Flexible, accurate and scalable time integration of multiphysics problems

Daniel R. Reynolds

reynolds@smu.edu

(collaborators are inserted where appropriate)

Department of Mathematics, Southern Methodist University

Computational and Applied Mathematics Seminar
Missisippi State University
25 October 2019

Multiphysics Scientific Simulations

In recent decades computation has rapidly assumed its role as the third pillar of the scientific method [Vardi, *Commun. ACM*, 53(9):5, 2010]:

- Simulation complexity has evolved from simplistic calculations of only 1 or 2 basic equations, to massive models that combine vast arrays of processes.
- Early algorithms could be analyzed using standard techniques, but mathematics has not kept up with the fast pace of scientific simulation development.
- Presently, many numerical analysts construct elegant solvers for models of limited practical use, while computational scientists "solve" highly-realistic systems using *ad hoc* methods with questionable reliability.

I work to bridge this gap between mathematical theory and computing practice.

Outline

Outline

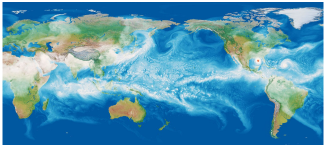
Climate – Energy Exascale Earth System Model (E3SM)

Motivation: 2013 DOE report on need for climate model predictions of energy sector impacts:

- air and water temperature trends
- water availability
- storms and heavy precipitation
- coastal flooding and sea-level rise

Mission (`https://e3sm.org/about/vision-and-mission`)

- integrate advanced models and algorithms to push the high-resolution frontier
- bridge the gap in modeling scales and processes to include natural, managed and man-made systems
- develop ensemble modeling strategies to quantify uncertainty



`https://e3sm.org`

[In collaboration w/ D. Gardner (LLNL), A. Steyer (SNL), M. Taylor (SNL), P. Ullrich (UC Davis), C. Vogl (LLNL) & C. Woodward (LLNL)]

SMU

ECP EXASCALE COMPUTING PROJECT

FASTMATH

sundials

Nonhydrostatic Atmospheric Models

- Increased computational power enables spatial resolutions beyond the hydrostatic limit.
- Nonhydrostatic models consider the 3D compressible Navier Stokes equations; these support acoustic (sound) waves.
- Acoustic waves have a negligible effect on climate, but travel much faster than convection (343 m/s vs 100 m/s horizontal and 1 m/s vertical), leading to overly-restrictive explicit stability restrictions.

- To overcome this stiffness, nonhydrostatic models utilize split-explicit, implicit-explicit, or fully implicit time integration.
- Additionally, climate "dycores" are coupled to myriad other processes (ocean, land/sea ice, . . .), each evolving on significantly different time scales.

Nonhydrostatic Formulation (Tempest) [Gardner, Guerra, Hamon, R., Ullrich & Woodward, 2018]

Tempest is an experimental dycore used for method development; it considers 5 governing [hyperbolic] equations in an arbitrary coordinate system:

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\frac{1}{J}\frac{\partial}{\partial \alpha}\left(J\rho u^{\alpha}\right) - \frac{1}{J}\frac{\partial}{\partial \beta}\left(J\rho u^{\beta}\right) - \frac{1}{J}\frac{\partial}{\partial \xi}\left(J\rho u^{\xi}\right)\\
\frac{\partial u_\alpha}{\partial t} &= -\frac{\partial}{\partial \alpha}(K+\Phi) - \theta\frac{\partial \Pi}{\partial \alpha} + (\eta\times\mathbf{u})_\alpha\\
\frac{\partial u_\beta}{\partial t} &= -\frac{\partial}{\partial \beta}(K+\Phi) - \theta\frac{\partial \Pi}{\partial \beta} + (\eta\times\mathbf{u})_\beta\\
\left(\frac{\partial r}{\partial \xi}\right)\frac{\partial w}{\partial t} &= -\frac{\partial}{\partial \xi}(K+\Phi) - \theta\frac{\partial \Pi}{\partial \xi} + u^{\alpha}\frac{\partial u_\alpha}{\partial \xi} + u^{\beta}\frac{\partial u_\beta}{\partial \xi} - u^{\alpha}\frac{\partial u_\xi}{\partial \alpha} - u^{\beta}\frac{\partial u_\xi}{\partial \beta}\\
\frac{\partial \theta}{\partial t} &= -u^{\alpha}\frac{\partial \theta}{\partial \alpha} - u^{\beta}\frac{\partial \theta}{\partial \beta} - u^{\xi}\frac{\partial \theta}{\partial \xi},
\end{aligned}
$$

where ρ is the density, (u_α, u_β) are the horizontal velocity, w is the vertical velocity, and θ is the potential temperature.

Key: horizontal propagation and vertical propagation.

Nonhydrostatic Formulation (HOMME-NH) [Vogl, Steyer, R., Ullrich & Woodward, 2019]

HOMME-NH will be the “production” dycore in E3SM v2 responsible for global atmospheric flow:

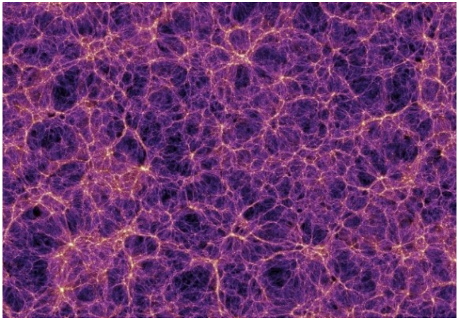
$$
\begin{aligned}
\frac{\partial}{\partial t}\left(\frac{\partial \pi}{\partial \eta}\right) &= -\nabla_\eta \cdot \left(\frac{\partial \pi}{\partial \eta}\mathbf{u}\right) - \frac{\partial}{\partial \eta}\left(\pi \frac{\mathrm{d}\eta}{\mathrm{d}t}\right)\\
\frac{\partial \mathbf{u}}{\partial t} &= -(\nabla_\eta \times \mathbf{u} + 2\boldsymbol{\Omega}) \times \mathbf{u} - \frac{1}{2}\nabla_\eta(\mathbf{u}\cdot\mathbf{u}) - \frac{\mathrm{d}\eta}{\mathrm{d}t}\frac{\partial \mathbf{u}}{\partial \eta} - \frac{1}{\rho}\nabla_\eta p\\
\frac{\partial w}{\partial t} &= -\mathbf{u}\cdot\nabla_\eta w - \frac{\mathrm{d}\eta}{\mathrm{d}t}\frac{\partial w}{\partial \eta} - g(1-\mu), \quad \mu = \left(\frac{\partial p}{\partial \eta}\right) \Big/ \left(\frac{\partial \pi}{\partial \eta}\right),\\
\frac{\partial \Theta}{\partial t} &= -\nabla_\eta \cdot (\Theta \mathbf{u}) - \frac{\partial}{\partial \eta}\left(\Theta \frac{\mathrm{d}\eta}{\mathrm{d}t}\right), \quad \Theta = \frac{\partial \pi}{\partial \eta}\theta,\\
\frac{\partial \phi}{\partial t} &= -\mathbf{u}\cdot\nabla_\eta \phi - \frac{\mathrm{d}\eta}{\mathrm{d}t}\frac{\partial \phi}{\partial \eta} + gw,
\end{aligned}
$$

where π is hydrostatic pressure, η is vertical coordinate, $\mathbf{u}$ and w are horizontal and vertical velocities, θ is potential temperature, and ϕ is geopotential.

Key: hydrostatic model and nonhydrostatic terms.

Cosmic Reionization – The Origins of the Universe

- After the Big Bang, primordial matter (96% dark matter, 2.92% H, 1% He) was strewn throughout the universe.
- Gravitational attraction condensed this into the "cosmic web," the large-scale structure that connects/creates galaxies.

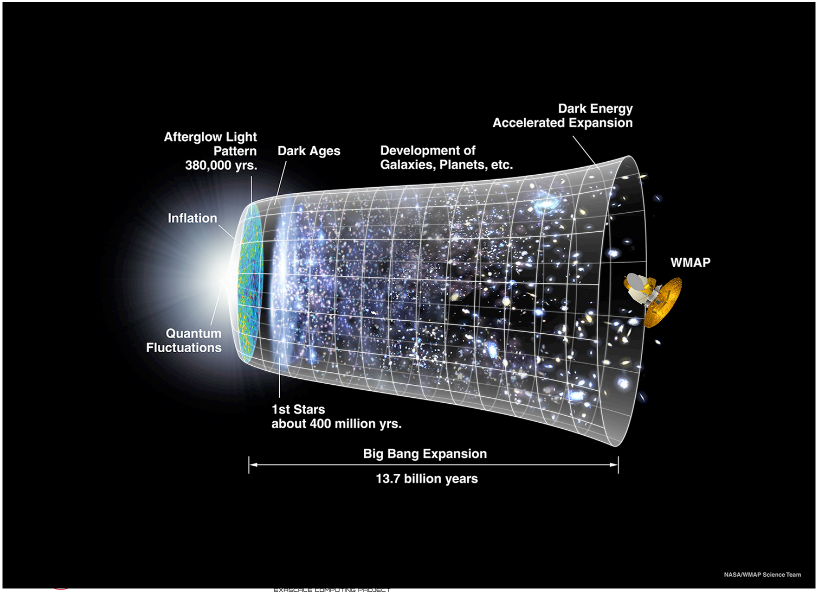


[http://svs.gsfc.nasa.gov/cgi-bin/details.cgi?aid=10118]

- Each bright spot above is an entire galaxy; purple filaments show where material connects these. To the eye, only the galaxies are visible.
- This visualization spans 134 Mpc (437 million light-years) per side.

[In collaboration w/ M. Norman (UCSD), B. O'Shea (MSU), J. Wise (GA Tech) and the *ENZO* team]

Cosmic Reionization – Timeline


Afterglow Light Pattern 380,000 yrs.
Dark Ages
Development of Galaxies, Planets, etc.
Dark Energy Accelerated Expansion
Inflation
Quantum Fluctuations
WMAP
1st Stars about 400 million yrs.
Big Bang Expansion
13.7 billion years
NASA/WMAP Science Team

Cosmology Multiphysics Model

[Bryan et al., 1995; R. et al., 2009; Norman, R. & So, 2009; Bryan et al., 2014]

We model early universe cosmology with the following equations (from slow $\to$ fast).

- Cold dark matter motion ($k = 1, \ldots, N_d$), cosmological expansion:

$$\dot{\mathbf{q}}_{d,k} = \mathbf{v}_{d,k}, \qquad \dot{\mathbf{v}}_{d,k} = -\frac{1}{m_{d,k}}\nabla\phi,$$
$$\nabla^2\phi = \frac{4\pi G}{a}\left[\rho_b + \rho_d(\mathbf{q_d}) - \rho_0\right],$$
$$\frac{\ddot{a}}{a} = -\frac{4\pi G}{3a^3}\left(\rho_0 + 3\frac{p_0}{c^2}\right) + \frac{\Lambda}{3}, \qquad \mathbf{x} \equiv \frac{\mathbf{r}}{a(t)}.$$

- Hydrodynamic motion (conservation of mass, momentum and energy):

$$\partial_t\rho_b + \frac{1}{a}\mathbf{v}_b\cdot\nabla\rho_b = -\frac{1}{a}\rho_b\nabla\cdot\mathbf{v}_b,$$
$$\partial_t\mathbf{v}_b + \frac{1}{a}\left(\mathbf{v}_b\cdot\nabla\right)\mathbf{v}_b = -\frac{\dot{a}}{a}\mathbf{v}_b - \frac{1}{a\rho_b}\nabla p - \frac{1}{a}\nabla\phi,$$
$$\partial_t e + \frac{1}{a}\mathbf{v}_b\cdot\nabla e = -\frac{2\dot{a}}{a}e - \frac{1}{a\rho_b}\nabla\cdot\left(p\mathbf{v}_b\right) - \frac{1}{a}\mathbf{v}_b\cdot\nabla\phi.$$

- Radiation chemistry ($\nu = 1, \ldots, N_f, \quad i,j = 1, \ldots, N_c$):

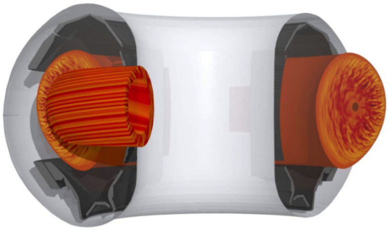
$$\partial_t E_\nu + \nabla\cdot\left(E_\nu\mathbf{v}_b\right) - \nabla\cdot\left(D\,\nabla E_\nu\right) = \frac{\nu\dot{a}}{a}\partial_\nu E_\nu - \frac{3\dot{a}}{a}E_\nu + \eta_\nu - c\kappa_\nu E_\nu,$$
$$\partial_t\mathbf{n}_i + \nabla\cdot\left(\mathbf{n}_i\mathbf{v}_b\right) = -\mathbf{n}_i\Gamma_i^{ph} + \alpha_{i,j}^{rec}\mathbf{n}_e\mathbf{n}_j.$$

SMU ECP EXASCALE COMPUTING PROJECT FASTMATH sundials

Fusion Plasma Simulations

Large-scale, nonlinear simulation of fusion plasmas is critical for the design of next-generation confinement devices.

- Fusion easy to achieve but difficult to *stabilize*, as needed to increase yield and protect device.
- Linear modes present in fluid models are typically well-controlled.
- Most current work focuses on disruptions due to nonlinear instabilities and kinetic effects.
- Turbulence in the sharp edge controls the core, but is difficult to simulate:
    - must accurately couple ions and electrons in high dimensions: $\mathbf{x} \in \mathbb{R}^d$, $\mathbf{v} \in \mathbb{R}^d$, $t \in \mathbb{R}$; $d = \{2, 3\}$
    - mass/velocity differences result in $100\times$ spatial/temporal scale separation.


GENE gyrokinetic simulation of core turbulence

[In collaboration w/ D. Ernst, M. Francisquez, MGK SciDAC Project]

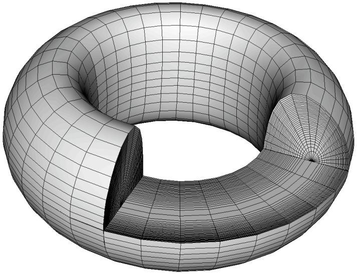
Visco-Resistive MHD Model

[R., Samtaney & Tiedeman, 2012; R. & Samtaney, 2012]

Even fluid-only models present multiphysics challenges. Consider a "simple" 4-dimensional model of tokamak plasma:

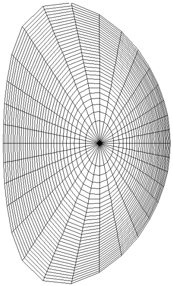
$$\partial_t \mathbf{U} + \frac{1}{r\mathcal{J}}\left[\partial_\xi(r\tilde{\mathbf{F}}(\mathbf{U})) + \partial_\eta(r\tilde{\mathbf{H}}(\mathbf{U})) + \partial_\varphi(\tilde{\mathbf{G}}(\mathbf{U}))\right] = \mathbf{S}(\mathbf{U}) + \nabla\cdot\tilde{\mathbf{F}}_d(\mathbf{U}),$$

where $\mathbf{U} = (\rho, \rho\mathbf{u}, \mathbf{B}, e)^T$.



Left: toroidal tokamak domain, with slice removed to show grid structure.

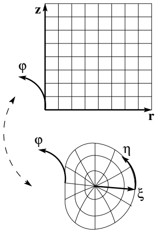
Right: poloidal cross-section.



Visco-Resistive MHD Model (continued)

The hyperbolic fluxes are given by

$$\begin{aligned}\tilde{\mathbf{F}} &= \mathcal{J}\left(\partial_r\xi\,\mathbf{F} + \partial_z\xi\,\mathbf{H}\right) = \partial_\eta z\,\mathbf{F} - \partial_\eta r\,\mathbf{H},\\ \tilde{\mathbf{H}} &= \mathcal{J}\left(\partial_r\eta\,\mathbf{F} + \partial_z\eta\,\mathbf{H}\right) = \partial_\xi z\,\mathbf{F} - \partial_\xi r\,\mathbf{H},\\ \tilde{\mathbf{G}} &= \mathcal{J}\mathbf{G},\end{aligned}$$



where $\xi = \xi(r,z)$, $\eta = \eta(r,z)$ and $\mathcal{J} = (\partial_\xi r)(\partial_\eta z) - (\partial_\eta r)(\partial_\xi z)$ map from cylindrical to tokamak coordinates. Here,

$$\begin{aligned}\mathbf{F} = \big(&\rho u_r,\;\; \rho u_r^2 + \tilde{p} - B_r^2,\;\; \rho u_r u_\varphi - B_r B_\varphi,\;\; \rho u_r u_z - B_r B_z,\;\; 0,\\ &u_r B_\varphi - u_\varphi B_r,\;\; u_r B_z - u_z B_r,\;\; (e+\tilde{p})u_r - (\mathbf{B}\cdot\mathbf{u})B_r\big),\\ \mathbf{G} = \big(&\rho u_\varphi,\;\; \rho u_r u_\varphi - B_r B_\varphi,\;\; \rho u_\varphi^2 + \tilde{p} - B_\varphi^2,\;\; \rho u_z u_\varphi - B_z B_\varphi,\\ &u_\varphi B_r - u_r B_\varphi,\, 0,\, u_\varphi B_z - u_z B_\varphi,\, (e+\tilde{p})u_\varphi - (\mathbf{B}\cdot\mathbf{u})B_\varphi\big),\\ \mathbf{H} = \big(&\rho u_z,\;\; \rho u_r u_z - B_r B_z,\;\; \rho u_z u_\varphi - B_z B_\varphi,\;\; \rho u_z^2 + \tilde{p} - B_z^2,\\ &u_z B_r - u_r B_z,\;\; u_z B_\varphi - u_\varphi B_z,\;\; 0,\;\; (e+\tilde{p})u_z - (\mathbf{B}\cdot\mathbf{u})B_z\big),\end{aligned}$$

$\tilde{p} = p + \frac{\mathbf{B}\cdot\mathbf{B}}{2}$, $e = \frac{p}{\gamma-1} + \frac{\rho\mathbf{u}\cdot\mathbf{u}}{2} + \frac{\mathbf{B}\cdot\mathbf{B}}{2}$, and $\nabla\cdot\mathbf{F}_d(\mathbf{U})$ adds a small amount of diffusion (viscosity, resistivity, heat conduction).

A Sampling of Multiphysics Challenges

These multiphysics problems exhibit key characteristics that challenge "traditional" numerical methods:

- "Multirate" structure: different processes evolve on distinct time scales, but these are too close to analytically reformulate (e.g., via steady-state approximation).
- The existence of stiff components prohibits fully explicit methods.
- Nonlinearity and insufficient differentiability prohibit fully implicit methods.
- "Multiscale" structure: some spatial regions may be well-modeled via coarse meshes, while others require high resolution.
- Need for extreme parallel scalability requires use of optimal algorithms. While robust and scalable algebraic solvers exist for some pieces (e.g., FMM for particles, multigrid for diffusion), none yet apply to the full combination.

I make no claim to have solved all of the above problems, only to point out these challenges and state that work is needed to address each.

Need for Flexible Methods and Robust Software Libraries

With respect to their temporal aspects, multiphysics problems have significant (and sometimes contradictory) needs:

- Stability/accuracy for each component, as well as inter-physics couplings
- Custom/flexible time step sizes for distinct components
- Robust temporal error estimation & adaptivity of step size(s)
- Built-in support for spatial adaptivity
- Ability to apply optimally efficient and scalable solver algorithms
- Support for experimentation and testing between methods and solution algorithms

"Classical" Time Integrators (and their deficiencies)

Historically, ODE research has focused on two simple problem types:

$$\dot{y}(t) = f(t, y(t)), \qquad y(t_0) = y_0 \qquad \text{[ODE]}$$

$$0 = F(t, y(t), \dot{y}(t)), \qquad y(t_0) = y_0, \quad \dot{y}(t_o) = \dot{y}_0 \qquad \text{[DAE]}$$

Corresponding solvers thus enforced overly-rigid standards:

- Treat all components implicitly or explicitly, without IMEX flexibility.
  - Fully explicit: "stiff" components require overly-small Δt for stability.
  - Fully implicit: highly nonlinear or nonsmooth terms present difficulties for scalable/robust algebraic solvers.
- Inflexible vector/matrix/solver data structures. While contiguous 1D vectors and matrices work well in LAPACK/MATLAB, these are rarely optimal for large-scale, multiphysics problems.
- Software was hard-coded for specific methods and parameters – while these are decent for most problems, they're rarely optimal for any.

Ad Hoc Algorithms Pervade Scientific Computing Applications

Practitioners, on the other hand, frequently "split" their problems and solve each component separately over a time step $[t_0, t_0 + \Delta t]$:

$$\dot{y}(t) = f_1(t,y) + \cdots + f_m(t,y), \quad y(t_0) = y_0$$

$$\approx$$

$$\dot{y}^{(1)}(t) = f_1\left(t, y^{(1)}\right), \quad y^{(1)}(t_0) = y_0,$$

$$\vdots$$

$$\dot{y}^{(m)}(t) = f_m\left(t, y^{(m)}\right), \quad y^{(m)}(t_0) = y^{(m-1)}(t_0 + \Delta t),$$

While each component may be tackled independently (or even subcycled) using, e.g., something from "Numerical Recipes," the resulting methods suffer from:

- Low accuracy – typically $\mathcal{O}(h)$-accurate; symmetrization/extrapolation may improve this but at significant cost [Ropp, Shadid & Ober 2005].
- Poor/unknown stability – even when each part utilizes a 'stable' step size, the combined problem may admit unstable modes [Estep et al., 2007].

Outline

Meeting in the middle – modern methods for split systems

In recent years, applied mathematics has begun to catch up to standard scientific computing practice, designing accurate and stable time integration methods for split multiphysics systems in both additively-partitioned form:

$$\dot{y}(t) = f_1(t,y) + \cdots + f_m(t,y), \quad y(t_0) = y_0$$

and in variable-partitioned form:

$$\begin{aligned} \dot{y}_1(t) &= f_1\left(t,y\right), \quad y_1(t_0) = y_{1,0} \\ &\vdots \\ \dot{y}_m(t) &= f_m\left(t,y\right), \quad y_m(t_0) = y_{m,0}, \end{aligned}$$

where $y = \begin{bmatrix} y_1 & \cdots & y_m \end{bmatrix}^T$.

In the next slides I highlight a number of these methods, discussing my work in each.

Additive Runge–Kutta (ARK) Methods [Ascher et al. 1997; Araújo et al. 1997; . . .]

In 2014, I released the ARKode package as a component of SUNDIALS, providing adaptive ARK methods that support up to two split components: *explicit* and *implicit*,

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

- M is any nonsingular linear operator (mass matrix, typically $M = I$),
- $f^E(t,y)$ contains the explicit terms,
- $f^I(t,y)$ contains the implicit terms.

Combine two s-stage RK methods; denoting $t^*_{n,j} = t_n + c^*_j h_n$, $h_n = t_{n+1} - t_n$:

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

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

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

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

Each stage is implicitly defined via a root-finding problem:

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

- If $f^I(t,y)$ is *linear* in y then we must solve a linear system for each z_i,
- Else G_i is nonlinear, requiring an iterative solver – `ARKode` options include:
  - *Newton*: inexact or 'standard' (depends on linear solver),
    - Scaled, preconditioned, Krylov; "matrix-free" available.
    - *user-supplied* to exploit linear system structure
  - Fixed-point, with optional Anderson acceleration.
  - User-supplied (new feature).

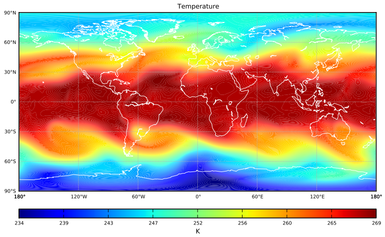
Multiphysics Flexibility Enhancements

ARKode includes numerous additional enhancements for multiphysics codes:

- All solvers (except direct linear) formulated via generic vector operations:
  - Numerous supplied vector implementations: serial, MPI, OpenMP, PETSc, *hypre*, CUDA, Raja, Trilinos, ...
  - Application-specific vectors may be easily supplied.
- Variety of built-in Butcher tables (ERK, DIRK, IMEX ARK) and temporal adaptivity controllers; either may be user-supplied.
- Variety of built-in implicit predictor algorithms.
- Ability to resize data structures based on changing IVP size.
- All internal solver parameters are user-modifiable.

Nonhydrostatic Climate [Gardner et al., 2018; Vogl et al., 2019; Ullrich et al., 2018]

We have applied ARKode to a variety of large-scale applications, including the climate codes *Tempest* & *HOMME-NH* (equations shown earlier).



Examined:

- 5 IMEX splittings & 21 published ARK methods for accuracy/stability.
- 13 custom ARK methods that optimize explicit stability along the imaginary axis.
- Various (non)linear solver algorithms for implicit components.
- Effects of "standard" stabilization approaches (hyperviscosity, vertical remap).

Findings:

- ARKode's modular structure allowed rapid exploration of "solver space."
- Stability $\propto$ implicitness, but horizontally implicit terms *significantly* increase cost.
- Best overall ARK methods were those we designed for this application.
- Linearly-implicit solves fine for current h_n, but nonlinear effects become more relevant at desired h_n.

SMU EXASCALE COMPUTING PROJECT FASTMATH sundials

Exponential Rosenbrock (ExpRB) Methods [Hochbruch et al., 2009; Luan & Ostermann, 2014]

Exponential Rosenbrock methods consider a specific additive splitting of the IVP:

$$\dot{y} = f(y) = \mathcal{J}(y)y + \mathcal{N}(y), \quad t \in [t_0, t_f], \quad y(t_0) = y_0,$$

- $\mathcal{J}(y) \equiv \frac{\partial f(y)}{\partial y}$ is the Jacobian of f, and
- $\mathcal{N}(y) \equiv f(y) - \mathcal{J}(y)y$ contains the remaining nonlinearities [assumed nonstiff].

This has analytical solution over $t \in [t_n, t_n + h]$:

$$y(t) = \mathrm{e}^{(t-t_n)\mathcal{J}(y_n)} y(t_n) + \int_0^t \mathrm{e}^{(t-\tau)\mathcal{J}(y_n)} \mathcal{N}(u(t_n+\tau))\,\mathrm{d}\tau.$$

By approximating the integral via quadrature, an s-stage ExpRB method may be written:

$$z_i = y_n + c_i h \varphi_1(c_i h \mathcal{J}_n) f(y_n) + h \sum_{j=2}^{i-1} a_{ij}(h\mathcal{J}_n)\left(\mathcal{N}_n(z_j) - \mathcal{N}_n(y_n)\right),$$

$$y_{n+1} = y_n + h\varphi_1(h\mathcal{J}_n) f(y_n) + h \sum_{i=2}^{s} b_i(h\mathcal{J}_n)\mathcal{N}_n(z_i) - \mathcal{N}_n(y_n)$$

where $z_1 = y_n$, $\mathcal{J}_n \equiv \mathcal{J}(y_n)$, $\mathcal{N}_n \equiv \mathcal{N}(y_n)$, and $\varphi_1(z) \equiv (e^z - 1)/z$.

ExpRB "Coefficients" [Luan & Ostermann, 2014; Gaudreault & Pudykiewicz, 2016; Niesen & Wright, 2012]

The matrix functions $a_{ij}(h\mathcal{J}_n)$ and $b_i(h\mathcal{J}_n)$ are usually linear combinations of the functions $\varphi_k(c_i h\mathcal{J}_n)$ and $\varphi_k(h\mathcal{J}_n)$, resp.; these are defined recursively via

$$\varphi_{k+1}(z) \equiv \frac{\varphi_k(z) - \frac{1}{k!}}{z}, \quad k \geq 1.$$

Thus the primary challenge in applying ExpRB methods is to efficiently compute linear combinations of these matrix functions multiplied by vectors,

$$w_k = \sum_{l=0}^{p} \varphi_l(c_k\, A) v_l, \quad k = 2, \ldots, s,$$

where the values $c_k \in (0,1]$ denote the "time" scaling factors used for each w_k output. These w_k may equivalently be computed as the solutions $u(c_k)$ of

$$u'(t) = Au(t) + v_1 + tv_2 + \cdots + \frac{t^{p-1}}{(p-1)!} v_p, \quad u(0) = v_0. \tag{1}$$

Efficient Methods for Computing $\{w_k\}$ [Luan, Pudykiewicz & R., 2019]

Since the $\{c_j\}$ are known *a priori*, we subcycle (1) over substeps $0 = t_0 < t_1 < \cdots < t_K = 1$ such that each c_j aligns with a t_k.

Exploiting various recursion relations, we compute each substep using only a single φ_k function,

$$u(t_{k+1}) = \tau_k^p \varphi_p(\tau_k A) x_p + \sum_{j=0}^{p-i} \frac{t_k^j}{j!} x_j,$$

where the vectors x_j satisfy another recurrence relation,

$$x_0 = u(t_k), \; x_j = A x_{j-1} + \sum_{\ell=0}^{p-j} \frac{t_k^\ell}{\ell!} v_{j+\ell}, \; j = 1, \ldots, p.$$

In 2018, Luan and I released a MATLAB implementation of this as the `phipm_simul_iom` algorithm. We hope to extend this to a parallel `ARKode` module in the near future.

Application to the 2D Shallow Water equations [Luan, Pudykiewicz & R., 2019]

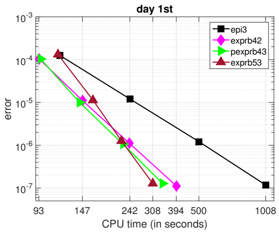
Simplified 2D model used in climate and weather prediction:

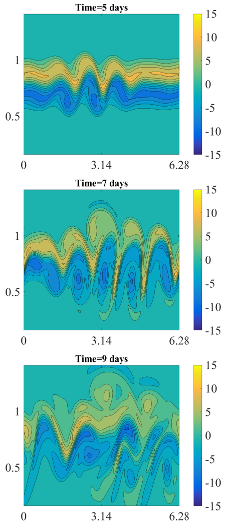
$$\frac{\partial \mathbf{u}}{\partial t} = -(\nabla_n \times \mathbf{u} + f\mathbf{n}) \times \mathbf{u} - \nabla\left(\frac{|\mathbf{u}|^2}{2} + g(h + h_s)\right),$$
$$\frac{\partial h}{\partial t} = -\nabla \cdot (h\mathbf{u}),$$

$\mathbf{u}$ is the velocity, h is the fluid thickness, h_s is the surface level, g is the gravitational acceleration, and f is the Coriolis parameter.

ExpRB methods achieved:

- $\sim 700x$ increase in usable step size over state-of-the-practice IMEX methods,
- $\sim 3x$ increase over previous state-of-the-art exponential RK methods [Gaudreault & Pudykiewicz, 2016].





Multirate Infinitesimal Step (MIS) methods [Knoth & Wolke 1998; Schlegel et al. 2009; ...]

MIS methods arose in the numerical weather prediction community. This generic infrastructure supports $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$ methods for multirate problems:

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

- $f^F(t,y)$ contains the "fast" terms; $f^S(t,y)$ contains the "slow" terms.
- $h^S > h^F$, with a time scale separation $h^S/h^F \approx m$.
- y is frequently partitioned as well, e.g. $y = \left[y^F \; y^S\right]^T$.
- The slow component may be integrated using an explicit "outer" RK method, $T_O = \{A, b, c\}$, where $c_i \le c_{i+1}$, $i = 1, \ldots, s-1$.
- The fast component is advanced between slow stages as the exact solution of a modified ODE.
- Practically, this fast solution is subcycled using an "inner" RK method.

MIS Algorithm

Denoting $y_n \approx y(t_n)$, a single MIS step $y_n \to y_{n+1}$ is as follows:

1. Set $z_1 = y_n$
2. For each outer Runge-Kutta stage $z_i, i = 2, \ldots, s+1$:
   a) Let $v(t_{n,i-1}) = z_{i-1}$ and $r = \sum_{j=1}^{i-1} \left(\frac{a_{i,j} - a_{i-1,j}}{c_i - c_{i-1}} \right) f^S(t_{n,j}, z_j)$
   b) Solve the fast ODE: $\dot{v}(\tau) = f^F(\tau, v) + r$, for $\tau \in [t_{n,i-1}, t_{n,i}]$
   c) Set $z_i = v(t_{n,i})$
3. Set $y_{n+1} = z_{s+1}$

where the outer stage times are $t_{n,j} = t_n + c_j h^S$ and $a_{s+1,j} = b_j$.

- When $c_i = c_{i+1}$, the IVP "solve" reduces to a standard Runge–Kutta update.
- Step 2b may use any applicable algorithm of sufficient accuracy.

MIS Properties

MIS methods satisfy a number of desirable multirate method properties:

- The MIS method is $\mathcal{O}(h^2)$ if both inner/outer methods are at least $\mathcal{O}(h^2)$.
- The MIS method is $\mathcal{O}(h^3)$ if both inner/outer methods are at least $\mathcal{O}(h^3)$, and T_O satisfies

$$\sum_{i=2}^{s} (c_i - c_{i-1})(e_i + e_{i-1})^T Ac + (1 - c_s)\left(\frac{1}{2} + e_s^T Ac\right) = \frac{1}{3}.$$

- The inner method may be a subcycled T_O, enabling a *telescopic* multirate method (i.e., n-rate problems supported via recursion).
- Both inner/outer methods can utilize problem-specific tables (SSP, etc.).
- h^F may be varied within a slow step to adapt the multirate structure.
- Highly efficient – only a single traversal of $[t_n, t_{n+1}]$ is required. To our knowledge, MIS are the most efficient $\mathcal{O}(h^3)$ multirate methods available.

MRIStep ARKode module [released Dec. 2018]

Our new MRIStep module in ARKode supports $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$ MIS-like methods:

- Slow scale integrated explicitly (*currently exploring implicit versions*).
- Fast scale is advanced by calling legacy ARKode solvers – i.e., may be explicit, implicit or IMEX.
- Applied to chemically reacting flow test (cosmology prototype), and achieved 90% weak scaling parallel efficiency using over 138k CPU cores on Summit. *We are currently expanding this to perform chemistry on the GPU*.
- Currently applying to kinetic fusion plasma turbulence code.
- Extensions to $\mathcal{O}(h^4)$ and higher are under investigation:
    - Our new *RMIS* method [w/ Jean Sexton, LBNL] computes y_{n+1} as a combination of $\{f(t_{n,i}, z_i)\}$;
    - Our new *MERK* method [w/ Vu Luan (MSState) & Rujeko Chinomona (SMU)] constructs fast IVP using exponential integrators;
    - etc.

Outline

Conclusions

Large-scale multiphysics problems:

- Challenges arise due to nonlinear, interacting models.
- Typically large data requirements, thus requiring scalable approximation methods.
- Often, each physical process admits an "optimal" solution algorithm and time scale, though unfortunately these approaches rarely agree.
- Most "classical" methods were invented for idealized problems, and may not work well (or at all) on "real world" applications.

My research aims to develop flexible solvers, that tune the algorithms to the problem (instead of vice-versa), and to implement these in high-quality, open-source software that directly impacts multiphysics applications:

- Method derivation:
  - High-order multirate methods.
  - Scalable/efficient methods for exponential integrators.
- Software:
  - Support ERK, DIRK, ARK IMEX single-rate methods; explicit+X multirate methods.
  - Strive for flexibility in data structures and solver algorithms, enabling user-supplied components that can be optimized for a given problem.

SMU

ECP EXASCALE COMPUTING PROJECT

FASTMATH

sundials

Thanks & Acknowledgements

Collaborators:

- SMU: R. Chinomona, T. Yan
- LLNL: C. Woodward, D. Gardner, C. Vogl, A. Hindmarsh
- LBL: J. Sexton
- Missippi State: V. Luan
- SNL: M. Taylor, A. Steyer
- UC Davis: P. Ullrich, J. Guerra
- MIT: D. Ernst
- PPPL: M. Francisquez
- Environment Canada: J. Pudykiewicz
- UCSD: M. Norman, J. Bordner

Grant/Computing Support:

- DOE SciDAC, ECP & INCITE Programs
- NSF AST & XSEDE Programs
- SMU Center for Scientific Computation
- DOD DURIP Program