Restarting algorithms for simulation problems with discontinuities

Fatemeh Mohammadi Carmen Arévalo Claus Führer[*]
Numerical Analysis, Center of Mathematical Sciences, Lund University
Sölvegatan 18, SE-22100 Lund, Sweden

Abstract

Modelica's language support includes so-called events for describing discontinuities. Modern integrating environments, like Assimulo, provide elaborate event detection and event handling methods. In addition, the overall performance of a simulation of models with discontinuities (hybrid models) depends strongly on the methods for restarting the integration after an event detection. The present paper reviews two restarting methods for multistep methods, both based on Runge–Kutta starters, and presents preliminary first experiments with Assimulo and LSODAR as a proof of concept, which motivates to apply the technique to hybrid systems described in Modelica and simulated by JModelica.org/PyFMI and Assimulo [1, 3, 2].

Keywords: events, discontinuities, hybrid systems, multistep method, Runge–Kutta method, simulation restart

1 Introduction

When dealing with hybrid systems, i.e. dynamic systems with state or time discontinuities, much emphasis has been put on the modeling aspect. Attempts to standardize the formulation of events and algorithms for event detection were in the focus of development and research, e.g. [4]. On the other hand, the question of restarting complex integration methods like multistep methods, with their sophisticated internal error and order control algorithms and internal data representations, did not attract much attention. In this paper we want to take up and review two early ideas for restarting and to present some experiments using the JModerlica.org - PyFMI - Assimulo toolchain.

A multistep method is classically started by stepwise increasing the order of the method, starting with a first order method (implicit Euler method) and leading to a method having the operational order of the problem at hand. Simulations are often done for a set of parameterized models for which the operational order and also good guesses of initial step sizes are available from other simulation runs. Thus, a goal for improving the integration performance is to avoid costly starting phases and directly start the integrator with a method already having the operational order. To start such a higher order method several internal values are required. Here we consider two ideas for providing these values. In both cases the starting values are obtained from the stage values of a single Runge–Kutta step of a specially designed method. One of them uses state values, while the other is geared to Nordsieck based multistep methods like LSODAR.

[*]partly supported by LCCC - Lund Center for Control of Complex Engineering Systems

2 Runge–Kutta starter with state values

We demonstrate the principle by constructing a Runge–Kutta starter for a third order multistep method, [8].

Furthermore, we construct two error estimates for determining the starting step of both the Runge–Kutta starter and a class of multistep methods, i.e. Adams methods.

Such a Runge–Kutta starter has to have an internal stage of order 3 as soon as possible and all subsequent stages need to be at least of order 3. In addition; the final result should be of order 4 for the purpose of error estimation. It is well-known that to get third-order accuracy at least three internal stages are necessary, and to conserve this accuracy for subsequent stages we need to aim for a Runge–Kutta method with at least six stages, [5]. We will thus consider a 6-stage Runge–Kutta method.

For the initial value problem

$$y' = f(t,y), \qquad y(0) = y_0, \tag{1}$$

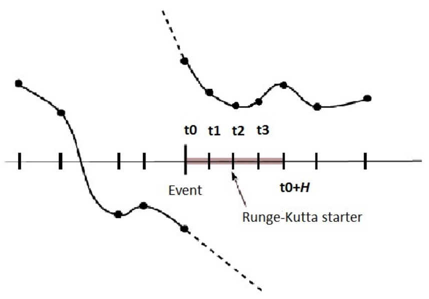

Figure 1: Runge–Kutta starter after a discontinuity

an s-stage explicit Runge–Kutta method can be written in the form,

$$\begin{aligned} k_i &:= f(t_0 + c_i H, g_{i-1}), \\ g_i &:= y_0 + H\sum_{j=1}^{i} a_{ij}k_j, \qquad i = 1,\dots,s, \\ y_1 &:= y_0 + H\sum_{j=1}^{s} b_j k_j, \end{aligned} \tag{2}$$

where y_1 is the numerical solution at $t_1 = t_0 + H$, H is the Runge–Kutta step size and k_i are stage derivatives.

f may be discontinuous but it is assumed to be piecewise smooth.

In the construction of an order 4, 6-stage explicit Runge–Kutta method, order conditions up to order four need to be satisfied. Let

$$\begin{aligned} b &:= (b_1, b_2, \cdots, b_s)^T, \\ a_i &:= (a_{i1}, a_{i2}, \cdots, a_{ii}), \\ C_i &:= \mathrm{diag}(c_1, \cdots, c_i), \\ A_i &:= (a_{jk})_{j,k=1}^{i}, \\ e_i &:= (1, 1, \cdots, 1)^T. \end{aligned} \tag{3}$$

When deriving the stage order conditions, we make use of the fact that in Eq. (2), the stage values g_i and y_i have structurally the same form. Therefore, we can derive the order conditions for internal stages in the same way.
The order conditions for a fourth-order Runge–Kutta method are

- order 1
$$b^T e_s = 1.$$
- order 2
$$b^T C_s e_s = \frac{1}{2}.$$
- order 3
$$b^T C_s^2 e_s = \frac{1}{3},$$
$$b^T A_s C_s e_s = \frac{1}{6}.$$
- order 4
$$\begin{aligned} b^T C_s^3 e_s &= \frac{1}{4}, \\ b^T C_s A_s C_s e_s &= \frac{1}{8}, \\ b^T A_s C_s^2 e_s &= \frac{1}{12}, \\ b^T A_s^2 C_s e_s &= \frac{1}{24}. \end{aligned} \tag{4}$$

Additionally we require

$$\sum_{j=1}^{s} a_{ij} = c_i. \tag{5}$$

The remaining order conditions for internal stages $i = 4, 5, 6$ are

- order 2
$$a_i^T C_i e_i = \frac{1}{2} c_i^2.$$
- order 3
$$\begin{aligned} a_i^T C_i^2 e_i &= \frac{1}{3} c_i^3, \\ a_i^T A_i C_i e_i &= \frac{1}{6} c_i^3. \end{aligned} \tag{6}$$

We want to both obtain third-order accuracy and minimize the truncation error bound. Raltson [6] showed that the third-order Runge-Kutta method which has the minimal error bound among all third-order Runge–Kutta methods is

$$\begin{aligned} k_1 &= hf(t_n, y_n), \\ k_2 &= hf(t_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1), \\ k_3 &= hf(t_n + \frac{3}{4}h, y_n + \frac{3}{4}k_2), \\ y_{n+1} &= y_n + \frac{2}{9}k_1 + \frac{1}{3}k_2 + \frac{4}{9}k_3. \end{aligned} \tag{7}$$

This implies the Butcher tableau for the first four stages is

$$\begin{array}{c|ccc} 0 & 0 & & \\ \frac{1}{2}c_4 & \frac{1}{2}c_4 & 0 & \\ \frac{3}{4}c_4 & 0 & \frac{3}{4}c_4 & 0 \\ \hline c_4 & \frac{2}{9}c_4 & \frac{1}{3}c_4 & \frac{4}{9}c_4 \end{array}$$

From condition (5) we find

$$a_{21} = c_2, \qquad a_{31} = (1-\theta)c_3, \;\; a_{32} = \theta c_3. \quad (8)$$

We now substitute (6) and (5) in (4) to calculate the values b_i for the six stage Runge–Kutta method.

$$\begin{aligned} b_2c_2^3 + b_3c_3^3 + b_4c_4^3 + b_5c_5^3 + b_6c_6^3 &= \frac{1}{4}, \\ b_3c_3\theta c_3c_2 + \frac{1}{2}b_4c_4^3 + \frac{1}{2}b_5c_5^3 + \frac{1}{2}b_6c_6^3 &= \frac{1}{8}, \\ b_3c_3\theta c_2^2 + \frac{1}{3}b_4c_4^3 + \frac{1}{3}b_5c_5^3 + \frac{1}{3}b_6c_6^3 &= \frac{1}{12}, \\ \frac{1}{6}b_4c_4^3 + \frac{1}{6}b_5c_5^3 + \frac{1}{6}b_6c_6^3 &= \frac{1}{24}. \end{aligned} \quad (9)$$

For the first three stages we require $c_2 \neq 0, c_3 \neq 0$ and $\theta \neq 0$, otherwise a third-order Runge–Kutta method cannot be obtained. So $b_2 = b_3 = 0$ and Equations (9) reduce to a single equation

$$b_4c_4^3 + b_5c_5^3 + b_6c_6^3 = \frac{1}{4}.$$

We repeat this process for order 2 and 3 conditions, getting

$$b_4c_4^2 + b_5c_5^2 + b_6c_6^2 = \frac{1}{3},$$

and

$$b_4c_4 + b_5c_5 + b_6c_6 = \frac{1}{2}.$$

respectively.

Here we have a system of equations for given c_4, c_5, c_6,

$$\begin{pmatrix} c_4 & c_5 & c_6 \\ c_4^2 & c_5^2 & c_6^2 \\ c_4^3 & c_5^3 & c_6^3 \end{pmatrix} \begin{pmatrix} b_4 \\ b_5 \\ b_6 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{3} \\ \frac{1}{4} \end{pmatrix}.$$

We obtain a Vandermonde type matrix, which has, for distinct c_4, c_5 and c_6, a unique solution:

$$\begin{aligned} b_1 &= 1 - b_4 - b_5 - b_6, \\ b_4 &= \frac{3 - 4c_5 - 4c_6 + 6c_5c_6}{12c_4(c_4-c_5)(c_4-c_6)}, \\ b_5 &= \frac{3 - 4c_4 - 4c_6 + 6c_4c_6}{12c_5(c_4-c_5)(c_5-c_6)}, \\ b_6 &= \frac{3 - 4c_4 - 4c_5 + 6c_4c_5}{12c_6(c_4-c_6)(c_5-c_6)}. \end{aligned}$$

In order to obtain an equidistant grid for starting multistep methods, we can choose

$$c_4 = \frac{1}{4}, \qquad c_5 = \frac{1}{2}, \qquad c_6 = \frac{3}{4},$$

which gives

$$b_1 = b_2 = b_3 = 0, \quad b_4 = \frac{2}{3}, \quad b_5 = -\frac{1}{3}, \quad b_6 = \frac{2}{3}.$$

Finally, by solving the equations that guarantee the remaining order conditions, we obtain the Butcher tableau for the Runge–Kutta starter:

$$\begin{array}{c|cccccc} 0 & 0 \\ \frac{1}{8} & \frac{1}{8} & 0 \\ \frac{3}{16} & 0 & \frac{3}{16} & 0 \\ \frac{1}{4} & \frac{1}{18} & \frac{1}{12} & \frac{1}{9} & 0 \\ \frac{1}{2} & \frac{5}{12} & -\frac{1}{3} & -\frac{4}{9} & 1 & 0 \\ \frac{3}{4} & -\frac{1}{4} & \frac{3}{4} & 1 & -\frac{3}{2} & \frac{3}{4} & 0 \\ \hline & 0 & 0 & 0 & \frac{2}{3} & -\frac{1}{3} & \frac{2}{3} \end{array}$$

We can apply the explicit Runge–Kutta starter to start $k = 3$-step Adams methods. We need k data points $(t_i, f_i), i = n-k+1, \ldots, n$ to compute the respective polynomials for either the Adams–Moulton corrector or the Adams–Bashforth predictor.

3 Error estimation and step size control

The error of the numerical solution depends on the function f and on the step size H. The step size influences the size of the global error increment. Thus, for a given tolerance the step size is chosen in such a way that the global error increment meets a user-supplied tolerance bound.

We use an embedded formula to obtain an error estimate for the Runge–Kutta starter of the Adams method. The estimation can be done by reusing the available stages to produce a formula of different order. To do so, we apply stages k_4, k_5, k_6 of the Runge–Kutta method in Section 2 and obtain $\hat{y}_1$ by the third-order Adams–Bashforth method. We generate the difference table

$$\begin{array}{llll} c_4 = h & k_4 & & \\ & & \nabla k_5 & \\ c_5 = 2h & k_5 & & \nabla^2 k_6 \\ & & \nabla k_6 & \\ c_6 = 3h & k_6 & & \end{array}$$

where $h = \frac{H}{4}$ and H is the Runge–Kutta step size. The third-order Adams–Bashforth method is

$$\hat{y}_1 = g_6 + h\sum_{i=1}^{3}\gamma_{i-1}\nabla^{i-1}k_6 = g_6 + h\sum_{i=1}^{3}\gamma_i^\star k_{i+3}, \quad (10)$$

The latter is the Lagrange form of the Adams–Bashforth method and

$$\begin{aligned}\gamma_1^\star &= \gamma_2 = \frac{5}{12},\\ \gamma_2^\star &= -\gamma_1 - 2\gamma_2 = -\frac{4}{3},\\ \gamma_3^\star &= \gamma_0 + \gamma_1 + \gamma_2 = \frac{23}{12}.\end{aligned}$$

As we have

$$g_6 = y_0 + H\sum_{j=1}^{5} a_{6j}k_j, \quad (11)$$

we can rewrite equation (10) as

$$\hat{y}_1 = y_0 + H\sum_{j=1}^{6}\hat{b}_j k_j$$

Thus, the error estimate is

$$y_1 - \hat{y}_1 = y_0 + H\sum_{j=1}^{6} b_j k_j - \left(y_0 + H\sum_{j=1}^{6}\hat{b}_j k_j\right) = h\sum_{j=1}^{6}\hat{e}_j k_j, \quad (12)$$

giving the following coefficients

j	1	2	3	4	5	6
$\hat{b}_j$	$-\frac{1}{4}$	$\frac{3}{4}$	1	$-\frac{67}{48}$	$\frac{5}{12}$	$\frac{23}{48}$
$\hat{e}_j$	$\frac{1}{4}$	$-\frac{3}{4}$	-1	$\frac{99}{48}$	$-\frac{3}{4}$	$\frac{3}{16}$

This error estimation is the difference of a third-order predictor and the fourth-order result of the Runge–Kutta method that is applied to determine the step size for the Runge–Kutta starter.

We will now develop a second error estimate, to determine a step size for Adams method. We evaluate the right-hand side function f at the solution value y_1 and call it k_7. Then we generate the third-order Adams–Moulton corrector using k_5, k_6, k_7,

$$\begin{array}{llll} c_5 = h & k_5 & & \\ & & \nabla k_6 & \\ c_6 = 2h & k_6 & & \nabla^2 k_7 \\ & & \nabla k_7 & \\ c_7 = 3h & k_7 & & \end{array}$$

The third-order approximation by the Adams–Moulton method is

$$\tilde{y}_1 = g_6 + h\sum_{i=1}^{3}\beta_{i-1}\nabla^{i-1}k_7 = g_6 + h\sum_{i=1}^{3}\beta_i^\star k_{i+4}, \quad (13)$$

where the latter is the Lagrange form of the Adams–Moulton corrector and

$$\begin{aligned}\beta_1^\star &= \beta_2 = -\frac{1}{12},\\ \beta_2^\star &= -\beta_1 - 2\beta_2 = \frac{2}{3},\\ \beta_3^\star &= \beta_0 + \beta_1 + \beta_2 = \frac{5}{12}.\end{aligned}$$

From Equation (11) we can rewrite the third-order corrector in Runge–Kutta form

$$\tilde{y}_1 = y_0 + H\sum_{j=1}^{7}\tilde{b}_j k_j.$$

resulting in the following table:

j	1	2	3	4	5	6	7
$\tilde{b}_j$	$-\frac{1}{4}$	$\frac{3}{4}$	1	$-\frac{3}{2}$	$\frac{35}{48}$	$\frac{1}{6}$	$\frac{5}{48}$
$\tilde{e}_j$	$\frac{1}{4}$	$-\frac{3}{4}$	-1	$\frac{13}{6}$	$-\frac{51}{48}$	$\frac{1}{2}$	$-\frac{5}{48}$

The error estimate is used in determining the step size for starting the third-order Adams-Moulton method.

4 Runge–Kutta starter as an extrapolation method

The starting values of a multistep method can also be stored as a differentiation array, which constitutes the Nordsieck vector of scaled derivatives $\frac{h^i y^{(i)}}{i!}$, $i = 0,\ldots,p$. It is possible to convert a vector of state values at consecutive grid points into a Nordsieck array and vice versa without loss of accuracy. Classical multistep codes like LSODAR are based on Nordsieck formulations.

Based on such a Nordsieck formulation an alternative way of constructing a Runge–Kutta starter was developed by Gear, [5]. Here, the asymptotic expansion of the global error of a base method is used to construct a Runge-Kutta method with higher order stage values.

We use the explicit Euler method as a base method to compute y_i^m(the super-script m refers to the corresponding step size, $h_m = \frac{H}{m}$) for $i = 1,\dots,m$, $m = p, p-1,\dots,1$. From these values the terms in the asymptotic expansion, [7],

$$y_i^m \; = \; y(ih_m) \; + \; \sum_{q=1}^{p} e_q(ih_m)h^q \; + \; \mathscr{O}(H^{p+1}). \quad (14)$$

can successively be eliminated by an extrapolation technique until a method of a required order is obtained. The resulting method is known to be a Runge–Kutta method.

We exemplify the approach by aiming for third-order accurate Nordsieck values and restricting ourselves to autonomous differential equations for simplicity. The same process can be employed to obtain higher order accuracy.

Let $h = \frac{H}{m}$, and integrate the autonomous form of the differential equation (1) on the interval $[y_0, y_0 + H]$ with Euler's method, using m steps of size $\frac{H}{m}$ for $m = 3,2,1$.

For $m = 3$

$$\begin{aligned} y_1^3 &= y_0 + hf(y_0) = y_0 + k_1, & k_1 &= hf(y_0),\\ y_2^3 &= y_1^3 + hf(y_1^3) = y_0 + k_1 + k_2, & k_2 &= hf(y_1^3),\\ y_3^3 &= y_2^3 + hf(y_2^3) = y_0 + k_1 + k_2 + k_3, & k_3 &= hf(y_2^3). \end{aligned} \quad (15)$$

For $m = 2$

$$\begin{aligned} y_1^2 &= y_0 + \frac{3}{2}hf(y_0) = y_0 + \frac{3}{2}k_1,\\ y_2^2 &= y_1^3 + \frac{3}{2}hf(y_1^3) = y_0 + \frac{3}{2}k_1 + \frac{3}{2}k_4, \quad k_4 = hf(y_1^2). \end{aligned} \quad (16)$$

For $m = 1$

$$y_1^1 = y_0 + 3hf(y_0) = y_0 + 3k_1. \quad (17)$$

with $h = \frac{H}{3}$. We use approxmation formulas for higher derivatives

$$h_m^k y^{(k)}(\frac{H}{2}) = \sum_{i=0}^{m} d_{ik} y(ih_m) + \sum_{s=k+1}^{p} c_{sk} h_m^s y^s(\frac{H}{2}) + \mathscr{O}(H^{p+1}), \;\; m \geq k$$

and (14) to obtain, after some algebraic manipulations,

$$\begin{aligned} D_3^3 &= y_3^3 - 3y_2^3 + 3y_1^3 - y_0 = h^3 y^{(3)},\\ D_2^3 &= y_3^3 - y_2^3 - y_1^3 + y_0 = 2h^2 y^{(2)} + 2h^3 e_1^{(2)},\\ D_2^2 &= y_2^2 - 2y_1^2 + y_0 = (\frac{3h}{2})^2 y^{(2)} + (\frac{3h}{2})^3 e_1^{(2)},\\ D_1^3 &= y_2^3 - y_1^3 = hy^{(1)} + h^2 e_1^{(1)} + h^3 e_2^{(1)} + \frac{h^3}{24} y^{(3)},\\ D_1^2 &= y_2^2 - y_0 = 3hy^{(1)} + \frac{9}{2}h^2 e_1^{(1)} + \frac{27}{4}h^3 e_2^{(1)} + \frac{9}{8}h^3 y^{(3)},\\ D_1^1 &= y_1^1 - y_0 = 3hy^{(1)} + 9h^2 e_1^{(1)} + 27h^3 e_2^{(1)} + \frac{27}{24}h^3 y^{(3)}. \end{aligned} \quad (18)$$

All derivatives are evaluated at $\frac{H}{2}$ and $\mathscr{O}(h^4)$ terms are dropped. Estimates of the derivatives can be derived at any point within a constant multiple of the interval H with the same accuracy. We solve the system (18) to remove the error terms for $h^k y^{(k)}(\frac{H}{2})$, for $k = 1,\dots,m$, to get

$$\begin{aligned} h^3 y^{(3)}(\frac{H}{2}) &= D_3^3 + \mathscr{O}(h^4),\\ h^2 y^{(2)}(\frac{H}{2}) &= \frac{3}{2}D_2^3 - \frac{8}{9}D_2^2 + \mathscr{O}(h^4),\\ hy^{(1)}(\frac{H}{2}) &= \frac{9}{2}D_1^3 - \frac{4}{3}D_1^2 + \frac{1}{6}D_1^1 + \frac{9}{8}D_3^3 + \mathscr{O}(h^4). \end{aligned} \quad (19)$$

The D_k^m can be expressed as combinations of stage values k_i. All $\mathscr{O}(h^4)$ terms are neglected.

$$\begin{aligned} D_3^3 &= k_1 - 2k_2 + k_3,\\ D_2^3 &= -k_1 + k_3,\\ D_2^2 &= -\frac{3}{2}k_1 + \frac{3}{2}k_4,\\ D_1^3 &= k_2,\\ D_1^2 &= \frac{3}{2}k_1 + \frac{3}{2}k_4,\\ D_1^1 &= 3k_1. \end{aligned} \quad (20)$$

From (19) and (20),

$$\begin{aligned} h^3 y^{(3)}(\frac{H}{2}) &= k_1 - 2k_2 + k_3 + \mathscr{O}(h^4),\\ h^2 y^{(2)}(\frac{H}{2}) &= -\frac{1}{6}k_1 + \frac{3}{2}k_3 - \frac{4}{3}k_4 + \mathscr{O}(h^4),\\ hy^{(1)}(\frac{H}{2}) &= -\frac{3}{8}k_1 + \frac{9}{4}k_2 + \frac{9}{8}k_3 - 2k_4 + \mathscr{O}(h^4),\\ y(\frac{H}{2}) &= y_0 + \frac{3}{16}k_1 + \frac{18}{8}k_2 + \frac{9}{16}k_3 - \frac{12}{8}k_4 + \mathscr{O}(h^4). \end{aligned} \quad (21)$$

The first element of the Nordsieck vector, $y(\frac{H}{2})$, is computed by Taylor expansion.

It follows that

$$\Gamma_{\frac{H}{2}} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \frac{3}{16} & -\frac{3}{8} & -\frac{1}{6} & 1 \\ \frac{18}{8} & \frac{9}{4} & 0 & -2 \\ \frac{9}{16} & \frac{9}{8} & \frac{3}{2} & 1 \\ -\frac{12}{8} & -2 & -\frac{4}{3} & 0 \end{pmatrix}. \tag{22}$$

If the derivatives are instead computed at the origin, the matrix above becomes

$$\Gamma_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & -\frac{5}{3} & 1 \\ 0 & 0 & 3 & -2 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{4}{3} & 0 \end{pmatrix}. \tag{23}$$

The matrix A of coefficients $\alpha_{i,j}$ in equation (2) is obtained from Equations (15) and (16)

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \frac{3}{2} & 0 & 0 & 0 \end{pmatrix}. \tag{24}$$

For a fourth-order method we need at least six function evaluations, [5], and the relevant matrices Γ_0 and A are

$$\Gamma_0 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -\frac{5}{6} & \frac{4}{9} & -\frac{1}{9} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{4}{9} & \frac{1}{9} \\ 0 & 0 & \frac{7}{3} & -\frac{19}{9} & \frac{7}{9} \\ 0 & 0 & -3 & \frac{10}{3} & -\frac{4}{3} \\ 0 & 0 & 1 & -\frac{11}{9} & \frac{5}{9} \end{pmatrix},$$

$$A = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ \frac{3}{4} & 0 & \frac{9}{4} & 0 & 0 & 0 \\ \frac{1}{2} & 1 & \frac{1}{2} & 2 & 0 & 0 \\ \frac{1}{12} & 2 & \frac{1}{4} & \frac{2}{3} & 2 & 0 \end{pmatrix}.$$

The cost of this process in terms of function evaluations is $1 + \frac{p(p-1)}{2}$, since the interval H is integrated by Euler's method m times with step size $\frac{H}{m}$ for $m = p, p-1, \ldots, 1$. For the first value of m we have p function evaluations because the initial value of y' has to be evaluated once, so for the next value of m we have $p-2$ function evaluations, and so on.

We constructed a Nordsieck vector with third-order accuracy. To do this we used four stages k_1, k_2, k_3, k_4 as in (15) and (16) with lower order and an extrapolation technique. It can be shown that there exists no method of the same order with less stages and thus less function evaluations.

5 Order tests

To verify that the starter indeed achieves the expected order we consider the harmonic oscillator

$$y'' = -4y, \qquad y_0 = 1, \; y_0' = 0.$$

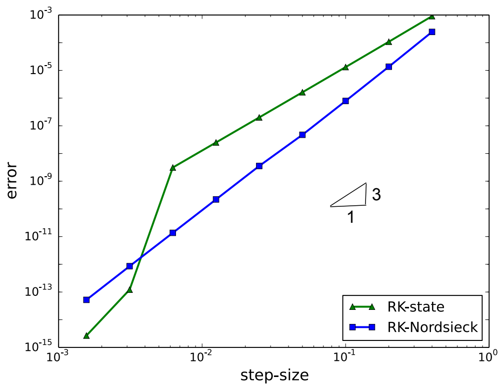

Figure 2: Both Runge–Kutta starters achieved third-order accuracy when solving the harmonic oscillator problem with the 3-step Adams-Moulton method

6 The bouncing ball test example

In this section we demonstrate the method on the example of a bouncing ball with linear damping $d = 0.1$:

$$\begin{aligned} \dot{y}_1 &= y_2 \\ \dot{y}_2 &= -dy_1 + 9.81 \end{aligned}$$

The bounces are modeled using a coefficient of restitution was chosen to be $c = 0.88$ to give the system sufficiently many impacts to be able to make a statement about the effect of restarting, see Fig. 3. The model includes two events, one to trigger bouncing and a second one which triggers the upper turning point. At the upper turning point the differential equation and its states remain unaltered, and only

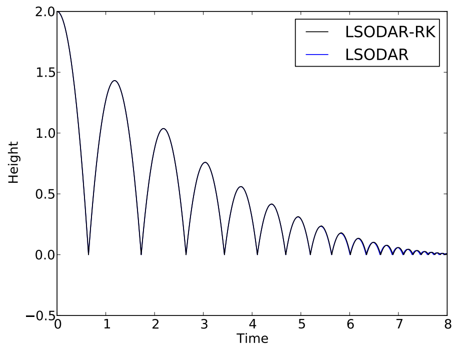

Figure 3: A simulation of a bouncing ball (damping: $d = 0.1$, coefficient of restitution $c = 0.88$).

the switch to control the bouncing event becomes activated. At the bouncing event the velocity $\dot{y}(t^-)$ is altered to $\dot{y}(t^+) = -c\dot{y}(t^-)$.

In Fig. 4 the step size and order history of both restarting techniques is compared. The classical starting procedure clearly shows a drop in order and step size. The method recovers quite quickly from the reduced step size as LSODAR allows exceptionally big step size changes during the initialization phase.

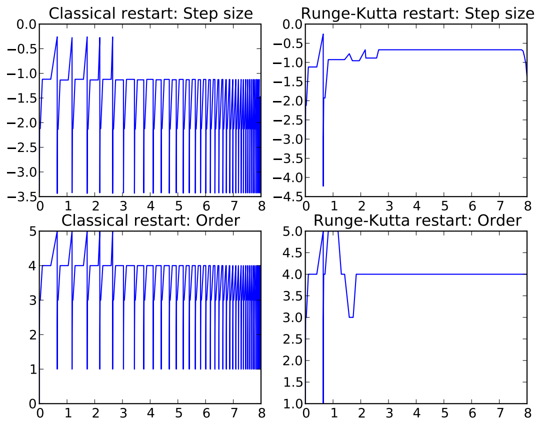

Figure 4: Comparison of the step size and order history for the two restarting approaches. A logarithmic scale is used for the step size plot.

The run statistics, cf. Tab. 1 show the effect of the Runge–Kutta starter in Assimulo. The gain in the number of function evaluations for this example is about 58%.

7 Conclusions

The aim of this paper is to study the effect of Runge–Kutta restarting techniques on the performance of the simulation of hybrid systems. Tests were made on a system with relatively small numbers of discontinuities. The tests give a clear indication that investigating a more sophisticated restarting procedure like the fourth-order Runge–Kutta starter presented here has a potential impact on the overall performance of an simulator.

The flexibility in selecting the order of the restarter as well as doing error control of the restarter is the topic of future research.

	Classic starter	Runge–Kutta starter
# steps	455	129
# function evals	1027	428
# event function evals	919	538
# events	38	37

Table 1: Run time statistics for the bouncing ball example with absolute and relative tolerance set to 10^{-8}.

References

[1] Johan Åkesson, Magnus Gäfvert, and Hubertus Tummescheit. JModelica—an open source platform for optimization of modelica models. In *Proceedings of MATHMOD 2009 - 6th Vienna International Conference on Mathematical Modelling*, Vienna, Austria, February 2009. TU Wien.

[2] Christian Andersson. Assimulo: a new Python based class for simulation of complex hybrid DAEs and its integration in JModelica.org. Master's thesis, Lund University, 2011.

[3] Christian Andersson, Johan Åkesson, Claus Führer, and Magnus Gäfvert. Import and export of functional mock-up units in JModelica.org. In *8th International Modelica Conference 2011*, Dresden, Germany, March 2011.

[4] Torsten Blochwitz, M. Otter, M. Arnold, C. Bausch, C. Clauß, H. Elmqvist, A. Junghanns, J. Mauss, M. Monteiro, T. Neidhold, et al. The functional mockup interface for tool independent exchange of simulation models. In *Modelica'2011 Conference, March*, pages 20–22, 2011.

[5] C. W. Gear. Runge–Kutta starters for multistep methods. *ACM Trans. Math. Softw.*, 6(3):263–279, September 1980.

[6] Anthony Ralston. Runge–Kutta methods with minimum error bounds. *Mathematics of computation*, 16(80):431–437, 1962.

[7] Hans J. Stetter. Asymptotic expansions for the error of discretization algorithms for non-linear functional equations. *Numerische Mathematik*, 7(1):18–31, 1965.

[8] Reinhold von Schwerin and Hans Georg Bock. A Runge–Kutta starter for a multistep method for differential-algebraic systems with discontinuous effects. *Applied numerical mathematics*, 18(1):337–350, 1995.