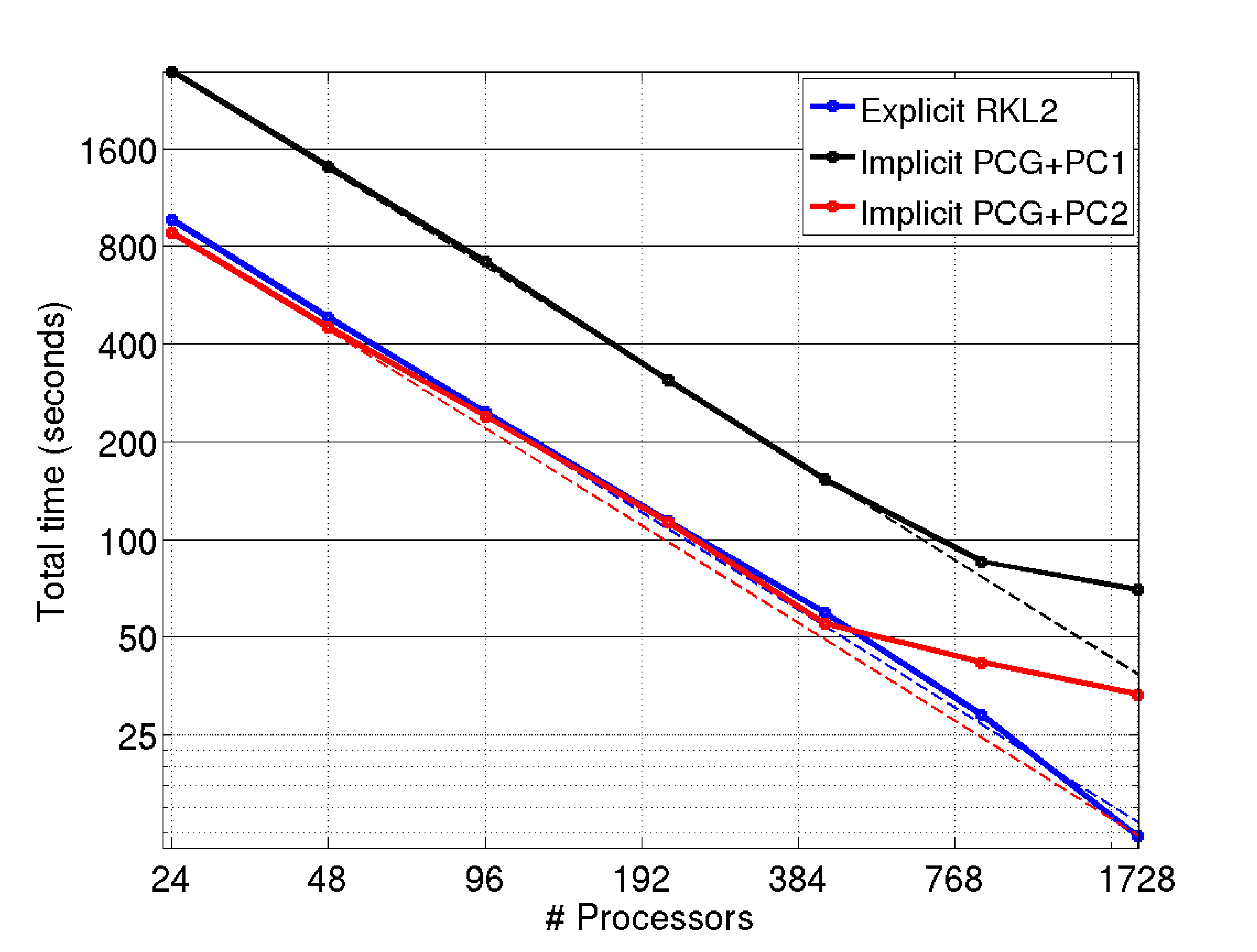
- Timings done using MPI timers
- Times for each routine are averaged over all processors
- Time recorded from the best run out of multiple runs

Validation



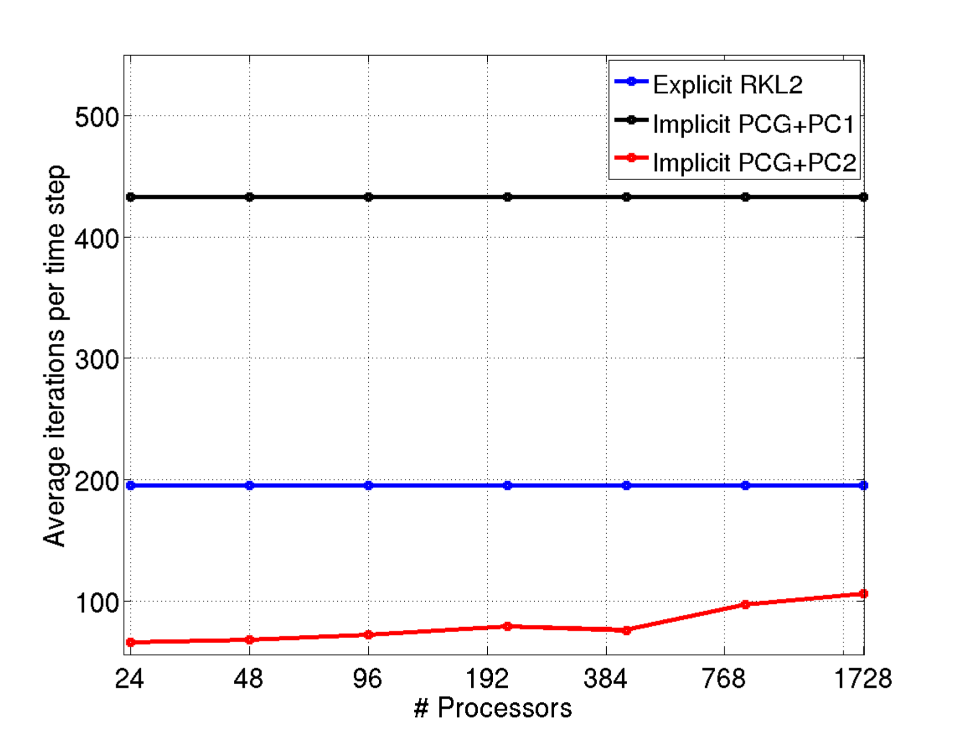
BE+PCG (PC2)
VS
RKL2
Uh oh...
ϕ=2.404
r cos(θ)
r sin(θ)
km/s
log10 g/cm3
MK
stat-Amp/cm2
ϕ=5.9062

Performance Results: Thermal Conduction

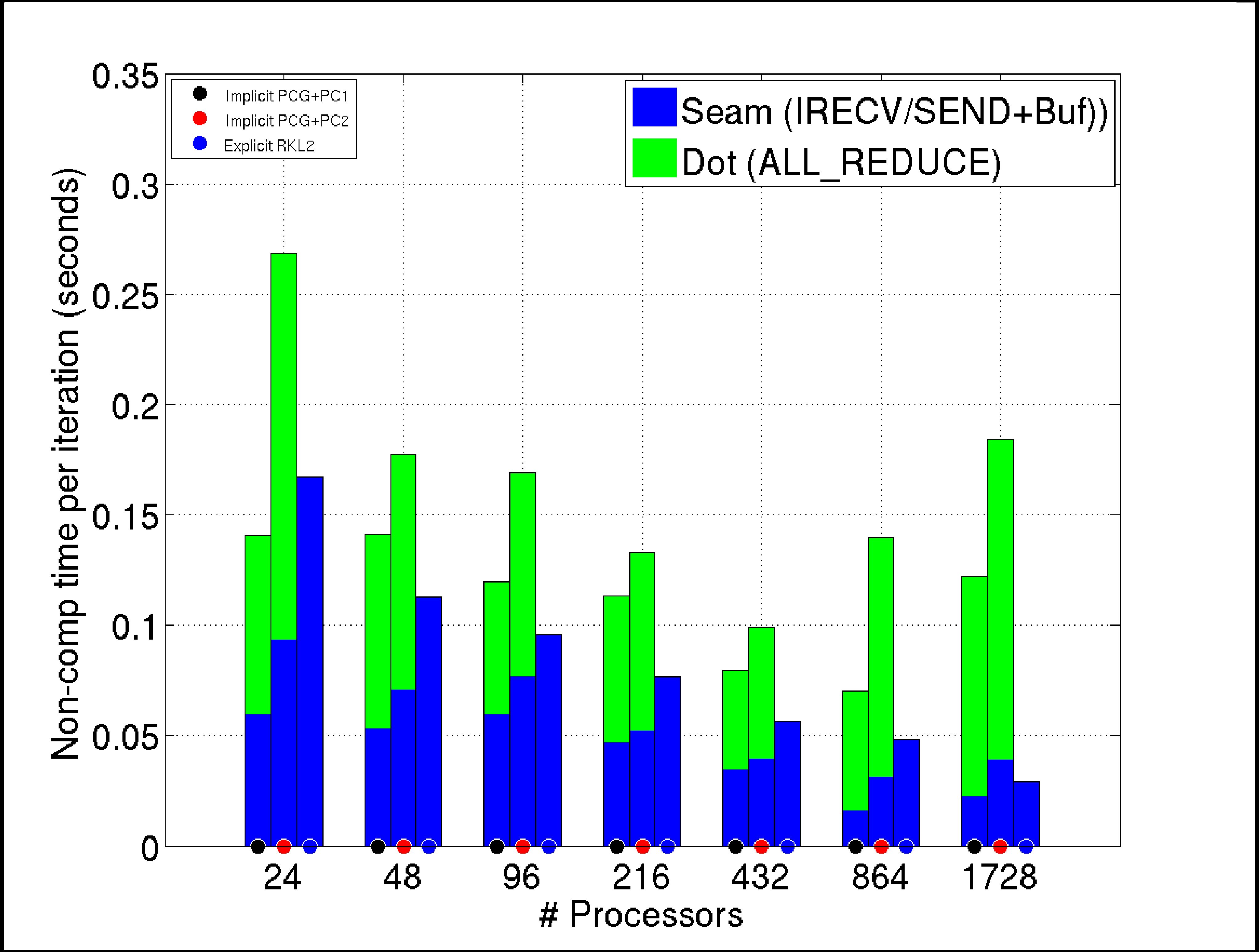


$$\nabla \cdot \left(\kappa_{\parallel} \nabla T\right)$$

Average Iterations per Step	
PCG+PC1 (DIAG)	433
PCG+PC2 (ILU0)	66 → 106
RKL2	195



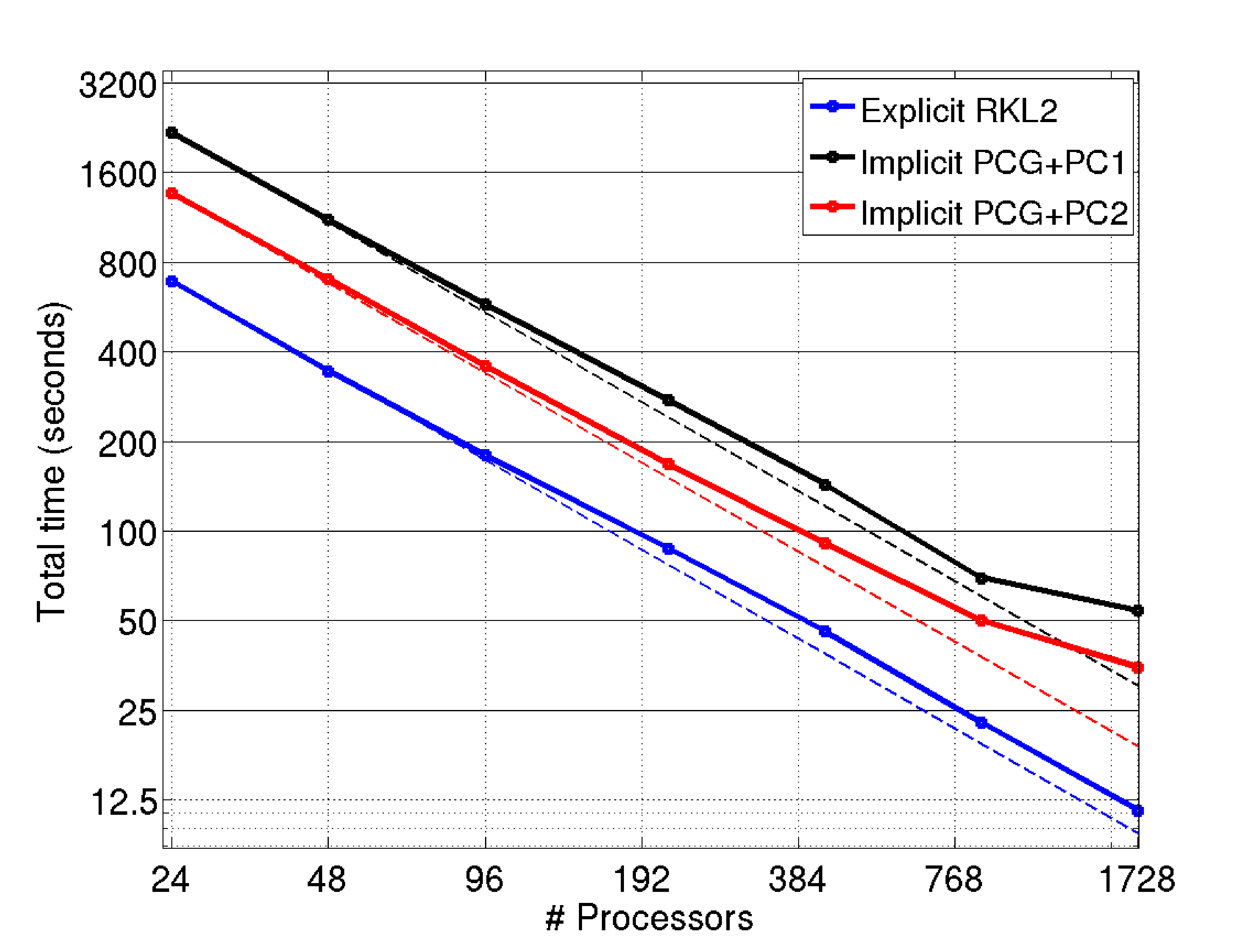
Performance Results: Thermal Conduction



$$\nabla \cdot \left(\kappa_{\parallel} \nabla T\right)$$

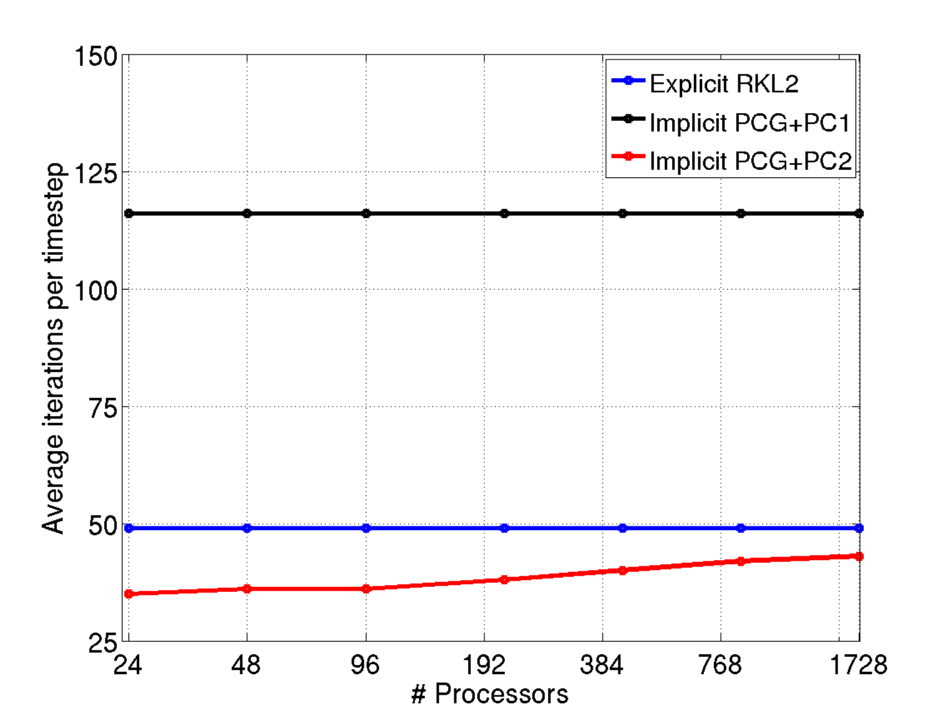
Average Iterations per Step	
PCG+PC1 (DIAG)	433
PCG+PC2 (ILU0)	$66 \rightarrow 106$
RKL2	195

Results: Viscosity

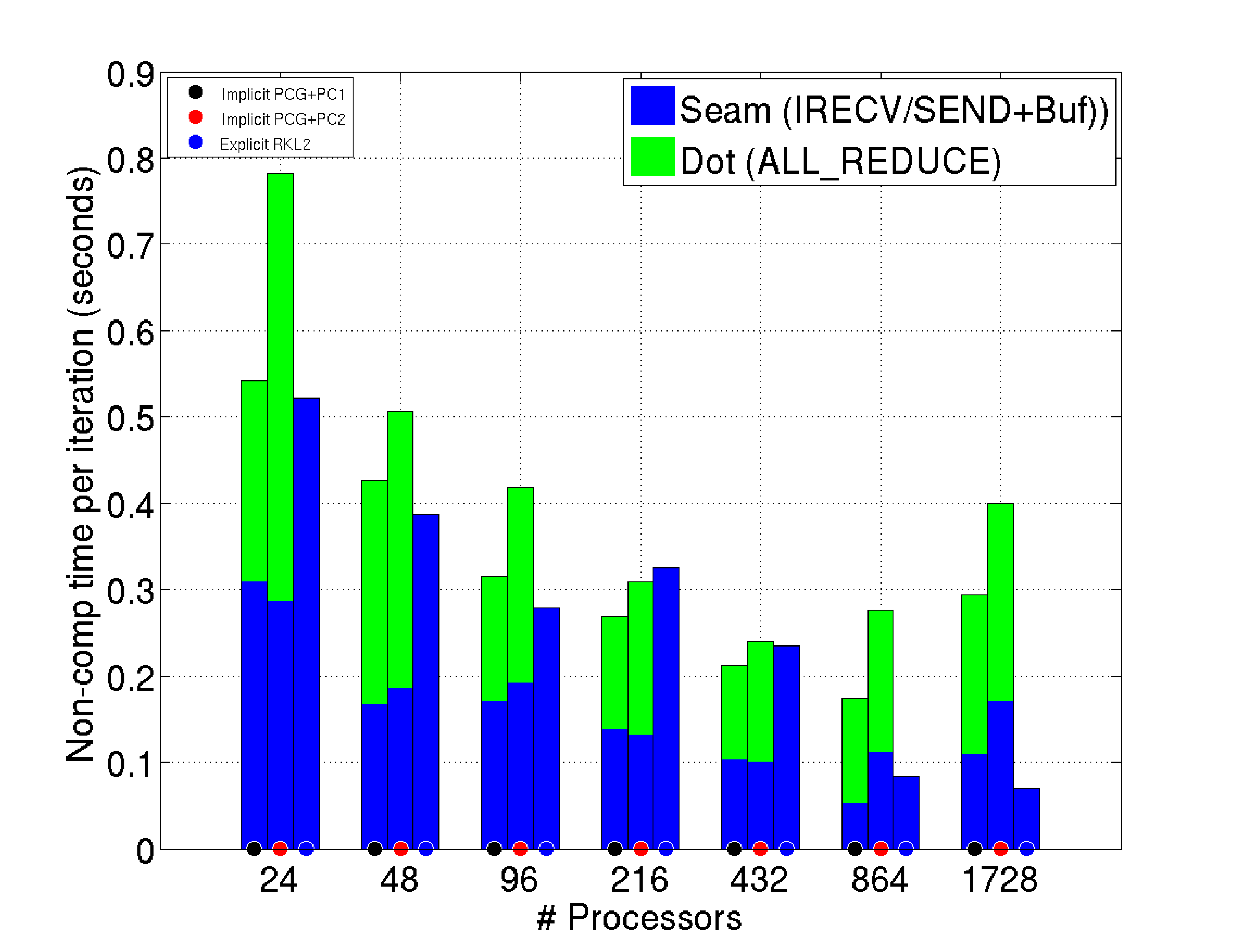


$$\nabla \cdot (\rho \nu \nabla \mathbf{v})$$

Average Iterations per Step	
PCG+PC1 (DIAG)	116
PCG+PC2 (ILU0)	$35 \rightarrow 43$
RKL2	49



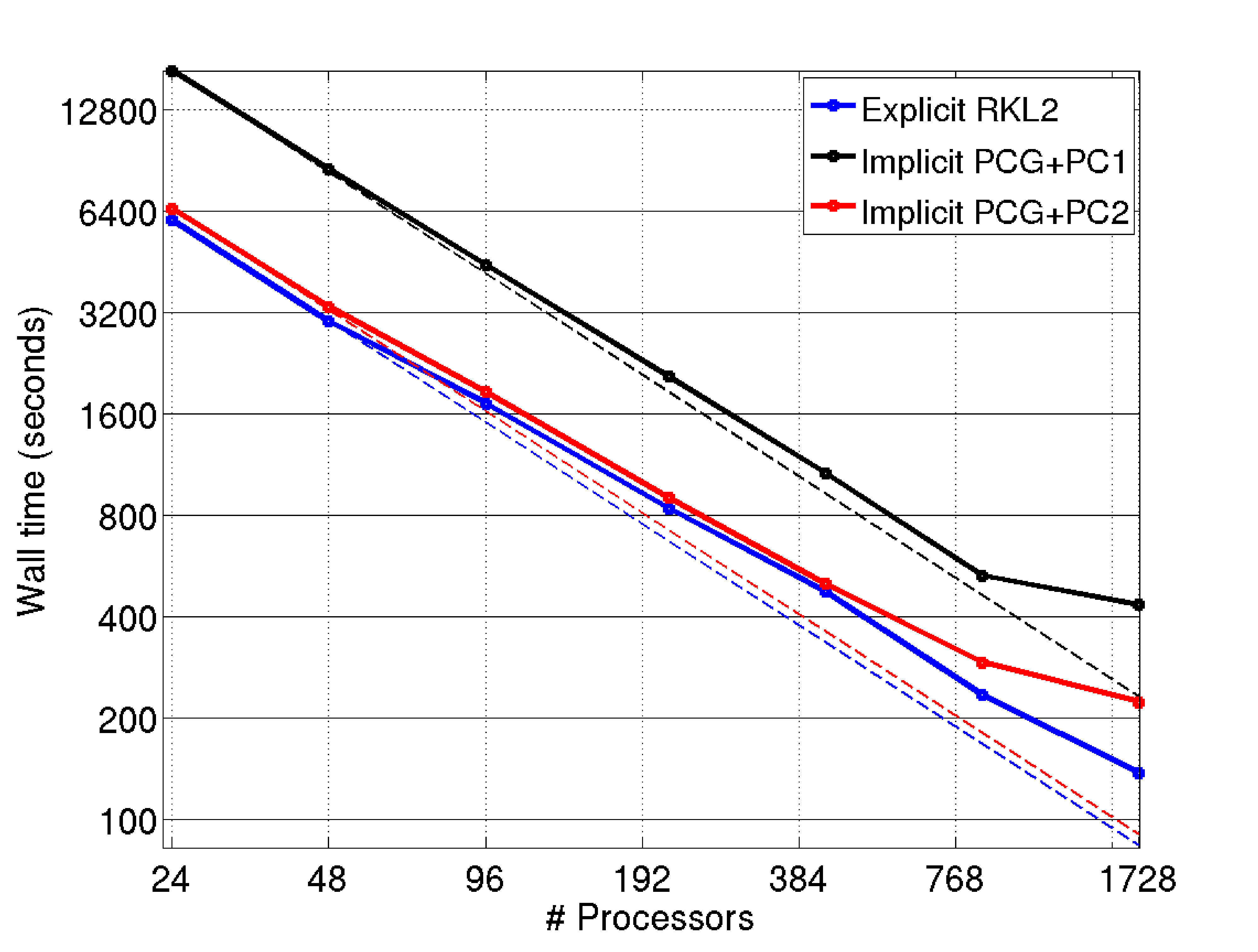
Performance Results: Viscosity



$$\nabla \cdot (\rho \nu \nabla \mathbf{v})$$

Average Iterations per Step	
PCG+PC1 (DIAG)	116
PCG+PC2 (ILU0)	$35 \to 43$
RKL2	49

Performance Results: Overall Code



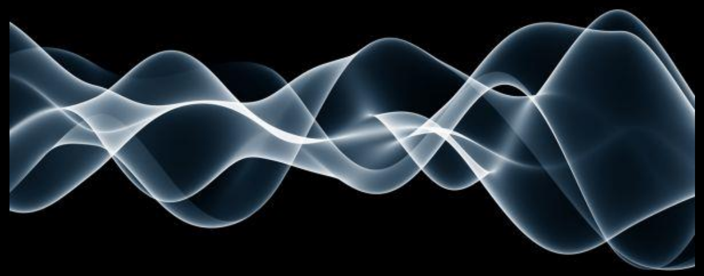
Time Breakdown from Relaxation	
Semi-implicit Pred+Corr	46%
Viscosity	16%
Thermal Conduction	14%
Alfven Waves	19%
Resistivity	1%
Rest of code	4%

Outlook

- Super Time-stepping algorithms show potential as an alternative to iterative solvers for parabolic terms in MHD models

- For the problem tested, the STS outperformed the iterative solver, especially in scaling to many CPUs

- For the viscosity term, there are some solution effects to work out
- Can STS methods be used for MHD waves?



Questions?

Slides available at:
www.predsci.com/~caplan/astro16

Predictive Science Inc.

Explicit Super Time-stepping: RKL2

- RKL needs the value of the Euler stability time-step to compute the number of required super-steps (s)
- In 3D spherical coordinates, simple upper-bound estimates can be too strict (especially for anisotropic diffusion)
- Can get much tighter bound using Gershgorin circle theorem:

Definition 1 *Given a square matrix* $\mathbf{A}$, *a Gershgorin disk for every row* j *is defined as a disk in the complex plane, centered at* A_{jj}, *with a radius of:*

$$R_j = \sum_{i \neq j}^{N} |A_{i,j}|$$

Theorem 1 *Every eigenvalue* λ *of a square matrix* $\mathbf{A}$ *lies in one of its Gershgorin discs:*

$$|\lambda - A_{jj}| \leq R_j$$

Since $\lambda \in \mathcal{R}$ and $A_{jj} \in \mathcal{R}$, we get a bound on $|\lambda|$ as:

$$|A_{jj}| - R_j \leq |\lambda_j| \leq |A_{jj}| + R_j$$

Therefore the maximum value of $|\lambda|$ is bound by:

$$|\lambda|_{\max} \leq \max\{|A_{jj}| + R_j, \forall j \text{ rows}\}$$

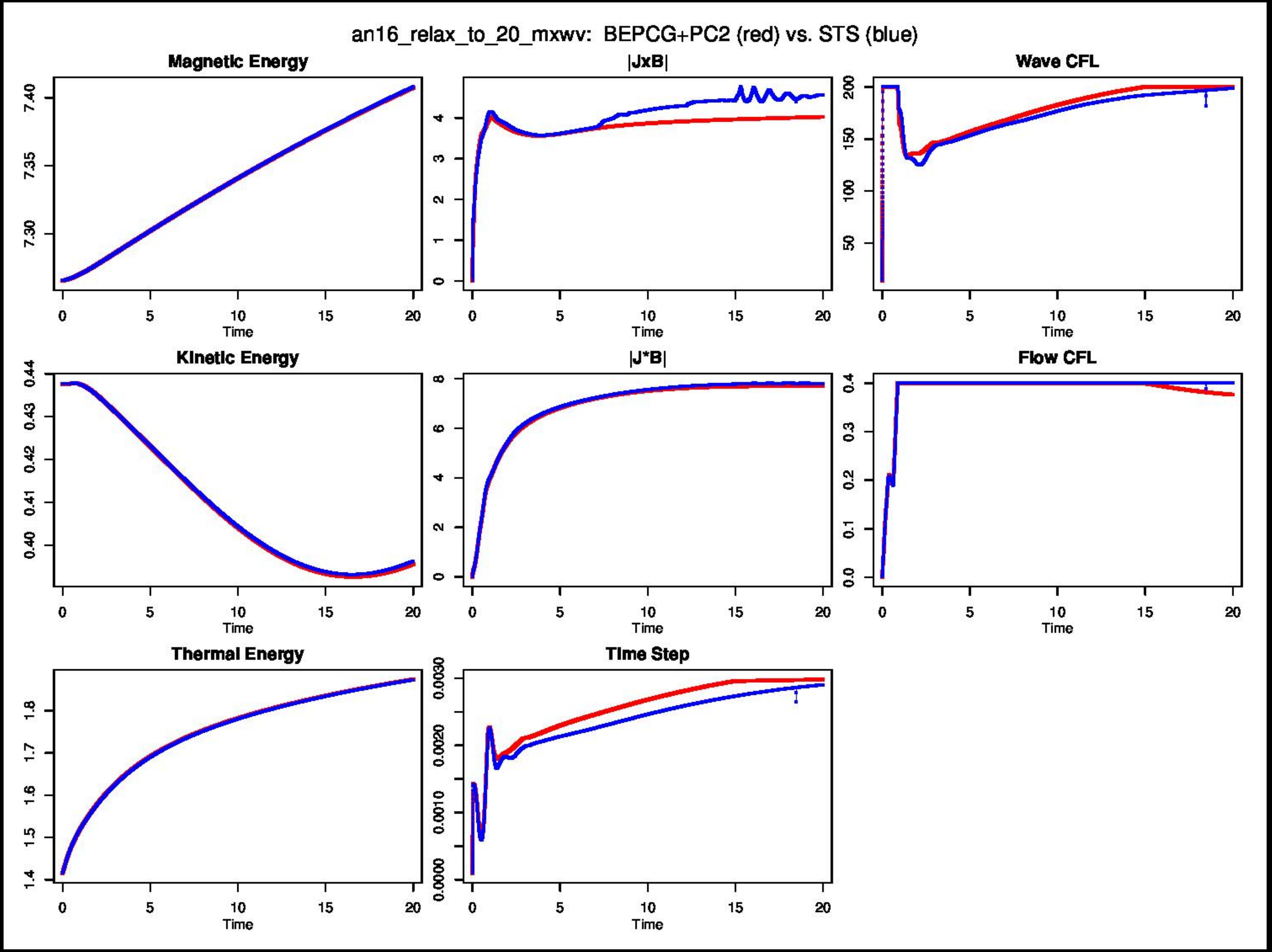
$$\Delta t_{\text{Euler}} \leq \frac{2}{|\lambda|_{\max}}$$

$$|\lambda|_{\max} \leq \max\left\{\sum_{i=1}^{N} |A_{i,j}|, \forall j \text{ rows}\right\}$$

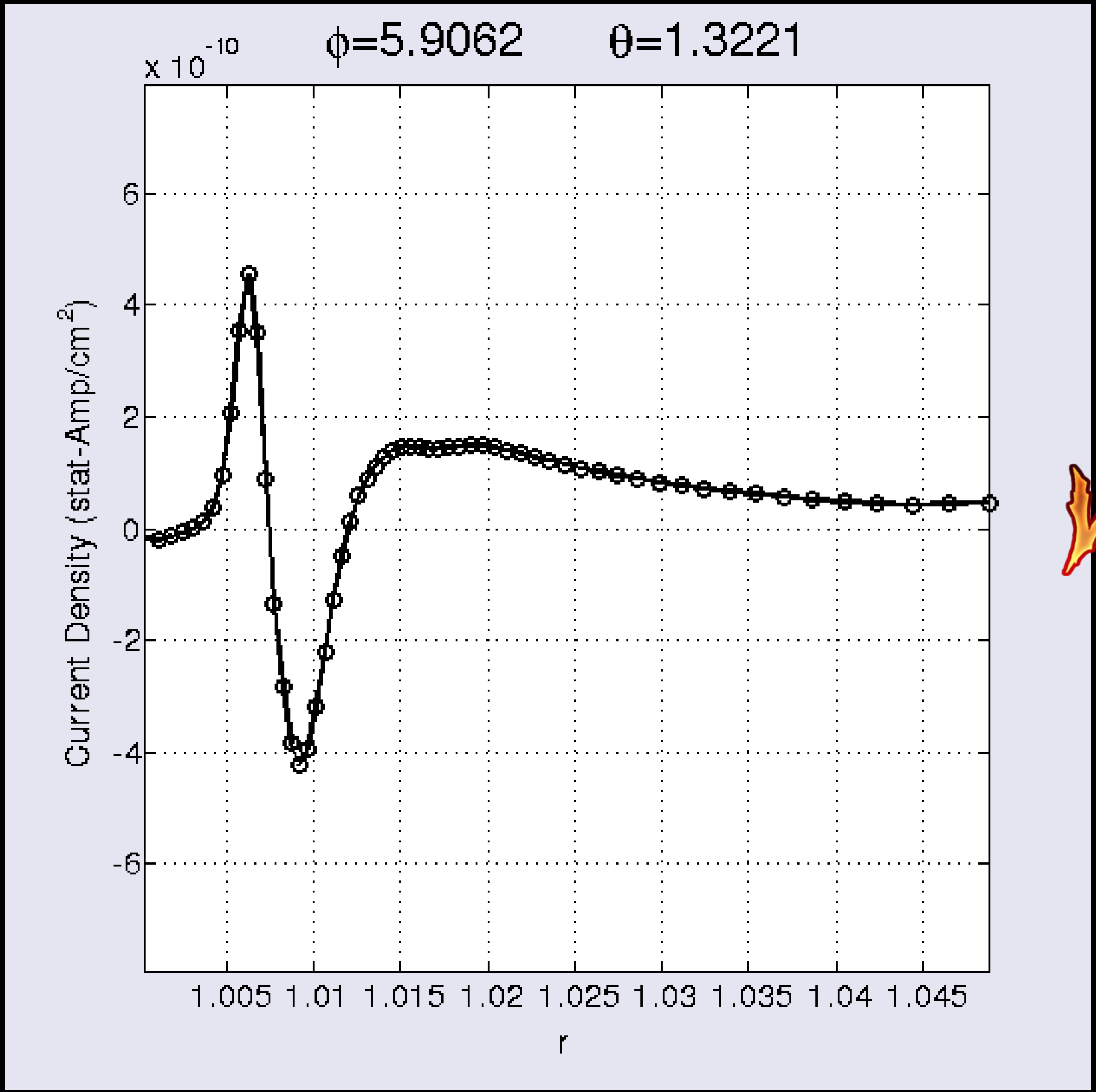
[Scott, 1985]: Max error in bound: $\sqrt{p+1}$

Validation

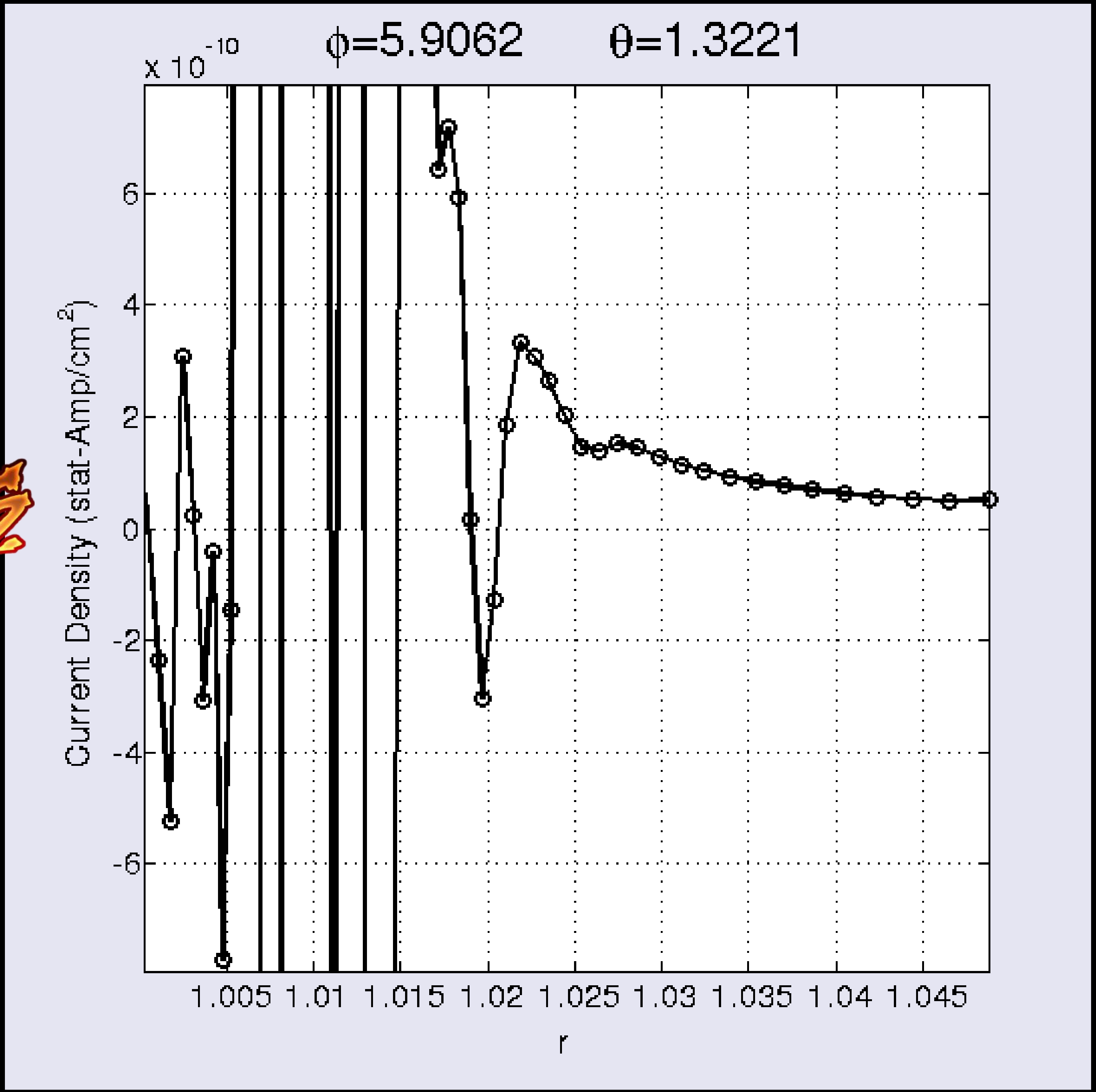
- First we compare BE+PCG2 versus RKL2 for thermal conduction only – match!
- Turning on RKL2 for viscosity has overall match (as seen in movies) but seems to have problems with the current density
- Maybe initial relaxation too violent? Try capping wave CFL to 200…
- Where is this happening?



Validation

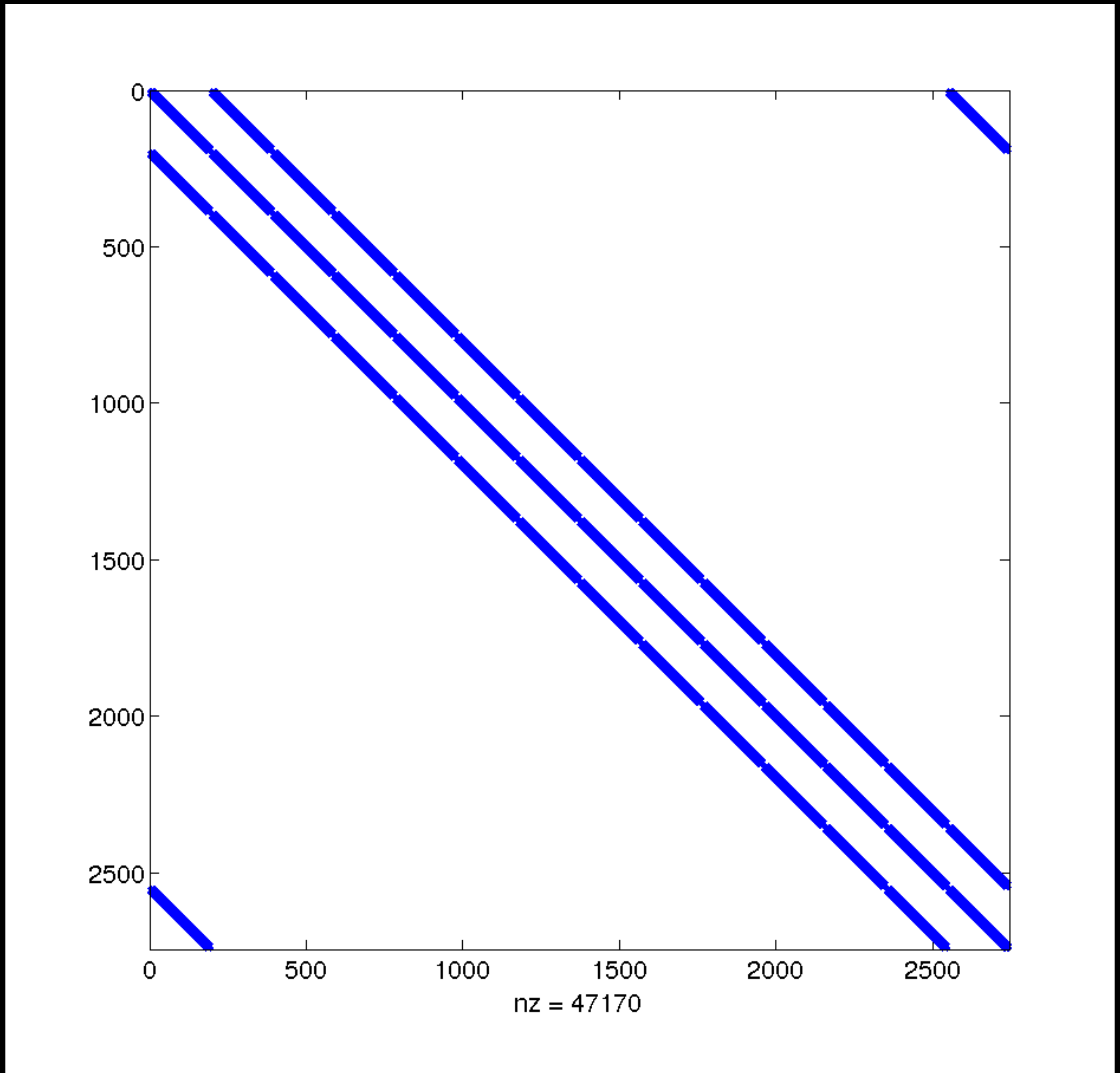


VS

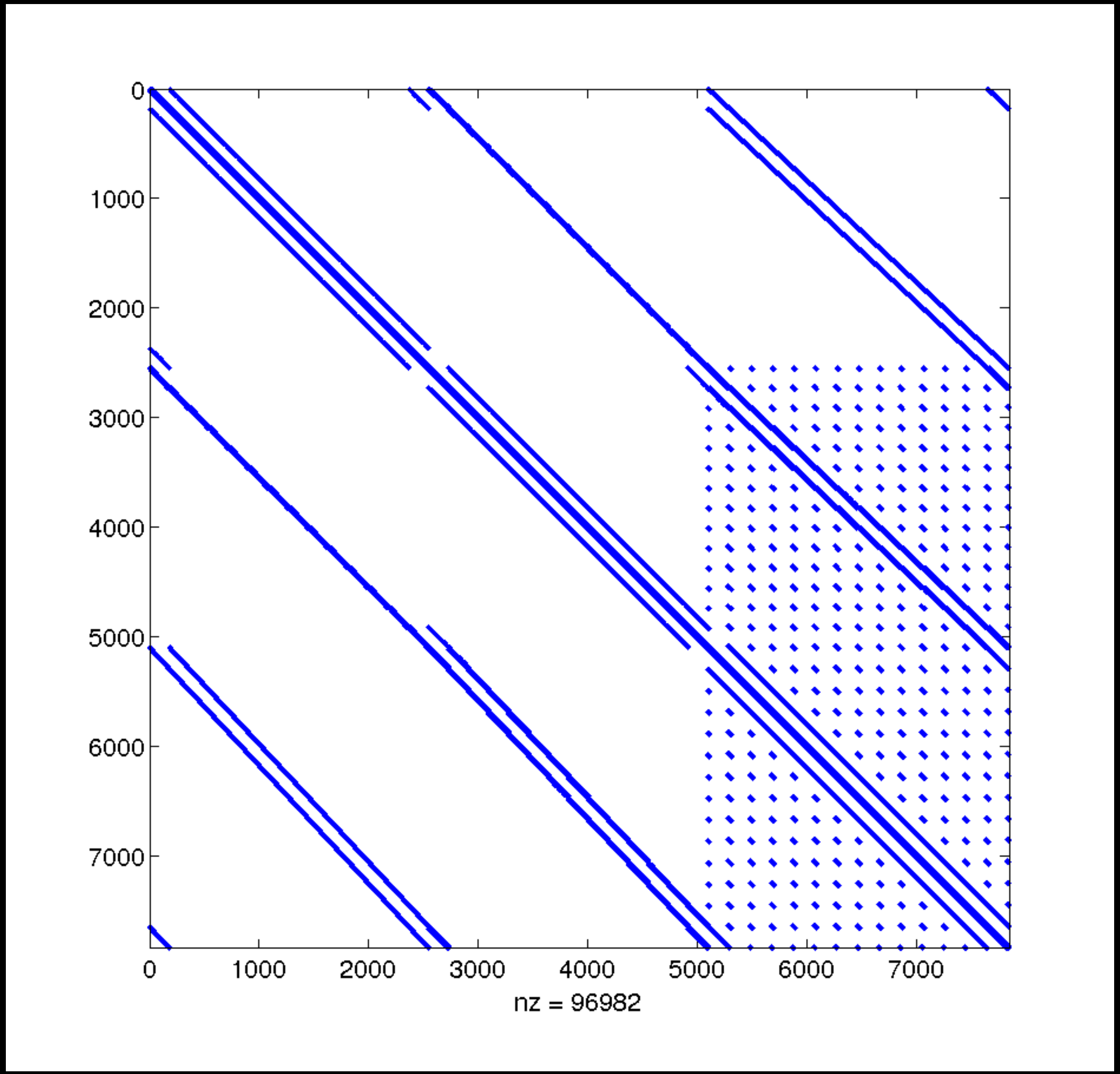


Validation

Thermal Conduction



Viscosity



Euler dt estimate:	0.0018
Euler dt estimate w/o pole bc:	0.0033