PART II

Review of numerical methods for
Ordinary Differential Equations

BOUNDARY VALUE PROBLEMS (I)

- Solving a two–point boundary value problem

$$\frac{d^2y}{dx^2} = g \qquad \text{for } x \in (0, 2\pi)$$
$$y(0) = y(2\pi) = 0$$

- Finite–difference approximation:
  - Second–order Central Difference formula for the interior nodes:

$$\frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} = g_j \text{ for } j = 1, \ldots, N$$

  where $h = \frac{2\pi}{N+1}$ and $x_j = jh$
  - Endpoint nodes:

$$y_0 = 0 \quad \Longrightarrow y_2 - 2y_1 = h^2 g_1$$
$$y_{N+1} = 0 \Longrightarrow -2y_N + y_{N-1} = h^2 g_N$$

  - Tridiagonal algebraic system

BOUNDARY VALUE PROBLEMS (II)

- Solving a two–point boundary value problem

$$\frac{d^2y}{dx^2} = g \qquad \text{for } x \in (0, 2\pi)$$
$$\frac{dy}{dx}(0) = \frac{dy}{dx}(2\pi) = 0$$

- Finite–difference approximation:
  - Second–order Central Difference formula for the interior nodes:

$$\frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} = g_j \text{ for } j = 1, \ldots, N$$

  - First–order Forward/Backward Difference formulae to re–express endpoint values:

$$\frac{y_1 - y_0}{h} = 0 \Longrightarrow y_0 = y_1$$
$$\frac{y_{N+1} - y_N}{h} = 0 \Longrightarrow y_{N+1} = y_N$$

  First–order only — degraded accuracy!
  - Tridiagonal algebraic system — Where is the problem?

BOUNDARY VALUE PROBLEMS (III)

- In order to retain second–order accuracy in the approximation of the Neumann problem need to use higher-order formulae at endpoints, e.g.

$$y_0' = \frac{-y_2 + 4y_1 - 3y_0}{2h} = 0 \Longrightarrow y_0 = \frac{1}{3}(-y_2 + 4y_1)$$

- The first row thus becomes

$$\frac{2}{3}y_2 - \frac{2}{3}y_1 = h^2 g_1$$

Second–order accuracy recovered!

Boundary Value Problems (IV)

- Compact Stencils — stencils based on three grid points only: $\{x_{j+1}, x_j, x_{j-1}\}$ at the $j-th$ node
- Is is possible to obtain higher (then second) order of accuracy on compact stencils? — YES!
- Consider the central difference approximation to the equation $\frac{d^2y}{dy^2} = g$

$$\frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} - \frac{h^2}{12} y_j^{(iv)} + O(h^4) = g_j$$

- Re-express the error term $\frac{h^2}{12} y_j^{(iv)}$ using the equation in question:

$$\frac{h^2}{12} y_j^{(iv)} = \frac{h^2}{12} g_j'' = \frac{h^2}{12}\left[\frac{g_{j+1} - 2g_j + g_{j-1}}{h^2} - \frac{h^2}{12} g_j^{(iv)} + O(h^4)\right]$$

- Inserting into the original finite–difference equation:

$$\frac{y_{j+1} - 2y_j + y_{j-1}}{h^2} = g_j + \frac{g_{j+1} - 2g_j + g_{j-1}}{12} + O(h^4)$$

- Slight modification of the RHS $\Longrightarrow$ fourth—order accuracy!!!

Boundary Value Problems (V)

- Compact Finite Difference Schemes — drawbacks:
  - need to be tailored to the specific equation solved
  - can get fairly complicated for more complex equations

Initial Value Problems — General Remarks

- Consider the following Cauchy problem :

$$\frac{dy}{dt} = f(y,t) \text{ with } y(t_0) = y_0$$

  The independent variable t is usually referred to as time .
- Equations with higher–order derivatives can be reduced to systems of first–order equations
- Generalizations to systems of ODEs straightforward
- When the RHS function doesn't depend on y, i.e., $f(y,t) = f(t)$, solution obtained via quadrature
- Assume uniform time–steps (h is constant)

Initial Value Problems — Characterization of Integration Methods

- ACCURACY — unlike in the Boundary Value Problems, there is no terminal condition and approximation errors may accumulate in time; consequently, a relevant characterization of accuracy is provided by the global error

$$(\text{global error}) = (\text{local error}) \times (\text{\# of time steps}),$$

  rather than the local error .
- STABILITY — unlike in the Boundary Value Problems, where boundedness of the solution at final time is enforced via a suitable terminal condition , in Initial Value Problems there is a priori no guarantee that the solution will remain bounded.

INITIAL VALUE PROBLEMS — MODEL PROBLEM

- Stability of various numerical schemes is usually analyzed by applying these schemes to the following linear model:

$$\frac{dy}{dt} = \lambda y = (\lambda_r + i\lambda_i)y \text{ with } y(t_0) = y_0,$$

which is stable when $\lambda_r <= 0$.

- Exact solution: $y(t) = y_0 e^{\lambda t} = \left(1 + \lambda h + \frac{\lambda^2 h^2}{2} + \frac{\lambda^3 h^3}{6} + \dots\right) y_0$

- Motivation — consider the following advection–diffusion PDE:

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} - a\frac{\partial^2 u}{\partial x^2} = 0$$

Taking Fourier transform yields (k is the wavenumber):

$$\frac{d\hat{u}_k}{dt} + cik\hat{u}_k + ak^2\hat{u}_k = 0$$

where
  - the real term $ak^2\hat{u}_k$ represents diffusion
  - the imaginary term $cik\hat{u}_k$ represents advection

INITIAL VALUE PROBLEMS — EXPLICIT EULER SCHEME (I)

- Consider a Taylor series expansion

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + \frac{h^2}{2}y''(t_n) + \dots$$

Using the ODE we obtain

$$y' = \frac{dy}{dt} = f$$
$$y'' = \frac{dy'}{dt} = \frac{df}{dt} = f_t + ff_y$$

- Neglecting terms proportional to second and higher powers of h yields the Explicit Euler Method

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

- Retaining higher–order terms is inconvenient, as it requires differentiation of f and does not lead to schemes with desirable stability properties.

INITIAL VALUE PROBLEMS — EXPLICIT EULER SCHEME (II)

- Local error analysis:

$$y_{n+1} = (1 + \lambda h)y_n + [O(h^2)]$$

- Global error analysis:

$$(\text{global error}) = Ch^2 \cdot N = Ch^2 \cdot \frac{T}{h} = C'h$$

Thus, the scheme is
  - locally second–order accurate
  - globally (over the interval $[t_0, t_0 + Nh]$) first–order accurate

INITIAL VALUE PROBLEMS — EXPLICIT EULER SCHEME (III)

- Stability (for the model problem)

$$y_{n+1} = y_n + \lambda h y_n = (1 + \lambda h)y_n$$

Thus, the solution after n time steps

$$y_n = (1 + \lambda h)^n y_0 \triangleq \sigma^n y_0 \implies \sigma = 1 + \lambda h$$

For large n, the numerical solution remains stable iff

$$|\sigma| \le 1 \implies (1 + \lambda_r h)^2 + (\lambda_i h)^2 \le 1$$

  - conditionally stable for real λ
  - unstable stable for imaginary λ

Initial Value Problems — Implicit Euler Scheme (I)

- Implicit Schemes — based on approximation of the RHS that involve $f(y_{n+1},t)$, where y_{n+1} is the unknown to be determined
- Implicit Euler Scheme — obtained by neglecting second and higher–order terms in the expansion:

$$y(t_n) = y(t_{n+1}) - hy'(t_{n+1}) + \frac{h^2}{2}y''(t_{n+1}) - \dots$$

Upon substitution $\left.\frac{dy}{dt}\right|_{t_{n+1}} = f(y_{n+1},t_{n+1})$ we obtain

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

The scheme is

- locally second–order accurate
- globally (over the interval $[t_0,t_0+Nh]$) first–order accurate

Initial Value Problems — Implicit Euler Scheme (II)

- Stability (for the model problem):

$$y_{n+1} = y_n + \lambda h y_{n+1} \implies y_{n+1} = (1-\lambda h)^{-1} y_n$$

$$y_{n+1} = \left(\frac{1}{1-\lambda h}\right)^n y_0 \triangleq \sigma^n y_0 \implies \sigma = \frac{1}{1-\lambda h}$$

$$|\sigma| \le 1 \implies (1-\lambda_r h)^2 + (\lambda_i h)^2 \ge 1$$

Implicit Euler scheme is thus stable for

- all stable model problems
- most unstable model problems

- When solving systems of ODEs of the form $\mathbf{y} = \mathcal{A}(t)\mathbf{y}$, each implicit step requires solution of an algebraic system: $\mathbf{y}_{n+1} = (I - h\mathcal{A})^{-1}\mathbf{y}_n$
- Implicit schemes are generally hard to implement for nonlinear problems

Initial Value Problems — Crank–Nicholson Scheme (I)

- Obtained by approximating the formal solution of the ODE $y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(y,t)\,dt$ using the trapezoidal quadrature:

$$y_{n+1} = y_n + \frac{h}{2}\left[f(y_n,t_n) + f(y_{n+1},t_{n+1})\right]$$

The scheme is

- locally third–order accurate
- globally (over the interval $[t_0,t_0+Nh]$) second–order accurate

- Stability (for the model problem):

$$y_{n+1} = y_n + \frac{\lambda h}{2}(y_{n+1}+y_n) \implies y_{n+1} = \left(\frac{1+\frac{\lambda h}{2}}{1-\frac{\lambda h}{2}}\right) y_n$$

$$y_{n+1} = \left(\frac{1+\frac{\lambda h}{2}}{1-\frac{\lambda h}{2}}\right)^n y_0 \triangleq \sigma^n y_0 \implies \sigma = \frac{1+\frac{\lambda h}{2}}{1-\frac{\lambda h}{2}}$$

$$|\sigma| \le 1 \implies \Re(\lambda h) \le 0$$

Stable for all model ODEs with stable solutions

Initial Value Problems — Leapfrog Scheme (I)

- Leapfrog as an example of a two–step method :

$$y_{n+1} = y_{n-1} + 2h\lambda y_n$$

- Characteristic equation for the amplification factor $(y_n = \sigma^n y_0)$

$$\sigma^2 - 2h\lambda\sigma - 1 = 0$$

where roots give the amplification factors:

$$\sigma_1 = \lambda h + \sqrt{1+\lambda^2h^2} \simeq \quad 1 + \lambda h + \frac{\lambda^2h^2}{2} + \dots \quad = \quad e^{\lambda h} + O(h^3)$$

$$\sigma_2 = \lambda h - \sqrt{1+\lambda^2h^2} \simeq -(1 - \lambda h + \frac{\lambda^2h^2}{2} - \dots) = -e^{-\lambda h} + O(h^3)$$

Thus, the scheme is

- locally third–order accurate
- globally (over the interval $[t_0,t_0+Nh]$) second–order accurate

Initial Value Problems — Leapfrog Scheme (II)

- Stability for diffusion problems ($\lambda = \lambda_r$):

$$\sigma_1 = \lambda h + \sqrt{1 + \lambda_r^2 h^2} > 1 \quad \text{for all} \quad h > 0$$

Thus the scheme is unconditionally unstable for diffusion problem!

- Stability for advection problems ($\lambda = i\lambda_i$):

$$\sigma_{1/2}^2 = 1 \text{ (!!!) for } \; h < \frac{1}{|\lambda_i|}$$

Thus the scheme is conditionally unstable and non–diffusive for advection problems!

- Question — analyze dispersive (i.e., related to $\arg(\sigma)$) errors of the leapfrog scheme.

Initial Value Problems — Multistep Procedures

- General form of a multistep procedure :

$$\sum_{j=1}^{p} \alpha_j y_{n+j} = h \sum_{j=1}^{q} \beta_j f(y_{n+j}, t_{n+j})$$

with characteristic polynomials

$$\xi_p(z) = \alpha_p z^p + \alpha_{p-1} z^{p-1} + \dots + \alpha_0$$
$$\zeta_q(z) = \beta_q z^q + \beta_{q-1} z^{q-1} + \dots + \beta_0$$

  – if $p > q$ — explicit scheme
  – if $p \le q$ — implicit scheme

- A (ξ, ζ) –procedure converges uniformly in $[a,b]$, i.e., $\lim_{h\to 0} \max_{t_n \in [a,b]} |y_n - y(t_n)| = 0$ if:
  – the following consistency conditions are verified: $\xi(1) = 0$ and $\xi'(1) = \zeta(1)$ (*consistency condition*)
  – all roots of the polynomial $\xi(z)$ are such that $|z_i| \le 1$ and the roots with $|z_k| = 1$ are simple (*stability condition*)

Initial Value Problems — Runge–Kutta Methods (I)

- General form of a fractional step method :

$$y_{n+1} = y_n + \gamma_1 h k_1 + \gamma_2 h k_2 + \gamma_3 h k_3 + \dots$$

where

$$k_1 = f(y_n, t_n)$$
$$k_2 = f(y_n + \beta_1 h k_1, t_n + \alpha_1 h)$$
$$k_3 = f(y_n + \beta_2 h k_1 + \beta_3 h k_2, t_n + \alpha_2 h)$$
$$\vdots$$

- Choose γ_i, β_i and α_i to match as many expansion coefficients as possible in

$$y(t_{n+1}) = y(t_n) + h y'(t_n) + \frac{h^2}{2} y''(t_n) + \frac{h^3}{6} y'''(t_n) \dots$$
$$y' = f$$
$$y'' = f_t + f f_y$$
$$y''' = f_{tt} + f_t f_y 2 f f_{yt} + f^2 f_{yt} + f^2 f_{yy}$$

- Runge—Kutta methods are self–starting with fairly good stability and accuracy properties.

Initial Value Problems — Runge–Kutta Methods (II)

- RK4 — an ODE workhorse:

$$y_{n+1} = y_n + \frac{h}{6} k_1 + \frac{h}{3}(k_2 + k_3) + \frac{h}{6} k_4$$
$$k_1 = f(y_n, t_n) \qquad k_2 = f(y_n + \frac{h}{2} k_1, t_{n+1/2})$$
$$k_3 = f(y_n + \frac{h}{2} k_2, t_{n+1/2}) \qquad k_4 = f(y_n + h k_3, t_{n+1})$$

- The amplification factor:

$$\sigma = 1 + \lambda h + \frac{\lambda^2 h^2}{2} + \frac{\lambda^3 h^3}{6} + \frac{\lambda^4 h^4}{24}$$

Thus, stability iff $|\sigma| \le 1$

- Accuracy:

$$e^{\lambda h} = \sigma + O(h^5)$$

Thus, the scheme is
  – locally fifth–order accurate
  – globally (over the interval $[t_0, t_0 + Nh]$) fourth–order accurate

INITIAL VALUE PROBLEMS — RUNGE'S PRINCIPLE

- Let $(k+1)$ be the local truncation error; denote $Y(t,h)$ an approximation of the exact solution $y(t)$ computed with the step size h; then at $t = t_0 + 2nh$:

$$y(t) - Y(t,h) \simeq C\,2\,n\,h^{k+1} \quad = C(t-t_0)h^k$$
$$y(t) - Y(t,2h) \simeq C\,n\,(2h)^{k+1} = C(t-t_0)2^k\,h^k$$

  Subtracting:

$$Y(t,2h) - Y(t,h) \simeq C(t-t_0)(1-2^k)h^k$$

  Thus we can obtain an estimate of the absolute error based on solution with two step–sizes only:

$$y(t) - Y(t,h) \simeq \frac{Y(t,h) - Y(t,2h)}{2^k - 1}$$

- Runge's principle is very useful for adaptive step size refinement