559

Numerical Integration Methods

Fish 559; Lecture 2

Some Preliminaries-I

- Numerical methods are numerical and hence subject to various types of errors:
  - Roundoff error: The consequences of a computer representing a real number using finite accuracy. Note:
    - roundoff errors accumulate; and
    - subtraction of two numbers that are nearly identical may lead to very large roundoff errors.
  - Truncation error: the difference between the true answer and the value from the numerical analysis.
  - e.g. approximation of a Taylor series which is an infinite series but we only use first two terms. E.g. delta method which uses the first term to estimate variance.

Some Preliminaries-II

- Truncation error:
  - There are often error bounds for methods (though we will not consider these in this course).
  - Truncation error can be reduced by increasing the number of steps / reducing tolerances or step sizes, but this can be costly in terms of computation time and, eventually, roundoff error will begin to impact the results.
  - E.g. setting low tolerance in TMB

Numerical Integration (or numerical quadrature)

- The objective of numerical integration is to solve the problem:

$$I = \int_a^b f(x)\,dx$$

- which is equivalent to finding $y(b)$ subject to:

$$\frac{dy}{dx} = f(x); \;\; y(a) = 0$$

- We will assume that the function *f* cannot be integrated analytically!
- Many integrals are not analytic

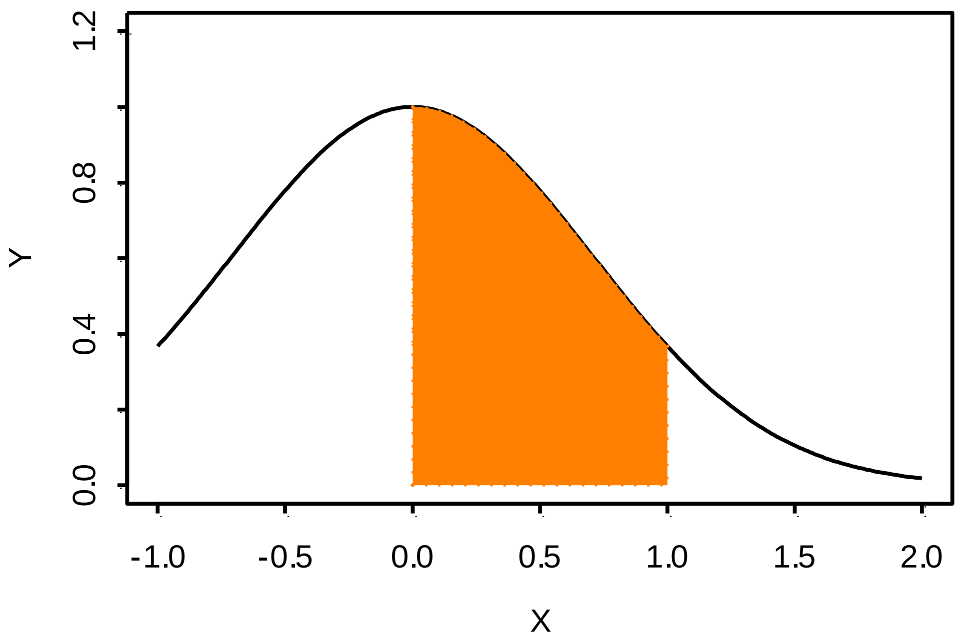
Numerical Integration (why do this)

- The density of larvae is given by the function *f* (as a function of distance along a river); how many larvae are there between points *a* and *b* along the river?
- Let *f* be the function that determines the rate of change in the number of animals of species *i* (which depends on the number of animals of species *i*). If there are 1,000 animals of species *i* at time t_0, how many are there at time t_0+1.

Numerical Integration (An example problem)

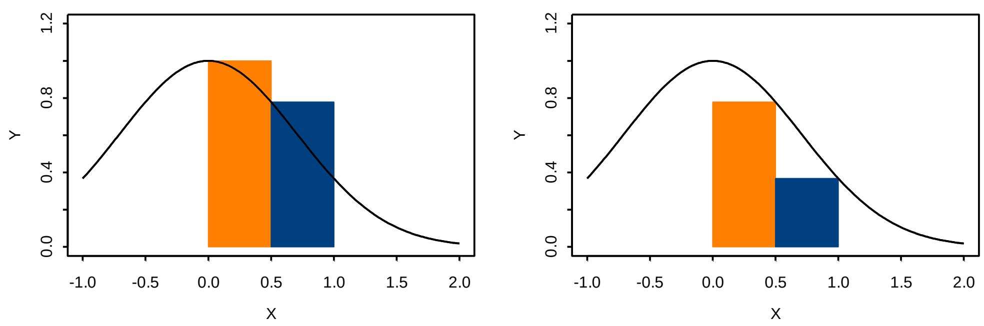
- We wish to evaluate the integral:

$$I = \int_0^1 e^{-x^2}\,dx$$



Numerical Integration (Single-step approaches-I)

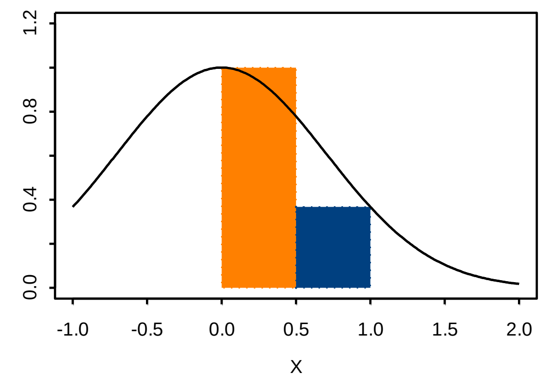
- We could approximate the integral by its height at its first point, at its last point, or at the middle.



- Approximating the integral by the average of the first two approaches is the "Trapezoidal rule" (only for this particular example).

Numerical Integration (Single-step approaches-II)

- We wish to integrate from *a* to *b*. We have formulae that get more accurate as more terms are added:



- Trapezoidal rule:

$$I \approx h\left[\tfrac{1}{2} f(a) + \tfrac{1}{2} f(b)\right]$$

- Simpson's Rule:

$$I \approx h\left[\tfrac{1}{3} f(a) + \tfrac{4}{3} f(a+h) + \tfrac{1}{3} f(b)\right]$$

- Boole's rule:

$$I \approx h\left[\tfrac{14}{45} f(a) + \tfrac{64}{45} f(a+h) + \tfrac{24}{45} f(a+2h) + \tfrac{64}{45} f(a+3h) + \tfrac{14}{45} f(b)\right]$$

- *h* is the step size. It is (b-a) divided by the number of function calls less 1.
- There are many variants on this theme!

http://en.wikipedia.org/wiki/George_Boole

Numerical Integration (Single-step approaches-III)

- Results:
  - Trapezodinal rule: 0.68394
  - Simpson’s rule: 0.74718
  - Boole’s rule: 0.74683
  - Exact: 0.74682!

  - Tradeoff between truncation error and computation time.
  - Acceptable level of accuracy depends on problem and rounding (accumulation of errors through multiple calculations)
  - Most use Simpson’s rule

Numerical Integration (Multi-step rules)

$$\text{Now}: \int_a^c f(x)dx = \int_a^b f(x)dx + \int_b^c f(x)dx$$

- Therefore, in order to numerically integrate over the interval [*a*, *c*], we can apply one of the single-step rules multiple times and add the results, i.e. for Simpson's rule:

$$I \approx h\left[\tfrac{1}{3} f_1 + \tfrac{4}{3} f_2 + \tfrac{2}{3} f_3 + \ldots + \tfrac{4}{3} f_{N-1} + \tfrac{1}{3} f_N\right]$$

Other Approaches (Gaussian quadrature-I)

- Generalizing the previous approaches:

$$I = \sum_i w_i \, f(x_i)$$

- The previous methods have all been based on equal spacing along abscissa. We can improve our accuracy (at the expense of a bit more programming) by allowing freedom in the location of the absciccas.

$$I = \int_a^b f(x)dx = \int_{-1}^{1} f(x(t))\, x^{'}(t)\, dt = \int_{-1}^{1} f(x(t)) \frac{b-a}{2} dt = \frac{b-a}{2} \int_{-1}^{1} f(x(t))\, dt$$

$$I \approx \frac{b-a}{2} \sum_i w_i \, f(x(t_i)) = \frac{b-a}{2} \sum_i w_i \, f\,''(t_i)$$

- Tradeoff between truncation error and amount of programming effort to get it right

Other Approaches (Gaussian quadrature-II)

- For two point Gaussian quadrature:

$$I \approx \frac{b-a}{2}\left[\tilde{f}(-0.57735026) + \tilde{f}(0.57735026)\right]$$

$$\int_0^1 e^{-x^2}\,dx \approx \tfrac{1}{2}[e^{-(x+1)^2/4} + e^{-(1-x)^2/4}];\;\; x = 0.5773502691$$

- This lead to an approximation to the integral of 0.74965. The three point formula leads to 0.74673.
- Abramowitz and Stegun provide details on the absiccas and weights for equations of different orders (numbers of terms in the approximating equation).

The Error Function

- Perhaps the most frequently required integral in fisheries is the "error function":

$$I = \int_{-\infty}^{a} e^{-x^2/2} dx$$

- It is not necessary to apply the methods of this lecture to compute this integral because very accurate approximations to it have been programmed and are available in software libraries (e.g. *NORMDIST* in EXCEL, *cumd_norm* in ADMB and *pnorm* in R and TMB).
- The lesson from this example, is that someone may have already solved your problem; before writing your own code check whether this is the case!

Multidimensional Integrals

- Multidimensional integrals are much more complicated than single-dimensional integrals. There are a few basic approaches to computing these integrals:
  - Split the problem into multiple single dimensional integrals.
  - Use Monte Carlo methods.
  - Use numerical methods (such as Laplace's method).
- The issue of computing multidimensional integrals using Monte Carlo methods will be discussed later.

Integration of ODEs

- The generic problem is the study of a set of *N* coupled *first order* differential equations (e.g. EcoSim x = time, y = biomass of different components): vector of equations

$$\frac{dy_i(x)}{dx} = f_i(x, y_1, .., y_N); \qquad i = 1, 2, ..., N$$

- The specification of ODEs generally includes initial conditions, i.e.:

$$y_i(0) = Y_i; \qquad i = 1, 2, ..., N$$

A Familiar Set of ODEs (Ecosim)

$$\frac{dB_i}{dt} = g_i \sum_j C_{j,i} - \sum_j C_{i,j} + I_i - (M_i + F_i + e_i)B_i$$

Net growth efficiency

Consumption of species *j* by species *i*

Immigration

Non-predation mortality, fishing mortality and emigration

- The initial conditions are determined from Ecopath.

Euler's Method

- Euler's method updates the solution from x_n to $x_{n+1}=x_{n+h}$ where x is time using the formula:

$$y_{n+1} = y_n + hf(x_n, y_n)$$

- This method is simple, but can be very inaccurate (unless a very small step-size is used).

Other Ways to Solve ODEs

- Runge-Kutta methods.
- Richardson extrapolation.
- Predictor-corrector methods.

- We will focus on the Runge-Kutta method.

Runge-Kutta Methods-I

- These tend to perform adequately for many problems (but more efficient algorithms do exist).
- The step size (h) can be monitored and modified automatically to improve performance.

Runge-Kutta Methods-II

- There are many versions of the Runge-Kutta method but the fourth-order version seems the most popular (evaluation of differential equations at different steps of x and y):

$$y_{n+1} = y_n + \tfrac{k_1}{6} + \tfrac{k_2}{3} + \tfrac{k_3}{3} + \tfrac{k_4}{6}$$

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n + \tfrac{h}{2}, y_n + \tfrac{k_1}{2})$$

$$k_3 = hf(x_n + \tfrac{h}{2}, y_n + \tfrac{k_2}{2})$$

$$k_4 = hf(x_n + h, y_n + k_3)$$

- There are multidimensional versions of the RK algorithm.

An Example-I

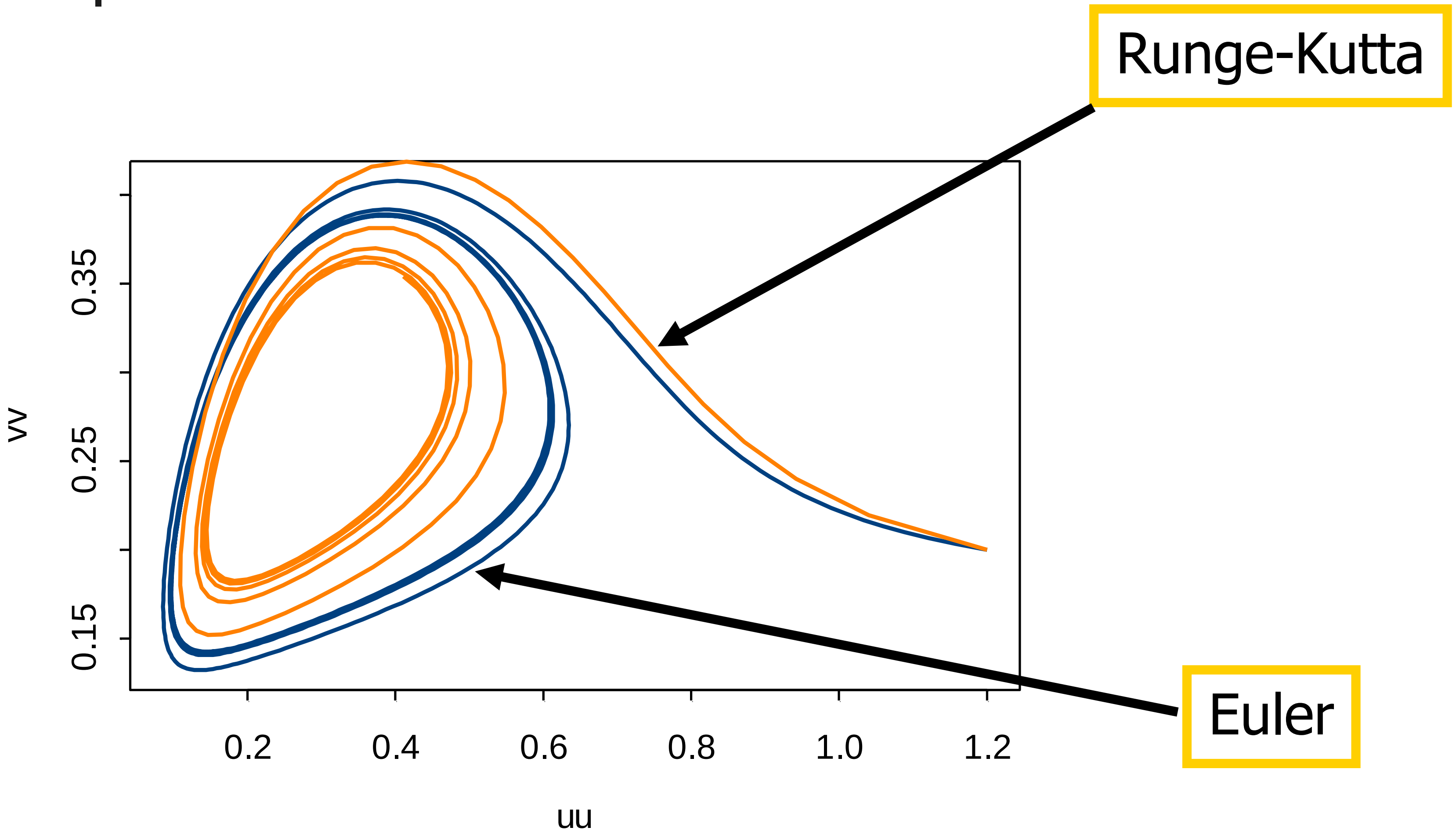
- Solve the following ODE using (a) Euler's method and (b) the Runge-Kutta fourth-order method:

$$\frac{du}{dt} = u(1-u) - \frac{a\,u\,v}{u+d}; \qquad \frac{dv}{dt} = bv(1-v/u)$$

- where:

$$u(0) = 1.2;\ \ v(0) = 0.2;\ \ a = 1;\ \ b = 0.2;\ \ d = 0.1$$

An Example -II



Moving to PDEs

- Partial Differential Equations (PDEs) are much more difficult to solve.
- Although algorithms do exist to solve these, the literature is vast and we will not discuss them in this course.

Numerical Instability-I

- We wish to solve the following initial value problem:

$$\frac{dy}{dx} = -2y + 1 \qquad y(0) = 1$$

- We can apply Euler's method:

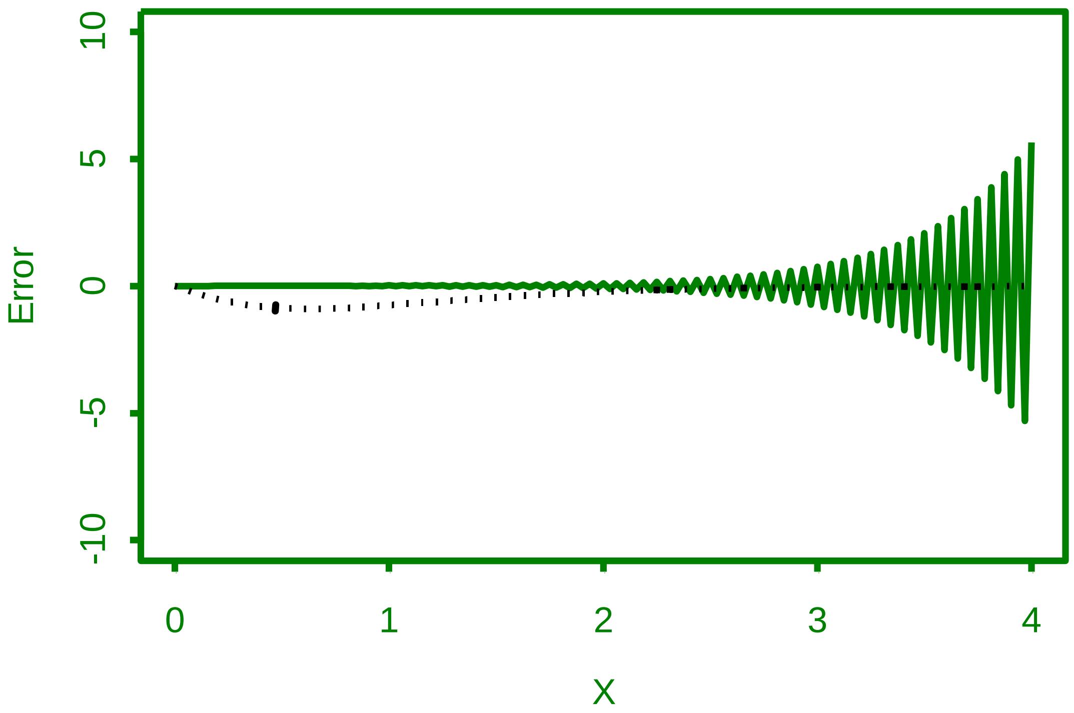
$$y_{n+1} = y_n + h\, f_n$$

- or Milne's method:

$$y_{n+1} = y_{n-1} + 2hf_n$$

Numerical Instability-II

Euler’s method is stable but Milne’s rule Becomes unstable after x=2!



In the case, we knew the correct solution, what would we do in reality?