Rythmos: Solution and Analysis Package for Differential-Algebraic Equations (DAEs) and Ordinary-Differential Equations (ODEs)

Roscoe A. Bartlett, Oak Ridge National Laboratories
Todd S. Coffey, Simulation Modeling Sciences
Roger P. Pawlowski, Multiphysics Simulation Technology
Curtis C. Ober, Numerical Analysis and Applications

August 14, 2012

Abstract

This document contains the basic theory on the time integrators; examples and usage; and software design and developer's guide for Rythmos.

Contents

I Theory Manual 5

1 Introduction 6

2 Mathematical Formulation of DAEs/ODEs for Basic Time Steppers 6

3 Formulation for Explicit Time Steppers for ODEs 8
3.1 Forward Euler . . . 8
3.2 Explicit Runge-Kutta Methods . . . 9
3.3 Explicit RK Forward Euler . . . 10
3.4 Explicit RK 2 Stage 2 Order by Runge (Explicit Midpoint) . . . 11
3.5 Explicit RK Trapezoidal . . . 11
3.6 Explicit RK 3 Stage 3 Order . . . 12
3.7 Explicit RK 3 Stage 3 Order by Heun . . . 12
3.8 Explicit RK 3 Stage 3 Order TVD . . . 13
3.9 Explicit RK 4 Stage 3 Order by Runge . . . 13
3.10 Explicit RK4 . . . 14
3.11 Explicit RK 3/8 Rule . . . 14

4 Formulation for Implicit Time Steppers for ODEs and DAEs 15
4.1 Backward Euler . . . 16
4.2 Backward Difference Formulas . . . 17
4.2.1 Convergence Test for Implicit BDF . . . 19
4.3 Implicit Runge-Kutta methods . . . 20
4.3.1 Implicit RK Backward Euler . . . 21
4.3.2 SDIRK 2 Stage 2 Order . . . 21
4.3.3 SDIRK 2 Stage 3 Order . . . 22
4.3.4 SDIRK 3 Stage 4 Order . . . 22
4.3.5 SDIRK 5 Stage 4 Order . . . 23
4.3.6 SDIRK 5 Stage 5 Order . . . 23
4.3.7 DIRK 2 Stage 3 Order . . . 24

II User's Manual 25

5 ParameterLists Description 26
5.1 Integrator Base . . . 28
5.2 Integrator Settings . . . 28
5.3 Integrator Selection . . . 28
5.4 Default Integrator . . . 28
5.5 VerboseObject . . . 28
5.6 Integration Control Strategy Selection . . . 30
5.7 Simple Integration Control Strategy . . . 30
5.8 Stepper Settings . . . 30
5.9 Stepper Selection . . . 31
5.10 Forward Euler . . . 31
5.11 Backward Euler . . . 31
5.12 Implicit BDF . . . 31
5.13 Explicit RK . . . 32
5.14 Implicit RK . . . 32
5.15 Step Control Settings . . . 32
5.16 Step Control Strategy Selection . . . 32
5.17 Implicit BDF Stepper Step Control Strategy . . . 32
5.18 magicNumbers . . . 33
5.19 Implicit BDF Stepper Ramping Step Control Strategy . . . 33
5.20 Error Weight Vector Calculator Selection . . . 34
5.21 Implicit BDF Stepper Error Weight Vector Calculator . . . 34

5.22 Interpolator Selection . . . 34
5.23 Linear Interpolator . . . 35
5.24 Hermite Interpolator . . . 35
5.25 Cubic Spline Interpolator . . . 35
5.26 Runge Kutta Butcher Tableau Selection . . . 35
5.27 Forward Euler . . . 37
5.28 Explicit 2 Stage 2nd order by Runge . . . 37
5.29 Explicit Trapezoidal . . . 38
5.30 Explicit 3 Stage 3rd order . . . 38
5.31 Explicit 3 Stage 3rd order by Heun . . . 38
5.32 Explicit 3 Stage 3rd order TVD . . . 39
5.33 Explicit 4 Stage 3rd order by Runge . . . 39
5.34 Explicit 4 Stage . . . 39
5.35 Explicit 3/8 Rule . . . 40
5.36 Backward Euler . . . 40
5.37 Singly Diagonal IRK 2 Stage 2nd order . . . 40
5.38 Singly Diagonal IRK 2 Stage 3rd order . . . 41
5.39 Singly Diagonal IRK 3 Stage 4th order . . . 41
5.40 Singly Diagonal IRK 5 Stage 4th order . . . 41
5.41 Singly Diagonal IRK 5 Stage 5th order . . . 42
5.42 Diagonal IRK 2 Stage 3rd order . . . 42
5.43 Implicit 1 Stage 2nd order Gauss . . . 43
5.44 Implicit 2 Stage 4th order Gauss . . . 43
5.45 Implicit 3 Stage 6th order Gauss . . . 43
5.46 Implicit 2 Stage 4th Order Hammer & Hollingsworth . . . 44
5.47 Implicit 3 Stage 6th Order Kuntzmann & Butcher . . . 44
5.48 Implicit 1 Stage 1st order Radau left . . . 44
5.49 Implicit 2 Stage 3rd order Radau left . . . 45
5.50 Implicit 3 Stage 5th order Radau left . . . 45
5.51 Implicit 1 Stage 1st order Radau right . . . 46
5.52 Implicit 2 Stage 3rd order Radau right . . . 46
5.53 Implicit 3 Stage 5th order Radau right . . . 46
5.54 Implicit 2 Stage 2nd order Lobatto A . . . 47
5.55 Implicit 3 Stage 4th order Lobatto A . . . 47
5.56 Implicit 4 Stage 6th order Lobatto A . . . 47
5.57 Implicit 2 Stage 2nd order Lobatto B . . . 48
5.58 Implicit 3 Stage 4th order Lobatto B . . . 48
5.59 Implicit 4 Stage 6th order Lobatto B . . . 48
5.60 Implicit 2 Stage 2nd order Lobatto C . . . 49
5.61 Implicit 3 Stage 4th order Lobatto C . . . 49
5.62 Implicit 4 Stage 6th order Lobatto C . . . 50
5.63 Interpolation Buffer Settings . . . 50
5.64 Trailing Interpolation Buffer Selection . . . 50
5.65 Interpolation Buffer . . . 50
5.66 Interpolation Buffer Appender Selection . . . 51
5.67 Pointwise Interpolation Buffer Appender . . . 51
5.68 Interpolator Selection . . . 51
5.69 Linear Interpolator . . . 52
5.70 Hermite Interpolator . . . 52
5.71 Cubic Spline Interpolator . . . 52

6 Convergence Test Examples **53**
6.1 Dahlquist Test Equation . . . 53
6.2 SinCos Problem . . . 53

III Developer's Guide **55**

IV Back Matter **55**

Index **56**

References **59**

Part I
Theory Manual

1 Introduction

Here we design and describe a set of C++ interfaces and concrete implementations for the solution of a broad class of transient ordinary differential equations (ODEs) and differential algebraic equations (DAEs) in a consistent manner.

2 Mathematical Formulation of DAEs/ODEs for Basic Time Steppers

Here we describe the basic mathematical form of a general nonlinear DAE (or ODE) for the purpose of presenting it to a forward time integrator. At the most general level of abstraction, we will consider the solution of fully implicit DAEs of the form

$$\begin{aligned} f(\dot{x}, x, t) &= 0, \text{ for } t \in [t_0, t_f], & (1) \\ x(t_0) &= x_0 & (2) \end{aligned}$$

where

$x \in \mathcal{X}$ is the vector of differential state variables,
$\dot{x} = d(x)/d(t) \in \mathcal{X}$ is the vector of temporal derivatives of x,
$t, t_0, t_f \in \mathbb{R}$ are the current, initial, and the final times respectively,
$f(\dot{x}, x, t) \in \mathcal{X}^2 \times \mathbb{R} \rightarrow \mathcal{F}$ defines the DAE vector function,
$\mathcal{X} \subseteq \mathbb{R}^{n_x}$ is the vector space of the state variables x, and
$\mathcal{F} \subseteq \mathbb{R}^{n_x}$ is the vector space of the output of the DAE function $f(\ldots)$.

Here we have been careful to define the Hilbert vector spaces $\mathcal{X}$ and $\mathcal{F}$ for the involved vectors and vector functions. The relevance of defining these vector spaces is that they come equipped with a definition of the space's scalar product (i.e. $< u, v >_{\mathcal{X}}$ for $u, v \in \mathcal{X}$) which should be considered and used when designing the numerical algorithms.

The above general DAE can be specialized to more specific types of problems based on the nature of Jacobian matrices $\partial f/\partial \dot{x} \in \mathcal{F}|\mathcal{X}$ and $\partial f/\partial x \in \mathcal{F}|\mathcal{X}$. Here we use the notation $\mathcal{F}|\mathcal{X}$ to define a linear operator space that maps vectors from the vector space $\mathcal{X}$ to the vector space $\mathcal{F}$. Note that the adjoint of such linear operators are defined in terms of these vector spaces (i.e. $< Au, v >_{\mathcal{F}} = < A^T v, u >_{\mathcal{X}}$ where $A \in \mathcal{F}|\mathcal{X}$ and A^T denotes the adjoint operator for A). Here we assume that these first derivatives exist for the specific intervals of $t \in [t_0, t_f]$ of which such an time integrator algorithm will be directly applied the the problem.

The precise class of the problem is primarily determined by the nature of the Jacobian $\partial f/\partial \dot{x}$:

- $\partial f/\partial \dot{x} = I \in \mathcal{F}|\mathcal{X}$ yields an explicit ODE
- $\partial f/\partial \dot{x}$ full rank yields an implicit ODE
- $\partial f/\partial \dot{x}$ rank deficient yields a general DAE.

In addition, the ODE/DAE may be linear or nonlinear and DAEs are classified by their index [3]. It is expected that a DAE will be able to tell a integrator what type of problem it is (*i.e.*, explicit ODE, implicit ODE, general DAE) and which, if any, of the variables are linear in the problem. This type of information can be exploited in a time integration algorithm.

Another formulation we will consider is the semi-explicit DAE formulation:

$$\begin{aligned} \dot{y} &= f(y, z, t) \text{ for } t \in [t_0, t_f], \\ 0 &= g(y, z, t) \text{ where } x = [y, z], \\ x(t_0) &= x_0. \end{aligned} \quad (3)$$

For each transient computation, the formulation will be cast into the general form in (1)–(2) to demonstrate how this general formulation can be exploited in many different contexts.

It is important for DAE initial-value problems that the initial values of the solution, x_0, and time derivative, $\dot{x}_0$, satisfy the DAE residual[4]

$$f(\dot{x}_0, x_0, t_0) = 0$$

3 Formulation for Explicit Time Steppers for ODEs

Explicit integration methods are primarily only attractive for non-stiff explicit and implicit ODEs but some classes of DAEs can be considered as well (*i.e.*, by nonlinearly eliminating the algebraic variables from the semi-explicit DAE formulation, Eq. (3) [3]). For this discussion, we will also assume that the DAE has been written in the explicit ODE form (*i.e.*, $\partial f / \partial \dot{x} = I$). Note that implicit ODEs can always be written as explicit ODEs by multiplying the implicit ODE from the left with $(\partial f / \partial \dot{x})^{-1}$ as

$$\begin{aligned}
f(\dot{x}, x, t) &= 0 \\
&\Rightarrow \\
\frac{\partial f}{\partial \dot{x}} \dot{x} + \hat{f}(x, t) &= 0 \\
&\Rightarrow \\
\left(\frac{\partial f}{\partial \dot{x}} \right)^{-1} \left(\frac{\partial f}{\partial \dot{x}} \dot{x} + \hat{f}(x, t) \right) &= 0 \\
&\Rightarrow \\
\dot{x} &= -\left(\frac{\partial f}{\partial \dot{x}} \right)^{-1} \hat{f}(x, t) \\
&= \bar{f}(x, t)
\end{aligned}$$

where $\dot{x} = \bar{f}(x, t)$ is the new explicit form of the ODE that is considered by the explicit time integration strategies below. The above transformation of course requires that the matrix $(\partial f / \partial \dot{x})^{-1}$ be fairly easy to invert.

3.1 Forward Euler

Forward Euler (Expicit euler) is simply obtained by first-order differencing the time derivative

$$\frac{x_n - x_{n-1}}{\Delta t} = \bar{f}(x, t)$$

or as an update formula

$$x_n = x_{n-1} + \Delta t \, \bar{f}(x, t).$$

Because of the first-order approximation, Forward Euler is first-order accurate as can be seen in the global-convergence plot in Fig. (1).

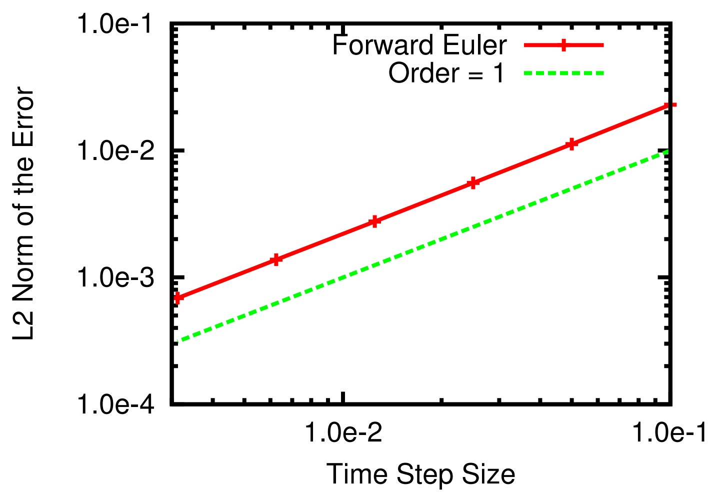


Figure 1: Order of accuracy for the SinCos Problem (Section 6.2) using Forward Euler.

3.2 Explicit Runge-Kutta Methods

The general Runge-Kutta method for s-stages, can be written as

$$X_i = x_{n-1} + \Delta t \sum_{j=1}^{s} a_{ij}\, \bar{f}(X_j, t_{n-1} + c_j \Delta t)$$

$$x_n = x_{n-1} + \Delta t \sum_{i=1}^{s} b_i\, \bar{f}(X_i, t_{n-1} + c_i \Delta t)$$

where X_i are intermediate approximations to the solution at times, $t_{n-1} + c_i \Delta t$, (*stage solutions*) which may be correct to a lower order of accuracy than the solution, x_n. We should note that these lower-order approximations are combined through b_i so that error terms cancel out and produce a more accurate solution [2, p. 80]. One can also write this in terms of $\dot{X}_i$ (or $\bar{f}(x,t)$)

$$\dot{X}_i = \bar{f}\left(x_{n-1} + \Delta t \sum_{j=1}^{s} a_{ij}\, \dot{X}_j, t_{n-1} + c_i \Delta t\right)$$

$$x_n = x_{n-1} + \Delta t \sum_{i=1}^{s} b_i\, \dot{X}_i$$

A convenient method to convey Runge-Kutta methods is to use the Butcher Tableau, which displays the coefficients in a "table" form.

Table 1: Schematic for a Butcher Tableau.

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2s} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\ \hline & b_1 & b_2 & \dots & b_s \end{array}$$

Notes:

1. c_i is the fractional timestep that the approximate solution X_i is known. It is possible for c_i to be outside the timestep range [0,1], however it is odd for a One-Step methods like Runge-Kutta to have stage solutions outside the current timestep.
2. $c_i = \sum_{j=1}^{s} a_{ij}$ for $i = 1, \dots, s$
3. For explicit methods, *e.g.*, Forward Euler and Explicit RK4, $c_1 = 0$ and $a_{1j} = 0$ for all j indicates that one needs the solution, x_{n-1}, and its time derivative, $\dot{x}_{n-1}$, (or basically an evaluation of $\bar{f}(x_{n-1}, t_{n-1})$) to start the timestep.
4. If $a_{ij} = 0$ for $j \geq i$, the Runge-Kutta (RK) method is explicit (also known as ERK), since each X_i is given in terms of known quantites.
5. If $a_{ij} \neq 0$ for any $j \geq i$, the Runge-Kutta (RK) method is implicit (also known as IRK), since an implicit solve is required for at least some of the stages.
6. If $a_{ij} = 0$ for $j > i$, the method is known as a Diagonally Implicit Runge-Kutta (DIRK) method.
7. If $a_{ij} = 0$ for $j > i$ and $a_{ii} = C$, the method is known as a Singly Diagonally Implicit Runge-Kutta (SDIRK) method.

8. b_i indicates the weighting of the intermediate solutions, X_i, to obtain the solution at the end time, x_n.

9. The stage solutions, X_i, are summed together to obtain the final solution, *i.e.*, the coefficients, b_i are a partion of unity, $\sum_{i=1}^{s} b_i = 1$.

10. If $a_{sj} = b_j$ for all j or $a_{i1} = b_1$ for all i, and a_{ij} is a nonsingular matrix, then A-stable RK methods are L-stable [5, p. 45][2, p. 103], *e.g.*, Backward Euler, Trapezoidal, and an SDIRK method. This is not a necessary condition for L-stability, but it is a sufficient condition.

3.3 Explicit RK Forward Euler

Forward Euler can also be written for Runge-Kutta methods, Fig. 2, and obtains convergence results similar to Forward Euler, Section 3.1.

a) $\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$ b)

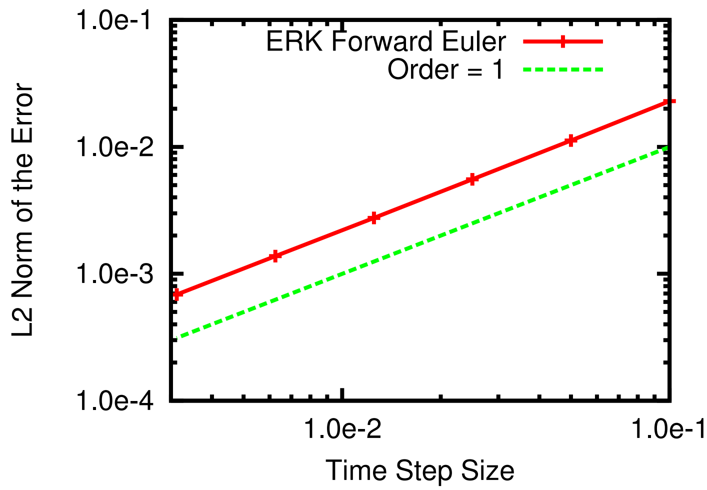


Figure 2: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK Forward Euler.

3.4 Explicit RK 2 Stage 2 Order by Runge (Explicit Midpoint)

a) $\begin{array}{c|cc} 0 & 0 & \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \end{array}$ b)

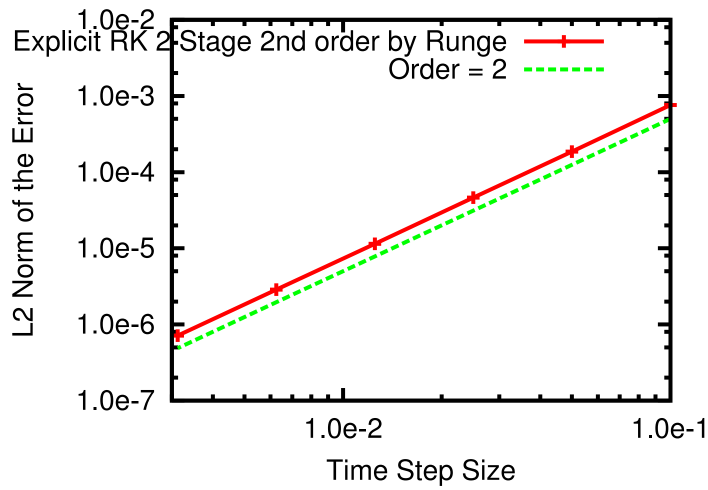


Figure 3: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 2 Stage 2nd Order by Runge..

3.5 Explicit RK Trapezoidal

a) $\begin{array}{c|cc} 0 & 0 & \\ 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array}$ b)

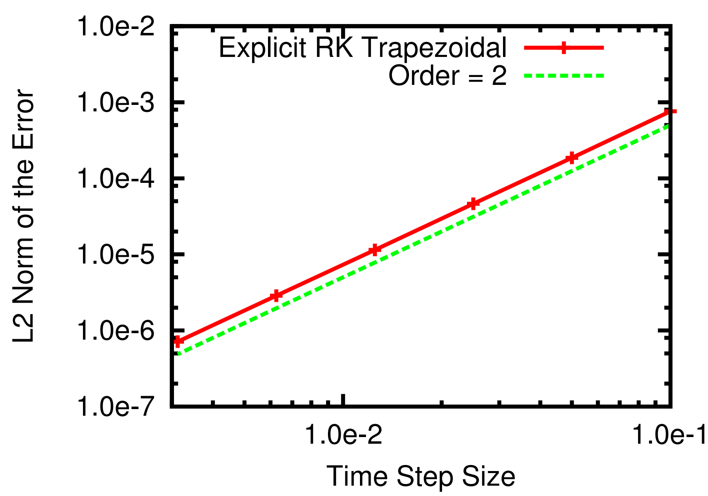


Figure 4: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK Trapezoidal.

3.6 Explicit RK 3 Stage 3 Order

a)
$$\begin{array}{c|ccc} 0 & 0 & & \\ 1/2 & 1/2 & 0 & \\ 1 & -1 & 2 & 0 \\ \hline & 1/6 & 4/6 & 1/6 \end{array}$$

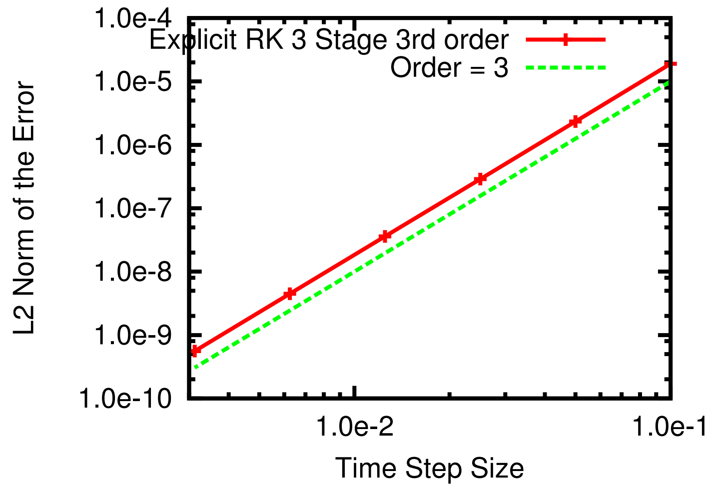
b)


Figure 5: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 3 Stage 3rd Order.

3.7 Explicit RK 3 Stage 3 Order by Heun

a)
$$\begin{array}{c|ccc} 0 & 0 & & \\ 1/3 & 1/3 & 0 & \\ 2/3 & 0 & 2/3 & 0 \\ \hline & 1/4 & 0 & 3/4 \end{array}$$

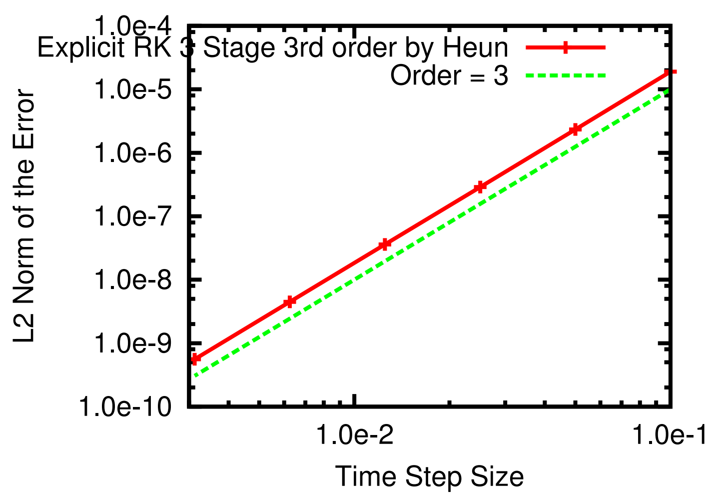
b)


Figure 6: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 3 Stage 3rd Order by Heun.

3.8 Explicit RK 3 Stage 3 Order TVD

a)
$$
\begin{array}{c|ccc}
0 & 0 & & \\
1 & 1 & 0 & \\
1/2 & 1/4 & 1/4 & 0 \\
\hline
 & 1/6 & 1/6 & 4/6
\end{array}
$$

b)

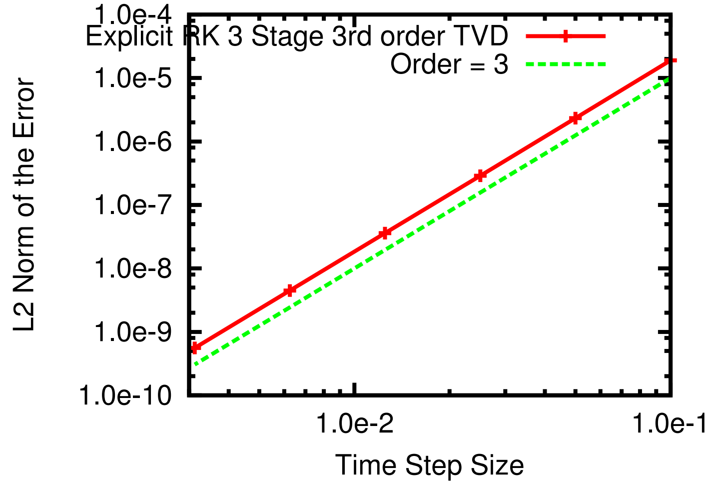


Figure 7: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 3 Stage 3rd Order TVD.

3.9 Explicit RK 4 Stage 3 Order by Runge

a)
$$
\begin{array}{c|cccc}
0 & 0 & & & \\
1/2 & 1/2 & 0 & & \\
1 & 0 & 1 & 0 & \\
1 & & 0 & 1 & 0 \\
\hline
 & 1/6 & 2/3 & 0 & 1/6
\end{array}
$$

b)

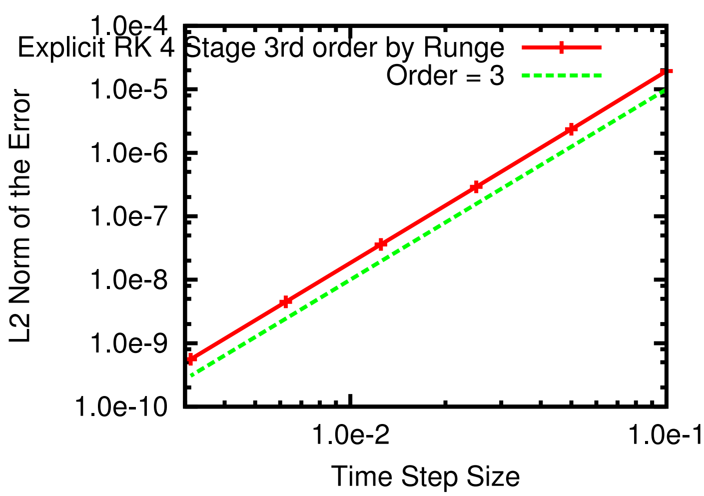


Figure 8: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 4 Stage 3rd Order by Runge.

3.10 Explicit RK4

a)
$$\begin{array}{c|cccc} 0 & 0 & & & \\ 1/2 & 1/2 & 0 & & \\ 1/2 & 0 & 1/2 & 0 & \\ 1 & 0 & 0 & 1 & 0 \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

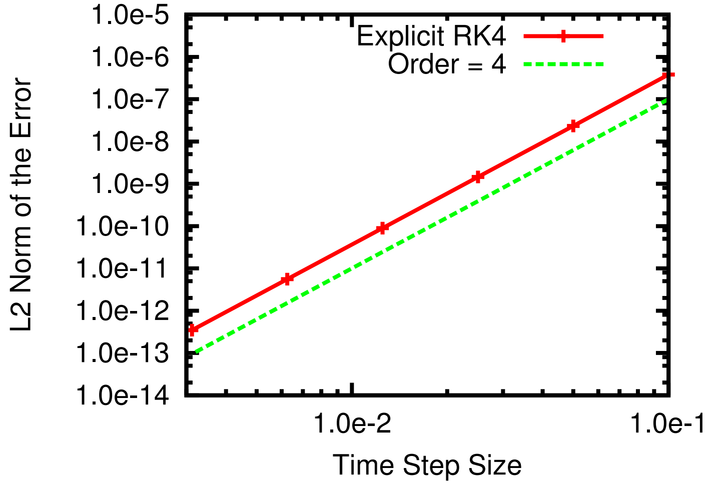
b) 

Figure 9: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 4..

3.11 Explicit RK 3/8 Rule

a)
$$\begin{array}{c|cccc} 0 & 0 & & & \\ 1/3 & 1/3 & 0 & & \\ 2/3 & -1/3 & 1 & 0 & \\ 1 & 1 & -1 & 1 & 0 \\ \hline & 1/8 & 3/8 & 3/8 & 1/8 \end{array}$$

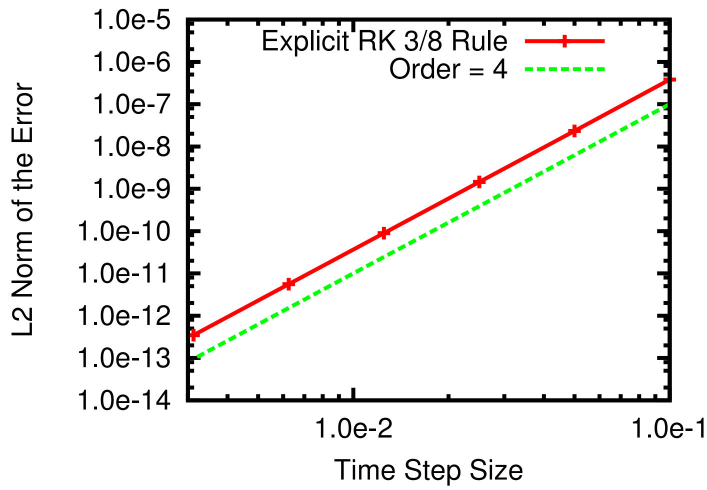
b) 

Figure 10: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Explicit RK 3/8 Rule.

4 Formulation for Implicit Time Steppers for ODEs and DAEs

Here we consider several different classes of implicit time stepping methods. For each class of method we show the set of general nonlinear equations that defines a single time step and then show how a linearized form of the equations may be formed to be solved by a Newton-type nonlinear equation solver.

In particular, for each method, we will show how to define a set of nonlinear equations of the form

$$r(z) = 0 \tag{4}$$

such that when solved will define an implicit time step from t_k to t_{k+1}, where $\Delta t = t_{k+1} - t_k$ denotes the time-step. In addition, for each method, we will show how to use the DAE residual evaluation $(\dot{x}, x, t) \rightarrow f$ to define the nonlinear time step equation (4). At the highest level, the time step method only requires very general convergence criteria for the time step equation (4) and therefore great flexibility is allowed in how the time step equation is solved. In general, the system in (4) must be solved such that $||x_{k+1} - x^*(t_{k+1})|| < \eta$, where $x_{k+1} \in \mathcal{X}$ is the computed step for the state, $x^*(t_{k+1}) \in \mathcal{X}$ is the exact state solution at t_{k+1}, and η is the maximum allowable local truncation error defined by the user.

Even though the time step equation can be solved by a variety of means, a large class of DAEs can also potentially provide support for a general Newton-type method for solving these equations and can therefore leverage general software for solving such problems (e.g. NOX). The foundation of Newton-like methods is the ability to solve linear systems similar to the Newton system

$$\frac{\partial r}{\partial z} \Delta z = -r(z_l) \tag{5}$$

where z_l is the current candidate solution of the nonlinear equations (which also defines the point where $\partial r/\partial z$ is evaluated) and $\Delta z = r_{l+1} - r_l$ is the update direction. Line-search Newton methods then define an update to the solution along the direction Δz. The essential functionality needed to perform a Newton-like method are the the abilities to evaluate the nonlinear residual $z \rightarrow r$ and to (approximately) solve linear systems involving the Jacobian matrix $\partial r/\partial z$. For each type of implicit time integration method, we will show, if possible, how to perform solves with $\partial r/\partial z$ by performing solves with the matrix

$$W = \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x}, \tag{6}$$

evaluated at points $(\dot{x}, x, t)$ selected by the time integration method and where $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}$ is some constants also defined by the time integration method. Note that the matrix W above in (9) may not necessarily exactly represent $\partial r/\partial z$ and z and r may not simply lie in the vector spaces $\mathcal{X}$ and $\mathcal{F}$ respectively; but in many cases, they will.

The iteration matrix, W, is defined to be

$$W \equiv \frac{df}{dx_n} = \frac{\partial \dot{x}}{\partial x_n} \frac{\partial f}{\partial \dot{x}} + \frac{\partial x}{\partial x_n} \frac{\partial f}{\partial x} = \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x}$$

where $\alpha = \partial \dot{x}/\partial x_n$ and $\beta = \partial x/\partial x_n$, and recalling $f\left(\dot{x}(x_n), x(x_n)\right) = 0$.

4.1 Backward Euler

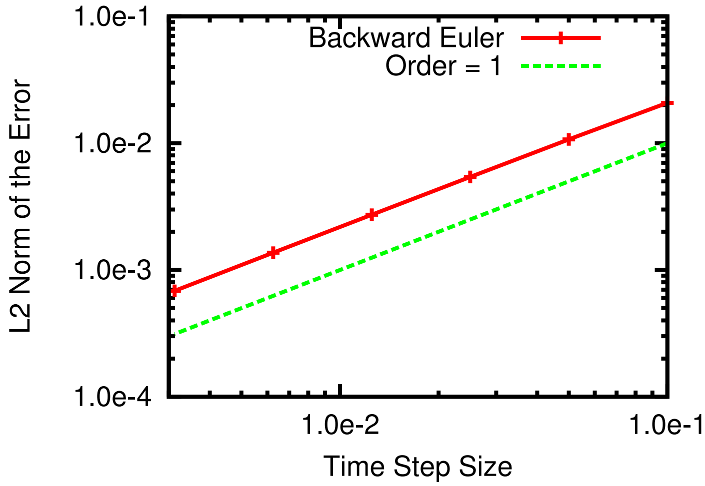


Figure 11: Order of accuracy for the SinCos Problem (Section 6.2) using Backward Euler.

4.2 Backward Difference Formulas

Backward Difference Formulas (BDF) are popular methods for stiff problems, and are derived by differentiating a polynomial representation of s past solutions, x_{n-i} for $i = 1, \ldots, s$, and setting the derivative $\dot{x}(t_n) = f(t_n, x_n)$. For evenly spaced intervals, $\Delta t = t_n - t_{n-1}$, an s-step BDF method (BDFs) is given by

$$\dot{x}_n = \bar{f}(t_n, x_n) = \frac{1}{\Delta t\, \beta_0} \sum_{i=0}^{s} \alpha_i\, x_{n-i} \tag{7}$$

where α_0 is normally scaled to one, and the order is equal to the number of steps, s.

The nonlinear time step equation to advance the solution from t_{n-1} to t_n is then formed by substituting $\dot{x} = \dot{x}_n$ in (7), $x = x_n$ and $t = t_n$ into (1) to obtain

$$f\left(\left[\frac{1}{\Delta t\, \beta_0} \sum_{i=0}^{s} \alpha_i\, x_{n-i}\right], x_n, t_n\right) = 0. \tag{8}$$

One can immediately identify the BDF time step equations (8) with the general form of the time step equations (4) and with unknown solution variables $z = x_n$. All of the other state variables x_{n-i}, for $i = 1 \ldots s$, are given.

Note that the first-order BDF method with $p = 1$, $\alpha_0 = 1$ and $\alpha_1 = -1$ is simply the standard backward Euler time integration method [2].

When considering a general Newton-like method for solving (8), note that the Newton Jacobian of these equations is

$$\frac{\partial r}{\partial z} = \frac{\alpha_0}{\Delta t\, \beta_0} \frac{\partial f}{\partial \dot{x}} + \frac{\partial f}{\partial x}, \tag{9}$$

which is evaluated at the point $\dot{x}$ in (7), $x = x_n$ and $t = t_n$. One can immediately identify (9) with the general form of the matrix M in (6) where $\alpha = \alpha_0/(\Delta t\, \beta_0)$ and $\beta = 1$. Note that the Jacobian (9) is the exact Jacobian for the nonlinear time step equations; this will not be true for some of the other methods.

There are three major variations of BDFs for variable time-step methods: fixed-coefficient formulas, variable-coefficient formulas, and fixed-leading-coefficient formulas. The fixed-coefficient formulas assume evenly spaced intervals so that the coefficients are fixed and can be precomputed for all orders[2, p. 130]. However when the time-step size changes the solution must be interpolated to evenly spaced intervals, and the amount of work for the interpolation is proportional to the number of equations in the system. Jackson and Sacks-Davis[6] have noted that fixed-coefficient formulas have worse stability properties than the other two variations. Also there can be an additional computational savings if $\partial f/\partial x$ is constant and the time step and order are consistently updated, fewer matrix factorizations and evaluations are possible.

For variable-coefficient formulas, the coefficients need to be recalculated if the time step has changed in the past s steps, which the amount of work depends on the number of steps but is independent of the number of equations. As noted earlier, variable-coefficient formulas have better stability properties which allow larger changes in time step and order during the solution of the problem. This can reduce the overall computational costs through larger time steps. It should be noted that the nonlinear matrix will need to be refactorized even if $\partial f/\partial x$ is constant, when the time step has changed in the past s steps thus increasing the of variable-coefficient formulas.

Variable-coefficient formulas would be the preferred approach except for the when $\partial f/\partial x$ is constant, when the time step has changed in the past s steps. Jackson and Sacks-Davis[6] noted that if the leading coefficient, α_0, is kept constant this exception can be removed, and these methods have been termed fixed-leading-coefficient formulas. thus is a hybrid between the two methods and tries to obtain the best of both worlds.

Fixed-Leading-Coefficient Formula[1] To begin, we need a predictor polynomial, $\phi_p(t)$, that uses the past solution values (possiblity at uneven intervals)

$$\phi_p(t_{n-i}) = x_{n-i} \quad \text{for } i = 1, \ldots, s$$

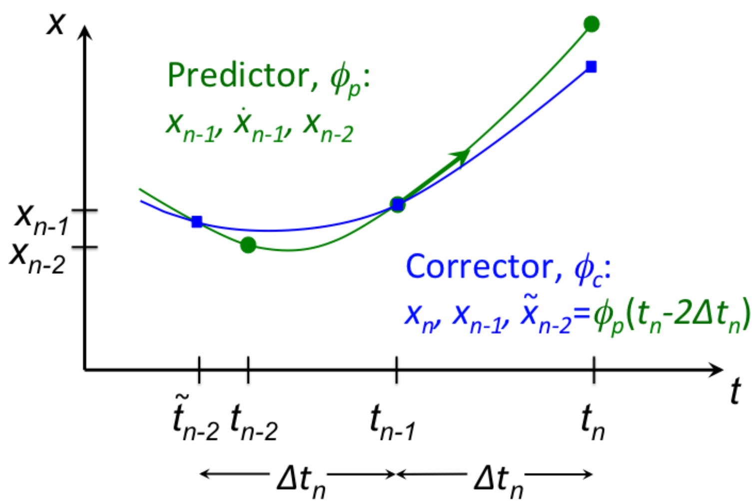


Figure 12: Schematic of a BDF2 method with predictor and corrector polynomials when the variable time stepping is increasing.

and the time derivative

$$\dot{\phi}_p(t_{n-i}) = \dot{x}_{n-i}$$

A corrector polynomial, $\phi_c(t)$, is constructed on even intervals of $\Delta t_n = t_n - t_{n-1}$, and is equal to $\phi_p(t)$ at those locations.

$$\begin{aligned} \phi_c(t_n) &= x_n \\ \phi_c(t_n - i\Delta t_n) &= \phi_p(t_n - i\Delta t_n) \quad \text{for} \quad i = 1, \ldots, s \end{aligned}$$

Thus ϕ_c passes through the unknown solution at t_n. The schematic of these polynomials are shown in Fig. 12 for a BDF2 method. Note that in this illustration, the variable time stepping is increasing, thus $t_n - 2\Delta t_n$ is further back in time than t_{n-2}.

With ϕ_p and ϕ_c being s-th order polynomials, they can be easily related through a new polynomial, $w(t)$, as

$$\phi_c(t) = \phi_p(t) + w(t)[\phi_c(t_n) - \phi_p(t_n)] \tag{10}$$

where

$$\begin{aligned} w(t_n) &= 1 \\ w(t_n - i\Delta t_n) &= 0 \quad \text{for} \quad i = 1, \ldots, s \end{aligned}$$

With these roots of a s-th order polynomial, one can write

$$w(t) = C \prod_{i=1}^{s} [t - (t_n - i\Delta t_n)]$$

and with $w(t_n) = 1$, the coefficient becomes $C = 1/(s!\, \Delta t_n^s)$.

Plugging the corrector, Eq. 10, into the ODE, Eq. 7, to generate the nonlinear equation to solve, and noting that $\phi_c(t_n) = x_n$ and $\phi_p(t_n) = x_{n(0)}$, one gets

$$\dot{\phi}_c(t) = \dot{\phi}_p(t) + \dot{w}(t)[x_n - x_{n(0)}].$$

Evaluating at $t = t_n$ to get the solution at the next time step and using $\dot{\phi}_c(t_n) = \dot{x}_n$, $\dot{\phi}_p(t_n) = \dot{x}_{n(0)}$, and $\dot{w}(t_n) = [1/1 + 1/2 + \ldots + 1/s]/\Delta t_n = 1/(\beta_0\, \Delta t_n)$, one gets

$$\dot{x}_n = \dot{x}_{n(0)} + \frac{1}{\beta_0\, \Delta t_n}[x_n - x_{n(0)}]$$

or

$$x_n = x_{n(0)} + \beta_0\, \Delta t_n[\dot{x}_n - \dot{x}_{n(0)}]$$

or

$$x_n - x_{n(0)} = \beta_0\, \Delta t_n[\dot{x}_n - \dot{x}_{n(0)}].$$

Note that the predictor and corrector are interwined. One should not try to use a different predictor as this will change the predictor polynomial, $\phi_p(t)$, and its presentation of past solution values. Also the predictor is dependent on the time derivative at the last time step, so one should be careful to keep the solution, x_{n-1}, and its time derivative, $\dot{x}_{n-1}$, used by the predictor in "sync" and consistent with previous time steps (e.g., clipping the solution between time steps will cause issues).

4.2.1 Convergence Test for Implicit BDF

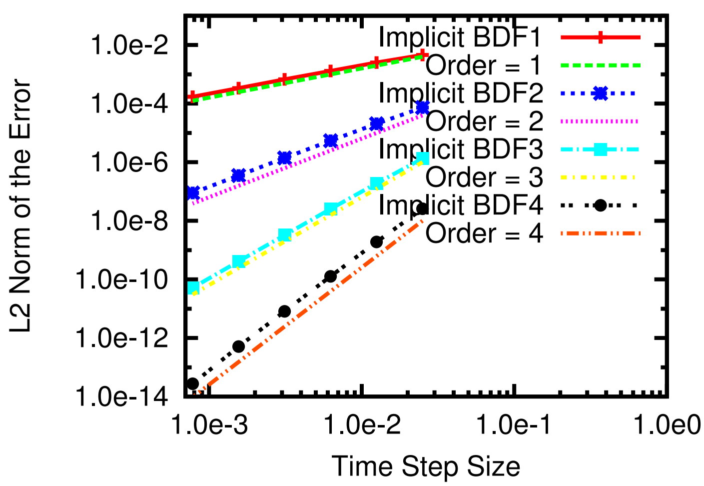


Figure 13: Order of accuracy for the SinCos Problem (Section 6.2) using Implicit BDF.

4.3 Implicit Runge-Kutta methods

We now consider a class of powerful and popular one-step methods for solving implicit DAEs, implicit Runge-Kutta (RK) methods. The most general form of implicit RK methods requires the simultaneous solution of s sets of coupled nonlinear equations that take the form

$$r_i(z) = f\left(\dot{X}_i,\ x_{n-1} + \Delta t \sum_{j=1}^{s} a_{ij}\, \dot{X}_j,\ t_{n-1} + c_i \Delta t\right) = 0 \tag{11}$$

for $i = 1, \ldots, s$ where $\dot{X}_i$ are essentially approximations to the derivatives $\dot{x}(t_{n-1} + c_i \Delta t)$ called *stage derivatives* and $z = [\dot{X}_1, \dot{X}_2, \ldots, \dot{X}_s]^T$ are the unknowns in this set of equations. After this set of coupled equations is solved, the state solution x_n is given as the linear combination

$$x_n = x_{n-1} + \Delta t \sum_{i=1}^{s} b_i\, \dot{X}_i$$

It is clear how to form the residual for the fully coupled system for $r(z)$ in Eq. (11) just from individual evaluations. How the Newton system for such a system is solved will vary greatly based on the structure and properties of the Butcher matrix, a_{ij}.

Fully implicit RK methods present somewhat of a problem for developing general software since they involve the need to solve a fully coupled system of s sets of equations of the form of Eq. (11). Each block $\partial r_i / \partial z_j = \partial r_i / \partial \dot{X}_j$ of the full Jacobian $\partial r / \partial z$ is represented as

$$\begin{aligned} W_{ij} &= \alpha \frac{\partial f}{\partial \dot{x}} + \beta \frac{\partial f}{\partial x} \\ &= \frac{\partial r_i}{\partial z_j} = \frac{\partial r_i}{\partial \dot{X}_j} \\ &= \frac{\partial \dot{x}}{\partial \dot{X}_j} \frac{\partial f}{\partial \dot{x}} + \frac{\partial x}{\partial \dot{X}_j} \frac{\partial f}{\partial x} \\ &= \delta_{ij} \frac{\partial f}{\partial \dot{x}} + \Delta t\, a_{ij} \frac{\partial f}{\partial x} \end{aligned} \tag{12}$$

for $i = 1, \ldots, s$ and $j = 1, \ldots, s$ which is evaluated at the points $(\dot{x}, x, t) = (\dot{X}_i,\ x_{n-1} + \Delta t \sum_{j=1}^{s} a_{ij}\, \dot{X}_j,$ $t_{n-1} + c_i \Delta t)$. Note that the iteration matrix, W, has $\alpha = \delta_{ij}$ and $\beta = \Delta t\, a_{ij}$.

When considering a iterative method for solving systems with the block operator matrix $\partial r / \partial z$, it is easy to see how to use the iteration matrix W in Eq. (6) to implement a matrix-vector product, but it is not obvious how to precondition such a system. Clearly a block diagonal preconditioner could be used but the effectiveness of such a preconditioning strategy is open to question. Other preconditioning strategies are also possible just given the basic block operators and this is an open area of research. In some cases, however, it may be possible to form a full matrix object for $\partial r / \partial z$ but this is not something that can be expected for most applications.

Semi-explicit IRK methods Semi-explicit IRK methods are those IRK methods where the Butcher matrix, A, is lower diagonal and therefore gives rise to a block lower triangular Jacobian matrix $\partial r / \partial z$. For these types of methods, the nonlinear equations in Eq. (12) can be solved one at a time for $i = 1, \ldots, s$ which is easily accommodated using a Newton-type method where the Newton Jacobian for each i is given by Eq. (12), which is of our basic general form, Eq. (6).

Singly-Diagonal-implicit IRK methods The next specialized class of IRK methods that we consider are singly-diagonal-implicit IRK methods where the Butcher coefficients in a_{ij} and c give rise to a lower triangular Jacobian $\partial r / \partial z$ (and hence are also semi-explicit IRK methods) that has the same nonsingular matrix block of the form in Eq. (12) along the diagonal. This, of course, requires that $a_{11} = a_{22} = \cdots = a_{ss}$ and $c_1 = c_2 = \cdots = c_s$. (I am not sure that this is possible, since $c_i = \sum_{j=1}^{s} a_{ij}$. The only method that satisifies this is IRK Backward Euler!? CCO) In this class of IRK methods, significant savings may be achieved since a single set of linear solver objects (*i.e.*, operators and preconditioners) can be utilized for the solution of the fully coupled system. In fact, it may even be possible to utilize multi-vector applications of the operator and preconditioner for matrices of the form Eq. (6) which can be supported by many applications.

4.3.1 Implicit RK Backward Euler

a) $\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}$ b)

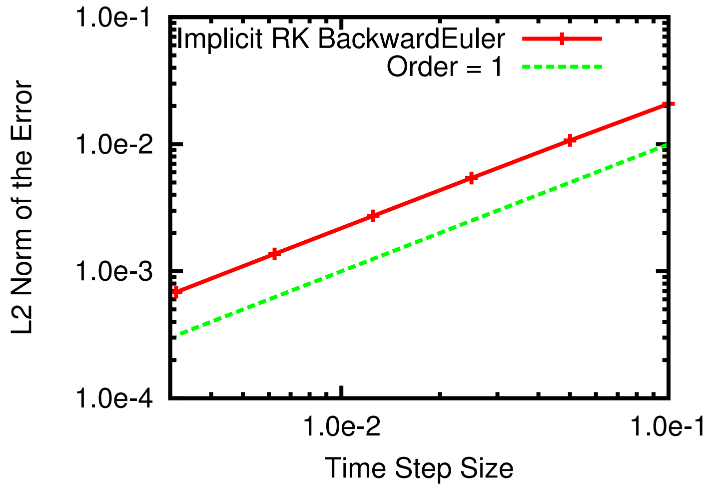


Figure 14: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for Implicit RK Backward Euler.

4.3.2 SDIRK 2 Stage 2 Order

For $\gamma = (2 \pm \sqrt{2})/2$, this method is 2nd order accurate and L-stable. Other values of γ will still produce an L-stable scheme, but with only be 1st order accurate.

a) $\begin{array}{c|cc} \gamma & \gamma & \\ 1 & 1-\gamma & \gamma \\ \hline & 1-\gamma & \gamma \end{array}$ b)

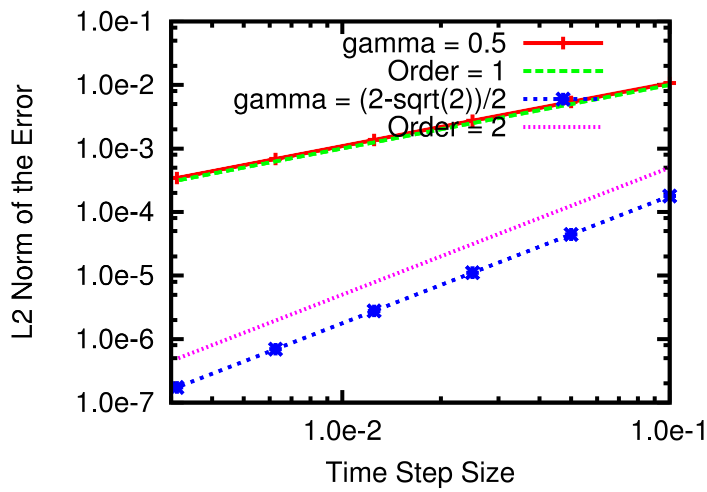


Figure 15: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for SDIRK 2 Stage 2 Order.

4.3.3 SDIRK 2 Stage 3 Order

For $\gamma = (3 \pm \sqrt{3})/6$, this method is 3rd order accurate and A-stable. For $\gamma = (2 \pm \sqrt{2})/2$, this method is only 2nd order accurate, but is then L-stable.

a) $\begin{array}{c|cc} \gamma & \gamma & \\ 1-\gamma & 1-2\gamma & \gamma \\ \hline & 1/2 & 1/2 \end{array}$ b)

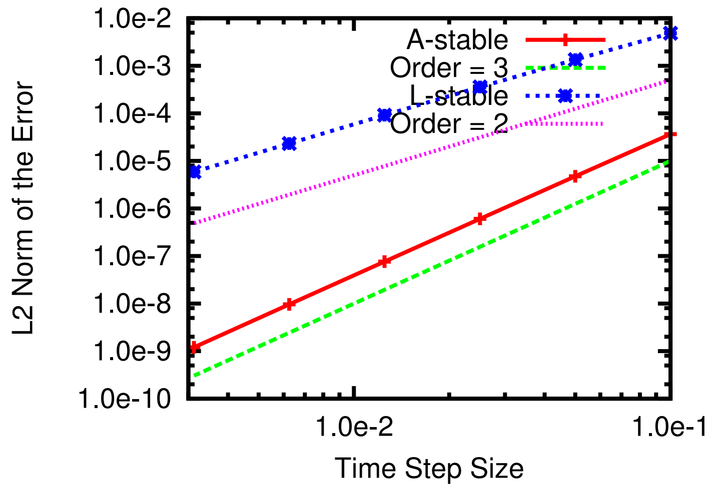


Figure 16: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for SDIRK 2 Stage 3 Order.

4.3.4 SDIRK 3 Stage 4 Order

The coefficients are $\gamma = 1/\sqrt{3}\cos(\pi/18) + 1/2$ and $\delta = (2\gamma - 1)^{-2}/6$.

a) $\begin{array}{c|ccc} \gamma & \gamma & & \\ 1/2 & 1/2-\gamma & \gamma & \\ 1-\gamma & 2\gamma & 1-4\gamma & \gamma \\ \hline & \delta & 1-2\delta & \delta \end{array}$ b)

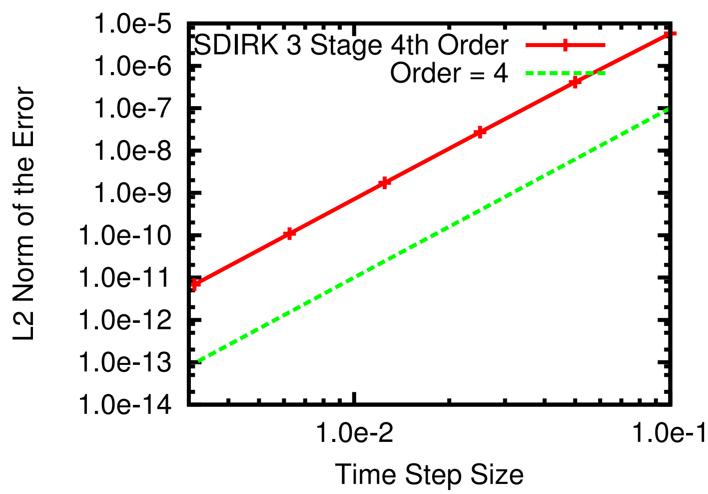


Figure 17: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for SDIRK 3 Stage 4 Order.

4.3.5 SDIRK 5 Stage 4 Order

$$
\begin{array}{c|ccccc}
1/4 & 1/4 & & & & \\
3/4 & 1/2 & 1/4 & & & \\
11/20 & 17/50 & -1/25 & 1/4 & & \\
1/2 & 371/1360 & -137/2720 & 15/544 & 1/4 & \\
1 & 25/24 & -49/48 & 125/16 & -85/12 & 1/4 \\
\hline
 & 25/24 & -49/48 & 125/16 & -85/12 & 1/4 \\
 & 59/48 & -17/96 & 225/32 & -85/12 & 0
\end{array}
$$

Figure 18: Butcher Tableau for SDIRK 5 Stage 4 Order.

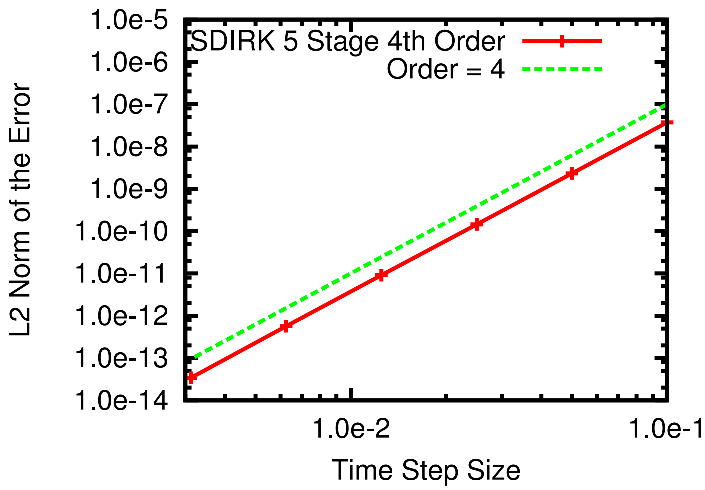


Figure 19: Order of accuracy for the SinCos Problem (Section 6.2) using SDIRK 5 Stage 4 Order.

4.3.6 SDIRK 5 Stage 5 Order

$$
\begin{array}{c|ccccc}
\frac{6-\sqrt{6}}{10} & \frac{6-\sqrt{6}}{10} & & & & \\
\frac{6+\sqrt{6}}{35} & \frac{-6+5\sqrt{6}}{14} & \frac{6-\sqrt{6}}{10} & & & \\
1 & \frac{888+607\sqrt{6}}{2850} & \frac{126-161\sqrt{6}}{1425} & \frac{6-\sqrt{6}}{10} & & \\
\frac{4-\sqrt{6}}{10} & \frac{3153-3082\sqrt{6}}{14250} & \frac{3213+1148\sqrt{6}}{28500} & \frac{-267+88\sqrt{6}}{500} & \frac{6-\sqrt{6}}{10} & \\
\frac{4+\sqrt{6}}{10} & \frac{-32583+14638\sqrt{6}}{71250} & \frac{-17199+364\sqrt{6}}{142500} & \frac{1329-544\sqrt{6}}{2500} & \frac{-96+131\sqrt{6}}{625} & \frac{6-\sqrt{6}}{10} \\
\hline
 & 0 & 0 & 1/9 & \frac{16-\sqrt{6}}{36} & \frac{16+\sqrt{6}}{36}
\end{array}
$$

Figure 20: Butcher Tableau for SDIRK 5 Stage 5 Order.

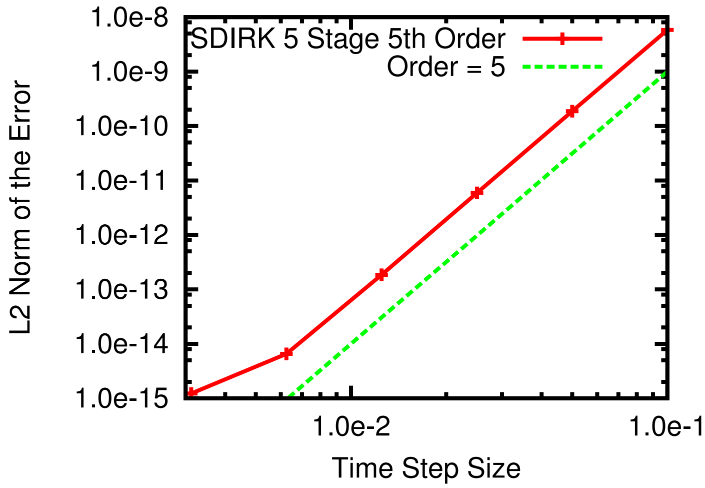


Figure 21: Order of accuracy for the SinCos Problem (Section 6.2) for SDIRK 5 Stage 5 Order.

4.3.7 DIRK 2 Stage 3 Order

a)

$$\begin{array}{c|cc} 0 & 0 & \\ 2/3 & 1/3 & 1/3 \\ \hline & 1/4 & 3/4 \end{array}$$

b)

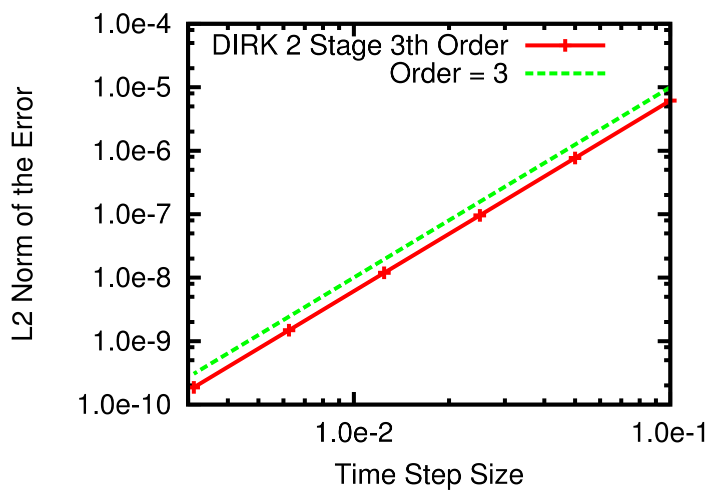


Figure 22: a) Butcher Tableau and b) Order of accuracy for the SinCos Problem (Section 6.2) for DIRK 2 Stage 3 Order.

Part II
User's Manual

5 ParameterLists Description

- Integrator Base (Section 5.1)
    - Integrator Settings (Section 5.2)
        - Integrator Selection (Section 5.3)
            - Default Integrator (Section 5.4)
                - VerboseObject (Section 5.5)
    - Integration Control Strategy Selection (Section 5.6)
        - Simple Integration Control Strategy (Section 5.7)
    - Stepper Settings (Section 5.8)
        - Stepper Selection (Section 5.9)
            - Forward Euler (Section 5.10)
            - Backward Euler (Section 5.11)
            - Implicit BDF (Section 5.12)
            - Explicit RK (Section 5.13)
            - Implicit RK (Section 5.14)
        - Step Control Settings (Section 5.15)
            - Step Control Strategy Selection (Section 5.16)
                - Implicit BDF Stepper Step Control Strategy (Section 5.17)
                    - magicNumbers (Section 5.18)
                - Implicit BDF Stepper Ramping Step Control Strategy (Section 5.19)
            - Error Weight Vector Calculator Selection (Section 5.20)
                - Implicit BDF Stepper Error Weight Vector Calculator (Section 5.21)
        - Interpolator Selection (Section 5.22)
            - Linear Interpolator (Section 5.23)
            - Hermite Interpolator (Section 5.24)
            - Cubic Spline Interpolator (Section 5.25)
        - Runge Kutta Butcher Tableau Selection (Section 5.26)
            - Forward Euler (Section 5.27)
            - Explicit 2 Stage 2nd order by Runge (Section 5.28)
            - Explicit Trapezoidal (Section 5.29)
            - Explicit 3 Stage 3rd order (Section 5.30)
            - Explicit 3 Stage 3rd order by Heun (Section 5.31)
            - ...
    - Interpolation Buffer Settings (Section 5.63)
        - Trailing Interpolation Buffer Selection (Section 5.64)
            - Interpolation Buffer (Section 5.65)
        - Interpolation Buffer Appender Selection (Section 5.66)
            - Pointwise Interpolation Buffer Appender (Section 5.67)
        - Interpolator Selection (Section 5.68)
            - Linear Interpolator (Section 5.69)
            - Hermite Interpolator (Section 5.70)
            - Cubic Spline Interpolator (Section 5.71)

Figure 23: Schematic of ParameterList heirarchy.

5.1 Integrator Base

Description:

Parent(s): ROOT

Child(ren): Integrator Settings (Section 5.2)
Integration Control Strategy Selection (Section 5.6)
Stepper Settings (Section 5.8)
Interpolation Buffer Settings (Section 5.63)

Parameters: None.

5.2 Integrator Settings

Description: These parameters are used directly in setting up the Integrator

Parent(s): Integrator Base (Section 5.1)

Child(ren): Integrator Selection (Section 5.3)

Parameters: **Final Time** = **1**

Land On Final Time = **1**

5.3 Integrator Selection

Description:

Parent(s): Integrator Settings (Section 5.2)

Child(ren): Default Integrator (Section 5.4)

Parameters: **Integrator Type** = **Default Integrator** Determines the type of Rythmos::Integrator object that will be built. The parameters for each Integrator Type are specified in this sublist

Valid std::string values:
"None"
"Default Integrator"

5.4 Default Integrator

Description:

Parent(s): Integrator Selection (Section 5.3)

Child(ren): VerboseObject (Section 5.5)

Parameters: **Max Number Time Steps** = **2147483647** Set the maximum number of integration time-steps allowed.

5.5 VerboseObject

Description:

Parent(s): Default Integrator (Section 5.4)
Simple Integration Control Strategy (Section 5.7)
Forward Euler (Section 5.10)
Backward Euler (Section 5.11)
Implicit BDF (Section 5.12)

Explicit RK (Section 5.13)
Implicit RK (Section 5.14)
magicNumbers (Section 5.18)
Implicit BDF Stepper Error Weight Vector Calculator (Section 5.21)
Linear Interpolator (Section 5.23)
Hermite Interpolator (Section 5.24)
Cubic Spline Interpolator (Section 5.25)
Forward Euler (Section 5.27)
Explicit 2 Stage 2nd order by Runge (Section 5.28)
Explicit Trapezoidal (Section 5.29)
Explicit 3 Stage 3rd order (Section 5.30)
Explicit 3 Stage 3rd order by Heun (Section 5.31)
Explicit 3 Stage 3rd order TVD (Section 5.32)
Explicit 4 Stage 3rd order by Runge (Section 5.33)
Explicit 4 Stage (Section 5.34)
Explicit 3/8 Rule (Section 5.35)
Backward Euler (Section 5.36)
Singly Diagonal IRK 2 Stage 2nd order (Section 5.37)
Singly Diagonal IRK 2 Stage 3rd order (Section 5.38)
Singly Diagonal IRK 3 Stage 4th order (Section 5.39)
Singly Diagonal IRK 5 Stage 4th order (Section 5.40)
Singly Diagonal IRK 5 Stage 5th order (Section 5.41)
Diagonal IRK 2 Stage 3rd order (Section 5.42)
Implicit 1 Stage 2nd order Gauss (Section 5.43)
Implicit 2 Stage 4th order Gauss (Section 5.44)
Implicit 3 Stage 6th order Gauss (Section 5.45)
Implicit 2 Stage 4th Order Hammer & Hollingsworth (Section 5.46)
Implicit 3 Stage 6th Order Kuntzmann & Butcher (Section 5.47)
Implicit 1 Stage 1st order Radau left (Section 5.48)
Implicit 2 Stage 3rd order Radau left (Section 5.49)
Implicit 3 Stage 5th order Radau left (Section 5.50)
Implicit 1 Stage 1st order Radau right (Section 5.51)
Implicit 2 Stage 3rd order Radau right (Section 5.52)
Implicit 3 Stage 5th order Radau right (Section 5.53)
Implicit 2 Stage 2nd order Lobatto A (Section 5.54)
Implicit 3 Stage 4th order Lobatto A (Section 5.55)
Implicit 4 Stage 6th order Lobatto A (Section 5.56)
Implicit 2 Stage 2nd order Lobatto B (Section 5.57)
Implicit 3 Stage 4th order Lobatto B (Section 5.58)
Implicit 4 Stage 6th order Lobatto B (Section 5.59)
Implicit 2 Stage 2nd order Lobatto C (Section 5.60)
Implicit 3 Stage 4th order Lobatto C (Section 5.61)
Implicit 4 Stage 6th order Lobatto C (Section 5.62)
Interpolation Buffer (Section 5.65)
Pointwise Interpolation Buffer Appender (Section 5.67)
Linear Interpolator (Section 5.69)
Hermite Interpolator (Section 5.70)
Cubic Spline Interpolator (Section 5.71)

Child(ren): None.

Parameters: **Verbosity Level = default** The verbosity level to use to override whatever is set in code. The value of "default" will allow the level set in code to be used.

Valid std::string values:

"default"	Use level set in code
"none"	Produce no output
"low"	Produce minimal output
"medium"	Produce a little more output
"high"	Produce a higher level of output
"extreme"	Produce the highest level of output

Output File = none The file to send output to. If the value "none" is used, then whatever is set in code will be used. However, any other std::string value will be used to create an std::ofstream object to a file with the given name. Therefore, any valid file name is a valid std::string value for this parameter.

5.6 Integration Control Strategy Selection

Description: Note that some settings conflict between step control and integration control. In general, the integration control decides which steps will be fixed or variable, not the stepper. When the integration control decides to take variable steps, the step control is then responsible for choosing appropriate step-sizes.

Parent(s): Integrator Base (Section 5.1)

Child(ren): Simple Integration Control Strategy (Section 5.7)

Parameters: **Integration Control Strategy Type = None** Determines the type of Rythmos::IntegrationControlStrategy object that will be built. The parameters for each Integration Control Strategy Type are specified in this sublist

Valid std::string values:

"None"
"Simple Integration Control Strategy"

5.7 Simple Integration Control Strategy

Description:

Parent(s): Integration Control Strategy Selection (Section 5.6)

Child(ren): None.

Parameters: **Take Variable Steps = 1** Take variable time steps or fixed time steps. If set to false, then the parameter "Fixed dt" or "Number of Time Steps" must be set!

Max dt = 1.79769e+308 Gives the max size of the variable time steps. This is only read and used if "Take Variable Steps" is set to true.

Number of Time Steps = -1 Gives the number of fixed time steps. The actual step size gets computed on the fly given the size of the time domain. This is only read and used if "Take Variable Steps" is set to false and "Fixed dt" is set to < 0.0.

Fixed dt = -1 Gives the size of the fixed time steps. This is only read and used if "Take Variable Steps" is set to false.

5.8 Stepper Settings

Description:

Parent(s): Integrator Base (Section 5.1)

Child(ren): Stepper Selection (Section 5.9)
Step Control Settings (Section 5.15)
Interpolator Selection (Section 5.22)
Runge Kutta Butcher Tableau Selection (Section 5.26)

Parameters: None.

5.9 Stepper Selection

Description:

Parent(s): Stepper Settings (Section 5.8)

Child(ren): Forward Euler (Section 5.10)
Backward Euler (Section 5.11)
Implicit BDF (Section 5.12)
Explicit RK (Section 5.13)
Implicit RK (Section 5.14)

Parameters: **Stepper Type = Backward Euler** Determines the type of Rythmos::Stepper object that will be built. The parameters for each Stepper Type are specified in this sublist

Valid std::string values:
"None"
"Forward Euler"
"Backward Euler"
"Implicit BDF"
"Explicit RK"
"Implicit RK"

5.10 Forward Euler

Description:

Parent(s): Stepper Selection (Section 5.9)

Child(ren): None.

Parameters: None.

5.11 Backward Euler

Description:

Parent(s): Stepper Selection (Section 5.9)

Child(ren): None.

Parameters: None.

5.12 Implicit BDF

Description:

Parent(s): Stepper Selection (Section 5.9)

Child(ren): None.

Parameters: None.

5.13 Explicit RK

Description:

Parent(s): Stepper Selection (Section 5.9)

Child(ren): None.

Parameters: None.

5.14 Implicit RK

Description:

Parent(s): Stepper Selection (Section 5.9)

Child(ren): None.

Parameters: None.

5.15 Step Control Settings

Description: Not all step control strategies are compatible with each stepper. If the strategy has the name of a stepper in its name, then it only works with that stepper.

Parent(s): Stepper Settings (Section 5.8)

Child(ren): Step Control Strategy Selection (Section 5.16)
Error Weight Vector Calculator Selection (Section 5.20)

Parameters: None.

5.16 Step Control Strategy Selection

Description:

Parent(s): Step Control Settings (Section 5.15)

Child(ren): Implicit BDF Stepper Step Control Strategy (Section 5.17)
Implicit BDF Stepper Ramping Step Control Strategy (Section 5.19)

Parameters: **Step Control Strategy Type = None** Determines the type of Rythmos::StepControlStrategy object that will be built. The parameters for each Step Control Strategy Type are specified in this sublist

Valid std::string values:
"None"
"Implicit BDF Stepper Step Control Strategy"
"Implicit BDF Stepper Ramping Step Control Strategy"

5.17 Implicit BDF Stepper Step Control Strategy

Description:

Parent(s): Step Control Strategy Selection (Section 5.16)

Child(ren): magicNumbers (Section 5.18)

Parameters: **minOrder = 1** lower limit of order selection, guaranteed

maxOrder = 5 upper limit of order selection, does not guarantee this order

relErrTol = **0.0001**

absErrTol = **1e-06**

constantStepSize = **0**

stopTime = **10**

failStepIfNonlinearSolveFails = **0** Power user command. Will force the function acceptStep() to return false ieven if the LET is acceptable. Used to run with loose tolerances but enforce a correct nonlinear solution to the step.

5.18 magicNumbers

Description: These are knobs in the algorithm that have been set to reasonable values using lots of testing and heuristics and some theory.

Parent(s): Implicit BDF Stepper Step Control Strategy (Section 5.17)

Child(ren): None.

Parameters: **h0_safety** = **2**

h0_max_factor = **0.001**

h_phase0_incr = **2** initial ramp-up in variable mode (stepSize multiplier)

h_max_inv = **0**

Tkm1_Tk_safety = **2**

Tkp1_Tk_safety = **0.5**

r_factor = **0.9** used in rejectStep: time step ratio multiplier

r_safety = **2** local error multiplier as part of time step ratio calculation

r_fudge = **0.0001** local error addition as part of time step ratio calculation

r_min = **0.125** used in rejectStep: how much to cut step and lower bound for time step ratio

r_max = **0.9** upper bound for time step ratio

r_hincr_test = **2** used in completeStep: if time step ratio > this then set time step ratio to r_hincr

r_hincr = **2** used in completeStep: limit on time step ratio increases, not checked by r_max

max_LET_fail = **15** Max number of rejected steps

minTimeStep = **0** bound on smallest time step in variable mode.

maxTimeStep = **10** bound on largest time step in variable mode.

5.19 Implicit BDF Stepper Ramping Step Control Strategy

Description:

Parent(s): Step Control Strategy Selection (Section 5.16)

Child(ren): None.

Parameters: **Number of Constant First Order Steps** = **10** Number of constant steps to take before handing control to variable stepper.

Initial Step Size = **0.001** Initial time step size and target step size to take during the initial constant step phase (could be reduced due to step failures).

Min Step Size = 1e-07 Minimum time step size.

Max Step Size = 1 Maximum time step size.

Step Size Increase Factor = 1.2 Time step growth factor used after a successful time step. dt_n+1 = (increase factor) * dt_n

Step Size Decrease Factor = 0.5 Time step reduction factor used for a failed time step. dt_n+1 = (decrease factor) * dt_n

Min Order = 1 Minimum order to run at.

Max Order = 5 Maximum order to run at.

Absolute Error Tolerance = 1e-05 abstol value used in WRMS calculation.

Relative Error Tolerance = 0.001 reltol value used in WRMS calculation.

Use LET To Determine Step Acceptance = FALSE If set to TRUE, then acceptance of step dependes on LET in addition to Nonlinear solver converging.

Valid std::string values:
"TRUE"
"FALSE"

5.20 Error Weight Vector Calculator Selection

Description: Not all ErrWtVec calculators are compatible with each step control strategy. If the calculator has the name of a stepper or another step control strategy in its name, then it only works with that step control strategy.

Parent(s): Step Control Settings (Section 5.15)

Child(ren): Implicit BDF Stepper Error Weight Vector Calculator (Section 5.21)

Parameters: **Error Weight Vector Calculator Type = None** Determines the type of Rythmos::ErrWtVecCalc object that will be built. The parameters for each Error Weight Vector Calculator Type are specified in this sublist

Valid std::string values:
"None"
"Implicit BDF Stepper Error Weight Vector Calculator"

5.21 Implicit BDF Stepper Error Weight Vector Calculator

Description:

Parent(s): Error Weight Vector Calculator Selection (Section 5.20)

Child(ren): None.

Parameters: None.

5.22 Interpolator Selection

Description: Note all Steppers accept an interpolator. Currently, only the BackwardEuler stepper does.

Parent(s): Stepper Settings (Section 5.8)

Child(ren): Linear Interpolator (Section 5.23)
Hermite Interpolator (Section 5.24)
Cubic Spline Interpolator (Section 5.25)

Parameters: **Interpolator Type = None** Determines the type of Rythmos::Interpolator object that will be built. The parameters for each Interpolator Type are specified in this sublist

Valid std::string values:
"None"
"Linear Interpolator"
"Hermite Interpolator"
"Cubic Spline Interpolator"

5.23 Linear Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.22)

Child(ren): None.

Parameters: None.

5.24 Hermite Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.22)

Child(ren): None.

Parameters: None.

5.25 Cubic Spline Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.22)

Child(ren): None.

Parameters: None.

5.26 Runge Kutta Butcher Tableau Selection

Description: Only the Explicit RK Stepper and the Implicit RK Stepper accept an RK Butcher Tableau.

Parent(s): Stepper Settings (Section 5.8)

Child(ren): Forward Euler (Section 5.27)
Explicit 2 Stage 2nd order by Runge (Section 5.28)
Explicit Trapezoidal (Section 5.29)
Explicit 3 Stage 3rd order (Section 5.30)
Explicit 3 Stage 3rd order by Heun (Section 5.31)
Explicit 3 Stage 3rd order TVD (Section 5.32)
Explicit 4 Stage 3rd order by Runge (Section 5.33)
Explicit 4 Stage (Section 5.34)
Explicit 3/8 Rule (Section 5.35)
Backward Euler (Section 5.36)
Singly Diagonal IRK 2 Stage 2nd order (Section 5.37)
Singly Diagonal IRK 2 Stage 3rd order (Section 5.38)

Singly Diagonal IRK 3 Stage 4th order (Section 5.39)
Singly Diagonal IRK 5 Stage 4th order (Section 5.40)
Singly Diagonal IRK 5 Stage 5th order (Section 5.41)
Diagonal IRK 2 Stage 3rd order (Section 5.42)
Implicit 1 Stage 2nd order Gauss (Section 5.43)
Implicit 2 Stage 4th order Gauss (Section 5.44)
Implicit 3 Stage 6th order Gauss (Section 5.45)
Implicit 2 Stage 4th Order Hammer & Hollingsworth (Section 5.46)
Implicit 3 Stage 6th Order Kuntzmann & Butcher (Section 5.47)
Implicit 1 Stage 1st order Radau left (Section 5.48)
Implicit 2 Stage 3rd order Radau left (Section 5.49)
Implicit 3 Stage 5th order Radau left (Section 5.50)
Implicit 1 Stage 1st order Radau right (Section 5.51)
Implicit 2 Stage 3rd order Radau right (Section 5.52)
Implicit 3 Stage 5th order Radau right (Section 5.53)
Implicit 2 Stage 2nd order Lobatto A (Section 5.54)
Implicit 3 Stage 4th order Lobatto A (Section 5.55)
Implicit 4 Stage 6th order Lobatto A (Section 5.56)
Implicit 2 Stage 2nd order Lobatto B (Section 5.57)
Implicit 3 Stage 4th order Lobatto B (Section 5.58)
Implicit 4 Stage 6th order Lobatto B (Section 5.59)
Implicit 2 Stage 2nd order Lobatto C (Section 5.60)
Implicit 3 Stage 4th order Lobatto C (Section 5.61)
Implicit 4 Stage 6th order Lobatto C (Section 5.62)

Parameters: **Runge Kutta Butcher Tableau Type = None** Determines the type of Rythmos::RKButcherTableau object that will be built. The parameters for each Runge Kutta Butcher Tableau Type are specified in this sublist

Valid std::string values:

"None"
"Forward Euler"
"Explicit 2 Stage 2nd order by Runge"
"Explicit Trapezoidal"
"Explicit 3 Stage 3rd order"
"Explicit 3 Stage 3rd order by Heun"
"Explicit 3 Stage 3rd order TVD"
"Explicit 4 Stage 3rd order by Runge"
"Explicit 4 Stage"
"Explicit 3/8 Rule"
"Backward Euler"
"Singly Diagonal IRK 2 Stage 2nd order"
"Singly Diagonal IRK 2 Stage 3rd order"
"Singly Diagonal IRK 3 Stage 4th order"
"Singly Diagonal IRK 5 Stage 4th order"
"Singly Diagonal IRK 5 Stage 5th order"
"Diagonal IRK 2 Stage 3rd order"
"Implicit 1 Stage 2nd order Gauss"
"Implicit 2 Stage 4th order Gauss"
"Implicit 3 Stage 6th order Gauss"
"Implicit 2 Stage 4th Order Hammer & Hollingsworth"
"Implicit 3 Stage 6th Order Kuntzmann & Butcher"
"Implicit 1 Stage 1st order Radau left"
"Implicit 2 Stage 3rd order Radau left"
"Implicit 3 Stage 5th order Radau left"
"Implicit 1 Stage 1st order Radau right"
"Implicit 2 Stage 3rd order Radau right"
"Implicit 3 Stage 5th order Radau right"
"Implicit 2 Stage 2nd order Lobatto A"
"Implicit 3 Stage 4th order Lobatto A"
"Implicit 4 Stage 6th order Lobatto A"
"Implicit 2 Stage 2nd order Lobatto B"
"Implicit 3 Stage 4th order Lobatto B"
"Implicit 4 Stage 6th order Lobatto B"
"Implicit 2 Stage 2nd order Lobatto C"
"Implicit 3 Stage 4th order Lobatto C"
"Implicit 4 Stage 6th order Lobatto C"

5.27 Forward Euler

Description:
```
Forward Euler
c = [ 0 ]'
A = [ 0 ]
b = [ 1 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.28 Explicit 2 Stage 2nd order by Runge

Description:
```
Explicit 2 Stage 2nd order by Runge
Also known as Explicit Midpoint
```

```
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S.P. Norsett, G. Wanner
Table 1.1, pg 135
c = [  0   1/2 ]'
A = [  0       ]
    [ 1/2  0   ]
b = [  0   1   ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.29 Explicit Trapezoidal

Description:

```
Explicit Trapezoidal
c = [  0    1  ]'
A = [  0       ]
    [  1    0  ]
b = [ 1/2 1/2 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.30 Explicit 3 Stage 3rd order

Description:

```
Explicit 3 Stage 3rd order
c = [  0   1/2  1   ]'
A = [  0            ]
    [ 1/2  0        ]
    [ -1   2    0   ]
b = [ 1/6 4/6 1/6 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.31 Explicit 3 Stage 3rd order by Heun

Description:

```
Explicit 3 Stage 3rd order by Heun
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S.P. Norsett, G. Wanner
Table 1.1, pg 135
c = [  0   1/3 2/3 ]'
A = [  0            ]
    [ 1/3  0        ]
    [  0   2/3  0   ]
b = [ 1/4  0   3/4 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.32 Explicit 3 Stage 3rd order TVD

Description:

```
Explicit 3 Stage 3rd order TVD
Sigal Gottlieb and Chi-Wang Shu
'Total Variation Diminishing Runge-Kutta Schemes'
Mathematics of Computation
Volume 67, Number 221, January 1998, pp. 73-85
c = [  0   1  1/2 ]'
A = [  0          ]
    [  1   0      ]
    [ 1/4 1/4  0  ]
b = [ 1/6 1/6 4/6 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.33 Explicit 4 Stage 3rd order by Runge

Description:

```
Explicit 4 Stage 3rd order by Runge
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S.P. Norsett, G. Wanner
Table 1.1, pg 135
c = [  0  1/2  1    1  ]'
A = [  0               ]
    [ 1/2  0           ]
    [  0   1   0       ]
    [  0   0   1    0  ]
b = [ 1/6 2/3  0  1/6 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.34 Explicit 4 Stage

Description:

```
Explicit 4 Stage
"The" Runge-Kutta Method (explicit):
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revised Edition
E. Hairer, S.P. Norsett, G. Wanner
Table 1.2, pg 138
c = [  0  1/2 1/2  1  ]'
A = [  0              ]
    [ 1/2  0          ]
    [  0  1/2  0      ]
    [  0   0   1   0  ]
b = [ 1/6 1/3 1/3 1/6 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.35 Explicit 3/8 Rule

Description:

```
Explicit 3/8 Rule
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S.P. Norsett, G. Wanner
Table 1.2, pg 138
c = [  0  1/3 2/3   1  ]'
A = [  0               ]
    [ 1/3  0           ]
    [-1/3  1   0       ]
    [  1  -1   1    0  ]
b = [ 1/8 3/8 3/8 1/8 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.36 Backward Euler

Description:

```
Backward Euler
c = [ 1 ]'
A = [ 1 ]
b = [ 1 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.37 Singly Diagonal IRK 2 Stage 2nd order

Description:

```
Singly Diagonal IRK 2 Stage 2nd order
Computer Methods for ODEs and DAEs
U. M. Ascher and L. R. Petzold
p. 106
gamma = (2+-sqrt(2))/2
c = [  gamma    1     ]'
A = [  gamma    0     ]
    [ 1-gamma  gamma ]
b = [ 1-gamma  gamma ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: **gamma = 0.292893** The default value is gamma = (2-sqrt(2))/2 = 0.29289321881345243. This will produce an L-stable 2nd order method with the stage times within the timestep. Other values of gamma will still produce an L-stable scheme, but will only be 1st order accurate.

5.38 Singly Diagonal IRK 2 Stage 3rd order

Description:

```
Singly Diagonal IRK 2 Stage 3rd order
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S. P. Norsett, and G. Wanner
Table 7.2, pg 207
gamma = (3+-sqrt(3))/6 -> 3rd order and A-stable
gamma = (2+-sqrt(2))/2 -> 2nd order and L-stable
c = [  gamma      1-gamma  ]'
A = [  gamma      0        ]
    [ 1-2*gamma  gamma    ]
b = [ 1/2        1/2      ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: **3rd Order A-stable = 1** If true, set gamma to gamma = (3+sqrt(3))/6 to obtain a 3rd order A-stable scheme. '3rd Order A-stable' and '2nd Order L-stable' can not both be true.

2nd Order L-stable = 0 If true, set gamma to gamma = (2+sqrt(2))/2 to obtain a 2nd order L-stable scheme. '3rd Order A-stable' and '2nd Order L-stable' can not both be true.

gamma = 0.788675 If both '3rd Order A-stable' and '2nd Order L-stable' are false, gamma will be used. The default value is the '3rd Order A-stable' gamma value, (3+sqrt(3))/6.

5.39 Singly Diagonal IRK 3 Stage 4th order

Description:

```
Singly Diagonal IRK 3 Stage 4th order
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
pg100
gamma = (1/sqrt(3))*cos(pi/18)+1/2
delta = 1/(6*(2*gamma-1)^2)
c = [ gamma      1/2        1-gamma ]'
A = [ gamma                         ]
    [ 1/2-gamma  gamma              ]
    [ 2*gamma    1-4*gamma  gamma   ]
b = [ delta      1-2*delta  delta   ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.40 Singly Diagonal IRK 5 Stage 4th order

Description:

```
Singly Diagonal IRK 5 Stage 4th order
L-stable
Solving Ordinary Differential Equations II:
```

```
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
pg100
c  = [ 1/4       3/4        11/20   1/2     1   ]'
A  = [ 1/4                                      ]
     [ 1/2       1/4                            ]
     [ 17/50     -1/25      1/4                 ]
     [ 371/1360  -137/2720  15/544  1/4         ]
     [ 25/24     -49/48     125/16  -85/12  1/4 ]
b  = [ 25/24     -49/48     125/16  -85/12  1/4 ]'
b' = [ 59/48     -17/96     225/32  -85/12  0   ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.41 Singly Diagonal IRK 5 Stage 5th order

Description:

```
Singly Diagonal IRK 5 Stage 5th order
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
pg101
c = [ (6-sqrt(6))/10                (6+9*sqrt(6))/35             1
A = [ (6-sqrt(6))/10
    [ (-6+5*sqrt(6))/14             (6-sqrt(6))/10
    [ (888+607*sqrt(6))/2850        (126-161*sqrt(6))/1425       (6-sqrt(6))/10
    [ (3153-3082*sqrt(6))/14250     (3213+1148*sqrt(6))/28500    (-267+88*sqrt(6))/5
    [ (-32583+14638*sqrt(6))/71250  (-17199+364*sqrt(6))/142500  (1329-544*sqrt(6))/
b = [ 0                             0                            1/9
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.42 Diagonal IRK 2 Stage 3rd order

Description:

```
Diagonal IRK 2 Stage 3rd order
Hammer & Hollingsworth method
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S. P. Norsett, and G. Wanner
Table 7.1, pg 205
c = [  0   2/3 ]'
A = [  0    0  ]
    [ 1/3  1/3 ]
b = [ 1/4  3/4 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.43 Implicit 1 Stage 2nd order Gauss

Description:
```
Implicit 1 Stage 2nd order Gauss
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.2, pg 72
Also:  Implicit midpoint rule
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S. P. Norsett, and G. Wanner
Table 7.1, pg 205
c = [ 1/2 ]'
A = [ 1/2 ]
b = [  1  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.44 Implicit 2 Stage 4th order Gauss

Description:
```
Implicit 2 Stage 4th order Gauss
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.2, pg 72
c = [ 1/2-sqrt(3)/6  1/2+sqrt(3)/6 ]'
A = [ 1/4            1/4-sqrt(3)/6 ]
    [ 1/4+sqrt(3)/6  1/4           ]
b = [ 1/2            1/2 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.45 Implicit 3 Stage 6th order Gauss

Description:
```
Implicit 3 Stage 6th order Gauss
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.2, pg 72
```

```
c = [ 1/2-sqrt(15)/10   1/2             1/2+sqrt(15)/10  ]'
A = [ 5/36              2/9-sqrt(15)/15  5/36-sqrt(15)/30 ]
    [ 5/36+sqrt(15)/24  2/9              5/36-sqrt(15)/24 ]
    [ 5/36+sqrt(15)/30  2/9+sqrt(15)/15  5/36             ]
b = [ 5/18              4/9              5/18             ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.46 Implicit 2 Stage 4th Order Hammer & Hollingsworth

Description:

```
Implicit 2 Stage 4th Order Hammer & Hollingsworth
Hammer & Hollingsworth method
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S. P. Norsett, and G. Wanner
Table 7.3, pg 207
c = [ 1/2-sqrt(3)/6  1/2+sqrt(3)/6 ]'
A = [ 1/4            1/4-sqrt(3)/6 ]
    [ 1/4+sqrt(3)/6  1/4           ]
b = [ 1/2            1/2           ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.47 Implicit 3 Stage 6th Order Kuntzmann & Butcher

Description:

```
Implicit 3 Stage 6th Order Kuntzmann & Butcher
Kuntzmann & Butcher method
Solving Ordinary Differential Equations I:
Nonstiff Problems, 2nd Revided Edition
E. Hairer, S. P. Norsett, and G. Wanner
Table 7.4, pg 209
c = [ 1/2-sqrt(15)/10   1/2             1/2-sqrt(15)/10  ]'
A = [ 5/36              2/9-sqrt(15)/15  5/36-sqrt(15)/30 ]
    [ 5/36+sqrt(15)/24  2/9              5/36-sqrt(15)/24 ]
    [ 5/36+sqrt(15)/30  2/9+sqrt(15)/15  5/36             ]
b = [ 5/18              4/9              5/18             ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.48 Implicit 1 Stage 1st order Radau left

Description:

```
Implicit 1 Stage 1st order Radau left
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
```

```
E. Hairer and G. Wanner
Table 5.3, pg 73
c = [ 0 ]'
A = [ 1 ]
b = [ 1 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.49 Implicit 2 Stage 3rd order Radau left

Description:

```
Implicit 2 Stage 3rd order Radau left
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.3, pg 73
c = [  0    2/3 ]'
A = [ 1/4  -1/4 ]
    [ 1/4  5/12 ]
b = [ 1/4  3/4  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.50 Implicit 3 Stage 5th order Radau left

Description:

```
Implicit 3 Stage 5th order Radau left
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.4, pg 73
c = [  0   (6-sqrt(6))/10      (6+sqrt(6))/10      ]'
A = [ 1/9  (-1-sqrt(6))/18     (-1+sqrt(6))/18     ]
    [ 1/9  (88+7*sqrt(6))/360  (88-43*sqrt(6))/360 ]
    [ 1/9  (88+43*sqrt(6))/360 (88-7*sqrt(6))/360  ]
b = [ 1/9  (16+sqrt(6))/36     (16-sqrt(6))/36     ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.51 Implicit 1 Stage 1st order Radau right

Description:
```
Implicit 1 Stage 1st order Radau right
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.5, pg 74
c = [ 1 ]'
A = [ 1 ]
b = [ 1 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.52 Implicit 2 Stage 3rd order Radau right

Description:
```
Implicit 2 Stage 3rd order Radau right
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.5, pg 74
c = [ 1/3     1    ]'
A = [ 5/12  -1/12 ]
    [ 3/4    1/4  ]
b = [ 3/4    1/4  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.53 Implicit 3 Stage 5th order Radau right

Description:
```
Implicit 3 Stage 5th order Radau right
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.6, pg 74
c = [ (4-sqrt(6))/10          (4+sqrt(6))/10              1                ]'
A = [ (88-7*sqrt(6))/360      (296-169*sqrt(6))/1800  (-2+3*sqrt(6))/225  ]
    [ (296+169*sqrt(6))/1800  (88+7*sqrt(6))/360      (-2-3*sqrt(6))/225  ]
    [ (16-sqrt(6))/36         (16+sqrt(6))/36         1/9                 ]
b = [ (16-sqrt(6))/36         (16+sqrt(6))/36         1/9                 ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.54 Implicit 2 Stage 2nd order Lobatto A

Description:
```
Implicit 2 Stage 2nd order Lobatto A
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.7, pg 75
c = [  0    1   ]'
A = [  0    0   ]
    [ 1/2  1/2  ]
b = [ 1/2  1/2  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.55 Implicit 3 Stage 4th order Lobatto A

Description:
```
Implicit 3 Stage 4th order Lobatto A
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.7, pg 75
c = [  0    1/2    1  ]'
A = [  0     0     0   ]
    [ 5/24  1/3  -1/24  ]
    [ 1/6   2/3   1/6   ]
b = [ 1/6   2/3   1/6   ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.56 Implicit 4 Stage 6th order Lobatto A

Description:
```
Implicit 4 Stage 6th order Lobatto A
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.8, pg 75
c = [ 0               (5-sqrt(5))/10      (5+sqrt(5))/10      1                  ]'
A = [ 0               0                   0                   0                  ]
    [ (11+sqrt(5)/120 (25-sqrt(5))/120    (25-13*sqrt(5))/120 (-1+sqrt(5))/120   ]
    [ (11-sqrt(5)/120 (25+13*sqrt(5))/120 (25+sqrt(5))/120    (-1-sqrt(5))/120   ]
    [ 1/12            5/12                5/12                1/12               ]
b = [ 1/12            5/12                5/12                1/12               ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.57 Implicit 2 Stage 2nd order Lobatto B

Description:
```
Implicit 2 Stage 2nd order Lobatto B
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.9, pg 76
c = [  0    1   ]'
A = [ 1/2   0   ]
    [ 1/2   0   ]
b = [ 1/2  1/2  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.58 Implicit 3 Stage 4th order Lobatto B

Description:
```
Implicit 3 Stage 4th order Lobatto B
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.9, pg 76
c = [  0    1/2    1   ]'
A = [ 1/6  -1/6    0   ]
    [ 1/6   1/3    0   ]
    [ 1/6   5/6    0   ]
b = [ 1/6   2/3   1/6  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.59 Implicit 4 Stage 6th order Lobatto B

Description:
```
Implicit 4 Stage 6th order Lobatto B
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.10, pg 76
c = [ 0     (5-sqrt(5))/10       (5+sqrt(5))/10        1     ]'
```

```
A = [ 1/12  (-1-sqrt(5))/24      (-1+sqrt(5))/24      0     ]
    [ 1/12  (25+sqrt(5))/120     (25-13*sqrt(5))/120  0     ]
    [ 1/12  (25+13*sqrt(5))/120  (25-sqrt(5))/120     0     ]
    [ 1/12  (11-sqrt(5))/24      (11+sqrt(5))/24      0     ]
b = [ 1/12  5/12                 5/12                 1/12  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.60 Implicit 2 Stage 2nd order Lobatto C

Description:

```
Implicit 2 Stage 2nd order Lobatto C
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.11, pg 76
c = [  0     1   ]'
A = [ 1/2 -1/2  ]
    [ 1/2  1/2  ]
b = [ 1/2  1/2  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.61 Implicit 3 Stage 4th order Lobatto C

Description:

```
Implicit 3 Stage 4th order Lobatto C
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.11, pg 76
c = [  0     1/2    1   ]'
A = [ 1/6   -1/3   1/6  ]
    [ 1/6    5/12 -1/12 ]
    [ 1/6    2/3   1/6  ]
b = [ 1/6    2/3   1/6  ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.62 Implicit 4 Stage 6th order Lobatto C

Description:
```
Implicit 4 Stage 6th order Lobatto C
A-stable
Solving Ordinary Differential Equations II:
Stiff and Differential-Algebraic Problems,
2nd Revised Edition
E. Hairer and G. Wanner
Table 5.12, pg 76
c = [ 0     (5-sqrt(5))/10      (5+sqrt(5))/10      1          ]'
A = [ 1/12  -sqrt(5)/12         sqrt(5)/12         -1/12       ]
    [ 1/12  1/4                 (10-7*sqrt(5))/60  sqrt(5)/60  ]
    [ 1/12  (10+7*sqrt(5))/60   1/4                -sqrt(5)/60 ]
    [ 1/12  5/12                5/12                1/12       ]
b = [ 1/12  5/12                5/12                1/12       ]'
```

Parent(s): Runge Kutta Butcher Tableau Selection (Section 5.26)

Child(ren): None.

Parameters: None.

5.63 Interpolation Buffer Settings

Description:

Parent(s): Integrator Base (Section 5.1)

Child(ren): Trailing Interpolation Buffer Selection (Section 5.64)
Interpolation Buffer Appender Selection (Section 5.66)
Interpolator Selection (Section 5.68)

Parameters: None.

5.64 Trailing Interpolation Buffer Selection

Description:

Parent(s): Interpolation Buffer Settings (Section 5.63)

Child(ren): Interpolation Buffer (Section 5.65)

Parameters: **Interpolation Buffer Type = None** Determines the type of Rythmos::InterpolationBuffer object that will be built. The parameters for each Interpolation Buffer Type are specified in this sublist

Valid std::string values:
"None"
"Interpolation Buffer"

5.65 Interpolation Buffer

Description:

Parent(s): Trailing Interpolation Buffer Selection (Section 5.64)

Child(ren): None.

Parameters: **InterpolationBufferPolicy = Keep Newest Policy** Interpolation Buffer Policy for when the maximum storage size is exceeded. Static will throw an exception when the storage limit is exceeded. Keep Newest will over-write the oldest data in the buffer when the storage limit is exceeded.

Valid std::string values:
"Invalid Policy"
"Static Policy"
"Keep Newest Policy"

StorageLimit = 0 Storage limit for the interpolation buffer.

5.66 Interpolation Buffer Appender Selection

Description:

Parent(s): Interpolation Buffer Settings (Section 5.63)

Child(ren): Pointwise Interpolation Buffer Appender (Section 5.67)

Parameters: **Interpolation Buffer Appender Type = None** Determines the type of Rythmos::InterpolationBufferAppender object that will be built. The parameters for each Interpolation Buffer Appender Type are specified in this sublist

Valid std::string values:
"None"
"Pointwise Interpolation Buffer Appender"

5.67 Pointwise Interpolation Buffer Appender

Description:

Parent(s): Interpolation Buffer Appender Selection (Section 5.66)

Child(ren): None.

Parameters: None.

5.68 Interpolator Selection

Description:

Parent(s): Interpolation Buffer Settings (Section 5.63)

Child(ren): Linear Interpolator (Section 5.69)
Hermite Interpolator (Section 5.70)
Cubic Spline Interpolator (Section 5.71)

Parameters: **Interpolator Type = None** Determines the type of Rythmos::Interpolator object that will be built. The parameters for each Interpolator Type are specified in this sublist

Valid std::string values:
"None"
"Linear Interpolator"
"Hermite Interpolator"
"Cubic Spline Interpolator"

5.69 Linear Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.68)

Child(ren): None.

Parameters: None.

5.70 Hermite Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.68)

Child(ren): None.

Parameters: None.

5.71 Cubic Spline Interpolator

Description:

Parent(s): Interpolator Selection (Section 5.68)

Child(ren): None.

Parameters: None.

6 Convergence Test Examples

6.1 Dahlquist Test Equation

The Dahlquist Test equation,

$$\dot{x} = \lambda x, \tag{13}$$

is used to investigate and classify the properties of integrators. Its solution is $x = ce^{\lambda t}$ where λ can be complex with positive, negative coefficients, and provides a variety of canonical problems (*e.g.*, exponentially increasing and decaying solutions, and oscillatory solutions). By implementing time integrators in the Dahlquist test equation, the amplification or stability factor, $R(z)$, can be determined by writing it in the update form

$$x_n = R(\lambda \Delta t)\, x_{n-1} = R(z)\, x_{n-1}$$

where $z = \lambda \Delta t$. A few important measures of stability are absolute stability, A-stability and L-stability and are defined by z and $R(z)$. First off, absolute stability requires the solution has no growth from one time step to the next,

$$|x_n| \leq |x_{n-1}|$$

and this true when

$$|R(z)| \leq 1$$

which defines the stability domain. A-stable methods have a stability domain for the negative half domain where $Re(z) \leq 0$. Although A-stability is useful for many non-stiff equations, stiff equations require additional stability properties to A-stability, $|R(z)| \to 0$ as $z \to -\infty$, and is known as L-stability.

6.2 SinCos Problem

This is a canonical Sine-Cosine differential equation

$$\ddot{\mathbf{x}} = -\mathbf{x}$$

with a few enhancements. We start with the exact solution to the differential equation

$$\begin{aligned} x_0(t) &= a + b * \sin((f/L) * t + \phi) \\ x_1(t) &= b * (f/L) * \cos((f/L) * t + \phi) \end{aligned}$$

then the form of the model is:

$$\begin{aligned} \frac{d}{dt} x_0(t) &= x_1(t) \\ \frac{d}{dt} x_1(t) &= \left(\frac{f}{L}\right)^2 (a - x_0(t)) \end{aligned}$$

where the default parameter values are $a = 0$, $f = 1$, and $L = 1$, and the initial conditions

$$\begin{aligned} x_0(t_0 = 0) &= \gamma_0 [= 0] \\ x_1(t_0 = 0) &= \gamma_1 [= 1] \end{aligned}$$

determine the remaining coefficients

$$\begin{aligned} \phi &= \arctan(((f/L)/\gamma_1) * (\gamma_0 - a)) - (f/L) * t_0 [= 0] \\ b &= \gamma_1 / ((f/L) * cos((f/L) * t_0 + \phi)) [= 1] \end{aligned}$$

Therefore this model has three model parameters and two initial conditions which effect the exact solution as above.

$$\mathbf{p} = (a, f, L)$$

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, t, \mathbf{p})$$

where

$$\begin{aligned} F_0 &= x_1 \\ F_1 &= \left(\frac{f}{L}\right)^2 (a - x_0) \end{aligned}$$

The exact sensitivities, $\mathbf{s} = \partial \mathbf{x}/\partial \mathbf{p}$, for the problem are specified as

$$\mathbf{s}(t) = \begin{bmatrix} 1 & 0 \\ \left(\frac{b}{L}\right) t \cos\left(\left(\frac{f}{L}\right) t + \phi\right) & \left(\frac{b}{L}\right) \cos\left(\left(\frac{f}{L}\right) t + \phi\right) - \frac{b\,f\,t}{L^2} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) \\ -\frac{b\,f\,t}{L^2} \cos\left(\left(\frac{f}{L}\right) t + \phi\right) & -\frac{b\,f}{L^2} \cos\left(\left(\frac{f}{L}\right) t + \phi\right) + \frac{b\,f^2\,t}{L^3} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) \end{bmatrix}$$

and for the default initial conditions, $\phi = 0$ and $b = 1$

$$\mathbf{s}(t = 0) = \begin{bmatrix} 1 & 0 \\ 0 & \frac{b}{L} \\ 0 & -\frac{f}{L^2} \end{bmatrix}$$

The time differentiated sensivtivities, $\dot{\mathbf{s}} = \partial \mathbf{s}/\partial t = \partial/\partial t(\partial \mathbf{x}/\partial \mathbf{p}) = \partial/\partial \mathbf{p}(\partial \mathbf{x}/\partial t)$ are

$$\dot{\mathbf{s}}(t) = \begin{bmatrix} 0 & 0 \\ \left(\frac{b}{L}\right) \cos\left(\left(\frac{f}{L}\right) t + \phi\right) - \frac{b\,f\,t}{L^2} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) & -\frac{2b\,f}{L^2} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) \left(\frac{b}{L}\right) - \frac{b\,f^2\,t}{L^3} \cos\left(\left(\frac{f}{L}\right) t + \phi\right) \\ -\frac{b\,f}{L^2} \cos\left(\left(\frac{f}{L}\right) t + \phi\right) + \frac{b\,f^2\,t}{L^3} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) & \frac{2b\,f^2}{L^3} \sin\left(\left(\frac{f}{L}\right) t + \phi\right) + \frac{b\,f^3\,t}{L^4} \cos\left(\left(\frac{f}{L}\right) t + \phi\right) \end{bmatrix}$$

Part III
Developer's Guide

Part IV
Back Matter

Index

2nd Order L-stable, 39
3rd Order A-stable, 39

absErrTol, 31
Absolute Error Tolerance, 32

Backward Euler, 25–27, 29, 33, 38

constantStepSize, 31
Cubic Spline Interpolator, 25, 27, 32, 33, 49, 50

Default Integrator, 25, 26
  Max Number Time Steps, 26
Diagonal IRK 2 Stage 3rd order, 27, 34, 40

Error Weight Vector Calculator Selection, 25, 30, 32
  Error Weight Vector Calculator Type, 32
Error Weight Vector Calculator Type, 32
Explicit 2 Stage 2nd order by Runge, 25, 27, 33, 35
Explicit 3 Stage 3rd order, 25, 27, 33, 36
Explicit 3 Stage 3rd order by Heun, 25, 27, 33, 36
Explicit 3 Stage 3rd order TVD, 27, 33, 37
Explicit 3/8 Rule, 27, 33, 38
Explicit 4 Stage, 27, 33, 37
Explicit 4 Stage 3rd order by Runge, 27, 33, 37
Explicit RK, 25, 27, 29, 30
Explicit Trapezoidal, 25, 27, 33, 36

failStepIfNonlinearSolveFails, 31
Final Time, 26
Fixed dt, 28
Forward Euler, 25–27, 29, 33, 35

gamma, 38, 39

h0_max_factor, 31
h0_safety, 31
h_max_inv, 31
h_phase0_incr, 31
Hermite Interpolator, 25, 27, 32, 33, 49, 50

Implicit 1 Stage 1st order Radau left, 27, 34, 42
Implicit 1 Stage 1st order Radau right, 27, 34, 44
Implicit 1 Stage 2nd order Gauss, 27, 34, 41
Implicit 2 Stage 2nd order Lobatto A, 27, 34, 45
Implicit 2 Stage 2nd order Lobatto B, 27, 34, 46
Implicit 2 Stage 2nd order Lobatto C, 27, 34, 47
Implicit 2 Stage 3rd order Radau left, 27, 34, 43
Implicit 2 Stage 3rd order Radau right, 27, 34, 44
Implicit 2 Stage 4th order Gauss, 27, 34, 41
Implicit 2 Stage 4th Order Hammer & Hollingsworth, 27, 34, 42
Implicit 3 Stage 4th order Lobatto A, 27, 34, 45
Implicit 3 Stage 4th order Lobatto B, 27, 34, 46
Implicit 3 Stage 4th order Lobatto C, 27, 34, 47
Implicit 3 Stage 5th order Radau left, 27, 34, 43
Implicit 3 Stage 5th order Radau right, 27, 34, 44
Implicit 3 Stage 6th order Gauss, 27, 34, 41
Implicit 3 Stage 6th Order Kuntzmann & Butcher, 27, 34, 42
Implicit 4 Stage 6th order Lobatto A, 27, 34, 45
Implicit 4 Stage 6th order Lobatto B, 27, 34, 46
Implicit 4 Stage 6th order Lobatto C, 27, 34, 48
Implicit BDF, 25, 26, 29
Implicit BDF Stepper Error Weight Vector Calculator, 25, 27, 32
Implicit BDF Stepper Ramping Step Control Strategy, 25, 30, 31
  Absolute Error Tolerance, 32
  Initial Step Size, 31
  Max Order, 32
  Max Step Size, 32
  Min Order, 32
  Min Step Size, 32
  Number of Constant First Order Steps, 31
  Relative Error Tolerance, 32
  Step Size Decrease Factor, 32
  Step Size Increase Factor, 32
  Use LET To Determine Step Acceptance, 32
Implicit BDF Stepper Step Control Strategy, 25, 30, 31
  absErrTol, 31
  constantStepSize, 31
  failStepIfNonlinearSolveFails, 31
  maxOrder, 30
  minOrder, 30
  relErrTol, 31
  stopTime, 31
Implicit RK, 25, 27, 29, 30
Initial Step Size, 31
Integration Control Strategy Selection, 25, 26, 28
  Integration Control Strategy Type, 28
Integration Control Strategy Type, 28
Integrator Base, 25, 26, 28, 48
Integrator Selection, 25, 26
  Integrator Type, 26
Integrator Settings, 25, 26
  Final Time, 26
  Land On Final Time, 26
Integrator Type, 26
Interpolation Buffer, 25, 27, 48

InterpolationBufferPolicy, 49
StorageLimit, 49
Interpolation Buffer Appender Selection, 25, 48, 49
Interpolation Buffer Appender Type, 49
Interpolation Buffer Appender Type, 49
Interpolation Buffer Settings, 25, 26, 48, 49
Interpolation Buffer Type, 48
InterpolationBufferPolicy, 49
Interpolator Selection, 25, 29, 32, 33, 48–50
Interpolator Type, 33, 49
Interpolator Type, 33, 49

Land On Final Time, 26
Linear Interpolator, 25, 27, 32, 33, 49, 50

magicNumbers, 25, 27, 30, 31
h0_max_factor, 31
h0_safety, 31
h_max_inv, 31
h_phase0_incr, 31
max_LET_fail, 31
maxTimeStep, 31
minTimeStep, 31
r_factor, 31
r_fudge, 31
r_hincr, 31
r_hincr_test, 31
r_max, 31
r_min, 31
r_safety, 31
Tkm1_Tk_safety, 31
Tkp1_Tk_safety, 31
Max dt, 28
Max Number Time Steps, 26
Max Order, 32
Max Step Size, 32
max_LET_fail, 31
maxOrder, 30
maxTimeStep, 31
Min Order, 32
Min Step Size, 32
minOrder, 30
minTimeStep, 31

Number of Constant First Order Steps, 31
Number of Time Steps, 28

Output File, 28

Pointwise Interpolation Buffer Appender, 25, 27, 49

r_factor, 31
r_fudge, 31
r_hincr, 31
r_hincr_test, 31
r_max, 31
r_min, 31
r_safety, 31
Relative Error Tolerance, 32
relErrTol, 31
Runge Kutta Butcher Tableau Selection, 25, 29, 33, 35–48
Runge Kutta Butcher Tableau Type, 35
Runge Kutta Butcher Tableau Type, 35

Simple Integration Control Strategy, 25, 26, 28
Fixed dt, 28
Max dt, 28
Number of Time Steps, 28
Take Variable Steps, 28
Singly Diagonal IRK 2 Stage 2nd order, 27, 33, 38
gamma, 38
Singly Diagonal IRK 2 Stage 3rd order, 27, 33, 39
2nd Order L-stable, 39
3rd Order A-stable, 39
gamma, 39
Singly Diagonal IRK 3 Stage 4th order, 27, 34, 39
Singly Diagonal IRK 5 Stage 4th order, 27, 34, 39
Singly Diagonal IRK 5 Stage 5th order, 27, 34, 40
Step Control Settings, 25, 29, 30, 32
Step Control Strategy Selection, 25, 30, 31
Step Control Strategy Type, 30
Step Control Strategy Type, 30
Step Size Decrease Factor, 32
Step Size Increase Factor, 32
Stepper Selection, 25, 29, 30
Stepper Type, 29
Stepper Settings, 25, 26, 28–30, 32, 33
Stepper Type, 29
stopTime, 31
StorageLimit, 49

Take Variable Steps, 28
Tkm1_Tk_safety, 31
Tkp1_Tk_safety, 31
Trailing Interpolation Buffer Selection, 25, 48
Interpolation Buffer Type, 48

Use LET To Determine Step Acceptance, 32

VerboseObject, 25, 26
Output File, 28
Verbosity Level, 28

Verbosity Level, 28

References

[1] Sundials website. `http://sundials.wikidot.com/bdf-flc-derivation`, April 2007.

[2] U. M. Ascher and L. R. Petzold. *Computer methods for ordinary differential equations and differential-algebraic equations.* SIAM, Philadelphia, 1998.

[3] K. E. Brennan, S. L. Campbell, and L. R. Petzold. *Numerical solution of initial-value problems in differential-algebraic equations*, volume 14 of *Classics in Applied Mathematics.* SIAM, Philadelphia, PA, first edition, 1996.

[4] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold. Consistent initial condition calculation for differential-algebraic systems. *SIAM J. Sci. Comput.*, 19:1495–1512, 1998.

[5] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, volume 14 of *Springer Series in Computational Mathematics.* Springer-Verlag, Berlin, Germany, second edition, 1996.

[6] K. R. Jackson and R. Sacks-Davis. An alternative implementation of variable step-size multistep formulas for stiff odes. *ACM Transactions on Mathematical Software*, 6(3):295–318, 1980.