xx

Introduction

[next] [tail] [up]

1 Introduction

1.1 What it does

Runge is an Interactive Solver for Systems of Ordinary Differential Equations. It solves initial value problem (aka Cauchy problem) defined as the following: for a given system of ordinary differential equations

.

$$\dot{x} = F(t, x), \quad (t, x) \in \mathbb{R}^{n+1}$$

and given initial values

.

$$x(t_0) = x_0$$

find solution

.

$$x(t_k) \in \mathbb{R}^n$$

at a given point of “time” i.e. for a given value t_k of *independent variable* t . Actually Runge produces solutions set

.

$$x(t_0), x(t_1), x(t_2), \ldots, x(t_k)$$

where k is the number of steps taken. This allows to build *trajectories* of solutions.

1.2 Why Runge?

It's fast. It utilizes BLAS and LAPACK FORTRAN libraries optimized for modern multi-core processors.

It's interactive. It allows you to start a solution by mouse click on a plane.

It's precise. It uses Runge Rule to adjust step length to satisfy required precision on each step.

It's effective. When it needs to compute derivatives (Jacobian matrix, for example) it does that *analytically*, i.e. without using numerical methods.

It's portable. It works on Windows and Linux (32 and 64 bit versions) and Mac OSX (64 bit only).

It's open. It allows you to implement and embed your own algorithms (aka "solvers").

It's easy to use. It allows to export results to MS Excel and MATLAB.

It's free. It's distributed under Boost Software License.

1.3 System Types and Solvers

Runge comes with pre-installed solvers optimized for solving differential equations of different types:

Type 1. Generic non-autonomous system

.

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

Type 2. Generic autonomous system (it's a subset of Type 1, i.e. you can use both types 1 and 2 for autonomous sytems — sometimes it makes sense for choosing appropriate solver)

.

$$\dot{x} = F(x)$$

Type 3. Pseudo-linear system (here it's assumed that $\phi(x)$ is relatively small compared to matrix $A(t)$)

.

$$\dot{x} = A(t)x + \phi(x)$$

Type 4. Pseudo-linear system with constant matrix B

.

$$\dot{x} = Bx + f(t, x)$$

The solvers (algorithms) coming in standard package are:

Runge-Kutta process modification developed by R. England. Fast and precise method of fifth order suitable for solving systems of Type 1. See [1].

Exponential method modification developed by J.D. Lawson. It's recommended for linear and quasi-linear systems of Type 1,3 and 4 (including stiff ones). The method is A-stable for linear systems, i.e.

residuals do not depend on step length. See [2].

Implicit process developed by H.H. Rosenbrock. It's recommended for non-linear systems of Type 1 and 2 (including stiff ones). See [3].

1.4 Expressions and Functions

The following functions and operators are supported for programming the systems mentioned above.

11	`+ - * / ^`	11	arithmetic operators: add, subtract, multiply, divide, power	
11	`exp(x)`	11	e^x	
11	`sqrt(x)`	11	$\sqrt{x}$	
11	`log(x)`	11	natural logarithm of x	
11	`log10(x)`	11	common (base 10) logarithm of x	
11	`sin(x)`	11	sine of x	
11	`cos(x)`	11	cosine of x	
11	`tan(x)`	11	tangent of x	
11	`asin(x)`	11	arc sine of x	
11	`acos(x)`	11	arc cosine of x	
11	`atan(x)`	11	arc tangent of x	
11	`sinh(x)`	11	hyperbolic sine of x	
11	`cosh(x)`	11	hyperbolic cosine of x	
11	`tanh(x)`	11	hyperbolic tangent of x	
11	`sinint(x)`	11	sine integral of x .	$\int_0^x \frac{\sin}{t}$
11	`cosint(x)`	11	cosine integral of x .	$-\int_x^\infty \frac{co}{}$
11	`sign(x)`	11	sign of x .	$\begin{cases} -1 \text{ if} \\ 0 \text{ if} \\ 1 \text{ if} \end{cases}$
11	`abs(x)`	11	$\lvert x\rvert$	
11	`iif(x,expr1,expr2)`	11	immediate if .	$\begin{cases} \mathrm{expr1} \text{ i} \\ \mathrm{expr2} \text{ i} \end{cases}$
11	`sat(x,y)`	11	satellite function of x and y .	$\begin{cases} 1 \text{ if } x > \lvert \\ 0 \text{ if } -\lvert y \\ -1 \text{ if } x < - \end{cases}$
11	`i`	11	1 (one)	
11		11	0 (empty field means zero)	
11				

Examples: `2*sin(t-1)+cos(t)-x^2`, `sqrt(abs(x))`, `iif(t,sin(x),cos(x))` etc.

[next] [front] [up]

Quick Tour

[next] [prev] [prev-tail] [tail] [up]

2 Quick Tour

When started, Runge looks like this:

Main Window

There are 4 areas here:

1. Menu and toolbar.
2. Left pane with four selectable sub-panes.
3. Solutions computed.
4. Right pane with equations you enter and solutions' values you compute.

If you click Start button Start Button at this moment it would solve $\dot{x} = 0,\ t \in [0,1]$ problem. That's because empty field for $F(t,x)$ system on the right pane is equivalent to 0 and default range for independent variable is $[0,1]$. Click Start button Start Button or choose "Start" item in "Run" menu

Run Menu

Select "Solutions" subpane and click on first solution to see results on the right pane:

Solutions

The columns you see out there are: values of independent variables $t_i, i = 0, 1, \ldots$, solution values $x(t_i), i = 0, 1, \ldots$ and *recommended* step values. You can also delete this solution or export it to MS Excel comma-separated values file or to MATLAB script using the following buttons:

Click 2D Draw button 2D Draw Button to see solution graph:

2D Draw Window

[next] [prev] [prev-tail] [front] [up]

Main Menu

[next] [prev] [prev-tail] [tail] [up]

3 Main Menu

3.1 File Menu

Here you can:

- Start new file (new system to solve) by using CtrlxN (CmdxN) combination.
- Open existing Runge file by using CtrlxO (CmdxO) combination.
- Save current Runge file by using CtrlxS (CmdxS) combination.
- Save current Runge file with different name.

Runge file stores the system you entered and all solutions and their properties.

3.2 Options Menu

Windows and Linux: Mac:

Here you can:

- Choose Runge configuration file. On Mac use main "Runge" menu, item "Preferences". Use it only if you want to add your own algorithms (aka “solvers”). Keep default Runge*.xml configuration otherwise.
- Choose font for entering expressions.

Choose language (restart required). Currently English and Russian are supported only.

3.3 Run Menu

Here you can:

Start solving your system.

Puse it if it's started.

Resume it if it's paused.

Stop it if it's started.

3.4 View Menu

Call it to invoke 2D Drawer window.

3.5 Help Menu

Call it to open Runge Manual.

[next] [prev] [prev-tail] [front] [up]

Programming and Solving

[next] [prev] [prev-tail] [tail] [up]

4 Programming and Solving

Before solving your system of ordinary differential equations you need to program it in Runge. This section describes the way you can do that.

4.1 System

1. Choose type of your system:

   System Type

2. Choose dimension:

   Dimension

3. Choose independent variable. It should be valid identifier (i.e. a string beginning with letter):

   Independent Variable

4. Enter the system you'd like to solve. First column is for dependent variables (valid identifiers, i.e. strings beginning with letter), second one is for equations. Equations should be valid expressions containing arithmetic operators and elementary functions. You can also use macros (like macro "p" shown below) for repeating expressions:

   System

   The following system is entered at this picture:

.

$$\begin{cases} \dot{x} = y + xp \\ \dot{y} = -x + yp \end{cases}$$

4.2 Parameters

1. Choose solver. Not every solver works for a given system type:

   Solver

2. Set start parameters, values and macros:

   Parameters

   Here you can set:

   Statr t That's the initial value of the independent variable.

   End t That's the end value of the independent variable — last solution will be computed at this point.

   H Initial step. Real step will be computed according to precision required.

   Hmin Minimum step allowed. If precision can't be reached even for this step, solver fails.

Hmax Maximum step allowed.

Eps Absolute precision required on each step.

P Relative precision measure. If some component's absolute value becomes greater than P, it gets normalized when precision compared to Eps.

Macros List with macros. Here we have

.

$$p = 2 - x^2 - y^2$$

therefore the final system turns to be

.

$$\begin{cases} \dot{x} = y + x(2 - x^2 - y^2) \\ \dot{y} = -x + y(2 - x^2 - y^2) \end{cases}$$

Dep. var. - Start Value Initial values for each dependent variable. In this particular case we have:

.

$$x(t_0) = 2,\ y(t_0) = 2,$$

where $t_0 = 0$, see Start t above.

4.3 Solution

1. Click Start button Start Button to start solving the system. For big systems or long time ranges you might see progress bar

   Progress Bar

   Here you can stop, pause and resume solving process by using appropriate buttons.

2. Explore solution

   Every solution has an id starting from zero.

3. Explore solution graphically (here you can select different variables for X and Y axes):

4.4 Export

Use Export feature to export solution to other programs like MS Excel or MATLAB

Excel should read it like this:

MATLAB should read it like this:

And after issuing a command like

```
>> plot(solution0(:,2), solution0(:,3));
```

it should render a plot:

[next] [prev] [prev-tail] [front] [up]

2D Graphics

[next] [prev] [prev-tail] [tail] [up]

5 2D Graphics

5.1 2D Drawer window

Click to bring 2D Drawer window which can also be used to start solving the system you entered interactively. You can open few windows by clicking this button multiple times.

1. Toolbar (each button function explained below).

2. X axis variable. This is independent variable by default.

3. Y axis variable.

4. Graphics plane

5. Y axis with marks. Mark step is 1 coordinate unit (adjustable), marks can be turned off.

6. X axis with marks. Mark step is 1 coordinate unit (adjustable), marks can be turned off.

7. Solution. It has a holder at the beginning looking like a small circle. Click on it to select a solution (color gets changed, *solution also gets selected in main window*). Click somewhere else to unselect.

8. Current mouse coordinates (this is handy to start new solutions).

9. Resizing handle. You can resize this window as much as you want.

5.2 Interactively starting solution from given point

Click “Run froim point” button to enter the mode (mouse cursor will be changed accordingly). Click on a plain where you want to run new solution from. You should get something like this:

Note that new solution gets added to the main window list and it’s also drawn on every other drawer open.

You can quickly create a set of solutions by clicking at different points:

5.3 Selecting solutions

Every solution has a first point where independent variable $t = t_0$. It’s marked by a small circle called “handle”:

Click it to select:

Note how it gets selected in main window (it works in both ways — you can select it in the list as well). You can explore, export and remove selected solution.

Selected solution is marked by different color for each second step. You choose the color using the color menu (since version 1.1, see below). Sometimes it’s handy to observe how steps change:

5.4 Zoom and Pan

Click Zoom button to bring zoom menu

Select Zoom In or press Ctrlx+ combination to increase the drawing in 2:1 ratio:

Zoom in level is virtually endless. Use “Zoom Out” to unzoom (Ctrlx–). Use “Zoom to rectangle” (CtrlxZ) to select a rectangle by mouse. This rectangle would be scaled to. Use “Fit all” (CtrlxF) to show everything.

Click Pan button to enter Panning mode (mouse cursor changes). You can drag your drawing for exploring its different parts. Panning is available only when there is something out of window borders.

5.5 Color

Click Palette button to bring the color menu:

By selecting first item you can:

- change color of selected solution (if one is selected)
- change *current* color for all subsequet solutions otherwise

Please note that said above is also valid for all odd steps.

By selecting second item you can:

- change color of each even step of selected solution (if one is selected)
- change *current* color of each even step for all subsequet solutions otherwise

For example:

5.6 Settings

Click Drawer Settings button to bring settings window:

Here you can:

1. Turn on/off axes. If axes are turned on you can:

   Click Adjust button to adjust axes lengths automatically according to current drawing.

   Click Color... button to set axes color.

   Edit semi-axes lengths manually (in coordinate units).

2. Turn on/off line widths for axes and solutions. When turned off, the lines are drawn without widths at any scale level (“hair” lines). Sometimes for better quality of exported images you’d need to set those widths manually.

3. Turn on/off axes marks. If axes marks are turned on you can:

   Change X axis step (default is 1).

   Change Y axis step (default is 1).

   Set length of marks *in pixels*.

4. Change diameter of solution handles *in pixels*.

5. Make the drawer to rescale each time new solution added. This mode (default) helps to keep seeing everything.

5.7 Export to File

Click Export to File button to open Export window:

Here you can:

1. Choose image type:

   This list is platform-dependent.

2. Set image width and height in pixels.

3. Set JPEG quality if it's selected.

4. Make the drawer to open newly created image using OS default viewer:

   Here is 300x300 result sample is shown:

5.8 Print to Document/Paper

Click Print button to open Print to Document/Paper window:

1. Choose where to print: to PDF, PS or paper.

2. Choose document/paper size:

   This list is platform-dependent.

3. Set different lines widths and handle diameter im millimeters.

4. Make the drawer to open newly created document using OS default viewer:

5.9 Help

Click Help button to open Runge Manual.

[next] [prev] [prev-tail] [front] [up]

3D Graphics

[next] [prev] [prev-tail] [tail] [up]

6 3D Graphics

6.1 3D Drawer window

Click to bring 2D Drawer window which can also be used to start solving the system you entered interactively. You can open few windows by clicking this button multiple times.

[next] [prev] [prev-tail] [front] [up]

References

[prev] [prev-tail] [tail] [up]

References

[1]xxx*R. England.* Error Estimates for Runge-Kutta Type Solutions to Systems of Ordinary Differential Equations // Research and Development Department, Pressed Steel Fisher Ltd., Cowley, Oxford, UK. October 1968

[2]xxx*Lawson J.D.* Generalized Runge-Kutta processes for stable systems with large Lipschitz constants // SIAM Journal on Numerical Analysis, 1967, V. 4, No 3.

[3]xxx*H.H. Rosenbrock* Some general implicit processes for the numerical solution of differential equations Comput. J., 5 (1963) pp.329-330.

[prev] [prev-tail] [front] [up]