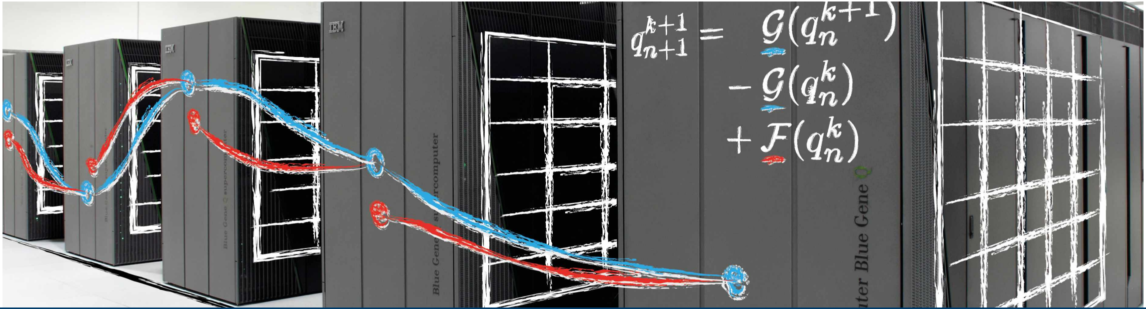


SDC for Rayleigh-Benard convection with spectral methods

December 17, 2024 | Thomas Baumann | Jülich Supercomputing Centre

Spectral methods

General idea

- Approximate solution via finite series expansion in some basis: $u(x) \approx \sum_{n=0}^{N-1} u_n f_n(x)$
- Known derivative relationships of the basis allows to compute derivatives of the solution
- Converges quickly with select bases and smooth solutions

Polynomial spectral method

- Basis functions: $p_n(x) = x^n$
- Write vectorially:
  $u(x) \approx \sum_{n=0}^{N-1} u_n x^n \rightsquigarrow \vec{u} = (u_0, u_1, \dots, u_{N-1})^T$
- Derivative: $\partial_x p_n(x) = n p_{n-1}(x)$
- Pretty useless in practice!

Derivative matrix

$$\partial_x \vec{u} = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 2 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & N-1 \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ \vdots \\ u_{N-2} \\ u_{N-1} \end{pmatrix}$$

Fourier spectral method

- Basis functions: $W_n(x) = \exp(-2\pi ixn/L)$
- Derivative: $\partial_x W_n(x) = \frac{-2\pi in}{L} W_n(x) \rightarrow$ diagonal
- Compute expansion via FFT
- Works only for periodic boundary conditions (BCs)

Derivative matrix

$$\frac{-2\pi i}{L}\begin{pmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & N \end{pmatrix}$$

JÜLICH Forschungszentrum

Chebychev spectral method

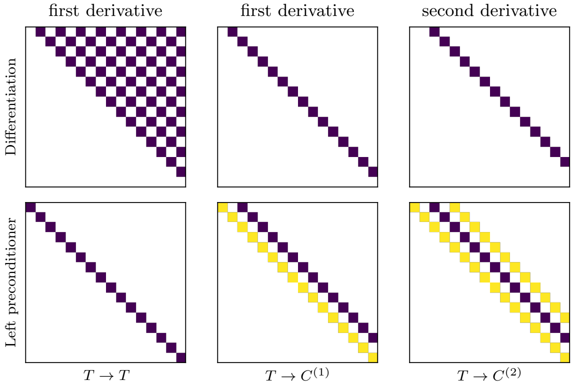
- Basis functions $T_n(x) = \cos(n \cos^{-1}(x))$
- Bounded $\| T_n(x) \|_\infty = 1$
- With change of variable $x = \cos(\theta)$: $T_n(x) = \cos(n\theta)$
  - Use DCT to compute series expansion
  - Change of variables defines grid on $x \in (-1,1)$, points clustered at the boundary
- Use non-periodic BCs with τ-method (see later)
- Downside: Derivative matrix is dense

$$\partial_x T_n(x) = \sum_{i=0}^{n-1} \frac{2n((n-i)\%2)}{1+\delta_{i0}} T_i(x)$$

$\rightarrow$ Use T_n in combination with other bases

Ultraspherical spectral method

Sparse preconditioned Chebychev method



- Compute expansion in Chebychev T polynomials
- Change basis while computing derivative, e.q. $\partial_x T_n(x) = nC^{(1)}_{n-1}(x)$
- New basis: Normalized Gegenbauer polynomials $C^{(\lambda)}$
- Render Chebychev matrices sparse with left preconditioner

τ-method for boundary conditions

Problem

Consider linear PDE $L\vec{u} = \vec{a}$ with BC $\vec{b}\vec{u} = c$ and N DoF.

τ-method: perturb PDE with τ term in the highest mode and add BC to the system

$$
\left(\begin{array}{ccc|c}
L_{0,0} & \dots & L_{0,N-1} & 0 \\
\vdots & \ddots & \vdots & \vdots \\
L_{N-2,0} & \dots & L_{N-2,N-1} & 0 \\
L_{N-1,0} & \dots & L_{N-1,N-1} & 1 \\
\hline
b_0 & \dots & b_{N-1} & 0
\end{array}\right)
\left(\begin{array}{c}
u_0 \\ \vdots \\ u_{N-2} \\ u_{N-1} \\ \hline \tau
\end{array}\right)
=
\left(\begin{array}{c}
a_0 \\ \vdots \\ a_{N-2} \\ a_{N-1} \\ \hline c
\end{array}\right)
$$

Get Dirichlet BCs by evaluating the elements at the boundary. E.g. $\mathrm{u}(\mathrm{x}) = \mathrm{c}$:

$$u(x) = \sum_{n=0}^{N-1} u_n f_n(x) \rightsquigarrow b_n = f_n(x).$$

τ-method: Example in polynomial base

Problem

PDE: $u_x = -1$, BC: $u(-1) = 1$, $N = 3$, $u^*(x) = -x$

Resulting discretization

$$\left(\begin{array}{ccc|c} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & -1 & 1 & 0 \end{array}\right) \left(\begin{array}{c} u_0 \\ u_1 \\ u_2 \\ \hline \tau \end{array}\right) = \left(\begin{array}{c} -1 \\ 0 \\ 0 \\ \hline 1 \end{array}\right) \rightarrow \vec{u} = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \quad \tau = 0.$$

In practice, we don't need to compute the τ terms and can simplify to

$$\left(\begin{array}{ccc} 0 & 1 & 0 \\ 0 & 0 & 2 \\ \hline 1 & -1 & 1 \end{array}\right) \vec{u} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

τ-method: Example in polynomial base

Problem

PDE: $u_x = -1$, BC: $u(-1) = 1$, $N = 3$, $u^*(x) = -x$

Resulting discretization

$$\left(\begin{array}{ccc|c} 0 & 1 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & -1 & 1 & 0 \end{array}\right) \left(\begin{array}{c} u_0 \\ u_1 \\ u_2 \\ \hline \tau \end{array}\right) = \left(\begin{array}{c} -1 \\ 0 \\ 0 \\ \hline 1 \end{array}\right) \rightarrow \vec{u} = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \quad \tau = 0.$$

In practice, we don't need to compute the τ terms and can simplify to

$$\left(\begin{array}{ccc} 0 & 1 & 0 \\ 0 & 0 & 2 \\ \hline 1 & -1 & 1 \end{array}\right) \vec{u} = \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix}.$$

τ-perturbations may show up in (SDC) residual

Poisson problem in regular polynomials

PDE: $\Delta u = 12x^2$, BCs: $u(-1) = -1$ and $u(1) = 1$, $u^*(x) = x^4 + x - 1$

Properly resolved $N = 5$

$$\begin{pmatrix} 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 12 \\ \hline 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 \end{pmatrix} \vec{u}_5 = \begin{pmatrix} 0 \\ 0 \\ 12 \\ \hline 1 \\ -1 \end{pmatrix}$$

$$\vec{u}_5 = (-1,1,0,0,1)^T$$

$$r = \|\Delta u_5 - 12x^2\| = \|\Delta(x^4 + x - 1) - 12x^2\| = 0$$

Under-resolved $N = 4$

$$\begin{pmatrix} 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 6 \\ \hline 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \vec{u}_4 = \begin{pmatrix} 0 \\ 0 \\ \hline 1 \\ -1 \end{pmatrix}$$

$$\vec{u}_4 = (0,1,0,0)^T$$

$$r = \|\Delta u_4 - 12x^2\| = \|\Delta x - 12x^2\| = \|12x^2\|$$

Rayleigh Benard convection (RBC)

Equations

$$
\begin{aligned}
u_t - \nu(u_{xx} + u_{zz}) + p_x &= -uu_x - vu_z, \\
v_t - \nu(v_{xx} + v_{zz}) + p_z - T &= -uv_x - vv_z, \\
T_t - \kappa(T_{xx} + T_{zz}) &= -uT_x - vT_z, \\
u_x + v_z &= 0.
\end{aligned}
$$

$$
\begin{aligned}
u(z=-1) = u(z=1) &= 0 \\
v(z=-1) = v(z=1) &= 0 \\
T(z=1) = 0, \quad T(z=-1) &= 2 \\
\int_\Omega p &= 0
\end{aligned}
$$

Discretization

- Fourier horizontally
- ultraspherical vertically

$$\Omega = [0,8) \times (-1,1)$$

Use Kronecker product to get 2D discretization from 2 1D discretizations

IMEX Euler solver

$$
\begin{aligned}
M\vec{u}_t + L\vec{u} &= f_{\mathrm{nonlin}}(\vec{u}) \\
\vec{u} &= (u,v,T,p)^T \\
M &= \mathrm{diag}(1,1,1,0) \\
L &= \begin{pmatrix} -\partial_x^2 - \partial_z^2 & 0 & 0 & \partial_x \\ 0 & -\partial_x^2 - \partial_z^2 & -1 & \partial_z \\ 0 & 0 & -\partial_x^2 - \partial_z^2 & 0 \\ \partial_x & \partial_z & 0 & 0 \end{pmatrix} \\
f_{\mathrm{nonlin}}(\vec{u}) &= (uu_x + vu_z, uv_x + vvz, uT_x + vT_z)^T
\end{aligned}
$$

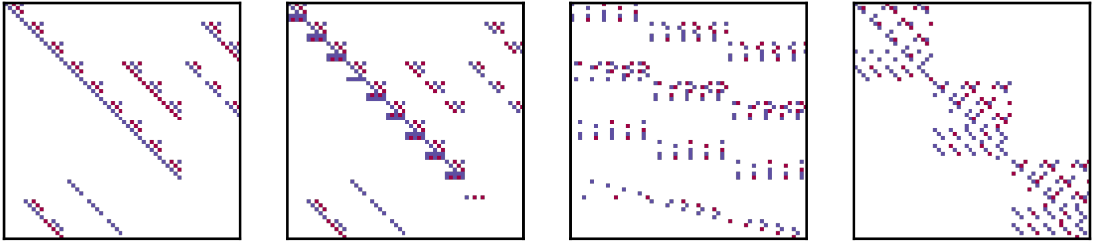
- M is not invertible $\rightarrow$ DAE
- $M + \Delta t L$ is invertible $\rightarrow$ no worries!
  $\rightarrow$ Need $\tau_M = 1$, though!
- Treat linear parts implicitly
- Treat convection explicitly

Resulting IMEX solver

$$(M + \Delta t L)\,\vec{u} = M\vec{u}_0 + \Delta t f_{\mathrm{nonlin}}(\vec{u}_0)$$

Resulting matrix for $N = 3 \times 5$

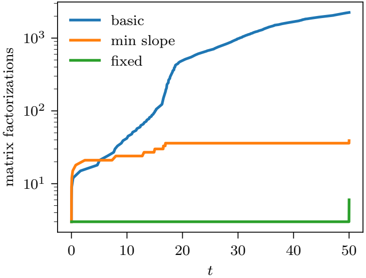
with preconditioners from Dedalus

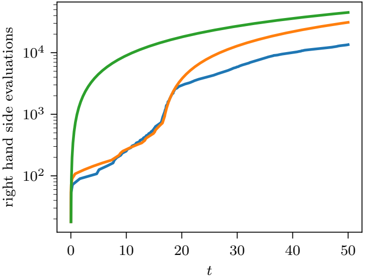


Left to right:

- $M + L$
- $+$ boundary conditions
- $+$ Dirichlet recombination: Right preconditioner
- $+$ "reverse Kronecker product": Sort by mode, not by component $\rightarrow$ left preconditioner

Adaptive step size selection

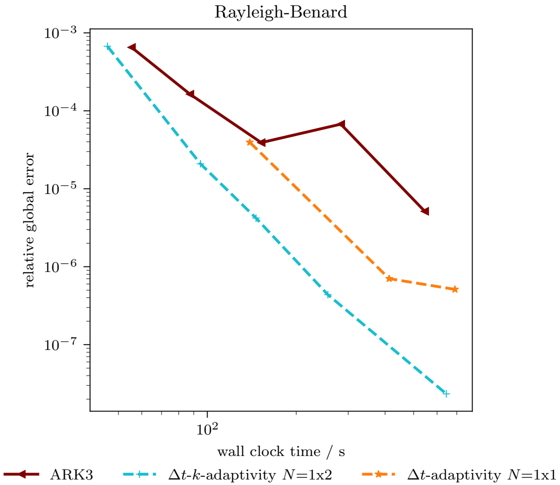




Avoiding LU factorizations via caching

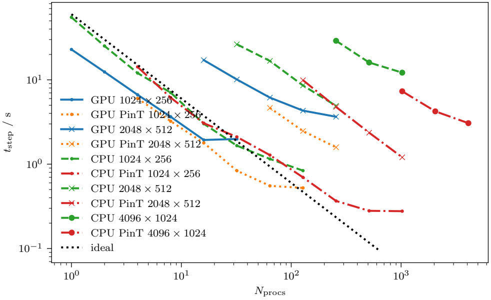
- Need to factorize when using new Δt
- Change step size only after exceeding relative threshold
- Round step size generously to increase cache hits after restart if close to stability limit

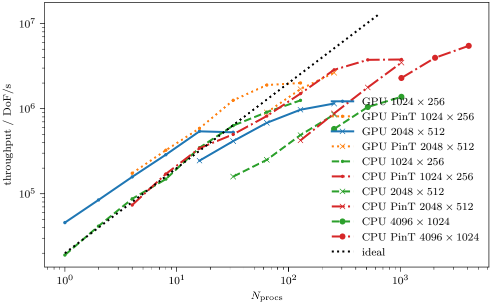
SDC is faster than DIRK for RBC



- Need globally stiffly accurate IMEX RKM for comparison
  $\rightarrow$ Best I could find is order 3
- SDC at order 3 on two tasks is much faster
- Can easily increase order of SDC to obtain even greater advantage

Parallel scaling of RBC implementation





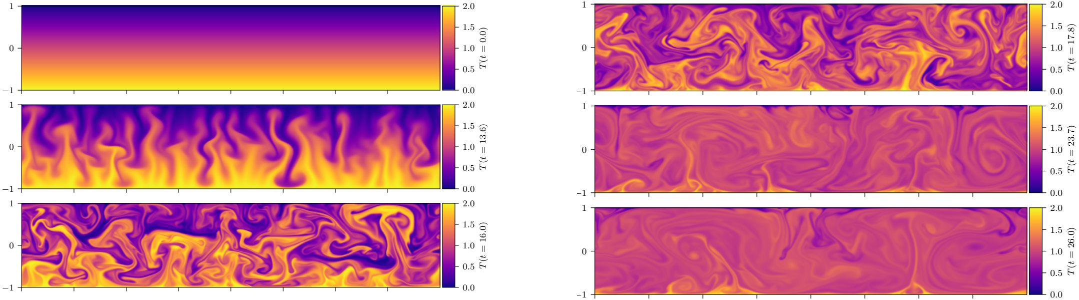
CPUs outperform GPUs

- GPUs perform better per task
- CPUs perform better per node
- LU takes most of the time!

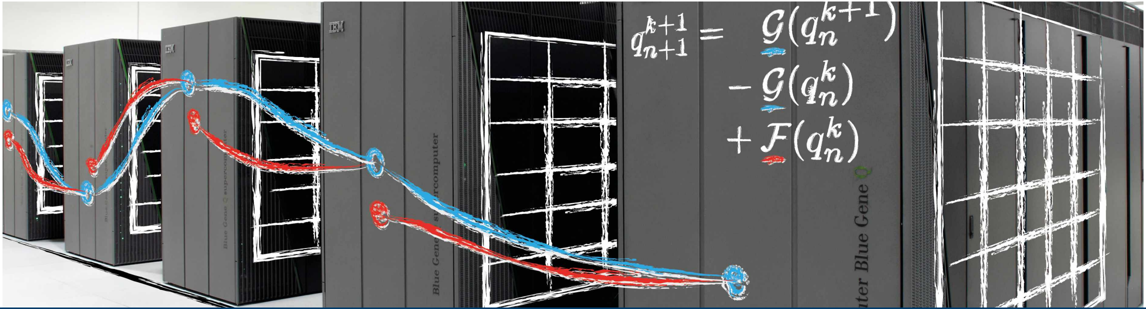
Use diagonal SDC to extend scaling

- Circumvents space-decomposition limit
- Improves strong scaling
- Enables scaling up to 4096 CPUs

Large space-time parallel Rayleigh-Benard simulation



- Use Rayleigh number 2×10^7 on $N = 4096 \times 1024$ grid
- Use Δt-k-adaptivity with four Gauß-Radau nodes for step size selection
- Use $4 \times 1024 = 4096$ CPUs on JURECA DC (32 nodes)
- Use approx. 140k CPU hours to reach $t = 26$ because IMEX stability limit requires $\Delta t \approx 10^{-3}$



SDC for Rayleigh-Benard convection with spectral methods

December 17, 2024 | Thomas Baumann | Jülich Supercomputing Centre