A Family of Runge-Kutta Restarters for Multistep Methods

Fatemeh Mohammadi    Carmen Arévalo    Claus Führer

Lund University
Centre for Mathematical Sciences
Numerical Analysis

August, 2015

Restarting a multistep method

- Classical self-starting multistep method
- Apply several Runge–Kutta steps
- A single Runge–Kutta step
Goals of construction

For starting a $k$-step multistep method, the explicit Runge–Kutta starter is required to have

- Minimal number of internal stages.
- At least $k$-stage values of order $k$.
- Error estimation.
- Equidistant stages for starting the multistep method.
Runge–Kutta starter after a discontinuity

$H$: Runge–Kutta step size
$h$: Multistep step size

$t_0$, $t_1$, $t_2$, $t_3$, $\ldots$, $t_{k-1}$

$t_0 + H$
Explicit Runge–Kutta method

For initial value problem

\[ y' = f(t, y), \quad y(0) = y_0, \]

an \( s \)-stage explicit Runge–Kutta method is,

\[
g_i := y_0 + H \sum_{j=1}^{i} a_{ij} k_j, \quad i = 1, \ldots, s, \]

\[
k_i := f(t_0 + c_i H, g_i), \]

\[
y_1 := y_0 + H \sum_{j=1}^{s} b_j k_j, \]
Internal order conditions

\(k = 4\)

\[\sum_j a_{ij}c_j = \frac{c_i^2}{2},\]

\[\sum_{jk} a_{ik}a_{kj}c_j = \frac{c_i^3}{6},\]

\[\sum_{jk} a_{ij}c_ja_{jk}c_k = \frac{c_i^4}{8},\]

\[\sum_{jkl} a_{ij}a_{jk}a_{kl}c_l = \frac{c_i^4}{24},\]

\[\sum_j a_{ij}c_j^2 = \frac{c_i^3}{3},\]

\[\sum_j a_{ij}c_j^3 = \frac{c_i^4}{4},\]

\[\sum_{jk} a_{ij}a_{jk}c_k^2 = \frac{c_i^4}{12},\]

For internal stage \(i\).
An embedded third order Runge–Kutta starter

\[
\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{6} & & & \\
\frac{3}{4} & 0 & \frac{3}{4} & & \\
1 & \frac{2}{9} & \frac{1}{3} & \frac{4}{9} & \\
\frac{1}{2} & \frac{17}{72} & \frac{1}{6} & \frac{2}{9} & -\frac{1}{8} \\
1 & \frac{1}{6} & 0 & 0 & \frac{1}{6} & \frac{2}{3} \\
\end{array}
\]
An embedded fourth order Runge–Kutta starter

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1/6</th>
<th>1/6</th>
<th>1/3</th>
<th>1/3</th>
<th>1/3</th>
<th>1</th>
<th>2/3</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25/10</td>
<td>10/45</td>
<td>-29/1062</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-3</td>
<td>-8/45</td>
<td>83/531</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-3</td>
<td>-8/45</td>
<td>83/531</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>225/100</td>
<td>-4/45</td>
<td>2/531</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>225/100</td>
<td>13/15</td>
<td>56/531</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>225/100</td>
<td>1/45</td>
<td>251/531</td>
</tr>
</tbody>
</table>
Order Plot

\[ y' = -y \]

- stage 5
- stage 6
- stage 7
- stage 8

Error vs. Step-size H
Order Plot

\[ y' = -y \]

- stage 7
- stage 10
- stage 12
- stage 13
- stage 14

Error vs. Step-size \( H \)
Implementation

Tests are conducted in Assimulo (Python):
(default simulation environment of JModelica.org)
Integrator: LSODAR (ODEPACK, FORTRAN)
The bouncing ball test example

the example of a bouncing ball with linear damping $d = 0.1$,\[ y'_1 = y_2 \]
\[ y'_2 = -dy_2 + 9.81 \]

At the bouncing event the velocity $y_2(t^-)$ is altered to
\[ y_2(t^+) = -cy_2(t^-). \]
Bouncing ball: simulation results

(damping: $d = 0.1$, coefficient of restitution $c = 0.88$)
Bouncing ball: Step size and order history

Classical restart

Runge–Kutta restart
Conclusion and future work

- RKrestarter is a promising alternative in case of frequent events with small or no changes in the dynamics.
- Various restarting orders have been derived. Goal: No order drop after restarting.
- Step size control for the starter.

How to continue....

- Various strategies for choosing the initial step sizes have to be tested.
- Tests with complex physical models.
Thank you!