Flexible time integration methods for multiphysics PDE systems

Daniel R. Reynolds[1], Rujeko Chinomona[2], David J. Gardner[3], Carol S. Woodward[3], Cody J. Balos[3]

reynolds@smu.edu, rujeko.chinomona@temple.edu, gardner48@llnl.gov, woodward6@llnl.gov, balos1@llnl.gov

[1]Department of Mathematics, Southern Methodist University
[2]Department of Mathematics, Temple University
[3]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory

Mathematics in Computation (MiC) Seminar
Oak Ridge National Laboratory
7 April 2022



Outline



Outline

Multiphysics Simulations [Keyes et al., 2013]

Multiphysics simulations couple different models either in the bulk or across interfaces.

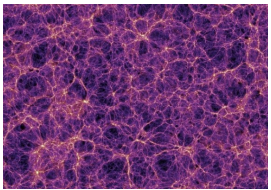
Climate:

- Atmospheric simulations combine fluid dynamics with local "physics" models for chemistry, condensation, ...
- Atmosphere is coupled at interfaces to myriad other processes (ocean, land/sea ice, ...), each using distinct models.

[https://e3sm.org]

Astrophysics/cosmology:

- Dark matter modeled using particles that give rise to large-scale gravitational structures (at right).
- Baryonic matter modeled by combining fluid dynamics, gravity, radiation transport, and reaction networks for chemical ionization states.



[http://svs.gsfc.nasa.gov/cgi-bin/details.cgi?aid=10118]

Multiphysics Challenges [Keyes et al., 2013]

These model combinations can challenge traditional numerical methods:

- "Multirate" processes evolve on different time scales but prohibit analytical reformulation.
- Stiff components disallow fully explicit methods.
- Nonlinearity and insufficient differentiability challenge fully implicit methods.
- Parallel scalability demands optimal algorithms – while robust/scalable algebraic solvers exist for parts (e.g., FMM for particles, multigrid for diffusion), none are optimal for the whole.

We may consider a prototypical problem as having m coupled evolutionary processes:

$$\dot{y}(t) = f^{\{1\}}(t,y) + \cdots + f^{\{m\}}(t,y), \quad t \in [t_0, t_f], \quad y(t_0) = y_0.$$

Each component $f^{\{k\}}(t,y)$:

- may act on all of y (in the bulk), or on only a subset of y (within a subdomain),
- may evolve on a different characteristic time scale,
- may be "stiff" or "nonstiff," thereby desiring implicit or explicit treatment.

Legacy Multiphysics Method 1: Lie–Trotter

"Operator-splitting" approaches have historically been used for multiphysics applications.

Lie–Trotter computes $y_n \to y_{n+1}$ (here, $y_n \approx y(t_n)$) via

$$\begin{aligned}
\dot{y}^{\{1\}}(t) &= f^{\{1\}}\left(t, y^{\{1\}}\right), && t \in [t_n, t_{n+1}], && y^{\{1\}}(t_n) = y_n, \\
\dot{y}^{\{2\}}(t) &= f^{\{2\}}\left(t, y^{\{2\}}\right), && t \in [t_n, t_{n+1}], && y^{\{2\}}(t_n) = y^{\{1\}}(t_{n+1}), \\
&\vdots \\
\dot{y}^{\{m\}}(t) &= f^{\{m\}}\left(t, y^{\{m\}}\right), && t \in [t_n, t_{n+1}], && y^{\{m\}}(t_n) = y^{\{m-1\}}(t_{n+1}),
\end{aligned}$$

and sets $y_{n+1} = y^{\{m\}}(t_{n+1})$. Each IVP is tackled independently using different "standard" approaches (e.g., implicit Euler, ERK-4, subcycling, ...).

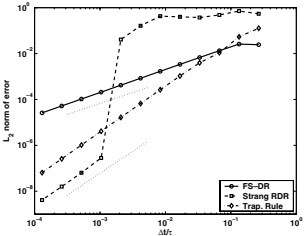
Legacy Multiphysics Method 2: Strang–Marchuk

$$
\begin{aligned}
\dot{y}^{\{1\}}(t) &= f^{\{1\}}\left(t, y^{\{1\}}\right), && t \in [t_n, t_{n+1/2}], && y^{\{1\}}(t_n) = y_n, \\
&\vdots \\
\dot{y}^{\{m-1\}}(t) &= f^{\{m-1\}}\left(t, y^{\{m-1\}}\right), && t \in [t_n, t_{n+1/2}], && y^{\{m-1\}}(t_n) = y^{\{m-2\}}(t_{n+1/2}), \\
\dot{y}^{\{m\}}(t) &= f^{\{m\}}\left(t, y^{\{m\}}\right), && t \in [t_n, t_{n+1}], && y^{\{m\}}(t_n) = y^{\{m-1\}}(t_{n+1/2}), \\
\dot{y}^{\{m-1\}}(t) &= f^{\{m-1\}}\left(t, y^{\{m-1\}}\right), && t \in [t_{n+1/2}, t_{n+1}], && y^{\{m-1\}}(t_{n+1/2}) = y^{\{m\}}(t_{n+1}), \\
&\vdots \\
\dot{y}^{\{1\}}(t) &= f^{\{1\}}\left(t, y^{\{1\}}\right), && t \in [t_{n+1/2}, t_{n+1}], && y^{\{1\}}(t_{n+1/2}) = y^{\{2\}}(t_{n+1}), \\
y_{n+1} &= y^{\{1\}}(t_{n+1}).
\end{aligned}
$$

Shorcomings of loose "initial condition" coupling

Generally poor accuracy:

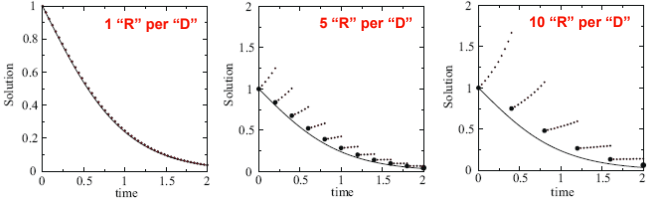
- *No matter the accuracy of each sub-solver*, Lie–Trotter is at best $\mathcal{O}(H)$ and Strang–Marchuk is $\mathcal{O}(H^2)$.
- Extrapolation or deferred correction can improve this but at significant cost.



Convergence of splitting approaches (brusselator) [Ropp & Shadid, 2005].

Poor stability:

- Even "stable" step sizes for each part can result in unstable modes.



Subcycling stability (reaction-diffusion) [Estep et al., 2008].

Outline

Outline

Additive Runge–Kutta (ARK) Methods [Ascher et al., 1997; Araújo et al., 1997; Kennedy & Carpenter, 2003; ...]

ARK methods allow high-order adaptive implicit-explicit time integration for additively-split *single rate* simulations:

$$M(t)\,\dot{y}(t) = f^E(t,y) + f^I(t,y), \quad t \in [t_0, t_f], \quad y(t_0) = y_0,$$

- M is any nonsingular linear operator (mass matrix, typically $M = I$, as used below),
- $f^E(t,y)$ contains the nonstiff terms to be treated explicitly,
- $f^I(t,y)$ contains the stiff terms to be treated implicitly.

Combine two s-stage RK methods; denoting $h_n = t_{n+1} - t_n$, $t^E_{n,j} = t_n + c^E_j h_n$, $t^I_{n,j} = t_n + c^I_j h_n$:

$$z_i = y_n + h_n \sum_{j=1}^{i-1} a^E_{i,j} f^E(t^E_{n,j}, z_j) + h_n \sum_{j=1}^{i} a^I_{i,j} f^I(t^I_{n,j}, z_j), \quad i = 1, \ldots, s,$$

$$y_{n+1} = y_n + h_n \sum_{j=1}^{s} \left[b^E_j f^E(t^E_{n,j}, z_j) + b^I_j f^I(t^I_{n,j}, z_j) \right] \quad \text{(solution)}$$

$$\tilde{y}_{n+1} = y_n + h_n \sum_{j=1}^{s} \left[\tilde{b}^E_j f^E(t^E_{n,j}, z_j) + \tilde{b}^I_j f^I(t^I_{n,j}, z_j) \right] \quad \text{(embedding)}$$

Solving each stage z_i, $i = 1, \ldots, s$

Per-stage cost is commensurate with implicit Euler for $\dot{y}(t) = f^I(t, y)$ – solve a root-finding problem:

$$0 = G_i(z) = \left[z - h_n a^I_{i,i} f^I(t^I_{n,i}, z)\right] - \left[y_n + h_n \sum_{j=1}^{i-1} \left(a^E_{i,j} f^E(t^E_{n,j}, z_j) + a^I_{i,j} f^I(t^I_{n,j}, z_j)\right)\right]$$

- If $f^I(t, y)$ is *linear* in y then this is a large-scale linear system for each z_i.
- Else this requires an iterative solver (e.g., Newton, accelerated fixed-point, or problem-specific).
- All operators in $f^E(t, y)$ are treated explicitly (do not affect algebraic solvers).

ARK methods are defined by compatible *explicit* $\left\{c^E, A^E, b^E, \tilde{b}^E\right\}$ and *implicit* $\left\{c^I, A^I, b^I, \tilde{b}^I\right\}$ tables.

- Derived in unison to satisfy order conditions arising from NB-trees.
- By selecting $A^I = 0$ and $f^I(t, y) = 0$, or $A^E = 0$ and $f^E(t, y) = 0$, ARK methods reduce to standard ERK or DIRK.

Outline

Multirate Infinitesimal Step (MIS/MRI) methods [Schlegel et al., 2009; Sandu, 2019; ...]

MRI methods arose in the numerical weather prediction community. This generic infrastructure supports up to $\mathcal{O}(h^4)$ methods for multirate problems:

$$\dot{y}(t) = f^S(t,y) + f^F(t,y), \quad t \in [t_0, t_f], \quad y(t_0) = y_0.$$

- $f^S(t,y)$ contains the "slow" dynamics, evolved with time step H.
- $f^F(t,y)$ contains the "fast" dynamics, evolved with time steps $h \ll H$.
- The slow component is defined by an "outer" RK method, while the fast component is advanced between slow stages by solving a modified IVP with a subcycled "inner" RK method.
- Extremely efficient – high order attainable with *only a single traversal* of $[t_n, t_{n+1}]$.

MIS/MRI Algorithm [Schlegel et al., 2009; Sandu, 2019; . . .]

Denoting $y_n \approx y(t_n)$ and $H = t_{n+1} - t_n$, a single step $y_n \to y_{n+1}$ proceeds as follows:

1. Let: $z_1 = y_n$.
2. For each slow stage $z_i,\ i = 2, \ldots, s$:
   a) Define: $r_i(\tau) = \sum_{j=1}^{i} \gamma_{i,j} \left(\frac{\tau}{(c_i - c_{i-1})H} \right) f^S \left(t_n + c_j H, z_j \right)$.
   b) Evolve: $\dot{v}(\tau) = f^F \left(t_n + \tau, v \right) + r_i(\tau)$, for $\tau \in [c_{i-1}H, c_i H],\ v(c_{i-1}H) = z_i$.
   c) Let: $z_i = v(c_i H)$.
3. Let: $y_{n+1} = z_s$.

- MIS: $\gamma_{i,j}(\theta)$ is independent of θ, with coefficients computed from the "outer" Runge–Kutta method.
- MRI: $\gamma_{i,j}(\theta)$ is polynomial in θ, coefficients satisfy GARK-based order conditions [Sandu & Günther, 2015].
- Step 2b may use any applicable algorithm of sufficient accuracy (including another MRI method).
- When $c_i = c_{i-1}$, step 2b reduces to a standard ERK/DIRK Runge–Kutta stage update.
- Implicitness at the slow scale depends on "diagonal" $\gamma_{i,i}(\theta)$, typically only used when $c_i = c_{i-1}$.

Other high-order infinitesimal methods

In the last few years multiple groups have made progress on higher-order MRI-like methods:

- *extMIS* [Bauer & Knoth, 2019] slightly modifies their MIS algorithm, and develops $\mathcal{O}\left(H^4\right)$ conditions.
- *RMIS* [Sexton & R., 2019] follows basic MIS stages by computing updated step y_{n+1} as $\sum_{j=1}^{s} b_j \left(f^S(t_n + c_j H, z_j) + f^F(t_n + c_j H, z_j)\right)$, enabling $\mathcal{O}\left(H^4\right)$ and conserv. linear invariants.
- *MERK* [Luan, Chinomona & R., 2020] constructs $r_i(\tau)$ to approximate the action of matrix φ-functions from Exponential Runge–Kutta methods, inheriting up to $\mathcal{O}\left(H^5\right)$ from base ExpRK method.
- *MERB* [Luan, Chinomona & R., 2021] constructs $r_i(\tau)$ to approximate the action of matrix φ-functions from Exponential Rosenbrock methods, inheriting up to $\mathcal{O}\left(H^6\right)$ from base ExpRB method.

Each of these methods focus on explicit treatment of the slow time scale $f^S(t, y)$.

Outline

Implicit-Explicit Multirate Infinitesimal GARK Methods [Chinomona & R., *SIAM J. Sci. Comput.*, 2021]

To better support the flexibility needs of multiphysics problems, we have extended Sandu's MRI-GARK methods to support implicit-explicit treatment of the slow time scale, for problems of the form:

$$\dot{y}(t) = f^I(t,y) + f^E(t,y) + f^F(t,y), \quad t \in [t_0, t_f], \quad y(t_0) = y_0.$$

These follow the same basic approach as the previous MRI algorithm, but with forcing function

$$r_i(\tau) = \sum_{j=1}^{i} \gamma_{i,j}\left(\tfrac{\tau}{(c_i - c_{i-1})H}\right) f^I(t_n + c_j H, z_j) + \sum_{j=1}^{i-1} \omega_{i,j}\left(\tfrac{\tau}{(c_i - c_{i-1})H}\right) f^E(t_n + c_j H, z_j),$$

where $\gamma_{i,j}(\theta) := \sum_{k=0}^{k_{max}} \gamma_{i,j}^{\{k\}} \theta^k$ and $\omega_{i,j}(\theta) := \sum_{k=0}^{k_{max}} \omega_{i,j}^{\{k\}} \theta^k$.

- Coefficients matrices $\Gamma^{\{k\}}, \Omega^{\{k\}} \in \mathbb{R}^{s\times s}$ are lower and strictly lower triangular, respectively.
- Order conditions up to $\mathcal{O}(H^4)$ leverage GARK framework.
- While theory supports "solve-coupled" methods; all current tables are solve-decoupled.

Joint Linear Stability – IMEX-MRI-GARK3a & IMEX-MRI-GARK3b (stability optimized)

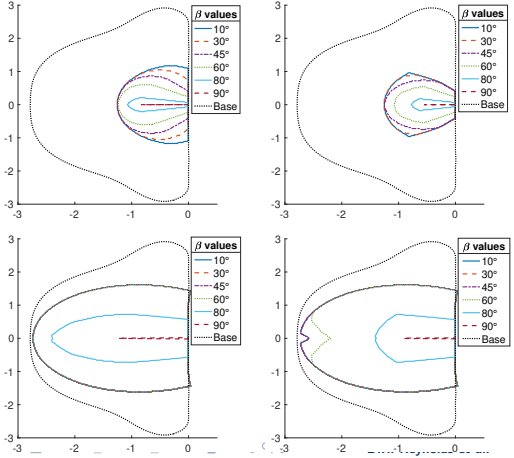
Multirate method stability is currently difficult to analyze. Examining "joint stability" [Zharovsky et al., 2015] for the Dahlquist-like test problem $\dot{y} = \lambda^I y + \lambda^E y + \lambda^F y$:

$$\mathcal{J}_{\alpha,\beta} = \left\{ z^E \in \mathbb{C}^- : \left| R\left(z^F, z^E, z^I\right) \right| \leq 1, \forall z^F \in \mathcal{S}^F_\alpha, \forall z^I \in \mathcal{S}^I_\beta \right\}, \quad \mathcal{S}^\sigma_\alpha = \left\{ z^\sigma \in \mathbb{C}^- : |\arg(z^\sigma) - \pi| \leq \alpha \right\}$$

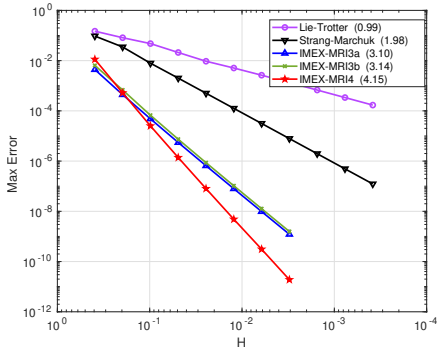
$\mathcal{J}_{\alpha,\beta}$ regions for various implicit sector angles β:

- IMEX-MRI-GARK3a (↑)
- IMEX-MRI-GARK3b (↓)
- fast $\alpha = 10^o$ (←)
- fast $\alpha = 45^o$ (→)

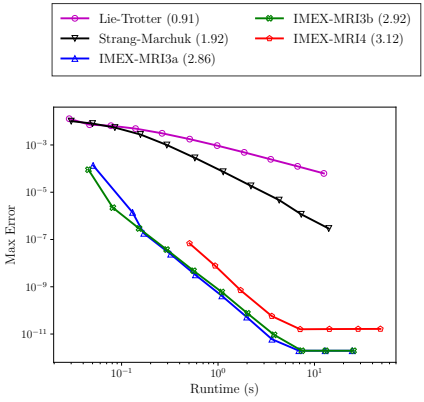
We have a $\mathcal{O}(H^4)$ IMEX-MRI-GARK4 table for convergence verification, though it has poor joint stability.



IMEX-MRI-GARK Convergence/Efficiency Results



Nonlinear Kværnø-Prothero-Robinson test problem convergence.



Stiff brusselator PDE test runtime efficiency.

$H = \left\{\frac{1}{40}, \frac{1}{80}\right\}$ runs were unstable for IMEX-MRI4.

Outline

Outline

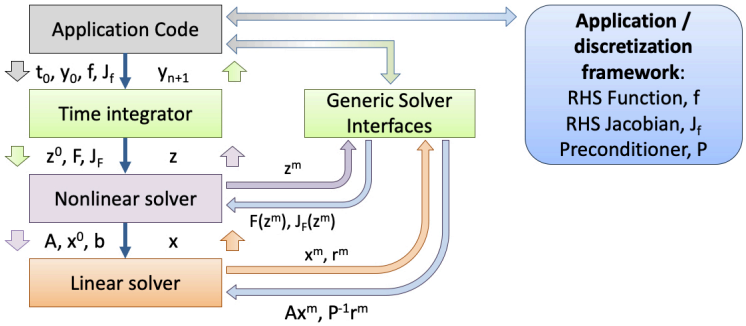
Software: ARKODE and SUNDIALS

ARKODE's initial release within SUNDIALS in 2014 provided adaptive IMEX-ARK methods. Since then we have enhanced ARKODE to include a variety of "steppers":

- `ARKStep`: supports all of ARKODE's original functionality (adaptive ARK, ERK, DIRK methods); includes an interface to XBraid for PinT (work by D. Gardner).
- `ERKStep`: tuned for highly efficient explicit Runge–Kutta methods.
- `MRIStep`: infinitesimal multirate time stepping module.
  - Includes explicit MIS method of $\mathcal{O}(H^3)$.
  - Includes explicit or solve-decoupled implicit MRI-GARK methods of $\mathcal{O}(H^2)$ to $\mathcal{O}(H^4)$.
  - Includes IMEX-MRI-GARK methods of $\mathcal{O}(H^3)$ to $\mathcal{O}(H^4)$.
  - Supports user-provided MRI-GARK tables $\Gamma^{\{k\}}$ or IMEX-MRI-GARK tables $\{\Gamma^{\{k\}}, \Omega^{\{k\}}\}$.
  - Slow time scale uses a user-defined H that can be varied between steps. Fast time scale evolved using `ARKStep` or any viable user-supplied IVP solver.
  - Robust multirate adaptivity (H and h) in development [Fish & R., arXiv:2202.10484, 2022].

ARKODE leverages SUNDIALS' Modular Design & Control Inversion [Gardner et al., 2021]

Control passes between integrator, solvers, and application code as the integration progresses:



Time integrators are agnostic as to the vector data layout and algebraic solvers used, leveraging application-specific implementations where possible, and providing native modules if desired.

Outline

Multirate reacting flow demonstration problem

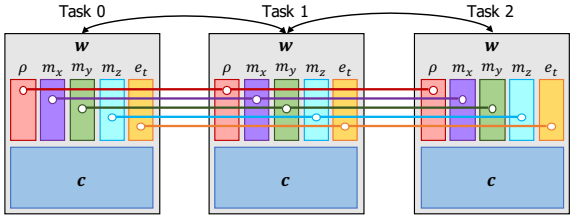
3D nonlinear compressible Euler equations combined with stiff chemical reactions for a low-density primordial gas (molecular & ionization states of H and He, free electrons, and internal gas energy), present in models of the early universe.

$$\partial_t \mathbf{w} = -\nabla \cdot \mathbf{F}(\mathbf{w}) + \mathbf{R}(\mathbf{w}), \quad \mathbf{w}(t_0) = \mathbf{w}_0,$$

$\mathbf{w}$: density, momenta, total energy, and chemical densities (10)
$\mathbf{F}$: advective fluxes (nonstiff/slow); and $\mathbf{R}$: reaction network (stiff/fast)

$\mathbf{w}$ is stored as an `MPIManyVector`:

- Software layer treating collection of vector objects as a single cohesive vector.
- Does not touch any vector data directly.
- Simplifies partitioning of data among computational resources (e.g., CPU vs GPU).
- May also combine distinct MPI intracommunicators together in a multiphysics simulation.

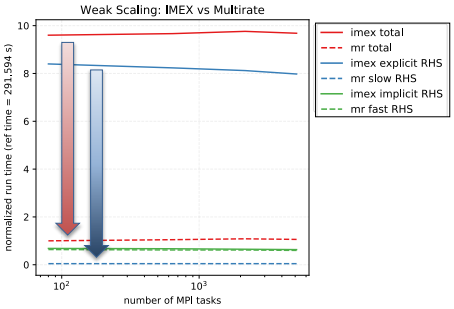


$\mathbf{w}$ is a collection of distributed vectors (density ρ, momentum m_i, and total energy e_T), and local vectors $\mathbf{c}$ (chemical densities).

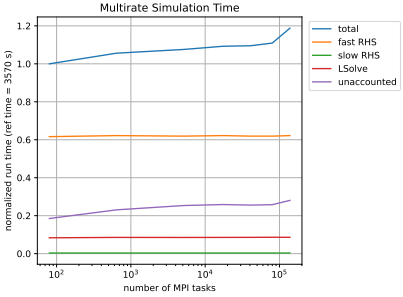
Multirate reacting flow solver strategy

- Method of lines: $(X,t) \in \Omega \times (t_0, t_f]$, with $\Omega = [x_l, x_r] \times [y_l, y_r] \times [z_l, z_r]$.
- Regular $n_x \times n_y \times n_z$ grid for Ω, parallelized using standard 3D MPI domain decomposition.
- $\mathcal{O}(\Delta x^5)$ FD-WENO flux reconstruction for $\mathbf{F}(\mathbf{w})$ [Shu, 2003].
- Resulting IVP system: $\dot{\mathbf{w}}(t) = f_1(\mathbf{w}) + f_2(\mathbf{w})$, $\mathbf{w}(t_0) = \mathbf{w}_0$, where $f_1(\mathbf{w})$ contains $-\nabla \cdot \mathbf{F}(\mathbf{w})$, and $f_2(\mathbf{w})$ contains spatially-local reaction network $\mathbf{R}(\mathbf{w})$.
- Compare two forms of temporal evolution:
  - (a) temporally-adaptive, $\mathcal{O}(H^3)$ ARK-IMEX method from `ARKStep`: f_1 explicit and f_2 implicit,
  - (b) fixed-step, $\mathcal{O}(H^3)$ MRI-GARK method from `MRIStep` (temporally-adaptive fast step h): f_1 slow/explicit and f_2 fast/DIRK.

Multirate reacting flow results (CPU) – GMRES for spatially-local linear solves in f_2



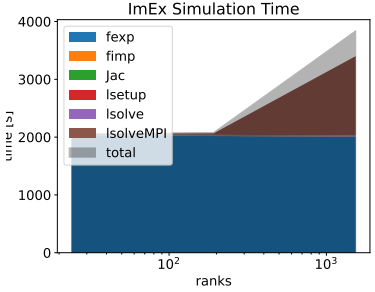
$\sim 10\times$ speedup with multirate



90% weak scaling efficiency using 80 to 138,240 CPU cores of OLCF Summit.

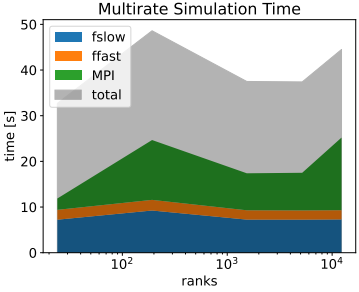
- Multirating allows advection (which requires MPI) to run at a far larger time step size than that required for the single rate ImEx method to maintain stability, leading to significant speedup.
- Multirate cost is dominated by fast RHS (which remains unchanged from ImEx); upturn at largest size due to serialized chemical rate table input (subsequently fixed).

Multirate reacting flow (CPU+GPU) – hydro data & f_1 on CPU, chem data & f_2 on GPU



ImEx and multirate results using hybrid CPU+GPU on OLCF Summit.

Larger-scale jobs are currently waiting in the queue.



- Both use robust MAGMA batched linear solver, enabling a much more challenging test setup.
- ImEx: although f_2 is spatially-local, synchronizations occur within every global time step.
- Multirate: f_2 integrated separately on each MPI rank, relaxing synchronizations to "slow" f_1 time scale.
- Huge reduction in f_1 evaluations allows multirate speedup of $\sim 50\times$ over ImEx.
- Both solvers show algorithmic scalability, but GPU over-synchronization hinders ImEx runtime scalability.

Outline

Conclusions

Large-scale multiphysics problems:

- Nonlinear, interacting models pose key challenges to stable, accurate and scalable simulation.
- Large data requirements require scalable solvers; while individual processes admit "optimal" algorithms & time scales, these rarely agree.
- Most classical methods derived for idealized problems perform poorly on "real world" applications.

Although simple operator-spliting remains standard, new & flexible methods are catching up, supporting high order accuracy (up to $\mathcal{O}(H^6)$) and multirate/ImEx flexibility.

The optimal choice of method depends on a variety of factors:

- whether the problem admits a natural and effective ImEx and/or multirate splitting,
- relative costs of $f^S(t,y)$ and $f^F(t,y)$ for multirate; availability of optimal algebraic solvers for $f^I(t,y)$,
- desired solution accuracy, ...



Future Work

Much work remains to be done:

- Incorporate slow/implicit fluid viscosity and/or heat conduction in demonstration code (to leverage IMEX-MRI-GARK methods).
- Support for additional infinitesimal multirate methods (e.g., MERK, MERB, etc.) within ARKODE's `MRIStep` module.
- New $\Gamma^{(k)}$ and $\Omega^{(k)}$ tables (with embeddings) for $\mathcal{O}(H^3)$-$\mathcal{O}(H^4)$ MRI-GARK and IMEX-MRI-GARK methods (and order conditions for $\mathcal{O}(H^5)$ or higher).
- Robust temporal controllers for nested multirating, $h_1 > h_2 > \cdots > h_m$.
- Rigorous stability theory for additively-partitioned ODE systems (not just $\dot{y} = \sum_k \lambda_k y$, that assumes *simultaneous diagonalizability*).
- Robust (perhaps automated) approaches for determining additive splittings $f(t,y) = \sum_k f^{\{k\}}(t,y)$.

Funding & Computing Support

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Scientific Discovery through Advanced Computing (SciDAC) Program through the FASTMath Institute, under Lawrence Livermore National Laboratory subcontract B626484 and DOE award DE-SC0021354.

This research was supported in part by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration.

This research used resources of the Oak Ridge Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC05-00OR22725.



References (all link to web versions)

- Keyes et al., *Int. J. High Perf. Comput. Appl.*, 2013.
- Strang, *SIAM J. Numer. Anal.*, 1968.
- Marchuk, *Aplikace Matematiky*, 1968.
- Ropp & Shadid, *J. Comput. Phys.*, 2005.
- Estep et al., *SINUM*, 2008.
- Gear & Wells, *BIT*, 1984.
- Günther et al., *BIT*, 2001.
- Constantinescu & Sandu, *J. Sci. Comput.*, 2007.
- Sandu & Constantinescu, *J. Sci. Comput.*, 2009.
- Fok, *J. Sci. Comput.*, 2016.
- Knoth & Wolke, *Appl. Numer. Math.*, 1998.
- Schlegel et al., *J. Comput. Appl. Math.*, 2009.
- Schlegel et al., *Appl. Numer. Math.*, 2012.

References (all link to web versions)

- Hairer & Ostermann, *Numer. Math.*, 1990.
- Engstler & Lubich, *Appl. Numer. Math.*, 1997.
- Constantinescu & Sandu, *SIAM J. Sci. Comput.*, 2010.
- Constantinescu & Sandu, *J. Sci. Comput.*, 2013.
- Bouzarth & Minion, *J. Comput. Phys.*, 2010.
- Sandu, *SIAM J. Numer. Anal.*, 2019.
- Bauer & Knoth, *J. Comput. Appl. Math.*, 2019.
- Sexton & Reynolds, *arXiv:1808.03718*, 2019.
- Luan, Chinomona & Reynolds, *SIAM J. Sci. Comput.*, 2020.
- Luan, Chinomona & Reynolds, *arXiv:2106.05385*, 2021.
- Sandu & Günther, *SIAM J. Numer. Anal.*, 2015.
- Chinomona & Reynolds, *SIAM J. Sci. Comput.*, 2021.
- Zharovsky et al., *SIAM J. Numer. Anal.*, 2015.