ELEC2146

Electrical Engineering Modelling and Simulation

Runge-Kutta Methods for Differential Equations

Dr Ray Eaton

S2, 2016

Lecture notes written by Dr Julien Epps

Overview

- Motivation
- Second-order Runge-Kutta
- $3^{rd}/4^{th}$ order Runge-Kutta
- Truncation errors
- Selection of step size T

Motivation

- Consider

$$\frac{dx(t)}{dt} + a_0 x(t) = f\left(x(t), u(t)\right)$$

- What if:
  - Forcing function takes different forms ?
  - Forcing function is unknown ?
  - I am having déjà vu ?

  - Analytical solutions not practical
    - Euler's method OK
    - Higher order methods: Starting to need analytical derivatives again . . .

Motivation

- Recall:
  - Higher-order Taylor series methods (general):

$$x(t_{i+1}) = x(t_i) + Tf(t_i, x(t_i)) + \frac{T^2}{2!} f'(t_i, x(t_i)) + \frac{T^3}{3!} f''(t_i, x(t_i)) + \ldots$$
$$\frac{T^n}{n!} f^{(n-1)}(t_i, x(t_i)) + \frac{T^{n+1}}{(n+1)!} x^{(n+1)}(\xi)$$

- Want:
  - Very good accuracy (= small truncation error)
- Do not want:
  - To derive expressions for derivatives

Runge-Kutta

- No derivatives
- All methods take the form

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

- $\varphi(t_i, x_i, T)$:
  - The increment function
  - Average slope over $[t_i, t_{i+1}]$
- One-step methods
  - Do not use values from $t < t_i$
- $f(t, x(t))$ evaluated at multiple points
  - n points, where n is the order of the method

Runge-Kutta

- Euler's method
  - Gradient estimated only at beginning of interval
  - Error is large
- Alternative
  - Compute value of *x* at fractional steps
  - Estimate gradient at each fractional step
  - Compute weighted average of gradients
- Increment function

$$\varphi(t_i, x_i, T) = c_1 k_1 + c_2 k_2 + ... + c_n k_n$$

  - $c_1, c_2, ..., c_n$ constant weights
  - $k_1, k_2, ..., k_n$ recurrence relations
    - gradients

Runge-Kutta

- Recurrence relations
  - General form

$$k_1 = f(t_i, x_i),$$
$$k_2 = f(t_i + p_2 T, x_i + a_{21} T k_1),$$
$$k_3 = f(t_i + p_3 T, x_i + a_{31} T k_1 + a_{32} T k_2),$$
$$\vdots$$
$$k_n = f(t_i + p_n T, x_i + a_{n1} T k_1 + a_{n2} T k_2 + \ldots + a_{n,n-1} T k_{n-1})$$

  - $p_2, p_3, \ldots, p_n$ step fractions

Runge-Kutta

- A family of methods
  - Order *n*
  - Each method determined by choice of constants

    $c_1, c_2, \ldots, c_n$

    $p_2, p_3, \ldots, p_n$

    $a_{21}, a_{31}, a_{32}, a_{41}, \ldots, a_{n,n-1}$

  - More symmetric
    - In terms of gradient approximation over the entire interval

First Order Runge-Kutta

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = c_1 k_1$$

$$k_1 = f(t_i, x_i)$$

- A lot of notation that simplifies to

$$x(t_{i+1}) = x(t_i) + Tc_1 f(t_i, x_i)$$

- Generally, methods choose $\sum_{l=1}^{n} c_l = 1$
- So $c_1 = 1$
  - i.e. Euler's method $x(t_{i+1}) = x(t_i) + Tf(t_i, x_i)$

Second Order Runge-Kutta

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = c_1 k_1 + c_2 k_2$$

$$k_1 = f(t_i, x_i),$$

$$k_2 = f(t_i + p_2 T, x_i + a_{21} T k_1)$$

- p's must be within the interval $[t_i, t_i+T]$
  - i.e.

$$p_l \in [0,1] \quad \forall l$$

Second Order Runge-Kutta

$$c_1 = 0,\ c_2 = 1,\ p_2 = \tfrac{1}{2},\ a_{21} = \tfrac{1}{2}$$

- Modified Euler's method

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = 1.k_2$$

$$k_1 = f(t_i, x_i),$$

$$k_2 = f(t_i + \tfrac{1}{2}T, x_i + \tfrac{1}{2}Tk_1)$$

- Simplifying

$$x(t_{i+1}) = x(t_i) + Tf\left(t_i + \tfrac{1}{2}T, x_i + \tfrac{1}{2}Tf(t_i, x_i)\right)$$

Second Order Runge-Kutta

$$c_1 = \tfrac{1}{2},\ c_2 = \tfrac{1}{2},\ p_2 = 1,\ a_{21} = 1$$

- Heun's method

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = \tfrac{1}{2}k_1 + \tfrac{1}{2}k_2$$

$$k_1 = f(t_i, x_i),$$

$$k_2 = f(t_i + 1.T, x_i + 1.Tk_1)$$

- Simplifying

$$x(t_{i+1}) = x(t_i) + \tfrac{T}{2}\left(f(t_i, x_i) + f\left(t_i + T, x_i + Tf(t_i, x_i)\right)\right)$$

Third Order Runge-Kutta

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = c_1 k_1 + c_2 k_2 + c_3 k_3$$

$$k_1 = f(t_i, x_i),$$
$$k_2 = f(t_i + p_2 T, x_i + a_{21} T k_1),$$
$$k_3 = f(t_i + p_3 T, x_i + a_{31} T k_1 + a_{32} T k_2)$$

Third Order Runge-Kutta

$$c_1 = \tfrac{1}{6},\ c_2 = \tfrac{4}{6},\ c_3 = \tfrac{1}{6},\ p_2 = \tfrac{1}{2},\ p_3 = 1,\ a_{21} = \tfrac{1}{2},\ a_{31} = -1,\ a_{32} = 2$$

- Popular version

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = \tfrac{1}{6}(k_1 + 4k_2 + k_3)$$

$$k_1 = f(t_i, x_i),$$

$$k_2 = f(t_i + \tfrac{1}{2}T, x_i + \tfrac{1}{2}Tk_1),$$

$$k_3 = f(t_i + T, x_i - Tk_1 + 2Tk_2)$$

- Simplifying (slightly)

$$x(t_{i+1}) = x(t_i) + \tfrac{T}{6}\left(k_1 + 4k_2 + k_3\right)$$

Fourth Order Runge-Kutta

$$c_1 = \tfrac{1}{6},\ c_2 = \tfrac{2}{6},\ c_3 = \tfrac{2}{6},\ c_4 = \tfrac{1}{6},\ p_2 = \tfrac{1}{2},\ p_3 = \tfrac{1}{2},\ p_4 = 1,$$

$$a_{21} = \tfrac{1}{2},\ a_{31} = 0,\ a_{32} = \tfrac{1}{2},\ a_{41} = 0,\ a_{42} = 0,\ a_{43} = 1$$

- Popular version

$$x(t_{i+1}) = x(t_i) + T\varphi(t_i, x_i, T)$$

$$\varphi(t_i, x_i, T) = \tfrac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = f(t_i, x_i), \qquad k_3 = f(t_i + \tfrac{1}{2}T, x_i + \tfrac{1}{2}Tk_2)$$

$$k_2 = f(t_i + \tfrac{1}{2}T, x_i + \tfrac{1}{2}Tk_1), \qquad k_4 = f(t_i + T, x_i + Tk_3)$$

- Simplifying (slightly)

$$x(t_{i+1}) = x(t_i) + \tfrac{T}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right)$$

Truncation Error

- Runge-Kutta method generation
  - Can be shown to be the same as a Taylor expansion of order *n*
  - Local truncation error is

$$CT^{n+1} + O(T^{n+2})$$

  - *C* determined by form of solution

Truncation Error

- Are errors similar to Taylor ?
- Example special case:

$$\frac{dx(t)}{dt} = f\left(t, x(t)\right) = cx$$

- RK methods:

$$x_{i+1} = x_i + Tcx_i = (1+Tc)x_i$$

$$x_{i+1} = x_i + Tc\left((1+\tfrac{T}{2}c)x_i\right) = \left(1+cT+\tfrac{1}{2!}(cT)^2\right)x_i$$

$$x_{i+1} = \left(1+cT+\tfrac{1}{2!}(cT)^2+\tfrac{1}{3!}(cT)^3\right)x_i$$

$$\frac{x_{i+1}}{x_i} = e^{cT} = 1+cT+\tfrac{1}{2!}(cT)^2+\tfrac{1}{3!}(cT)^3+\tfrac{1}{4!}(cT)^4+\ldots$$

Stability

- A numerical method is unstable
  - If the errors at each step are propagated without bound
- Stability regions for RK
  - Complex
  - Rule of thumb:
    - Find solution of equation using different step sizes – if resulting solutions are sufficiently similar, assume the RK method is stable

Multistep methods

- Rather than using only x_i to estimate x_{i+1}
- Multistep methods use
  - $x_i, x_{i-1}, \ldots, x_{i-k}$
- Interpolate
  - E.g. using a polynomial through $x_i, x_{i-1}, \ldots, x_{i-k}$
  - Then extrapolate to x_{i+1}
- Requires one derivative evaluation per step
  - Faster than RK (requires n)
- Problem
  - Might know x_0 but not $x_{-1}, x_{-2}, \ldots, x_{-k}$
  - Often not practical

Choice of *T*

- Previously
  - Too large: truncation error
  - Too small: round-off error
- In practise
  - Determined by dynamics of system
  - e.g. RLC underdamped response
    - Need to sample multiple times faster than the damped frequency
  - $T = \frac{\tau_{\min}}{10}$, $\tau_{\min}$: shortest recognisable period in signal
- One option
  - Start with a large step size
  - Decrease it until solution does not change “much”
    - e.g. less than some pre-determined tolerance

Choice of *T*

- For high-order RK ($\geq 4^{th}$)
  - Characteristic equation of system

$$\alpha_n \lambda^n + \alpha_{n-1} \lambda^{n-1} + \ldots + \alpha_1 \lambda + \alpha_0 = 0$$

  - Choose step size as $T = \dfrac{1}{10|\lambda_{\max}|}$
    - $\lambda_{\max}$: largest eigenvalue
  - Eigenvalues unknown
    - Approximate by $T = \dfrac{1}{10} \dfrac{1}{\sqrt[n]{\frac{\alpha_0}{\alpha_n}}}$
- Lower order: need smaller *T*
  - For comparable accuracy

References

- Extensive use made of
  - Rao, S. S. (2002). *Applied numerical methods for engineers and scientists*, Prentice-Hall, Upper Saddle River, NJ.

Tuesday tutorial: Approximation of $\frac{dx}{dt} = x - t^2 + 1$

Time steps t = 0, 0.2, 0.4, 0.6, 0.8, 1s

Exact	Euler	error	Heun	error	Mod. Euler	error
0.5000	0.5000	0	0.5000	0	0.5000	0
0.8293	0.8000	0.0293	0.8260	0.0033	0.8280	0.0013
1.2141	1.1520	0.0621	1.2069	0.0072	1.2114	0.0027
1.6489	1.5504	0.0985	1.6372	0.0117	1.6447	0.0043
2.1272	1.9885	0.1387	2.1102	0.0170	2.1213	0.0059
2.6409	2.4582	0.1827	2.6177	0.0232	2.6332	0.0077