New Features in SUNDIALS (v7.2.0+)

SUNDIALS User Experiences Birds of a Feather Session
2026 CASS BoF Days

12 February 2026

Daniel R. Reynolds [UMBC]

U.S. DEPARTMENT OF ENERGY | Office of Science

SciDAC
Scientific Discovery through Advanced Computing

Lawrence Livermore National Laboratory

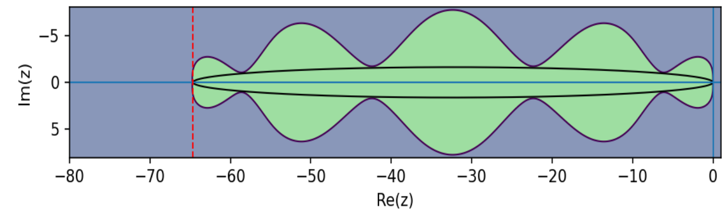
UMBC

Outline

- Low-Storage Runge—Kutta Methods [`LSRKStep`]
- Dominant Eigenvalue Estimation [`SUNDomEigEstimator`]
- Operator Splitting Methods [`SplittingStep`, `ForcingStep`]
- Discrete Adjoint Sensitivity Analysis in ARKODE [`ERKStep`, `ARKStep`]
- Command-line Control [all]

Low-Storage Runge—Kutta Methods in LSRKStep

The new LSRKStep in ARKODE provides methods with a low-storage implementation. This currently includes two families of temporally-adaptive explicit methods:



Above: RKC2 with 10 stages

- Super-Time-Stepping (STS) methods target diffusion equations:

  $$y'(t) = f(t,y),\ \ y(t_0) = y_0,\ \ \lambda\left(\partial_y f\right) \subset \mathbb{R}^-,\ \ \lambda_{max} \geq O(\Delta x^{-2})$$

  - Circumvent stability limitations by adding stages, *s*.
  - Includes two 2nd order methods: *Runge—Kutta—Chebyshev (RKC) and Runge—Kutta—Legendre (RKL)* [van der Houwen & Sommeijer 1980; Meyer, Balsara & Aslam 2012].
- Strong-Stability-Preserving (SSP) methods target hyperbolic problems (including shocks):
  - Includes Ketcheson's "optimal" methods: SSP(*s*,2), SSP(*s*,3), SSP(10,4), with embeddings by Fekete et al [Ketcheson 2008; Fekete et al 2022].
  - Our tests show that these can adaptively track Δt_{SSP} when using loose relative tolerance of ~10^{-2}.
- All LSRKStep methods require no storage beyond existing ARKODE infrastructure (independent of *s*).
- Adaptivity algorithms enable application to nonlinear problems with non-constant time scales.

Dominant Eigenvalue Estimation

STS methods require the dominant eigenvalue of the Jacobian, $\lambda_{max}(\partial_y f)$, to determine the number of stages:

- For $\lambda_{max} \in \mathbb{R}^-$, for linear stability RKL2 and RKC2 with "standard" damping require $(s^2 + s - 2) > 2\,\Delta t\,\lambda_{max}$ and $s^2 > 1.54\,\Delta t\,\lambda_{max}$, resp.
- SUNDIALS v7.2.0 (Dec. 2024) required a user-supplied function to provide λ_{max}
- SUNDIALS v7.5.0 (Sept. 2025) added the `SUNDomEigEstimator` class, that estimates λ_{max} iteratively, using only evaluations of f and standard vector operations.
  - `SUNDomEigEst_Power`: for $\lambda \in \mathbb{R}$, performs power iterations (cf. Google's PageRank) until convergence
    - Only need ~2 digits for STS methods; Gkeyll-based DG tests converge in ≤ 5 iterations
  - `SUNDomEigEst_Arnoldi`: for $\lambda \in \mathbb{C}$, we follow-up power iterations with ~3 Arnoldi iterations
    - Each Arnoldi iteration requires 1 storage vector
- Both are in active development, with additional upgrades forthcoming: standalone usage for IVP analysis, stability-based explicit method Δt selection, support for $\lambda \in \mathbb{C}$ in `SUNDomEigEst_Power`, …

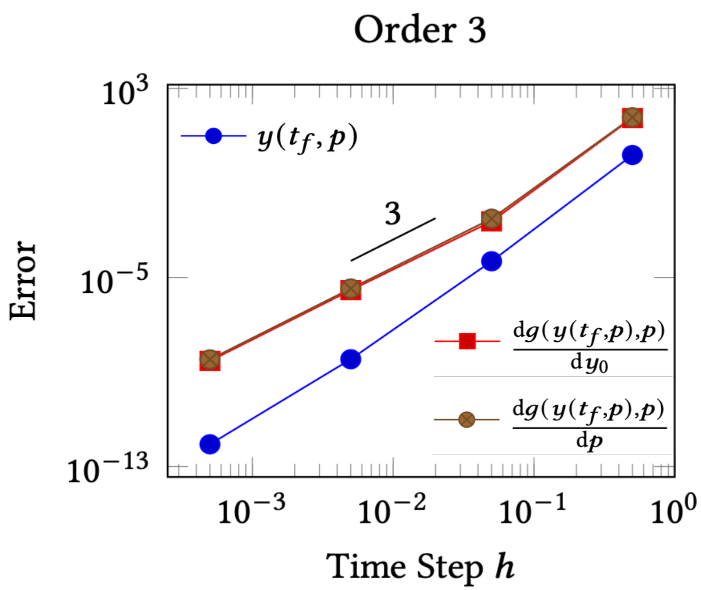
Operator-Splitting Methods in `SplittingStep`

- The new SplittingStep module for ARKODE implements operator splitting methods for an arbitrary number of partitions
$$y'(t) = f_1(t, y) + f_2(t, y) + \cdots + f_P(t, y)$$
- Operator splitting is an old and simple idea: integrate partitions individually in a sequence. Coupling is weak but unintrusive.
- We support many standard methods
  - Lie-Trotter (first order)
  - Strang (second order)
  - Triple- and Quintuple-jump (arbitrary order)
  - Custom coefficients
- The solution of inner ODEs, $y'(t) = f_\mathrm{i}(t, y)$, is very flexible. Users can control the order in which partitions are integrated and chose different integrators for different partitions, e.g., SUNDIALS integrators or custom solution procedures.
- A related `ForcingStep` module handles $P = 2$, coupling them through a constant forcing term.

Discrete Adjoint Sensitivity Analysis (ASA) [ERKStep, ARKStep]

Optimization problems often require computing gradients of a functional, $\partial_p g\big(t_f, y(t_f, p), p\big)$.

- Continuous ASA (as in CVODES & IDAS) may not recover the exact gradients of the discrete-time problem which can cause the optimizer to fail.
- We extended ARKODE's *explicit* Runge—Kutta methods to include discrete ASA, that provides the **exact gradient of the discrete-time problem**.
- Built on new base classes for discrete & continuous ASA, as well as different methods for managing checkpoints from the forward solution.
- Providing differentiable codes is key to leveraging gradient-based methods for optimizing parameters, e.g., for machine learning algorithms.



Results from the new discrete adjoint sensitivity analysis capability in ARKODE on the Lotka-Volterra model. The methods converge at the expected order for the forward solution as well as the adjoint solution.

Command-line Control [all packages]

- All SUNDIALS packages and modules have been upgraded to allow their scalar-valued options to be set from an array of strings (e.g., the command line).

- Each solver or module now offers an optional `SetOptions` routine to submit the array for processing, e.g.

```
retval = CVodeSetOptions(cvode_mem, NULL, NULL, argc, argv);
retval = SUNLinSolSetOptions(LS, “gmres”, NULL, argc, argv);
retval = SUNNonlinSolSetOptions(NLS, “aafp”, NULL, argc, argv);
```

- Each module has a unique prefix and reserved keys that it uses to parse its options, e.g.

```
$ a.out cvode.scalar_tolerances 1e-6 1e-8 \
  cvode.init_step 1e-2 \
  cvode.max_order 3 \
  cvode.max_num_steps 10000 \
  gmres.prec_type SUN_PREC_LEFT \
  aafp.max_iters 10
```

- Currently, only string processing is supported, but file-based processing is planned.