Advances on an asymptotic parallel-in-time algorithm

Terry Haut and Beth Wingate and Gunnar Martinnson

Outline

Time-stepping method for PDEs that exhibit highly oscillatory time scales

a) Motivation and background

b) Brief recap of asymptotic parallel-in-time method

Key technical component: applying the exponential e^{tL} $(L^* = -L)$

a) Standard methods and their limitations

b) New parallel-in-time method

c) Examples on 2D shallow water equations with spectral element discretization

Motivation for time parallelism

- Future trend: more processors available than can be efficiently used by spatial parallelization alone
- Once gains from spatial refinement are saturated, higher processor counts will not increase speed
- For problems with fast temporal oscillations, standard methods generally require small time steps
- Time-stepping constraints (small time steps, lack of time parallelization) represent a fundamental bottleneck

Model equations

- Focus on PDEs of the form

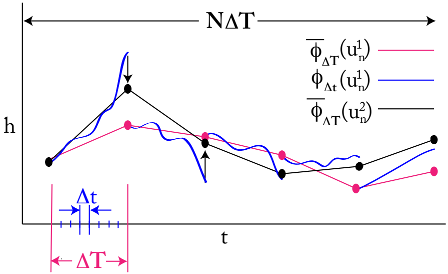
$$\frac{\partial \mathbf{u}}{\partial t} + \varepsilon^{-1} L\mathbf{u} = N(\mathbf{u}) + D\mathbf{u},$$

  where $L^* = -L$; L has pure imaginary eigenvalues
- Includes the primitive equations, Boussinesq equations, etc.
- $\varepsilon^{-1}L$ results in rapid time oscillations (think of $e^{i\omega t/\varepsilon}$)
- Generally, even implicit and linearly exact methods require $\Delta t = \mathscr{O}(\varepsilon)$ (exception: $\|L\mathbf{u}\| \ll 1$)

Asymptotic parallel-in-time method

- Take many big time steps $n\Delta T$, $n = 1, 2, \dots, N$ on an asymptotic approximation $(\Delta T \gg \varepsilon)$
- Refine the solution *in parallel* on $[n\Delta T, (n+1)\Delta T]$ using small time steps Δt on the full equation

Figure 1 : Schematic of parallel-in-time algorithm



Applying the operator exponential

- Asymptotic parallel-in-time method extends to general domains if $e^{tL}\mathbf{u}_0$ can be applied efficiently (here $L^* = -L$)
- Developed a method for applying $e^{tL}\mathbf{u}_0$ (with Gunnar Martinnson and Beth Wingate)

The exponential of a Skew-Hermitian matrix

- From eigenvalue decomposition $L = U\,(i\Omega)\,U^*$,

$$e^{tL} = U \begin{pmatrix} e^{i\omega_1 t} & 0 & \cdots & 0 \\ 0 & e^{i\omega_1 t} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & e^{i\omega_N t} \end{pmatrix} U^*$$

- Approximation $R_M\,(ix)$ of e^{ix} yields approximation of e^{tL},

$$\left\| e^{tL} - R_M\,(tL) \right\|_2 = \max_{1 \le k \le N} \left| e^{i\omega_k t} - R_M\,(i\omega_k) \right|$$

Standard methods for operator exponential

- Standard methods build polyomial or rational approximations $R_M(tL)$ to e^{tL} iteratively
- Forward Euler: $R_M(tL)\,\mathbf{u}_0 = (\Delta tL + I)^M\,\mathbf{u}_0,\ t = M\Delta t$
- Backward Euler: $R_M(tL)\,\mathbf{u}_0 = (-\Delta tL + I)^{-M}\,\mathbf{u}_0,\ t = M\Delta t$
- Other common approaches: Krylov methods, scaling and squaring, Chebyshev polynomials, etc.
- All of these methods rely on polynomial or rational approximations that are inherently serial

Optimal rational approximations for exponential

- Construct (near) optimal rational approximation, $R_M(ix)$, to e^{ix} on the interval $-t|\omega_N| \leq x \leq t|\omega_N|$ with ε error
- Leads to approximation of e^{tL},

$$\left\| e^{tL}\mathbf{u}_0 - \sum_{m=1}^{M} a_m (tL - \alpha_m)^{-1}\mathbf{u}_0 \right\|_2 \leq \varepsilon \|\mathbf{u}_0\|_2 + 2\left\| P_{\omega_{M+1}/t}\mathbf{u}_0 - \mathbf{u}_0 \right\|_2.$$

- The inverses $(tL - \alpha_m)^{-1}\mathbf{u}_0$ can be applied in parallel
- Near optimality even when $t|\omega_N| \gg 1$; can parallelize over many characteristic wavelengths

Rational approximations, continued

- Rational approximation $R_M(ix)$ has (near) optimally small error in the L^∞ norm (with Beylkin et al, (2013))
- This results in high efficiency relative to standard methods
- The same poles (and inverses) for $e^{tL}\mathbf{u}_0$ can be used to apply $e^{sL}\mathbf{u}_0$ for all $0 \leq s \leq t$
- Can apply e^{tL} with $M \ll |t\omega_N|$ terms if there is a priori knowledge of spectral gaps
- Also works for general functions $f(tL)$ (e.g. exponential integrators, filters, etc.)

Applying the inverse

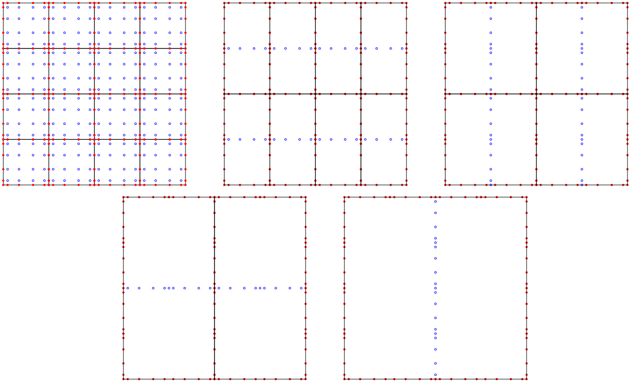
- Need to apply $(tL - \alpha_m)^{-1}\mathbf{u}_0$; typically reduces to an elliptic solve in one variable
- For e.g. RSW equations (with constant Coriolis term f), can be reduced to applying $\left(\Delta - \left(\alpha_m^2 + f^2\right)/c^2\right)^{-1}$, $c^2 = gH$
- Since the shift α_m is complex-valued, multigrid should efficient
- Take another approach: precompute an efficient direct solver (Martinsson, 2012)

Direct solver

- Spatial discretization uses the spectral element method
- Direct solver related to Nested Dissection, i.e. "Gaussian elimination for sparse matrices"
- In 2D, can apply $(tL - \alpha_m)^{-1} \mathbf{u}_0$ in $\mathcal{O}(N \log(N))$ operations
- In 2D experiments, time for applying $(tL - \alpha_m)^{-1}$ is $4 - 5$ times more expensive than applying L
- In theory, can be accelerated to $\mathcal{O}(N)$ operations in 2D and 3D

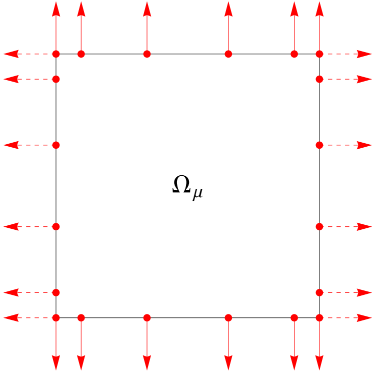
Flow of direct solver

Figure 2 : Interior variables are eliminated and boxes merged (top to bottom, left to right)



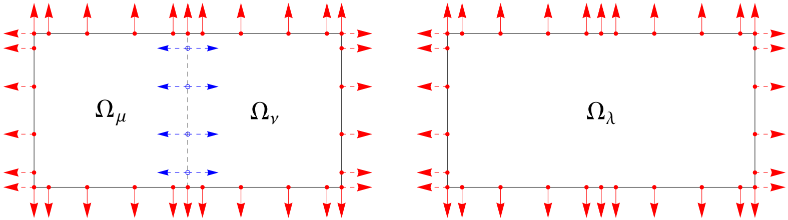
Dirichlet-to-Neumann operator

Uses Dirichlet-to-Neumann (DtN) matrices, which are pre-computed hierarchically



Eliminating variables hierarchically

1. Eliminate internal variables (blue points) by using continuity of the normal derivative
2. "Merge" DtN matrices of neighboring boxes Ω_μ and Ω_λ to get DtN matrix for parent box Ω_λ



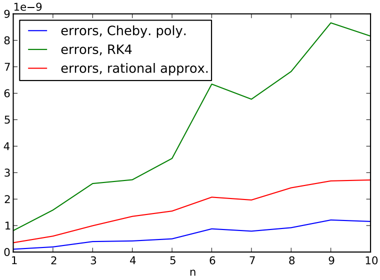
Numerical expample for matrix exponential

- Apply to 2D shallow water equations on $[0,1] \times [0,1]$
- Test $e^{ntL}\mathbf{u}_0$ with $t = 1.5$, $n = 1, \ldots, 10$; compare against RK4 and the use of Chebyshev polynomials
- $6 \times 6 = 36$ elements, with 16×16 quadrature nodes per element (same for all three methods)
- Use 379 inverses $(tL - \alpha_m)^{-1}$ for $R_M(tL)\mathbf{u}_0$
- Similar relative speeds obtained with $12 \times 12 = 144$ elements, with 16×16 quadrature nodes per element

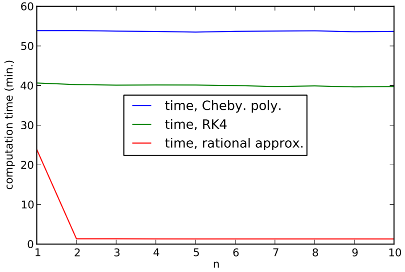
Numerical experiment, continued

Figure 3 : (a) Errors, $\left\|\mathbf{u}(nt) - e^{ntL}\mathbf{u}_0\right\|_\infty$, $1 \leq n \leq 10$ from three methods (a) Timings for three methods

(a)

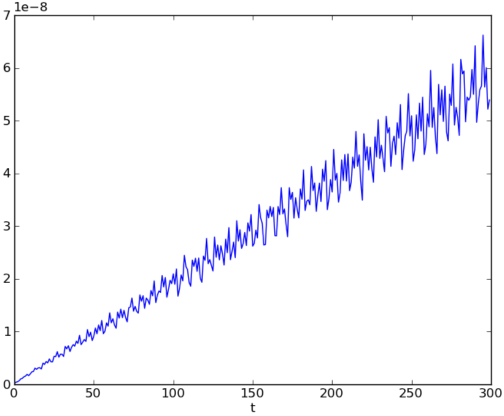


(b)



Numerical experiment: long time integrator

Figure 4 : Errors, $\left\|\mathbf{u}(nt) - e^{ntL}\mathbf{u}_0\right\|_\infty$, $t = 1.5$ and $1 \leq n \leq 300$ using (near) optimal rational approximations



Summary

- New method for applying operator exponential e^{tL}
- Is close to optimal among all rational or polynomial approximations
- Can be parallelized over as many characteristic wavelengths as resources permit
- Can take advantage of known scale separation between fast and slow waves
- Generalizes to (near) optimal rational approximations of $f(tL)$

Optimal approximations of operator exponential

- For skew-Hermitian operator $\mathscr{L}$,

$$\left\| e^{\tau \mathscr{L}} \mathbf{u}_0 - \sum_{m=-M}^{M} a_m \left(\tau \mathscr{L} - \alpha_m\right)^{-1} \mathbf{u}_0 \right\|_2 \leq \varepsilon \left\| \mathbf{u}_0 \right\|_2 + 2 \left\| P_\Lambda \mathbf{u}_0 - \mathbf{u}_0 \right\|,$$

  where P_Λ denotes projection associated with eigenvalues less than or equal to Λ.

- Spatial discretization L of $\mathscr{L}$ may not be skew-Hermitian, but above bound leads to ε accuracy (up to spatial discretization)

Notation for parallel-in-time algorithm

- Write $\mathbf{U}_n = \mathbf{u}(n\Delta T)$
- Let $\varphi_{\Delta T}(\mathbf{u}_0)$ and $\overline{\varphi}_{\Delta T}(\mathbf{u}_0)$ denote results of evolving the full equation and the asymptotic equation by a time step ΔT
- So $\mathbf{U}_n = \varphi_{\Delta T}(\mathbf{U}_{n-1})$ and

$$\|\varphi_{\Delta T}(\mathbf{U}_n) - \overline{\varphi}_{\Delta T}(\mathbf{U}_n)\| = \mathcal{O}(\varepsilon).$$

Parallel in time algorithm

- Use a variation of the so-called parareal method
- Rewrite $\mathbf{U}_n = \varphi_{\Delta T}(\mathbf{U}_{n-1})$ as

$$\mathbf{U}_n = \overline{\varphi}_{\Delta T}(\mathbf{U}_{n-1}) + (\varphi_{\Delta T}(\mathbf{U}_n) - \overline{\varphi}_{\Delta T}(\mathbf{U}_n)).$$

- Solve iteratively: if $\mathbf{U}_n^k \approx \varphi_{\Delta T}(\mathbf{U}_{n-1})$ approximation at iteration k, then more accurate approx. $\mathbf{U}_n^{k+1}$ given by

$$\mathbf{U}_n^{k+1} = \overline{\varphi}_{\Delta T}\left(\mathbf{U}_{n-1}^{k+1}\right) + \left(\varphi_{\Delta T}\left(\mathbf{U}_n^k\right) - \overline{\varphi}_{\Delta T}\left(\mathbf{U}_n^k\right)\right), \quad n = 1, \ldots, N.$$

- At level $k+1$, expensive $\varphi_{\Delta T}\left(\mathbf{U}_n^k\right)$ can be computed *in parallel* using small time steps $\Delta t \ll \Delta T$ on time intervals $[n\Delta T, (n+1)\Delta T]$.

Shallow water equations

The 1D shallow water equations:

$$
\begin{aligned}
\frac{\partial v_1}{\partial t} + \frac{1}{\varepsilon}\left(-v_2 + \frac{\partial \eta}{\partial x}\right) + v_1 \frac{\partial v_1}{\partial x} &= \mu \frac{\partial^4 v_1}{\partial x^4}, \\
\frac{\partial v_2}{\partial t} + \frac{1}{\varepsilon} v_1 + v_1 \frac{\partial v_2}{\partial x} &= \mu \frac{\partial^4 v_2}{\partial x^4}, \\
\frac{\partial \eta}{\partial t} + \frac{1}{\varepsilon}\frac{\partial v_1}{\partial x} + \frac{\partial}{\partial x}(h v_1) &= \mu \frac{\partial^4 \eta}{\partial x^4},
\end{aligned}
$$

Asymptotic solution

- The solution has a slowly varying asymptotic approximation (Majda et al., 1998):

$$\mathbf{u}(t) = e^{-(t/\varepsilon)L}\overline{\mathbf{u}}(t) + \mathscr{O}(\varepsilon),$$

where

$$\frac{\partial \overline{\mathbf{u}}}{\partial t} = \overline{N}(\overline{\mathbf{u}}) + \overline{D}\overline{\mathbf{u}}, \;\; \overline{\mathbf{u}}(0) = \mathbf{u}_0.$$

Here e.g.

$$\overline{N}(\overline{\mathbf{u}}(t)) = \lim_{T\to\infty} \frac{1}{T}\int_0^T e^{sL} N\left(e^{-sL}\overline{\mathbf{u}}(t)\right) ds.$$

- Since $\partial_t^p \overline{\mathbf{u}} = \mathscr{O}(1)$ for all p, can take large time steps $\Delta T \gg \varepsilon$

Computing the asymptotic approximation numerically

- Use a variation of HMM (E and Engquist, 2003) to compute the asymptotic approximation *on the fly*:

$$\begin{aligned}\overline{N}(\overline{\mathbf{u}}(t)) &= \lim_{T\to\infty} \frac{1}{T}\int_0^T e^{sL} N\left(e^{-sL}\overline{\mathbf{u}}(t)\right) ds \\ &\approx \frac{1}{M}\sum_{m=1}^{M} \rho\left(\frac{s_m}{\varepsilon T_0}\right) e^{(s_m/\varepsilon)L} N\left(e^{-(s_m/\varepsilon)L}\overline{\mathbf{u}}(t)\right)\end{aligned}$$

- M is (essentially) independent of ε
- The time average can be computed in parallel
- Subtle point: local time average yields accuracy even when $\varepsilon = \mathcal{O}(1)$

Intuition behind HMM

- Expand $\mathbf{u}(x,t)$ in basis of eigenvectors $\mathbf{u}_k(x)$ of $\mathscr{L}$ (corresponding to eigenvalues $i\omega_k$), so

$$e^{-s\mathscr{L}}\mathbf{u}(x,t) = \sum_k e^{-i\omega_k s} c_k(t)\,\mathbf{u}_k(x).$$

- Then can (in theory) expand nonlinear term

$$e^{s\mathscr{L}}\mathscr{N}\left(e^{-s\mathscr{L}}\mathbf{u}(t)\right) = \sum_{\lambda_n} e^{i\lambda_n s}\mathscr{N}_n(\mathbf{u}(t)),$$

where $i\lambda_n$ is some linear combinations of the eigenvalues $i\omega_k$

- Therefore,

$$\lim_{T\to\infty}\frac{1}{T}\int_0^T e^{s\mathscr{L}}\mathscr{N}\left(e^{-s\mathscr{L}}\mathbf{u}(t)\right)ds = \sum_{\lambda_n=0}\mathscr{N}_n(\mathbf{u}(t)).$$

Intuition behind HMM

- Want to choose T_0 and $\rho(s)$ such that

$$\frac{1}{T_0}\int_0^{T_0}\rho\left(\frac{s}{T_0}\right)e^{s\mathscr{L}}\mathscr{N}\left(e^{-s\mathscr{L}}\mathbf{u}(t)\right)ds =$$
$$\sum_{\lambda_n}\mathscr{N}_n(\mathbf{u}(t))\int_0^1 e^{iT_0\lambda_n s}\rho(s)\,ds \approx$$
$$\sum_{\lambda_n=0}\mathscr{N}_n(\mathbf{u}(t)) =$$
$$\lim_{T\to\infty}\frac{1}{T}\int_0^T e^{s\mathscr{L}}\mathscr{N}\left(e^{-s\mathscr{L}}\mathbf{u}(t)\right)ds.$$

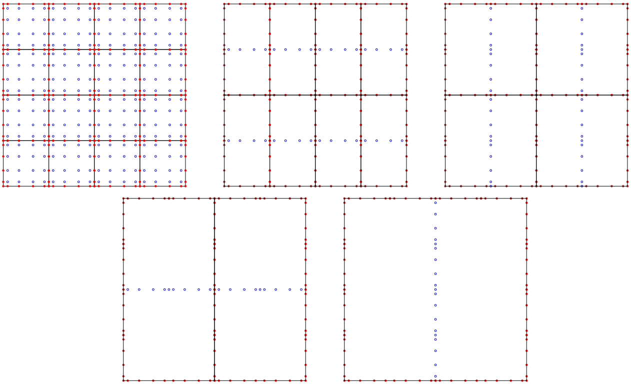
- Therefore, need

$$\int_0^1 e^{iT_0\lambda_n s}\rho(s)\,ds \approx 0, \text{ if } \lambda_n \neq 0.$$

- Repeated integration by parts shows that above integral is smaller than T_0^{-m} for any m

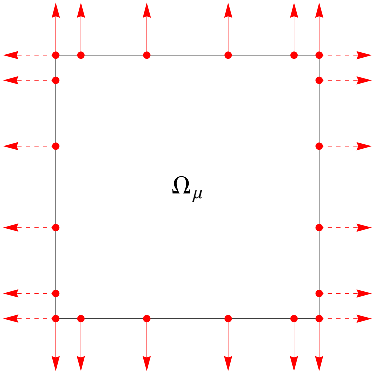
Flow of direct solver

Figure 5 : Interior variables are eliminated and boxes merged (top to bottom, left to right)



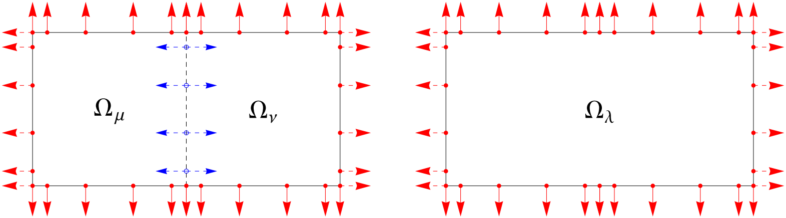
Dirichlet-to-Neumann operator

Uses Dirichlet-to-Neumann (DtN) matrices, which are pre-computed hierarchically



Eliminating variables hierarchically

1. Eliminate internal variables (blue points) by using continuity of the normal derivative
2. "Merge" DtN matrices of neighboring boxes Ω_μ and Ω_λ to get DtN matrix for parent box Ω_λ



2D shallow water equations

Solve 2D shallow water equations,

$$\begin{pmatrix} u_t \\ v_t \\ \eta_t \end{pmatrix} = \begin{pmatrix} 0 & f & \partial_x \\ -f & 0 & \partial_y \\ \partial_x & \partial_y & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ \eta \end{pmatrix}.$$

References