Multirate and IMEX methods in ARKode/SUNDIALS

Daniel R. Reynolds[1], David J. Gardner[2], Carol S. Woodward[2]

reynolds@smu.edu, gardner48@llnl.gov, cswoodward@llnl.gov

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

MGK SciDAC Meeting
Austin, Texas
21-22 March 2019

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

ARKode was initially designed to implement adaptive ARK methods for initial value problems (IVPs), supporting 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 $t^*_{n,j} = t_n + c^*_j h_n$, $h_n = t_{n+1} - t_n$:

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

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

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

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

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

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

- 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 – all generic SUNDIALS nonlinear solvers avaialble (*or user supplied*).

SMU
ECP EXASCALE COMPUTING PROJECT
sundials
FASTMATH

Reconfiguring ARKode into an infrastructure

Over the last year, we have overhauled ARKode to serve as an infrastructure for general, adaptive, one-step time integration methods:

- ARKode provides the outer time integration loop and generic usage modes (interpolation vs "tstop"; one-step versus time interval).
- Time-stepping modules handle problem-specific components: definition of the IVP, algorithm for a single time step.
- Time-stepping modules may leverage shared ARKode infrastructure:
  - SUNDIALS' vector, matrix, linear solver and nonlinear solver objects,
  - translation between SUNDIALS' generic matrix/solver structures ($\mathcal{A}x = b$) and IVP-specific linear systems ($\mathcal{A} \approx M - \gamma \frac{\partial f^I}{\partial y}(t, y)$),
  - time-step adaptivity controllers: PID, PI, I, *user-supplied*,
  - . . .

SMU

ECP EXASCALE COMPUTING PROJECT

sundials

FASTMATH

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

MIS/RFSMR 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:

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

- $f^{\{f\}}(t,y)$ 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 $c_i \le c_{i+1}$, $i = 1, \ldots, s$;
- the fast component is advanced between slow stages by solving 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 $y_n \to y_{n+1}$ has the generic form:

Set $z_1 = y_n,$

For $i = 1, \ldots, s:$

Let $t_{n,i} = t_n + c_i h_s$ and $v(t_{n,i}) = z_i$, then for $\tau \in [t_{n,i}, t_{n,i+1}]$ solve:

$$\dot{v}(\tau) = f^{\{f\}}(\tau, v) + \sum_{j=1}^{i} \alpha_{i+1,j} f^{\{s\}}(t_{n,j}, z_j),$$

Set $z_{i+1} = v(t_{n,i+1})$

Set $y_{n+1} = z_{s+1},$

where the coefficients $\alpha_{i,j}$ are defined appropriately.

The IVP for $v(\tau)$ may be solved using any applicable algorithm.

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 table (SSP, etc.).
- Highly efficient – only a single traversal of $[t_n, t_n + h]$ is required. To our knowledge, MIS are the most efficient $\mathcal{O}(h^3)$ multirate methods available.

MRIStep ARKode stepper

David Gardner has implemented a new *MRIStep* module to support $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$ MIS-like methods [released Dec. 2018].

- Currently requires user-defined h_s and h_f (may be varied between outer steps). *We are currently expanding this to support temporal adaptivity*.
- Slow time scale is integrated with an ERK method. *We are currently exploring methods with an implicit slow component*.
- Fast scale is advanced by calling the ARKStep module. Current release requires ERK fast scale, but *implicit and ImEx will be released soon*.
- Extensions to $\mathcal{O}(h^4)$ and higher are under investigation:
  - J.M. Sexton's *RMIS* computes y_{n+1} as a combination of $\{f(t_{n,i}, z_i)\}$;
  - V.T. Luan's *MERK* constructs fast IVP using exponential integrators;
  - A. Sandu's *MRI-GARK* modifies the fast IVP:

$$\dot{v}(\tau) = f^{\{f\}}(\tau, v) + \sum_{j=1}^{i+1} \gamma_{i,j}\left(\frac{\tau - t_{n,i}}{h_s}\right) f^{\{s\}}(t_{n,j}, z_j).$$

Conclusions

The ARKode infrastructure flexibly supports extensive studies of optimal algorithms for multiphysics problems:

- Numerous built-in ERK, DIRK, and ARK methods; supports user-supplied.
- Numerous vector/matrix data structures, support for user-supplied and data partitioned.
- Numerous algebraic solver algorithms, support for user-supplied.
- Actively developing state-of-the-art flexible time integration methods for multi-physics applications:
  - Additive partitioning – break apart physical processes based on stiffness (implicit/explicit/IMEX) or time scale (fast/slow).
  - Variable partitioning – break apart solution based on time scales (fast/slow) or solvers (algebraic, computing hardware).
  - Focus on ease-of-use and support for user-supplied components, so that critical methods can be highly optimized for a given problem.

Thanks & Acknowledgements

Collaborators/Students:

- Rujeko Chinomona [SMU, PhD]
- Vu Thai Luan [SMU, postdoc]
- John Loffeld [LLNL]
- Jean M. Sexton [LBL]

Current Grant/Computing Support:

- DOE SciDAC & ECP Programs
- SMU Center for Scientific Computation

Software:

- ARKode – http://faculty.smu.edu/reynolds/arkode
- SUNDIALS – https://computation.llnl.gov/casc/sundials

Support for this work was provided by the Department of Energy, Office of Science project "Frameworks, Algorithms and Scalable Technologies for Mathematics (FASTMath)," under Lawrence Livermore National Laboratory subcontract B626484.

References

- Ropp & Shadid, *J. Comput. Phys.*, 203, 2005.
- Estep et al., *Comput. Meth. Appl. Mech. Eng.*, 196, 2007.
- Ascher et al., *Applied Numerical Mathematics*, 25, 1997.
- Araújo et al., *SIAM J. Numer. Anal.*, 34, 1997.
- Gardner et al., *Model. Simul. Mater. Sci. Eng.*, 23, 2015.
- Gardner et al., *Geosci. Model Dev.*, 11, 2018.
- Vogl et al., *in preparation*, 2019.
- Knoth & Wolke, *Appl. Numer. Math.*, 1998.
- Schlegel et al., *J. Comput. Appl. Math.*, 2009.
- Sandu & Günther, *SINUM.*, 2015.
- Sexton & Reynolds, *arXiv:1808.03718*, 2018.
- Sandu, *arXiv:1808.02759*, 2018.
- Luan, Chinomona & Reynolds, *in preparation*, 2019.