COMS30017
COMPUTATIONAL NEUROSCIENCE

LECTURE: INTRODUCTION TO NUMERICAL SOLUTIONS OF DIFFERENTIAL EQUATIONS

PART-2

Dr. Rahul Gupta
xv20319@bristol.ac.uk

How do we set the timestep?

- The only user-defined parameter in Euler and RK is the timestep size. How do we choose it?

- We could use the Taylor series to estimate the magnitude of the error, decide what error we are comfortable living with, and set the timestep accordingly.

- However the size of the error likely varies at different points in the solution.

- In practise we can just test a range of timestep sizes and see at what value the numerical solution starts to stabilise.

- This choice can be informed by examining the timescales in the DEs. Usually we would want our timestep to be smaller than the fastest time constant in the differential equation. < 1

ODE solving in practise

• Euler and RK are two (families of) ODE solver methods from a large set.

• They are both examples of “fixed timestep” solvers: the user chooses the timestep and leaves it the same for the duration of the computation.

• However many ODE problems are “stiff”, meaning that they need a painfully small timestep to avoid numerical instability.

• In systems where the solution move between periods of fast changes and periods of slow changes, we can do better with “adaptive timestep” methods.

• Adaptive methods look at the magnitude of the derivatives online while solving, and take larger timesteps while the derivatives are small. The size of the timestep is computed to match some user-specified “tolerance” for the error.

ODE solving in practise

- Most of the time we hand the job of solving to pre-written packages.

- Common languages used in scientific computing often have ODE solver packages with many method options:

Python	MATLAB	Julia
vode	ode45	Euler
zvode	ode23	Midpoint
lsoda	ode113	SSPRK22
dopri5	ode15s	SSPRK33
dop853	ode23s	SSPRK104
	ode23t	RK4
	ode23tb	BS3
	ode15i	DP5
		Tsit5
		BS5
		Vern6
		Vern7
		TanYam7
		DP8
		TsitPap8
		Vern8
		Vern9
		Feagin10
		Feagin12
		Feagin14
		ImplicitEuler
		Trapezoid
		Rosenbrock23
		Rosenbrock32

Further Reading

Conor’s notes: https://github.com/coms30127/2019_20/ notes/numerical