Goethe-Center for Scientific Computing (G-CSC)
Goethe-Universität Frankfurt am Main

Modelling and Simulation I

(Practical SIM1, WS 2016/17)

M. Hoffer, Dr. S. Reiter, Dr. A. Vogel

Exercise sheet 5 (Due: Mo., 12.12.2016, 10h)

Until now, we only used constant step sizes h for our explicit and implicit ODE solvers. Although it is relatively easy to find acceptable step sizes for many problems that have been discussed so far, this is not always the case. Especially, if solutions vary strongly in parts of the time interval while they are almost constant in others it is impossible to find a suitable, uniformly distributed step size. Therefore, we will develop an adaptive step size control which ensures that the local discretization error of the solver is bounded by a user-defined, problem-dependent tolerance, usually referred to as TOL, i.e. we implement the variant (V1) for the local error control presented in the lecture. As error estimation strategy we use method (S2) based on the idea to use a numerical method of order $p+1$ to estimate the error of a method of order p which ensures that the estimator is asymptotically correct for $h \to 0$. For the algorithmic treatment in the control loop we employ (Algo. 1) presented in the lecture, i.e. halving the time step size if required.

Exercise 1 (7P)
We are going to realize the adaptive step size control using the so called Dormand-Prince 4/5 method (DOPRI). This methods is of embedded Runge-Kutta type, i.e., it consists of two Runge-Kutta methods that share the same function evaluations per step, such that the error estimation can be implemented highly efficiently. Remember that explicit, embedded Runge-Kutta methods are conveniently denoted in a Butcher tableau

$$
\begin{array}{c|ccccc}
0 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & & \ddots & & \\
c_L & a_{L1} & a_{L2} & \cdots & a_{L,L-1} & 0 \\
\hline
 & b_1^{(p)} & b_2^{(p)} & \cdots & b_{L-1}^{(p)} & b_L^{(p)} \\
\hline
 & b_1^{(p+1)} & b_2^{(p+1)} & \cdots & b_{L-1}^{(p)} & b_L^{(p+1)}
\end{array}
$$

and the computation for the next timepoint is carried out for both choices $\mathbf{b}^{(p)}$ and $\mathbf{b}^{(p+1)}$ using

$$t_{k-1,l} := t_{k-1} + c_l h_k, \quad \mathbf{k}_l := \mathbf{f}(t_{k-1,l}, \mathbf{y}_{k-1,l}), \qquad l = 1, \dots, L$$
$$\mathbf{y}_{k-1,l} = \mathbf{y}_{k-1} + h_k \sum_{s=1}^{l-1} a_{ls} \mathbf{k}_s, \qquad \mathbf{y}_k = \mathbf{y}_{k-1} + h_k \sum_{l=1}^{L} b_l \, \mathbf{k}_l,$$

where the single stages $l = 1, ..., L$ can be explicitly computed via

$$\begin{aligned} \mathbf{k}_1 &= f(t_{k-1}, \mathbf{y}_{k-1}), \\ \mathbf{k}_2 &= f(t_{k-1} + c_2 h_k, \mathbf{y}_{k-1} + h_k(a_{21}\mathbf{k}_1)), \\ &\vdots \\ \mathbf{k}_L &= f(t_{k-1} + c_s h_k, \mathbf{y}_{k-1} + h_k(a_{L1}\mathbf{k}_{L1} + a_{L2}\mathbf{k}_2 + \ldots + a_{L,L-1}\mathbf{k}_{L-1})) \end{aligned}$$

The Butcher tableau for the DOPRI method is given by

$$\begin{array}{c|ccccccc}
0 & & & & & & & \\
\frac{1}{5} & \frac{1}{5} & & & & & & \\
\frac{3}{10} & \frac{3}{40} & \frac{9}{40} & & & & & \\
\frac{4}{5} & \frac{44}{45} & \frac{-56}{15} & \frac{32}{9} & & & & \\
\frac{8}{9} & \frac{19372}{6561} & \frac{-25360}{2187} & \frac{64448}{6561} & \frac{-212}{729} & & & \\
1 & \frac{9017}{3168} & \frac{-355}{33} & \frac{46732}{5247} & \frac{49}{176} & \frac{-5103}{18656} & & \\
1 & \frac{35}{384} & 0 & \frac{500}{1113} & \frac{125}{192} & \frac{-2187}{6784} & \frac{11}{84} & \\
\hline
 & \frac{35}{384} & 0 & \frac{500}{1113} & \frac{125}{192} & \frac{-2187}{6784} & \frac{11}{84} & 0 \\
\hline
 & \frac{5179}{57600} & 0 & \frac{7571}{16695} & \frac{393}{640} & \frac{-92097}{339200} & \frac{187}{2100} & \frac{1}{40}
\end{array}$$

In order to control the step size, the local error is computed for a test step size h_{test} and it is checked if the error $h_{\text{test}}\|\tilde{\tau}_k(h_{\text{test}})\| \leq TOL$ is below a prescribed tolerance. The test step size is halved $h_{\text{test}} := \frac{1}{2}h_{\text{test}}$ until we can accept the computed solution (or the computation is aborted if a minimum step size $h_{\min}$ is reached). The estimation for the local error can be computed using the difference of the solutions of the two Runge-Kutta methods of order p and $p+1$, namely

$$\begin{aligned} h_{\text{test}}\tilde{\tau}_k(h_{\text{test}}) &= h_{\text{test}} \left\{ \Phi^{(p+1)}(\mathbf{y}_{k-1}) - \Phi^{(p)}(\mathbf{y}_{k-1}) \right\} \\ &= h_{\text{test}} \sum_{l=1}^{L} (b_l^{(p)} - b_l^{(p+1)}) \, \mathbf{k}_l. \end{aligned}$$

For the norm $\|\cdot\| : \mathbb{R}^d \to \mathbb{R}$ we use the euclidean norm.
Once a step size $h_k := h_{\text{test}}$ has been accepted with an error $\tilde{\tau}_k \in \mathbb{R}^d$ below the tolerance, $\|\tilde{\tau}\|_k \leq TOL$, the test step size for the next timestep is predicted using the optimal test step size

$$h_{\text{test}} := h_{k,opt} = \min(\max(0.9 \cdot \sqrt[p+1]{\frac{TOL}{h_k \|\tilde{\tau}_k\|}} \cdot h_k, h_{\min}), h_{\max}).$$

Hints:

- The `EmbeddedRKTemplate` available on the webpage shows how to return multiple values from a single method.
- Introduce a `stop()` method that allows to stop the solver in case the simulation runs with wrong parameters and to prevent infinite loops introduced by programming errors.
- To save solution trajectories, use the `VectorTrajectoryToFile.groovy` component that can be downloaded from the webpage.

Tasks/Questions:

(a) Analogously to the previously implemented one-step methods, implement the Dormand-Prince 4/5 scheme with adaptive time step control as specified above. In addition to the solution, return the local error and the step size. See the `EmbeddedRKTemplate` that has been published on how to accomplish that.

Exercise 2 (3P + 2P)
To test the implementation of your Dormand-Prince 4/5 scheme, we will solve the following problems:

(1)

$$\begin{cases} \text{Find } u : [t_0, t_n] \mapsto \mathbb{R}, \text{ such that} \\ u'(t) = -2ctu^2, c = 100 \\ u(0) = 0.00039968 \\ t_0 = -5, t_n = 5 \end{cases}$$

(2) Let us revisit the Lotka-Volterra model from `Sheet03, Exercise 2c`. Instead of an inhibition factor of 0.001 use 0.003. The initial values of $N_0(t_0) = 1000$ and $P_0(t_0) = 100$ shall be used. For the time step control use $TOL =$ `1e-8`, $h_{\text{min}} =$ `1e-4` and $h_{\text{max}} =$ `1.0`.

Tasks/Questions:

(a) Plot the solution, the local error and the step size for both, 1) and 2). For 1), simulate and plot your results with $TOL =$ `1e-8`, $TOL =$ `1e-10` and $TOL =$ `1e-12`.

(b) Do you see correlations between the solution and the step sizes chosen by the time step control? Discuss your results.

Exercise 3 (4P + 2P + 2P)

In `Sheet03, Exercise 3` we solved the two-body system, consisting of the earth and an artificial satellite. Now, with far more capable numerical solvers, we can finally solve a much more realistic N-Body system, such as our own solar system. Analogous to `Sheet03, Exercise 3`, the force between two bodies $body_i$ and $body_j$ is defined as:

$$F_{ij} = G m_i m_j \frac{\mathbf{r}_j - \mathbf{r}_i}{\|\mathbf{r}_j - \mathbf{r}_i\|^3} \tag{1}$$

where G is the gravitational constant, $\mathbf{r}_i$ is defined as the location of $body_i$ and $\|\mathbf{r}_j - \mathbf{r}_i\|$ is the magnitude of the distance between $\mathbf{r}_i$ and $\mathbf{r}_j$. The system of ODEs that describes the behavior of n bodies in space is defined as follows:

$$\frac{d^2\mathbf{r}_1}{dt^2} = \sum_{\substack{j=1 \\ j\neq 1}}^{n} Gm_j \frac{\mathbf{r}_j - \mathbf{r}_1}{\|\mathbf{r}_j - \mathbf{r}_1\|^3}$$

$$\vdots$$

$$\frac{d^2\mathbf{r}_i}{dt^2} = \sum_{\substack{j=1 \\ j\neq i}}^{n} Gm_j \frac{\mathbf{r}_j - \mathbf{r}_i}{\|\mathbf{r}_j - \mathbf{r}_i\|^3}$$

$$\vdots$$

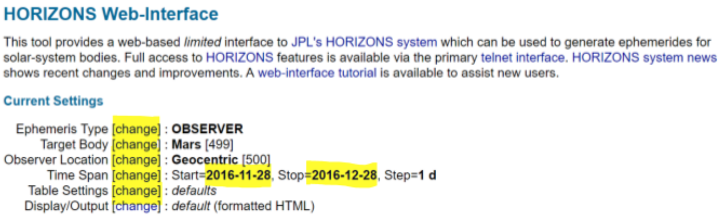
$$\frac{d^2\mathbf{r}_n}{dt^2} = \sum_{\substack{j=1 \\ j\neq n}}^{n} Gm_j \frac{\mathbf{r}_j - \mathbf{r}_n}{\|\mathbf{r}_j - \mathbf{r}_n\|^3}$$

Our simulation will start at 13.10.2015 and ends exactly 31558149 seconds later which is the duration of one earth orbit. The following bodies shall be simulated:

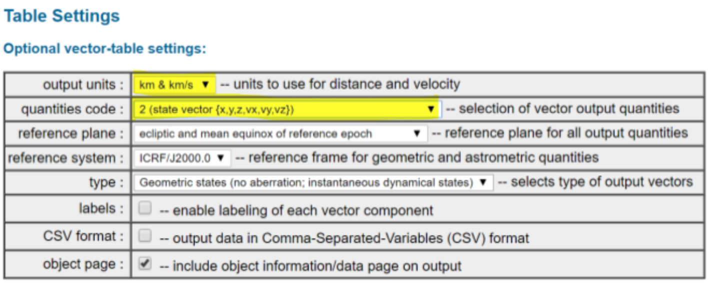
1. Sun
2. Mecrury
3. Venus
4. Earth
5. Moon (Luna)
6. Mars
7. Jupiter
8. Saturn
9. Uranus
10. Neptune
11. Comet Halley
12. Eros

13. Churiomov Gerasimenko (67P)

To acquire the initial values (mass, **r** and **v**), visit the NASA HORIZONS webpage (http://ssd.jpl.nasa.gov/horizons.cgi). To get data in the required format, change the following settings that are marked in yellow:



Necessary changes in table settings:



The necessary changes are listed below:

Ephemeris:

- Ephemeris Type: Vector

Time Span:

- Time Span: 2015-10-13 to 2016-10-13 (yyyy-mm-dd)

Table Settings:

- Output Units: km & km/s
- Quantities Code: 2 (state vector {x,y,z,vx,vy,vz})

After adjusting the settings, click "Generate Ephemeris". The result is shown as html output, which will look like this:

Results

```
*******************************************************************************
Ephemeris / WWW_USER Mon Nov 28 02:16:20 2016 Pasadena, USA      / Horizons
*******************************************************************************
Target body name: Mars (499)                      {source: mar097}
Center body name: Solar System Barycenter (0)     {source: DE431mx}
Center-site name: BODY CENTER
*******************************************************************************
Start time      : A.D. 2016-Nov-28 00:00:00.0000 TDB
Stop  time      : A.D. 2017-Dec-28 00:00:00.0000 TDB
Step-size       : 1440 minutes
*******************************************************************************
Center geodetic : 0.00000000,0.00000000,0.0000000 {E-lon(deg),Lat(deg),Alt(km)}
Center cylindric: 0.00000000,0.00000000,0.0000000 {E-lon(deg),Dxy(km),Dz(km)}
Center radii    : (undefined)
Output units    : KM-S
Output type     : GEOMETRIC cartesian states
Output format   : 2 (position and velocity)
Reference frame : ICRF/J2000.0
Coordinate systm: Ecliptic and Mean Equinox of Reference Epoch
*******************************************************************************
JDTDB
   X     Y     Z
   VX    VY    VZ
*******************************************************************************
$$SOE
2457720.500000000 = A.D. 2016-Nov-28 00:00:00.0000 TDB
   2.071675464648208E+08 -1.811620232212564E+07 -5.486466737662795E+06
   3.103630032791307E+00  2.621635586547728E+01  4.729487716331313E-01
2457721.500000000 = A.D. 2016-Nov-29 00:00:00.0000 TDB
   2.074242509695789E+08 -1.585011708426737E+07 -5.445302188771992E+06
   2.838589615065836E+00  2.623883361410372E+01  4.799244169017030E-01
2457722.500000000 = A.D. 2016-Nov-30 00:00:00.0000 TDB
   2.076580528109857E+08 -1.358221604507309E+07 -5.403537511979058E+06
   2.573482027676767E+00  2.625838876427120E+01  4.868404768498937E-01
```

The object properties, such as mass, can be found in the “Object Data Page”:

Object Data Page

```
Revised: Sep 28, 2012                Mars                             499 / 4

GEOPHYSICAL DATA (updated 2009-May-26):
 Mean radius (km)      = 3389.9(2+-4)    Density (g cm^-3)      =  3.933(5+-4)
 Mass (10^23 kg )      =    6.4185       Flattening, f          =  1/154.409
 Volume (x10^10 km^3)  =   16.318        Semi-major axis        =  3397+-4
 Sidereal rot. period  =   24.622962 hr  Rot. Rate (x10^5 s)    =  7.088218
 Mean solar day        =    1.0274907 d  Polar gravity ms^-2    =  3.758
 Mom. of Inertia       =    0.366        Equ. gravity  ms^-2    =  3.71
 Core radius (km)      =  ~1700          Potential Love # k2    =  0.153 +-.017

 Grav spectral fact u  =   14 (x10^5)    Topo. spectral fact t = 96 (x10^5)
 Fig. offset (Rcf-Rcm) = 2.50+-0.07 km   Offset (lat./long.)   = 62d / 88d
 GM (km^3 s^-2)        = 42828.3         Equatorial Radius, Re = 3394.0 km
 GM 1-sigma (km^3 s^-2)= +- 0.1          Mass ratio (Sun/Mars) = 3098708+-9

 Atmos. pressure (bar) =    0.0056       Max. angular diam.     =  17.9"
 Mean Temperature (K)  =  210            Visual mag. V(1,0)     =  -1.52
 Geometric albedo      =    0.150        Obliquity to orbit     =  25.19 deg
 Mean sidereal orb per =    1.88081578 y Orbit vel.  km/s       =  24.1309
 Mean sidereal orb per =  686.98 d       Escape vel. km/s       =   5.027
 Hill's sphere rad. Rp =  319.8          Mag. mom (gauss Rp^3) = < 1x10^-4
```

Hints:

- To allow an intuitive interpretation of the results, a Java program called `VRL-Solar-System-Viewer` has been published on the webpage. It allows the visulization of your simulated solar system. The program requires a recent Java installation (Version $> 1.8.70$). To start the program, either double-click on the .jar-file or run the program via `java -jar VRL-Solar-System-Viewer.jar`. The source code is also available on GitHub.

- Stick to the order of the planetary objects as specified above. Otherwise, results cannot be interpreted by the `VRL-Solar-System-Viewer`. Use the time slider at the bottom of the program window to visualize planetary movements.

Tasks/Questions:

(a) Setup the right-hand side for the N-Body system with the 13 bodies listed above and simulate the time interval $[t_0 = \texttt{0}, t_n = \texttt{3.1558149E7 sec}]$ with TOL = `1e-4`, h_{min} = `0.01` and h_{max} = `1e6`. Use the "VectorTrajectoryToFile" component to save your results as text file. To visualize your results, use the `VRL-Solar-System-Viewer.jar`. Compare the location of the earth at t_0 and t_n. Since the specified time-span is exactly one earth orbit, this is a good estimation for the global error of your simulation.

(b) Change the simulation interval to $[t_0 = \texttt{0}, t_n = \texttt{1 year}]$ and compare your results with the numbers published by NASA. Specify the global error.

(c) Simulate the time intervals $[t_0 = \texttt{0}, t_n = \texttt{10 years}]$ and $[t_0 = \texttt{0}, t_n = \texttt{100 years}]$. Please note that on older machines the computation can take a while. It is adviced to filter the solution trajectory, e.g., only add every n-th value to the solution. For the last interval, it is allowed to increase the TOL value to `1e-2` if your computer is too slow. If you do so, specify the TOL value that has been used for the computation.

Exercise 4 (10 Bonus Points)
Write a VRL component that reads the Ephemeris data from the HORIZONS web interface. Plot the difference between the computed trajectory by your ODE solver for planet earth and the trajectories published by NASA. The result is a good estimate for the global error.

Caution: Please implement all programs with VRL-Studio. Programs written in other languages and/or environments will not be accepted.

Remark: Send your implemented source code (.vrlp file). Append the pdfs produced with the VectorTrajectoryPlotter and the trajectory files produced with the `VectorTrajectoryToFile` component to the email.
Send your solution to `practical.sim1@gcsc.uni-frankfurt.de` until Monday, 12.12.2016, 10h.