High-order multirate infinitesimal methods for tokamak turbulence

Daniel R. Reynolds[1], Cody J. Balos[2], Darin R. Ernst[3], Manaure Francisquez[4], Carol S. Woodward[2]

reynolds@smu.edu, balos1@llnl.gov, dernst@psfc.mit.edu, mfrancis@pppl.gov, woodward6@llnl.gov

[1]Department of Mathematics, Southern Methodist University
[2]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory
[3]Plasma Science and Fusion Center, Massachusetts Institute of Technology
[4]Department of Computational Sciences, Princeton Plasma Physics Laboratory

SciCA∂E International Conference on Scientific Computation and Differential Equations
University of Iceland, Reykjavk
27 July 2022

Outline

Multiphysics simulations [Keyes et al., 2013]

Multiphysics simulations couple together different physical models, either *in the bulk* or *across interfaces*. For example in climate:

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

[https://e3sm.org]

Multiphysics challenges [Keyes et al., 2013]

These 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.

Multirate Infinitesimal Step (MIS/MRI) methods [Schlegel et al., 2009; Sandu, 2019; Chinomona & R., 2021; . . .]

- Multirate infinitesimal methods arose in numerical weather prediction, with dramatic recent advances.
- Generic infrastructure supports additively-split multirate problems:

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

- $f^S(t,y) := f^I(t,y) + f^E(t,y)$ contains the "slow" dynamics, evolved with time step H.
- $f^F(t,y)$ contains the "fast" dynamics, evolved with time steps $h \ll H$.
- Fast time scale is evolved using any desired solver (of sufficient accuracy), while slow time scale is advanced through solving a sequence of modified "fast" IVPs.
- Achieve higher-order through:
  - appropriate specification of initial conditions for each fast IVP, and
  - temporal interpolation of f^S onto the fast time scale through definition of each fast IVP.
- Extremely efficient – $\mathcal{O}(H^4)$ attainable with *only a single traversal* of $(t_n, t_{n+1}]$, unlike extrapolation or deferred correction approaches that bootstrap Lie–Trotter operator splittings at significantly higher cost.

MRI method skeleton

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

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

- Step 2b may use any applicable algorithm of sufficient accuracy (including another MRI method).
- When $\Delta c_i = 0$, step 2 reduces to an additive Runge–Kutta-like update,

$$z_i = z_{i-1} + H\sum_{j=1}^{i}\left(\int_0^1 \gamma_{i,j}(\theta)\,\mathrm{d}\theta\right) f^I(t_{n,j}, z_j) + H\sum_{j=1}^{i-1}\left(\int_0^1 \omega_{i,j}(\theta)\mathrm{d}\theta\right) f^E(t_{n,j}, z_j)$$

- Slow time scale is implicit when $\gamma_{i,i}(\theta) \neq 0$, only used when $\Delta c_i = 0$ (a.k.a., “solve decoupled”).

MRI variants

- Seminal up to $\mathcal{O}(H^3)$ MIS methods set $\gamma_{i,j}(\theta) = 0$, $\omega_{i,j}(\theta) = \begin{cases} 0 & \text{if } i = 1, \\ A^O_{i,j} - A^O_{i-1,j} & \text{if } 1 < i < s, \\ b^O_j - A^O_{s-1,j} & \text{if } i = s. \end{cases}$

  (A^O, b^O, c^O) is an "outer" explicit Butcher table with $s-1$ stages and $c^O_j \le c^O_{j+1}$.

- Sandu's MRI-GARK methods [*SIAM J. Numer. Anal.*, 2019] support solve-decoupled implicit methods, setting

$$\gamma_{i,j}(\theta) = \omega_{i,j}(\theta) = \sum_{k=0}^{k_{max}} \gamma^{\{k\}}_{i,j} \theta^k$$

  where order conditions on $\Gamma^{\{k\}}$ up to $\mathcal{O}(H^4)$ leverage GARK framework [Sandu & Günther, 2015].

- Chinomona & R.'s IMEX-MRI-GARK methods [*SIAM J. Sci. Comput.*, 2021] extend further to set

$$\gamma_{i,j}(\theta) = \sum_{k=0}^{k_{max}} \gamma^{\{k\}}_{i,j} \theta^k, \qquad \omega_{i,j}(\theta) = \sum_{k=0}^{k_{max}} \omega^{\{k\}}_{i,j} \theta^k,$$

  again leveraging GARK framework for up to $\mathcal{O}(H^4)$ order conditions on $\Gamma^{\{k\}}$ and $\Omega^{\{k\}}$.

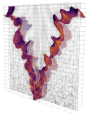
- Luan, Chinomona & R.'s MERK and MERB methods [*SIAM J. Sci. Comput.*, 2020 & 2022] instead leverage exponential Runge–Kutta and Rosenbrock methods for up to $\mathcal{O}(H^6)$ accuracy with similar structure.

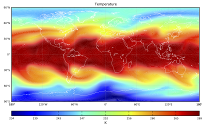
Outline

SUNDIALS – SUite of Nonlinear and DIfferential-ALgebraic equation Solvers

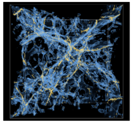
- Software library consisting of ODE and DAE integrators and nonlinear solvers
  - Consists of six independent packages: CVODE(S), ARKODE, IDA(S), KINSOL
  - Written in C with interfaces to Fortran (Python coming soon)
  - *Designed to be easily incorporated into existing codes*
- Modular implementation
  - Data use is fully encapsulated by vector and matrix APIs
  - Nonlinear and linear solvers are fully encapsulated from the integrators
  - All parallelism is encapsulated in vectors, solvers, and user-supplied functions
  - Includes data structures and solvers for *serial, threaded, MPI, and GPU*
  - *Vector, matrix, and solver modules can all be user-supplied*
- Availability and support
  - Freely available (BSD 3-Clause license); >120k downloads in 2021
  - Detailed user manuals at sundials.readthedocs.io
  - Active user community supported by sundials-users email list

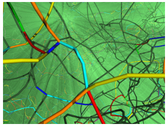
For more information visit github.com/LLNL/sundials or computing.llnl.gov/sundials

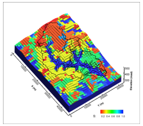

Combustion

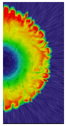


Atmospheric dynamics


Cosmology


Dislocation dynamics


Subsurface flow


Supernovae

ARKODE: a flexible infrastructure for one-step integration methods

- Originally designed to provide adaptive implicit-explicit (IMEX) ARK methods for IVPs, but recently overhauled to serve as an infrastructure for general, adaptive, one-step methods:
  - ARKODE provides outer time integration loop and generic use modes e.g., interpolation vs "tstop"
  - Time-stepping modules handle problem-specific components: IVP definition, single step algorithm
  - The step modules leverage ARKODE's and SUNDIALS' shared infrastructure e.g.,
    - SUNDIALS vector, matrix, linear solver, and nonlinear solver objects
    - Translation between generic solvers and IVP-specific algebraic systems
    - Time-step adaptivity controllers (PID, PI, I, or user-supplied), temporal interpolation modules, implicit predictors, . . .
- The new framework provides increased agility for implementing advanced algorithms in production software
  - **ARKStep**: ARK, DIRK, and ERK methods for $M(t)\,y' = f^E(t,y) + f^I(t,y),\ y(t_0) = y_0,$
  - **ERKStep**: A streamlined module with ERK methods for $y' = f(t,y),\ y(t_0) = y_0,$
  - **MRIStep**: Multirate infinitesimal methods for $y' = f^I(t,y) + f^E(t,y) + f^F(t,y),\ y(t_0) = y_0.$
- Design to allow users to explore "algorithm space," easily testing different methods for their application.

MRIStep

The current MRIStep release (SUNDIALS v6.2.0) supports explicit MIS and MRI-GARK, and solve-decoupled implicit MRI-GARK and IMEX-MRI-GARK methods

- Built-in methods of $\mathcal{O}(H^2)$ through $\mathcal{O}(H^4)$; supports user-provided coupling tables $\{\Gamma^{\{k\}}, \Omega^{\{k\}}\}$
- The slow time scale requires a user-defined fixed step size H that can be varied between steps
- The fast time scale can be evolved using any viable user-supplied IVP solver (a "custom" inner stepper)
    - Utility routine to wrap ARKStep for this role: adaptive or fixed-step explicit, implicit, or IMEX treatment of the fast time scale
    - ARKStep includes embedded methods of various orders (ARK 3 – 5, DIRK 2 – 5, and ERK 2 – 6, 8); user-provided Butcher tables supported
    - Example problems are even provided to show use of CVODE as a custom inner stepper
- Solve-decoupled implicit methods can utilize the full ARKStep solver infrastructure
- Robust multirate adaptivity (H and h) is under development [Fish & R., arXiv:2202.10484, 2022]

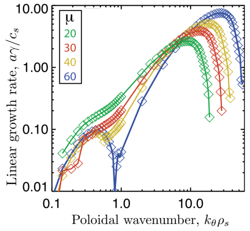
Outline

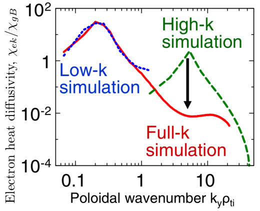
Multiscale gyrokinetic simulations indicate cross-scale ITG/ETG turbulence

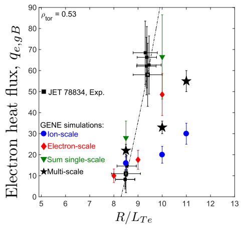
Initial studies with reduced ion/electron mass ratios $\left(\mu := \sqrt{m_i/m_e} < 60\right)$ found interactions between ion- and electron-scale turbulence.
[Toda & Itoh, 2001; Li & Kishimoto, 2002; . . .]

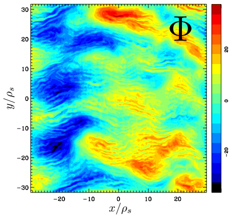
Gyrokinetic studies with realistic $\mu = 60$ indicated different growth rates and energy transport, but require resolving 2 orders of magnitude in both space & time.
[Howard et al., 2014 & 2021; Maeyama et al., 2015]

Realistic mass ratio simulations are required to accurately predict fluxes in current/future reactors, but each require $\mathcal{O}(10)$ million CPU-hours.
[Bonanomi et al., 2018]









MuSHrooM: reduced 2D toroidal fluid model [Francisquez, Ernst, R., & Balos, 2021]

Developed reduced fluid model as accurate test-bed for algorithms, with same physics as 5D gyrokinetic simulations. Model consists of two nonlinear, interacting, PDEs, $\{\tilde{n}, \tilde{T}_\perp\}$, for each species $s = \{e, i\}$:

$$\begin{aligned}
&\frac{\partial n}{\partial t} + \frac{c}{B}\left[\Psi, n\right] + \frac{n_0}{T_{\perp 0}}\frac{c}{B}\left[\frac{1}{2}\hat{\nabla}_\perp^2\Psi, T_\perp\right] - n_0\left(1 + \eta_\perp\frac{1}{2}\hat{\nabla}_\perp^2\right) i\omega_* \frac{e\Psi}{T_0} \\
&\quad + n_0\frac{q}{|q|}\left(2 + \frac{1}{2}\hat{\nabla}_\perp^2\right) i\omega_d\frac{e\Psi}{T_0} + \frac{i\omega_d}{m v_t^2}\left[\left(T_{\parallel 0} + T_{\perp 0}\right) n + n_0 T_\perp\right] = -\alpha_n n + \mathcal{D}_n,
\end{aligned}$$

$$\begin{aligned}
&\frac{n_0}{B}\frac{\partial T_\perp}{\partial t} + \frac{T_{\perp i0}}{B}\left(\frac{\partial n}{\partial t} + \frac{c}{B}\left[\Psi, n\right]\right) - \frac{p_{\perp 0}}{B}\left[\left(1 + \eta_\perp\right)\left(1 + \frac{1}{2}\hat{\nabla}_\perp^2\right) + \eta_\perp \hat{\hat{\nabla}}_\perp^2\right] i\omega_* \frac{e\Psi}{T_0} \\
&\quad + \frac{cT_{\perp 0}}{B^2}\left[\frac{1}{2}\hat{\nabla}_\perp^2\Psi, n\right] + \frac{cn_0}{B^2}\left[\left(1 + \frac{1}{2}\hat{\nabla}_\perp^2\right)\Psi, T_\perp\right] + \frac{p_{\perp 0}}{B}\frac{q}{|q|}\left(3 + \frac{3}{2}\hat{\nabla}_\perp^2 + \hat{\hat{\nabla}}_\perp^2\right) i\omega_d\frac{e\Psi}{T_0} \\
&\quad + \frac{cn_0}{B^2}\left[\hat{\hat{\nabla}}_\perp^2\Psi, T_\perp\right] + \frac{i\omega_d}{v_t^2}\frac{1}{B}\left(r_{\parallel,\perp} + r_{\perp,\perp}\right) = -\alpha_T T_\perp + \mathcal{D}_T,
\end{aligned}$$

where $\Psi_s = \Gamma_0^{1/2}(b_s)\phi$ and $b_s = k_\perp v_{ts}/\Omega_s$. Retains full Bessel functions for FLR effects, using gyrofluid tricks:

$$\frac{1}{2}\hat{\nabla}_\perp^2\Gamma_0^{1/2} = b_0\frac{\partial \Gamma_0^{1/2}}{\partial b_0} \qquad\qquad i\omega_{*s} = -\frac{cT_{s0}}{eBn_0}\nabla n_0 \cdot \hat{\mathbf{b}} \times \nabla$$

$$\hat{\hat{\nabla}}_\perp^2\Gamma_0^{1/2} = b_0\frac{\partial}{\partial b_0}\left(\Gamma_0^{1/2} + b_0\frac{\partial \Gamma_0^{1/2}}{\partial b_0}\right) \qquad\qquad i\omega_{ds} = \frac{v_{ts}L_n}{\Omega_s B}\hat{\mathbf{b}} \times \nabla B \cdot \nabla$$

Pseudospectral discretization

- Discretize spatial domain $[L_x, L_y]$ using Fourier basis with $N_x N_y$ uniformly spaced grid points, e.g.,

$$n_i(t,x,y) = \sum_{k_x,k_y} \tilde{n}_{i,k_x,k_y}(t)\ \exp\left(\frac{2\pi i k_x}{L_x}x + \frac{2\pi i k_y}{L_y}y\right)$$

- Standard 2D MPI domain decomposition for $[L_x, L_y]$
- Evolve PDE system in the frequency domain: coupled system of $4N_xN_y$ IVPs for the time-dependent coefficients $\left\{\tilde{n}_{i,k_x,k_y}(t), \tilde{n}_{e,k_x,k_y}(t), \tilde{T}_{i,k_x,k_y}(t), \tilde{T}_{e,k_x,k_y}(t)\right\}$
- While spatial derivatives correspond with simple scalar multiplication, evaluation of Poisson brackets $[\phi,\psi] := \partial_x\phi\partial_y\psi - \partial_y\phi\partial_x\psi$ requires FFT/IFFT.
- Ion temperature gradient (ITG) modes occur at low k_y, whereas ETG modes occur at higher k_y.
- Large-scale energy fluxes occur at ion scales and should be accurately resolved, but electron scale transport need only be captured in an averaged sense.
- Electron scale transport induces limits on explicit time integration: (60x faster) $\times$ (60x finer resolution)

Multirate formulation: exploit inherent time/space scale separation

Partition wavenumber space into non-overlapping sets $\mathcal{K}_i = \{(k_x, k_y) \,:\, k_y \le k_{y,c}\}$ and $\mathcal{K}_e = \{(k_x, k_y) \,:\, k_y > k_{y,c}\}$. The full multiscale MuSHrooM model may be written

$$y'(t) = \begin{bmatrix} y^{\{i\}}(t) \\ y^{\{e\}}(t) \end{bmatrix}' = \begin{bmatrix} f^{\{i\}}(y^{\{i\}}, y^{\{e\}}) \\ f^{\{e\}}(y^{\{i\}}, y^{\{e\}}) \end{bmatrix} = f(t), \quad t \in (t_0, t_f], \quad y(t_0) = y_0,$$

We define the "time averaging" operator:

$$\overline{f}^{\{e\}}(t, y) := \frac{1}{\Delta t - \Delta t_{min}} \int_{t+\Delta t_{min}}^{t+\Delta t} f^{\{e\}}(\hat{y}(\tau))\, \mathrm{d}\tau,$$

where $\hat{y}(\tau)$ solves the full multiscale IVP $\hat{y}'(\tau) = f(\hat{y})$ for $\tau \in (t, t+\Delta t]$ with $\hat{y}(t) = y(t)$.

Assumptions:

- The fastest components that we must accurately capture are in $\mathcal{K}_i$ – our "fast" time scale.
- The dynamics in $\mathcal{K}_e$ (that are accurately tracked by $\hat{y}$) are "microscale" – do not need to be resolved.
- Moments $\overline{f}^{\{e\}}(t, y)$ evolve on a considerably slower time scale than the "fast" dynamics within $\mathcal{K}_i$.

Multirate formulation: exploit inherent time/space scale separation

We consider a partially-time-averaged version of the original MuSHrooM model,

$$\begin{bmatrix} y^{\{i\}}(t) \\ \overline{y}^{\{e\}}(t) \end{bmatrix}' = \begin{bmatrix} f^{\{i\}}(y^{\{i\}}, \overline{y}^{\{e\}}) \\ \overline{f}^{\{e\}}(y^{\{i\}}, \overline{y}^{\{e\}}) \end{bmatrix}, \quad t \in (t_0, t_f], \quad y(t_0) = y_0,$$

that may be evolved using an explicit MRI-GARK algorithm with the partitioning

$$f^F = \begin{bmatrix} f^{\{i\}}(y^{\{i\}}, \overline{y}^{\{e\}}) \\ 0 \end{bmatrix}, \qquad f^S = \begin{bmatrix} 0 \\ \overline{f}^{\{e\}}(y^{\{i\}}, \overline{y}^{\{e\}}) \end{bmatrix}.$$

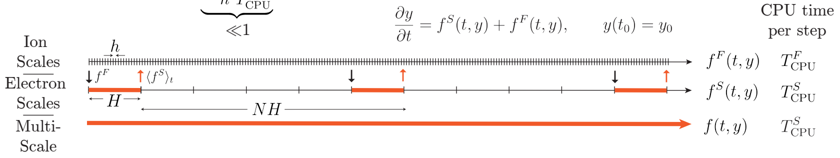
- In the limit as $\Delta t_{min}, \Delta t \to 0$, the homogenized IVP converges to the original IVP.
- The MRI-GARK method will use slow/fast time steps H and h, corresponding with the dynamics of $\overline{y}^{\{e\}}$ and $y^{\{i\}}$, respectively.
- Evaluation of f^S requires short bursts of the full multiscale model for $\hat{y}$ over $[t, t + \Delta t]$, using microscale time steps $\delta t \ll h$.

Multirate splitting parameters

- $k_{y,c}$: defines the frequency threshold for resolved vs unresolved modes. Asymptotic arguments estimate this as roughly $k_{y,c}/N_y \approx 1/60$.
- Δt_{min}: each short simulation for $\hat{y}$ must first integrate past initial transients before constructing the time average.
- Δt: since $\overline{f}^{\{e\}}$ is averaged over $[\Delta t_{min}, \Delta t]$, this must be large enough to construct a good average, but small enough to achieve overall cost savings, e.g., $2\Delta t_{min} < \Delta t < H/100$.
- δt and h: both may be computed adaptively by ARKStep; we expect that $\delta t < h/1000$.
- H: hope to eventually use multirate adaptivity, but we must currently determine this experimentally.

$$\text{MRI Speedup} = \frac{N}{1 + \underbrace{N \frac{H}{h} \frac{T^F_{\text{CPU}}}{T^S_{\text{CPU}}}}_{\ll 1}} \approx N \text{ times faster than full multi-scale}$$

$$\frac{\partial y}{\partial t} = f^S(t,y) + f^F(t,y), \qquad y(t_0) = y_0$$



Experimental parameter identification from full multiscale model

Currently running $\hat{y}(\tau)$ over a subset $\tau \in [t_0, \hat{t}_f]$ to determine appropriate parameter values.

- Δt_{min}: examine $\overline{f}^{\{e\}}$ as $\Delta t_{min} \to 0$. This should converge to a point, followed by stagnation.
- Δt: using a "best" Δt_{min} from above, examine $\overline{f}^{\{e\}}$ as $\Delta t \to \hat{t}_f$. This should converge as $1/(\Delta t - \Delta t_{min})$, illuminating potential $\overline{f}^{\{e\}}$ accuracy (and corresponding cost).
- $k_{y,c}$: perform above tests for multiple $k_{y,c}$ near $N_y/60$. As $k_{y,c} \to 0$, "optimal" values of both Δt_{min} and Δt should increase to better capture ion-scale dynamics.
- H: using "best" candidates for Δt and Δt_{min} from above, examine temporal autocorrelation function

$$G(\theta) = \frac{\left(\overline{f}^{\{e\}}(t,y) - \langle f^{\{e\}} \rangle\right) \cdot \left(\overline{f}^{\{e\}}(t+\theta,y) - \langle f^{\{e\}} \rangle\right)}{\overline{f}^{\{e\}}(t,y) \cdot \overline{f}^{\{e\}}(t,y)},$$

  where $\langle f^{\{e\}} \rangle$ is the time average of $\overline{f}^{\{e\}}(t,y)$ over $t \in [t_0, \hat{t}_f]$. Should find an H such that autocorrelation is high for $\theta < H$ and low for $\theta > H$.

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:

- Complete investigation of appropriate multirate splitting parameters for MuSHrooM.
- Investigate multirate temporal adaptivity within MuSHrooM.
- Investigate performance and accuracy of MuSHrooM multirate splitting for ITG/ETG turbulence.
- Expand ARKODE's `MRIStep` module to support additional multirate infinitesimal methods (e.g., MERK, MERB, etc.).
- Derive new $\Gamma^{(k)}$ and $\Omega^{(k)}$ tables (with embeddings) for MRI-GARK, IMEX-MRI-GARK, MERK and MERB methods.

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.

MIT work supported under Subaward No. UTA18-000276 under DOE SciDAC Partnership for Multiscale Gyrokinetic Turbulence, DOE award DE-SC0018429.

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231.

References (all link to web versions)

Multirate:

- Keyes et al., *Int. J. High Perf. Comput. Appl.*, 2013.
- Knoth & Wolke, *Appl. Numer. Math.*, 1998.
- Schlegel et al., *J. Comput. Appl. Math.*, 2009.
- Schlegel et al., *Appl. Numer. Math.*, 2012.
- Sandu, *SIAM J. Numer. Anal.*, 2019.
- Sandu & Günther, *SIAM J. Numer. Anal.*, 2015.
- Chinomona & Reynolds, *SIAM J. Sci. Comput.*, 2021.
- Luan, Chinomona & Reynolds, *SIAM J. Sci. Comput.*, 2020.
- Luan, Chinomona & Reynolds, *SIAM J. Sci. Comput.*, 2022.

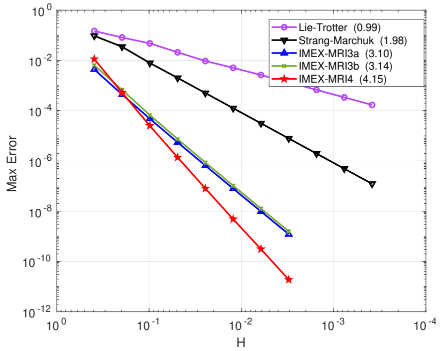
References (all link to web versions)

Plasma turbulence:

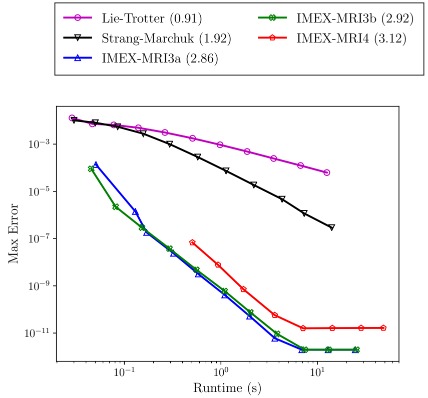
- Toda & Itoh, *Plasma Physics and Controlled Fusion*, 2001.
- Li & Kishimoto, *Physical Review Letters*, 2002.
- Candy, Waltz, Fahey & Holland, *Plasma Physics and Controlled Fusion*, 2007.
- Waltz, Candy & Fahey, *Physics of Plasmas*, 2007.
- Görler & Jenko, *Physical Review Letters*, 2008.
- Howard et al., *Physics of Plasmas*, 2014.
- Howard et al., *Nuclear Fusion*, 2021.
- Maeyama et al., *Physical Review Letters*, 2015.
- Bonanomi et al., *Nuclear Fusion*, 2018.
- Francisquez et al., *Bulletin of the APS*, 2021.
- Ernst et al., *Sherwood International Fusion Theory Conference*, 2022.

Outline

IMEX-MRI-GARK convergence/efficiency results [Chinomona & R., *SIAM J. Sci. Comput.*, 2021]



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

Multirate reacting flow demonstration problem

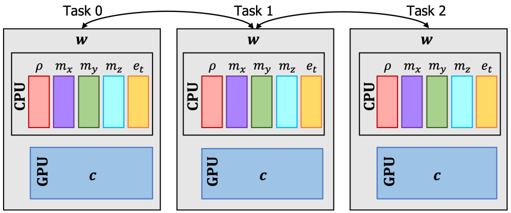
3D nonlinear compressible Euler equations combined with stiff chemical reactions for a low-density primordial gas (molecular & ionization states of H and He, free electrons, and internal gas energy), present in models of the early universe.

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

$\mathbf{w}$: 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.
- Fluid species (density, momenta, total energy) each stored in main memory.
- Chemical densities stored in GPU memory, using NVECTOR_RAJA interface.
- ManyVector handles MPI collectives; manual point-to-point communication for fluxes.



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 is evaluated on the CPU, while $f_2(\mathbf{w})$ contains spatially-local reaction network $\mathbf{R}(\mathbf{w})$ and is evaluated on the GPU.
- 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, $\mathcal{O}(H^3)$ explicit MRI-GARK method from `MRIStep` (temporally-adaptive fast step h): f_1 slow/explicit and f_2 fast/DIRK.

IMEX approach

- At each stage z_i within the ARK-IMEX method, we must solve a nonlinearly implicit system

$$z_i - hA^I_{i,i} f_2(z_i) - y_n - h\sum_{j=1}^{i-1}\left(A^E_{i,j} f_1(z_j) + A^I_{i,j} f_2(z_j)\right) = 0,$$

- Since f_2 contains only spatially-local reaction terms, Newton's method applied to this results in block-diagonal linear systems

$$J = \begin{bmatrix} J_1 & & & \\ & J_2 & & \\ & & \ddots & \\ & & & J_{n_p} \end{bmatrix}, \quad J_p = \begin{bmatrix} J_{p,1,1,1} & & & \\ & J_{p,2,1,1} & & \\ & & \ddots & \\ & & & J_{p,n_{xloc},n_{yloc},n_{zloc}} \end{bmatrix}, \quad J_{p,i,j,k} \in \mathbb{R}^{10\times 10}.$$

- We construct a custom SUNLinearSolver that solves each $J_p x_p = b_p$ using SUNDIALS' new GPU-enabled SUNLinSol_MagmaDense batched solver interface. The only communication required is a single `MPI_Allreduce` to gauge success/failure of the overall linear solve with J, along with norms associated with Newton's method.

Multirate approach

- The $\mathcal{O}(H^3)$ explicit MRI-GARK method evaluates f_1 three times *per slow step*, and requires three modified fast IVPs:

$$v_i'(\tau) = f_2(v) + r_i(\tau), \quad \tau \in (c_{i-1}H, c_i H], \quad v(c_{i-1}H) = z_i$$

corresponding with a system of $n_x n_y n_z$ *decoupled* 15-variable IVPs.

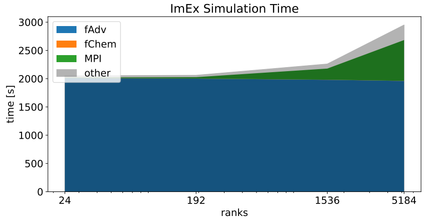
- We construct a custom MRIStepInnerStepper that evolves these separately on each MPI rank.
  - The MRIStep-provided z_i and $r_i(\tau)$ use MPIManyVectors.
  - Custom stepper repackages as rank-local ManyVectors, calling ARKStep to evolve each:

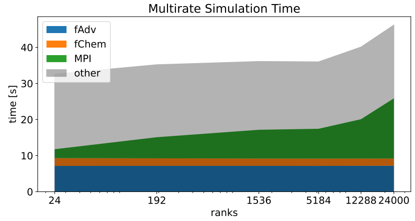
```
// create ManyVector version of input MPIManyVector (reuse y's context object)
N_Vector ysubvecs[6];
for (int ivec=0; ivec<6; ivec++)
   ysubvecs[ivec] = N_VGetSubvector_MPIManyVector(y, ivec);
N_Vector yloc = N_VNewManyVector(6, ysubvecs, y->sunctx);
```

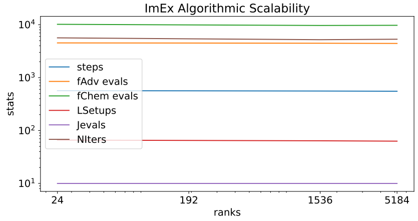
  - Implicit solves at the fast time scale involve rank-local Newton solvers, with nearly identical GPU-enabled SUNLinSol_MagmaDense batched solver interface.
  - `MPI_Allreduce` call to gauge success/failure of fast IVP solves [at slow time scale].

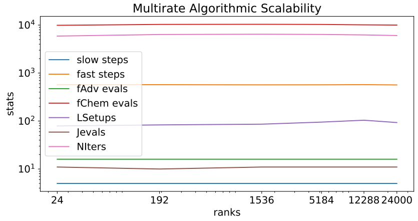
Multirate reacting flow weak scaling results (Summit: CPU+GPU)

- Weak scaling runs with 1 MPI rank per GPU.
- Multirate H chosen proportional to CFL condition on f_1.
- Both approaches show excellent alg. scalability.
- Huge reduction in f_1 evals allows MR / IMEX speedup of $\sim$70x.
- GPU synchronization more severely hinders runtime scalability of IMEX than MR, due to increased frequency (fast vs slow stages).


ImEx Simulation Time
fAdv
fChem
MPI
other
time [s]
ranks


Multirate Simulation Time
fAdv
fChem
MPI
other
time [s]
ranks


ImEx Algorithmic Scalability
steps
fAdv evals
fChem evals
LSetups
Jevals
NIters
stats
10^1
10^2
10^3
10^4
ranks


Multirate Algorithmic Scalability
slow steps
fast steps
fAdv evals
fChem evals
LSetups
Jevals
NIters
stats
10^1
10^2
10^3
10^4
ranks