Shaping a Sustainable SUNDIALS:
Applying Software Sustainability Practices to a CSE Library

Cody J. Balos, David J. Gardner, Daniel R. Reynolds, Carol S. Woodward, Slaven Peles, and Alan C. Hindmarsh

The SUNDIALS Suite of Nonlinear Differential-Algebraic Equation Solvers is an open source software library including highly robust and adaptive time integration methods for ODEs and DAEs as well as robust nonlinear solvers. SUNDIALS has a long history of user deployment with over 25,000 downloads in 2018. We describe the sustainability practices adopted by the SUNDIALS team, including modular code design, investment in performance portability, continuous testing, and multi-faceted support of the user community.

Overview of SUNDIALS

Suite of ODE and DAE time integrators and nonlinear solvers
- Adaptive time integrators with forward and adjoint sensitivity capabilities
- Written in C with interfaces to Fortran and C++
- Designed to be incorporated into existing codes

Modular implementation
- Packages are built on shared vector, matrix, solver, and preconditioner APIs
- Users can supply their own data structures and solvers or use SUNDIALS supplied modules

Availability and support
- Used worldwide with more than 25,000 downloads in 2018
- Available from the LLNL software site, GitHub, and Spack
- Released under the BSD 3-Clause license
- Extensive user documentation, and active user support email list

SUNDIALS Packages

CVODE(S): adaptive order and step implicit linear multistep methods for **ODEs**

$$\frac{dy}{dt} = f(t,y), \quad y(t_0) = y_0$$

ARKode: adaptive step, explicit, implicit, additive, and multirate Runge-Kutta methods for **ODEs**

$$M\frac{dy}{dt} = f_1(t,y) + f_2(t,y), \quad y(t_0) = y_0$$

IDA(S): adaptive order and step BDF (implicit linear multistep) methods for **DAEs**

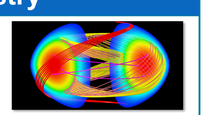
$$F(t,y,y') = 0, \quad y(t_0) = y_0, \quad y'(t_0) = y'_0$$

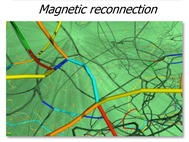
KINSOL: Newton and fixed-point methods for **nonlinear systems**
- Modified Newton and inexact Newton-Krylov methods
- Picard and fixed-point with Anderson acceleration
- Globalization: linesearch, inexact Newton

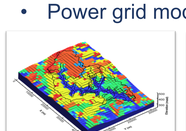
(S) Variant with forward and adjoint sensitivity analysis capabilities

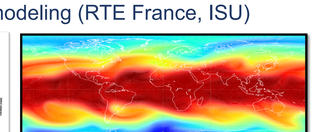
Applications in Research and Industry

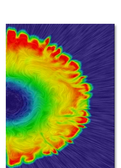
Used in application codes and in research worldwide:
- Computational Cosmology (Nyx)
- Combustion (PELE)
- Astrophysics (CASTRO)
- Atmospheric dynamics (DOE E3SM)
- AMReX (LBNL) and MFEM (LLNL)
- Dislocation dynamics (LLNL)
- 3D parallel fusion (U. York, LLNL)
- Power grid modeling (RTE France, ISU)

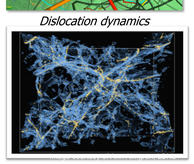

Magnetic reconnection


Dislocation dynamics


Subsurface flow


Atmospheric Dynamics

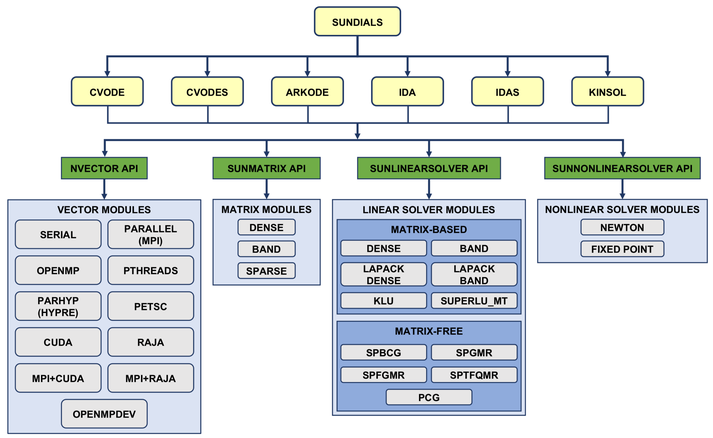

Core collapse supernovae


Cosmology

Modular Code Design

Modular code design in SUNDIALS provides extensibility that enables SUNDIALS to evolve as new paradigms and challenges arise

SUNDIALS packages are built on common modules for vectors, matrices, and solvers. The *NVector*, *SUNMatrix*, *SUNLinearSolver*, and *SUNNonlinearSolver* modules follow the same object oriented design pattern:
- Each module can be thought of as an abstract class that defines a generic API
- The *NVector* module defines routines for dot products, norms, etc., while the *SUNLinearSolver* module defines routines such as setup and solve
- Packages only interface with the generic APIs
- Module implementations provide the definitions of the generic routines
- Enables rapid development of new integration schemes by decoupling SUNDIALS packages from the underlying data structures and solvers



Modular code design provides flexibility that enables the use of SUNDIALS in an incredibly diverse set of applications

SUNDIALS ships with module implementations that interface with many popular math libraries to provide robust out-of-the-box functionality, but users can also supply their own module implementations:
- SUNDIALS-provided modules increase the number of applications in which it can be incorporated with minimal effort
- User-provided modules allow SUNDIALS to be flexible enough to work in almost any code base and makes it possible to develop highly-specialized, problem-specific module implementations once and reuse them across packages

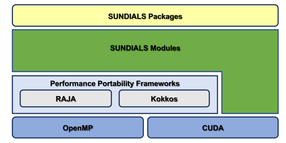
SuperLU_DIST · TRILINOS · PETSc · hypre

Modular code design in the ARKode package enables the development/implementation of cutting-edge one-step methods
- ARKode structure separates time-integration from algorithm for a single step
- New time-steppers may be more easily constructed within ARKode
- Leveraged by the SUNDIALS team to develop a multirate additive Runge-Kutta stepper and a generalized additive Runge-Kutta stepper – more planned

Performance Portability → Performance Sustainability

Parallelism is fully-encapsulated in SUNDIALS modules
- Most parallelism is in vector and linear solver modules
- Adapt to new target architectures without rewriting the whole suite – simply create a new module implementation for the target architecture

SUNDIALS provides *NVector* implementations that utilize popular performance portability frameworks/paradigms
- RAJA
- Tpetra/Kokkos
- OpenMP 4.5+ device offloading



Continuous Testing Assures Quality

Continuous integration tests run with every commit/pull-request and nightly
- Unit tests cover individual SUNDIALS modules
- Example problems cover the integrators and modules as a whole
- Driven by Jenkins running on a Linux workstation
- Allows for rapid feature development
- Ensures that the code is always in a releasable state
- Promotes incremental releases
- In the future, SUNDIALS CI will also run on target supercomputers

Users are Fundamental to Sustainability

SUNDIALS has an active user-base supported via a mailing list and GitHub
- The SUNDIALS team is active in providing assistance
- As of February 2019, SUNDIALS now accepts user-contributions
- Attractive to prospective users

SUNDIALS is easier than ever to obtain and use
- Downloadable from source and buildable via CMake
- Installable through the Spack package manager: `% spack install sundials`
- Member of the Extreme-scale Scientific Software Development Kit (xSDK)
- Extensive user-documentation that is kept up-to-date with each release
- Web-based documentation and CMake target exports are future goals

Spack · xSDK

https://computation.llnl.gov/projects/sundials

More Information:
Carol S. Woodward, LLNL, woodward6@llnl.gov
Daniel Reynolds, SMU, reynolds@smu.edu

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344. LLNL-POST-767922.

CASC Center for Applied Scientific Computing · Lawrence Livermore National Laboratory · SMU