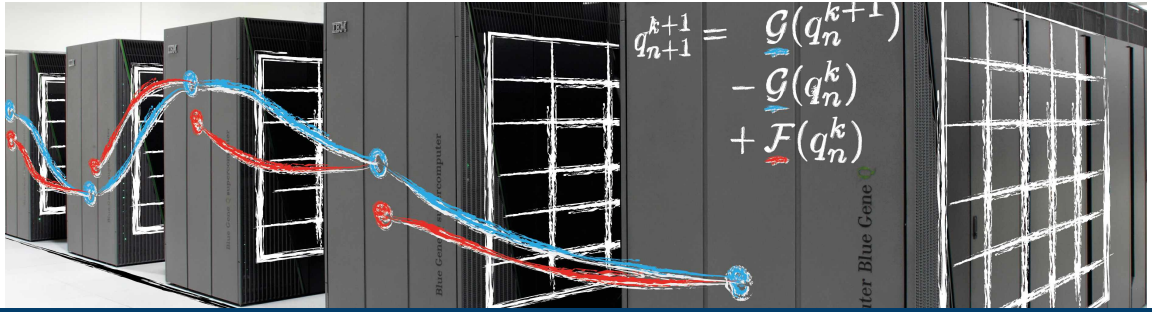


Diagonal spectral deferred correction for 3D Rayleigh-Benard convection

July 8, 2025 | Thomas Saupe, Thibaut Lunet, Sebastian Götschel, Daniel Ruprecht, Robert Speck | JSC & TUHH

Disclaimer: Very much work-in-progress

Rayleigh-Benard convection (RBC)

Equations

$$
\begin{aligned}
T_t - \kappa \Delta T &= -\vec{u} \nabla T \\
\vec{u}_t - \nu \Delta \vec{u} + \nabla p - T \vec{e}_z &= -\vec{u} \nabla \vec{u} \\
\nabla \vec{u} &= 0 \\
\Omega &= [0,1)^2 \times (0,1)
\end{aligned}
$$

Boundary conditions

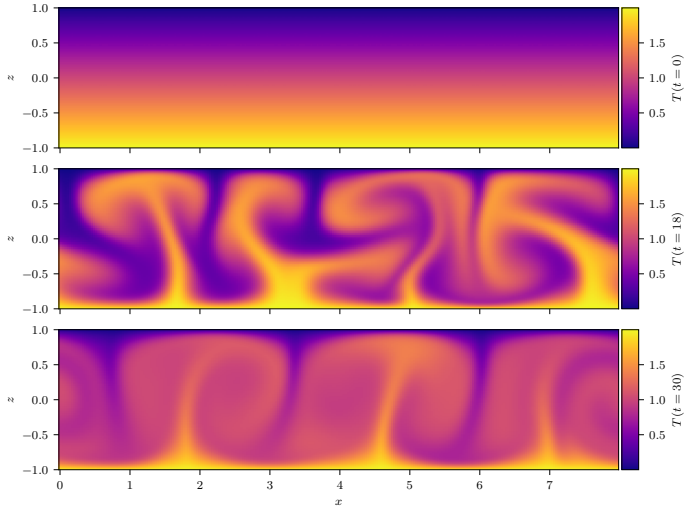
- $\vec{u}(z=0) = \vec{u}(z=1) = 0$
- $T(z=0) = 1,\ T(z=1) = 0$
- periodic in x and y

Dynamics mainly determined by Rayleigh number $Ra = \frac{\alpha g L_z^3 \Delta T}{\kappa \nu}$, turbulence develops from $Ra > 1709$

We pick $Ra > 10^7 \rightarrow$ complicated convection patterns $\rightarrow$ difficult to speed up with PinT

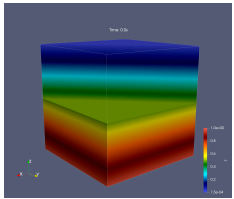
Dynamics in 2D

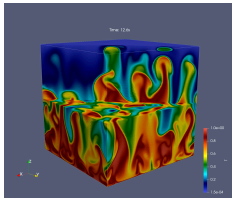
- Start with stable linear temperature gradient
- Perturb with random noise on the order of 10^{-3}
- Perturbations grow
- Flow settles into quasi-stationary rolling pattern

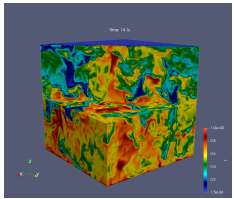


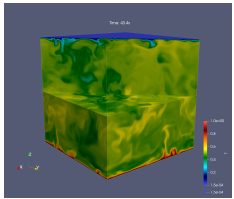
JÜLICH Forschungszentrum

Dynamics in 3D









Same thing, just looks more chaotic!

Spatial discretization and solver

- Use global spectral method with Fourier $\times$ Fourier $\times$ ultraspherical discretization
  $\rightarrow$ ultraspherical discretization from [Olver and Townsend, 2013] is fancy Chebychev discretization
- Treat convection ($F(\vec{u})$) explicitly and only linear terms implicitly:

$$\underbrace{\begin{pmatrix} I & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}}_{M} \begin{pmatrix} u \\ v \\ w \\ T \\ p \end{pmatrix}_t + \underbrace{\begin{pmatrix} -\nu\Delta & 0 & 0 & 0 & \partial_x \\ 0 & -\nu\Delta & 0 & 0 & \partial_y \\ 0 & 0 & -\nu\Delta & -I & \partial_z \\ 0 & 0 & 0 & -\kappa\Delta & 0 \\ \partial_x & \partial_y & \partial_z & 0 & 0 \end{pmatrix}}_{L} \begin{pmatrix} u \\ v \\ w \\ T \\ p \end{pmatrix} = F\left(\begin{pmatrix} u \\ v \\ w \\ T \\ p \end{pmatrix}\right)$$

- M is not invertible $\rightarrow$ differential algebraic equation (DAE)
- No worries: $M + \Delta t L$ is invertible $\rightarrow$ no DAE-specific method needed

Space-parallelism

Parallelize Fourier spectral methods

- Need distributed FFT $\rightarrow$ very common, no problem
- Differentiation matrices are diagonal $\rightarrow$ easily distributed

Parallelize ultraspherical spectral methods

- Need distributed DCT $\rightarrow$ express via FFTs, no problem
- Differentiation matrices are not diagonal $\rightarrow$ can't easily distribute inversion

Parallelize RBC

- Matrices are built from 1D matrices using Kronecker product
- Use pencil decomposition, aligned in z for matrix application
- FFTs require expensive all-to-all communication

Space-parallelism

Parallelize Fourier spectral methods

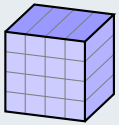
- Need distributed FFT $\rightarrow$ very common, no problem
- Differentiation matrices are diagonal $\rightarrow$ easily distributed

Parallelize ultraspherical spectral methods

- Need distributed DCT $\rightarrow$ express via FFTs, no problem
- Differentiation matrices are not diagonal $\rightarrow$ can't easily distribute inversion

Parallelize RBC

- Matrices are built from 1D matrices using Kronecker product
- Use pencil decomposition, aligned in z for matrix application
- FFTs require expensive all-to-all communication



Time discretization: Spectral deferred correction (SDC)

[Dutt et al., 2000, Weiser, 2015]

- Start with initial value problem

$$u(\Delta t) = \int_0^{\Delta t} f(u)dt + u(0)$$

- Choose M collocation nodes τ_m and discretize the integral with spectral quadrature rule Q

$$(I - \Delta t Q f)(\vec{u}) = \vec{u}_0,$$

  where $\vec{u} = (u(\tau_1), \dots u(\tau_M))$

- Solve iteratively with preconditioner Q_Δ

$$(I - \Delta t Q_\Delta f)(\vec{u}^{k+1}) = \vec{u}_0 + \Delta t(Q - Q_\Delta) f(\vec{u}^k)$$

See Thibaut's talk tomorrow morning for loads more detail!

Time discretization: Spectral deferred correction (SDC)

[Dutt et al., 2000, Weiser, 2015]

- Start with initial value problem

$$u(\Delta t) = \int_0^{\Delta t} f(u)dt + u(0)$$

- Choose M collocation nodes τ_m and discretize the integral with spectral quadrature rule Q

$$(I - \Delta t Q f)(\vec{u}) = \vec{u}_0,$$

  where $\vec{u} = (u(\tau_1), \dots u(\tau_M))$

- Solve iteratively with preconditioner Q_Δ

$$(I - \Delta t Q_\Delta f)(\vec{u}^{k+1}) = \vec{u}_0 + \Delta t(Q - Q_\Delta) f(\vec{u}^k)$$

See Thibaut's talk tomorrow morning for loads more detail!

Time parallelisation: Diagonal SDC

[van der Houwen and Sommeijer, 1991, Speck, 2018, Čaklović et al., 2024]

Q_Δ lower triangular

- Solve SDC iteration with forward substitution
- Can write as kM stage DIRK method

Q_Δ diagonal

- Removes coupling within SDC iteration
- Perform implicit solves and f evaluations in parallel
- Looks like k stage (S)DIRK method on each task

Butcher tableau for third order SDC

τ_1	$\tilde{q}_{11}$								
τ_2	$\tilde{q}_{21}$	$\tilde{q}_{22}$							
τ_3	$\tilde{q}_{31}$	$\tilde{q}_{32}$	$\tilde{q}_{33}$						
τ_1	a_{11}	a_{12}	a_{13}	$\tilde{q}_{11}$					
τ_2	a_{21}	a_{22}	a_{23}	$\tilde{q}_{21}$	$\tilde{q}_{22}$				
τ_3	a_{31}	a_{32}	a_{33}	$\tilde{q}_{31}$	$\tilde{q}_{32}$	$\tilde{q}_{33}$			
τ_1				a_{11}	a_{12}	a_{13}	$\tilde{q}_{11}$		
τ_2				a_{21}	a_{22}	a_{23}	$\tilde{q}_{21}$	$\tilde{q}_{22}$	
τ_3				a_{31}	a_{32}	a_{33}	$\tilde{q}_{31}$	$\tilde{q}_{32}$	$\tilde{q}_{33}$
	0	0	0	0	0	0	0	0	1

Time parallelisation: Diagonal SDC

[van der Houwen and Sommeijer, 1991, Speck, 2018, Čaklović et al., 2024]

Q_Δ lower triangular

- Solve SDC iteration with forward substitution
- Can write as kM stage DIRK method

Q_Δ diagonal

- Removes coupling within SDC iteration
- Perform implicit solves and f evaluations in parallel
- Looks like k stage (S)DIRK method on each task

Butcher tableau for third order diagonal SDC on three tasks

τ_1	$\tilde{q}_1$								
τ_2		$\tilde{q}_2$							
τ_3			$\tilde{q}_3$						
τ_1	a_{11}	a_{12}	a_{13}	$\tilde{q}_1$					
τ_2	a_{21}	a_{22}	a_{23}		$\tilde{q}_2$				
τ_3	a_{31}	a_{32}	a_{33}			$\tilde{q}_3$			
τ_1				a_{11}	a_{12}	a_{13}	$\tilde{q}_1$		
τ_2				a_{21}	a_{22}	a_{23}		$\tilde{q}_2$	
τ_3				a_{31}	a_{32}	a_{33}			$\tilde{q}_3$
	0	0	0	0	0	0	0	0	1

Measure PinT speedup

1. Establish that SDC is faster than time-serial reference method
2. Demonstrate that space-time-parallel scaling is better than space-only scaling
3. Implement in "production" code

Implement RBC in pySDC

- pySDC: Prototyping library for SDC and PFASST
- Demonstrate good performance of SDC for RBC
- Supports GPUs
- See [Speck, 2019]

Implement SDC in Dedalus

- Framework for spatial discretization with pseudo-spectral methods
- Demonstrate SDC implementation in existing code
- Is somewhat optimized
- See [Burns et al., 2020]

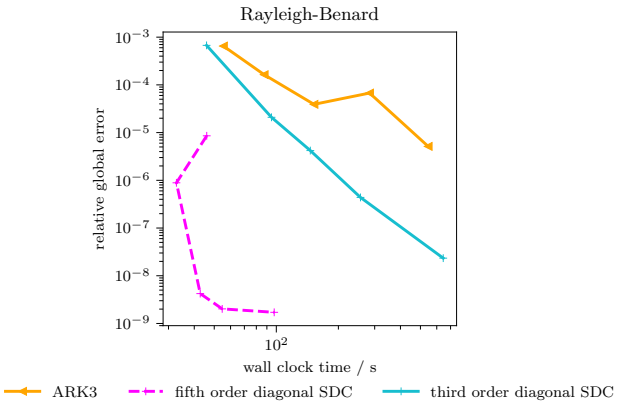
Comparison against DIRK-type method for 2D RBC

Reference method

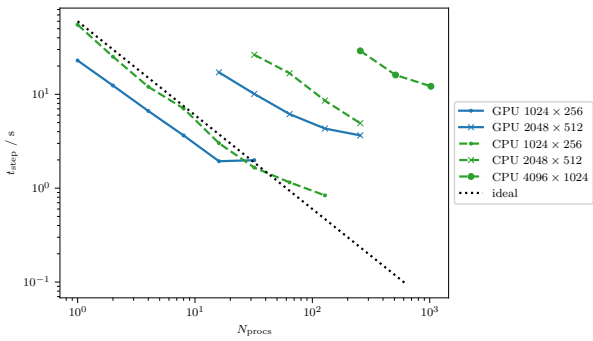
- Need stiffly accurate IMEX method to deal with algebraic constraint
- Best high-order method we could find: Four-stage, third order method ARK3 [Ascher et al., 1997, Section 2.8]

Comparison results

- Third order diagonal SDC on 2 tasks is faster than ARK3
- Increase order of SDC to 5 simply by adding one collocation node
- Fifth-order diagonal SDC on 3 tasks is much faster



3D scaling with pySDC on JURECA / JUWELS booster



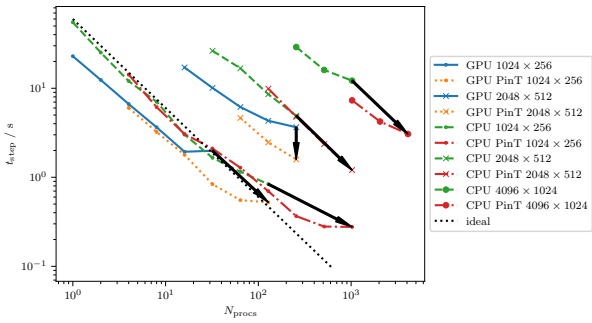
CPUs vs GPUs

- Sparse linear solves not accelerated on GPU
- FFTs faster on GPU

Use diagonal SDC to extend scaling

- Uses 4 tasks in time
- Circumvents space-decomposition limit
- Improves strong scaling
- Enables scaling up to 4096 CPUs

3D scaling with pySDC on JURECA / JUWELS booster



CPUs vs GPUs

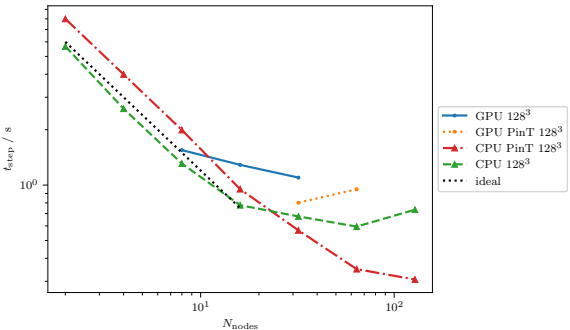
- Sparse linear solves not accelerated on GPU
- FFTs faster on GPU

Use diagonal SDC to extend scaling

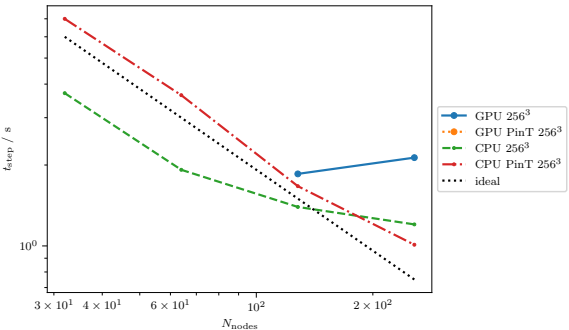
- Uses 4 tasks in time
- Circumvents space-decomposition limit
- Improves strong scaling
- Enables scaling up to 4096 CPUs

3D scaling with pySDC on JURECA / JUWELS booster

$N = 128^3$



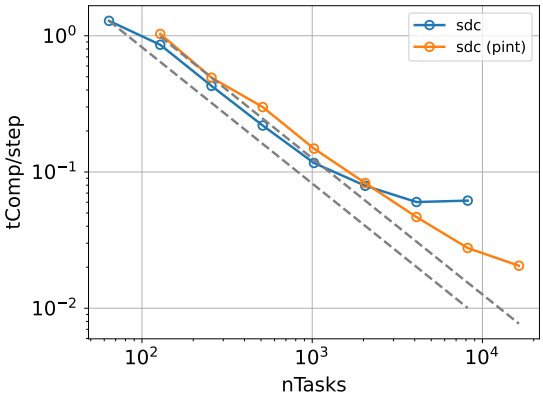
$N = 256^3$



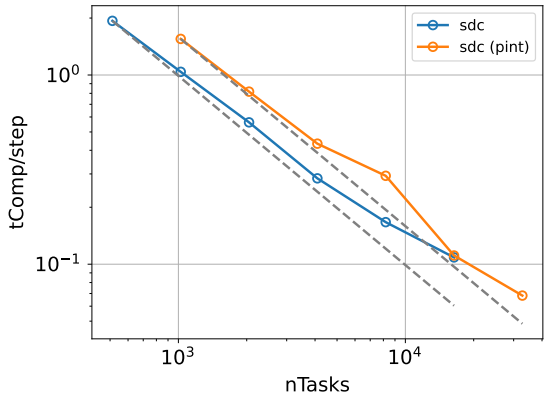
- PinT can again be used to extend scaling limits
- Don't yet know why PinT is slower for few tasks
- GPU implementation suffers from huge memory footprint

3D parallel scaling with Dedalus on JUWELS cluster

$N = 128^3$



$N = 256^3$



JÜLICH Forschungszentrum

Next stop: Jupiter

Implementation

- Reduce memory footprint
- Improve GPU performance

Measurements

- Measure performance for $N = 512^3$
- Measure performance on JUPITER

Conclusions

- Diagonal preconditioners from Gaya and Thibaut ([Čaklović et al., 2024]) are excellent
- Diagonal SDC can extend parallel scaling limits
- No coarsening involved
- Works even for complicated benchmark with heavy convection

pySDC

SCAN ME

Sources I

Ascher, U. M., Ruuth, S. J., and Spiteri, R. J. (1997).

Implicit-explicit runge-kutta methods for time-dependent partial differential equations.
Applied Numerical Mathematics, 25(2):151–167.
Special Issue on Time Integration.

Burns, K. J., Vasil, G. M., Oishi, J. S., Lecoanet, D., and Brown, B. P. (2020).

Dedalus: A flexible framework for numerical simulations with spectral methods.
Physical Review Research, 2(2):023068.

Čaklović, G., Lunet, T., Götschel, S., and Ruprecht, D. (2024).

Improving efficiency of parallel across the method spectral deferred corrections.

Dutt, A., Greengard, L., and Rokhlin, V. (2000).

Spectral deferred correction methods for ordinary differential equations.
BIT Numerical Mathematics, 40(2):241–266.

Olver, S. and Townsend, A. (2013).

A fast and well-conditioned spectral method.
SIAM Review, 55(3):462–489.

Speck, R. (2018).

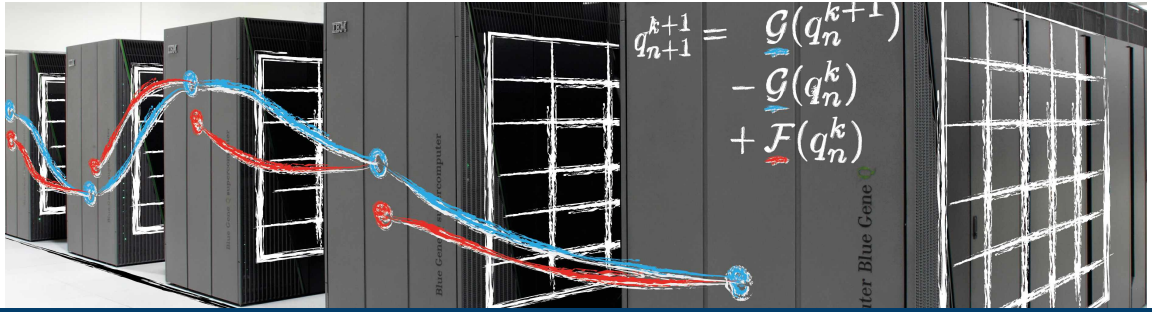
Parallelizing spectral deferred corrections across the method.
Computing and Visualization in Science, 19(3):75–83.

Sources II

Speck, R. (2019).
Algorithm 997: pySDC—Prototyping spectral deferred corrections.
ACM Trans. Math. Softw., 45(3).

van der Houwen, P. J. and Sommeijer, B. P. (1991).
Iterated runge–kutta methods on parallel computers.
SIAM Journal on Scientific and Statistical Computing, 12(5):1000–1028.

Weiser, M. (2015).
Faster SDC convergence on non-equidistant grids by DIRK sweeps.
BIT Numerical Mathematics, 55(4):1219–1241.



Diagonal spectral deferred correction for 3D Rayleigh-Benard convection

July 8, 2025 | Thomas Saupe, Thibaut Lunet, Sebastian Götschel, Daniel Ruprecht, Robert Speck | JSC & TUHH