Stability

While the accuracy (truncation error) of a method can be determined by using Taylor series, in some cases a particular method for a particular problem may be unstable:

small deviations from the true solution (what the algorithm ideally gives) grow as the algorithm progresses.

Consider Euler algorithm for $\dot{y} = f(y,t)$

1

Assume that there is an error, , arising initially from e.g. roundoff error, associated with each step

$$f_n = f(y_n, t_n) \to f(y_n + \delta y_n, t_n)$$

$$=$$

Euler becomes

$$y_{n+1} + \delta y_{n+1} \doteq$$

Error δy_n will at each step if

Euler is stable if $|g| < 1$

Consider three archetypal cases (α positive)

growth $y = y_0 e^{\alpha t}$ $\rightarrow$

decay $y = y_0 e^{-\alpha t}$ $\rightarrow$

oscillation $y = y_0 e^{i\alpha t}$ $\rightarrow$

For growth $g = 1 + \Delta t \dfrac{\partial f}{\partial y} =$

decay $g =$

oscillation $g =$

$|g|^2 =$

What about 2nd order RK (midpoint method)?

Midpoint Method

$$y_{n+1/2} = y_n + \frac{1}{2}\,\Delta t\, f(y_n, t_n)$$
$$y_{n+1} = y_n + \Delta t\, f(y_{n+1/2}, t_{n+1/2})$$

$$y_{n+1} = y_n + \Delta t\, f\left(y_n + \tfrac{1}{2}\Delta t\, f(y_n, t_n),\ t + \Delta t/2\right)$$

$$= y_n + \Delta t\left[f(y_n, t_n) + \frac{1}{2}\Delta t\, f_n \frac{\partial f}{\partial y}\Big|_n + \frac{\Delta t}{2}\frac{\partial f}{\partial t}\Big|_n\right] + \mathcal{O}(h^3)$$

$$y_{n+1} \doteq y_n + \Delta t\, f_n + \tfrac{1}{2}\Delta t^2\left(f_n \frac{\partial f}{\partial y}\Big|_n + \frac{\partial f}{\partial t}\right)$$

As before, $y_n \rightarrow$
$f_n \rightarrow$

growth: $\dfrac{\partial f_n}{\partial y} = \alpha$

decay: $\dfrac{\partial f_n}{\partial y} = -\alpha$

$-1 < g < 1 \rightarrow$

oscillation $\frac{\partial f_n}{\partial y} = i\alpha$

errors will accumulate
quite slowly

Monte Carlo Simulations

- based on taking averages of many different realizations of a system. These configurations are generated using (pseudo) random numbers.

- Most random number generators use a chaotic sequence.

e.g. multiplicative congruent method
- based on large integers that have no

$$x_{n+1} =$$

where is the remainder after dividing

$x \% y =$

e.g. $\mathrm{mod}(12,5) =$

This sequence generates integers less than in a "random" order.
- first value
- for a given
- for a new
- not at all

For this simple algorithm, quality of pseudorandom sequence depends a lot on choice of

A good pair is (Lewis, Goodman, Miller 1969)

$a =$ (largest 32-bit unsigned

$b =$ integer is)

When implementing, need to take care that integer product can be stored ( )

- either use unsigned long integer (64-bit integer)
  or use computational tricks (see Numerical Recipes)

A slightly more general class of pseudo-RNG is the

$$x_{n+1} =$$

(congruent map

linear:)

Basic idea: multiply two big integers →

To get result on [0, 1), simply take

Many algorithms exist for generating RNs on
see NR, www, google