

CS 450: Numerical Analysis¹

Initial Value Problems for Ordinary Differential Equations

University of Illinois at Urbana-Champaign

¹*These slides have been drafted by Edgar Solomonik as lecture templates and supplementary material for the book “Scientific Computing: An Introductory Survey” by Michael T. Heath ([slides](#)).*

Ordinary Differential Equations

- ▶ An *ordinary differential equation (ODE)* usually describes time-varying system by a function $\mathbf{y}(t)$ that satisfies a set of equations in its derivatives. The general *implicit* form is

$$\mathbf{g}(t, \mathbf{y}, \mathbf{y}', \mathbf{y}'', \dots, \mathbf{y}^{(k)}) = \mathbf{0},$$

but we restrict focus on the *explicit form*, $\mathbf{y}^{(k)} = \mathbf{f}(t, \mathbf{y}, \mathbf{y}', \mathbf{y}'', \dots, \mathbf{y}^{(k-1)})$.

- ▶ An ODE of any *order* k can be transformed into a first-order ODE,

$$\mathbf{u}' = \begin{bmatrix} \mathbf{u}'_1 \\ \vdots \\ \mathbf{u}'_{k-1} \\ \mathbf{u}'_k \end{bmatrix} = \begin{bmatrix} \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_k \\ \mathbf{f}(t, \mathbf{u}_1, \dots, \mathbf{u}_k) \end{bmatrix} \quad \text{where} \quad \mathbf{u}_i(t) = \mathbf{y}^{(i-1)}(t).$$

Consequently, we restrict our focus to systems of first-order ODEs. Of particular importance are *linear ODEs*, which have the form $\mathbf{y}' = \mathbf{A}(t)\mathbf{y}$, whose *coefficients* are said to be *constant* if $\mathbf{A}(t) = \mathbf{A}$ for all t .

Example: Newton's Second Law

- ▶ $F = ma$ corresponds to a second order ODE,

$$F = my''(t),$$
$$y''(t) = F/m.$$

- ▶ We can transform it into a first order ODE in two variables:

$$\mathbf{u} = \begin{bmatrix} y(t) \\ y'(t) \end{bmatrix},$$

$$\begin{bmatrix} u_1' \\ u_2' \end{bmatrix} = \mathbf{u}' = \mathbf{f}(t, \mathbf{u}) = \begin{bmatrix} u_2 \\ F/m \end{bmatrix}.$$

Initial Value Problems

- ▶ Generally, a first order ODE specifies only the derivative, so the solutions are non-unique. An *initial condition* addresses this:

$$\mathbf{y}(t_0) = \mathbf{y}_0$$

*This condition yields an initial value problem (IVP), which is the simplest example of a *boundary condition*.*

- ▶ Given an initial condition, an ODE must satisfy an integral equation for any given point t :

$$\mathbf{y}(t) = \mathbf{y}_0 + \int_{t_0}^t \mathbf{f}(s, \mathbf{y}(s)) ds,$$

In the special case that $\mathbf{y}' = \mathbf{f}(t)$, the integral can be computed directly by numerical quadrature to solve the ODE.

Existence and Uniqueness of Solutions

- ▶ For an ODE to have a unique solution, it must be defined on a closed domain D and be *Lipschitz continuous*:

$$\forall \mathbf{y}, \hat{\mathbf{y}} \in D, \quad \|\mathbf{f}(t, \hat{\mathbf{y}}) - \mathbf{f}(t, \mathbf{y})\| \leq L\|\hat{\mathbf{y}} - \mathbf{y}\|,$$

i.e. the rate of change of the ODE should itself change continuously. Any differentiable function f is Lipschitz continuous with

$$L = \max_{(t, \mathbf{y}) \in D} \|\mathbf{J}_f(t, \mathbf{y})\|,$$

where \mathbf{J}_f is Jacobian of f with respect to \mathbf{y} .

- ▶ The solutions of an ODE can be stable, unstable, or asymptotically stable: *Perturbation to the input causes a perturbation to the solution that*
 - ▶ *has bounded growth for a stable ODE,*
 - ▶ *unbounded growth for an unstable ODE, and*
 - ▶ *shrinks for an asymptotically stable ODE.*

Stability of 1D ODEs

- ▶ The solution to the scalar ODE $y' = \lambda y$ is $y(t) = y_0 e^{\lambda t}$, with stability dependent on λ :

$$\lim_{t \rightarrow \infty} y(t) = \begin{cases} \infty & : \lambda > 0 \text{ (unstable)} \\ y_0 & : \lambda = 0 \text{ (stable)} \\ 0 & : \lambda < 0 \text{ (asymptotically stable)} \end{cases}$$

- ▶ A constant-coefficient linear ODE has the form $\mathbf{y}' = \mathbf{A}\mathbf{y}$, with stability dependent on the real parts of the eigenvalues of \mathbf{A} :
 - ▶ *At a point (t, \mathbf{y}) , any ODE can be approximated by a linear ODE of the form $\mathbf{z}' = \mathbf{J}_f(t, \mathbf{y})\mathbf{z}$.*
 - ▶ *For general ODEs, stability can be ascertained locally by considering the eigenvalues of $\mathbf{J}_f(t, \mathbf{y})$.*

Numerical Solutions to ODEs

- ▶ Methods for numerical ODEs seek to approximate $\mathbf{y}(t)$ at $\{t_k\}_{k=1}^m$.
Compute \mathbf{y}_k for $k \in \{1, \dots, m\}$ so as to approximate $\mathbf{y}(t_k) \approx \mathbf{y}_k$. For an IVP, typically form \mathbf{y}_{k+1} using \mathbf{y}_k or additionally (for multistep methods) \mathbf{y}_{k-1}, \dots

- ▶ Euler's method provides the simplest method (attempt) for obtaining a numerical solution:

Approximation solution to ODE at $t_k + h$ by linear segment from (t_k, \mathbf{y}_k) with slope $\mathbf{f}(t_k, \mathbf{y}_k)$,

$$\mathbf{y}_{k+1} = \mathbf{y}_k + h_k \mathbf{f}(t_k, \mathbf{y}_k).$$

This approximation is the first order form of various models (Taylor series, finite differences, interpolation, quadrature, undetermined coefficients).

Error in Numerical Methods for ODEs

- ▶ Truncation error is typically the main quantity of interest, which can be defined *globally* or *locally*:

- ▶ *Global error is measured at all points*

$$e_k = \mathbf{y}_k - \mathbf{y}(t_k).$$

- ▶ *Local error measures the deviation from the exact solution $\mathbf{u}_{k-1}(t)$ passing through the previous point $(t_{k-1}, \mathbf{y}_{k-1})$,*

$$\mathbf{l}_k = \mathbf{y}_k - \mathbf{u}_{k-1}(t_k).$$

- ▶ The *order of accuracy* of a given method is one less than the order of the leading order term in the local error \mathbf{l}_k :

- ▶ *Accuracy is of order p if $\mathbf{l}_k = O(h_k^{p+1})$, for forward Euler $p = 1$ since*

$$\mathbf{y}(t_{k+1}) = \mathbf{y}(t_k) + h_k \mathbf{f}(t_k, \mathbf{y}(t_k)) + O(h_k^2),$$

$$\text{so } \mathbf{l}_k = O(h_k^2).$$

Accuracy and Taylor Series Methods

- ▶ By taking a degree- r Taylor expansion of the ODE in t , at each consecutive (t_k, \mathbf{y}_k) , we achieve r th order accuracy.

We can bound the local approximation error as the error in the Taylor expansion,

$$\mathbf{y}(t_k + h) = \mathbf{y}(t_k) + \mathbf{y}'(t_k)h + \cdots + \mathbf{y}^{(r)}(t_k)h^r/r!$$

which is $O(h^{r+1})$, leading to $O(h^r)$ accuracy in the approximation to $\mathbf{f}(t, \mathbf{y})$. Euler's method is a first-order Taylor series method.

- ▶ Taylor series methods require high-order derivatives at each step:
 - ▶ *Analytic differentiation is expensive, so seek to approximate.*
 - ▶ *Can perform a finite-differencing approximation by evaluating at points near (t_k, \mathbf{y}_k) (multi-stage methods) or simply using previous points, e.g. $(t_{k-1}, \mathbf{y}_{k-1})$ (multi-step methods).*

Growth Factors and Stability Regions

- ▶ Stability of an ODE method discerns whether local errors are amplified, deamplified, or stay constant:
 - ▶ A method is stable if the *growth factor* of the error is less than or equal to one.
 - ▶ The *stability region* for a method describes the step-size conditions necessary for stability in terms of
 - ▶ the step size h (assuming its constant) and
 - ▶ eigenvalues λ , usually as a function of $h\lambda$.
- ▶ Basic stability properties follow from analysis of linear scalar ODE, which serves as a local approximation to more complex ODEs.

- ▶ Consider forward Euler for the ODE $y' = \lambda y$, where

$$y_{k+1} = y_k + h\lambda y_k = \underbrace{(1 + h\lambda)}_{\text{growth factor}} y_k.$$

- ▶ Euler's method requires $|1 + h\lambda| \leq 1$ to be stable, which implies $-2 \leq h\lambda \leq 0$
- ▶ The global error then satisfies

$$e_k = l_k + (1 + h\lambda)e_{k-1}.$$

Stability Region for Forward Euler

- ▶ The stability region of a general ODE constrains the eigenvalues of $h\mathbf{J}_f$

- ▶ *The Mean Value Theorem implies that*

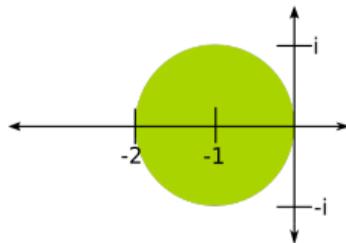
$$\exists \boldsymbol{\xi}, \text{ such that } \mathbf{f}(t_k, \mathbf{y}_k) - \mathbf{f}(t_k, \mathbf{y}(t_k)) = \mathbf{J}_f(t_k, \boldsymbol{\xi})(\mathbf{y}_k - \mathbf{y}(t_k))$$

and we know $\|\mathbf{J}_f(t_k, \boldsymbol{\xi})\|_2$ is bounded by Lipschitz continuity.

- ▶ *Consequently the growth factor for Forward Euler is $\mathbf{I} + h_k\mathbf{J}_f(t_k, \boldsymbol{\xi})$.*
 - ▶ *The global error then satisfies*

$$\mathbf{e}_k = (\mathbf{I} + h_k\mathbf{J}_f(t_k, \boldsymbol{\xi}))\mathbf{e}_{k-1} + \mathbf{l}_k.$$

- ▶ *Forward Euler is asymptotically stable if the spectral radius of the growth factor is less than one, which implies that the eigenvalues of $h_k\mathbf{J}_f(t_k, \boldsymbol{\xi})$ must always lie within a stability region that is a circle on the complex plane centered at -1 with radius 1.*



Backward Euler Method

Demo: Backward Euler stability

Activity: Backward Euler Method

- ▶ Implicit methods for ODEs form a sequence of solutions that satisfy conditions on a local approximation to the solution:

*The most basic implicit method is the **backward Euler** method*

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{h}_k \mathbf{f}(t_{k+1}, \mathbf{y}_{k+1}),$$

which solves for \mathbf{y}_{k+1} so that a linear approximation of the solution at t_{k+1} passes through the point (t_k, \mathbf{y}_k) . Just like forward Euler, first-order accuracy is achieved by the linear approximation.

- ▶ The stability region of the backward Euler method is the left half of the complex plane:

*Such a method is called **unconditionally stable**. Note that the growth factor can be derived via*

$$y_{k+1} = y_k + h\lambda y_{k+1} = \frac{1}{1 - h\lambda} y_k,$$

and satisfies $|1/(1 - h\lambda)| \leq 1$ so long as $h\lambda \leq 0$.

- ▶ *Stiff* ODEs are ones that contain components that vary at disparate time-scales:
 - ▶ *These are challenging since they require both high accuracy and stability*
 - ▶ *A linear ODE is stiff if the eigenvalues of A are disparate in magnitude*
 - ▶ *Explicit methods must use small step size h to ensure stability*
 - ▶ *Implicit methods are stable with any step size and hence effective for stiff ODEs*

Trapezoid Method

- ▶ A second-order accurate implicit method is the *trapezoid method*

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{h}_k(\mathbf{f}(t_k, \mathbf{y}_k) + \mathbf{f}(t_{k+1}, \mathbf{y}_{k+1}))/2,$$

- ▶ *This method takes the average of the backward and forward Euler steps.*
- ▶ *Its growth factor is $\frac{1+h\lambda/2}{1-h\lambda/2}$.*
- ▶ *Since $\left| \frac{1+h\lambda/2}{1-h\lambda/2} \right| \leq 1$ for any $\lambda < 0$, the method is unconditionally stable.*
- ▶ Generally, methods can be derived from quadrature rules:
 - ▶ *Evaluate or approximate f at a set of points near (t_k, \mathbf{y}_k) .*
 - ▶ *Use weights from a given quadrature rule to approximate solution to local integral equation.*
 - ▶ *Finding appropriate quadrature nodes is hard, implicit methods in effect solve for them.*

Multi-Stage Methods

- ▶ *Multi-stage methods* construct \mathbf{y}_{k+1} by approximating \mathbf{y} between t_k and t_{k+1} :
 - ▶ *Runge-Kutta methods* are the most well-known family of these, simple example is *Heun's method*,

$$\mathbf{y}_{k+1} = \mathbf{y}_k + h \left[\underbrace{\mathbf{f}(t_k, \mathbf{y}_k)}_{\mathbf{v}_1} / 2 + \mathbf{f} \left(t_k + h, \mathbf{y}_k + h \underbrace{\mathbf{f}(t_k, \mathbf{y}_k)}_{\mathbf{v}_1} \right) / 2 \right].$$

- ▶ We can think of the above method as employing the trapezoid quadrature rule.
- ▶ The difference between Heun's method and the (implicit) trapezoid method is that we evaluate at $\mathbf{f}(t_k + h, \mathbf{y}_k + h\mathbf{v}_1)$ rather than working with the implicit value of $\mathbf{f}(t_k + h, \mathbf{y}_{k+1})$.
- ▶ The 4th order Runge-Kutta scheme is particularly popular:
This scheme uses Simpson's rule,

$$\mathbf{y}_{k+1} = \mathbf{y}_k + (h/6)(\mathbf{v}_1 + 2\mathbf{v}_2 + 2\mathbf{v}_3 + \mathbf{v}_4)$$

$$\mathbf{v}_1 = \mathbf{f}(t_k, \mathbf{y}_k),$$

$$\mathbf{v}_2 = \mathbf{f}(t_k + h/2, \mathbf{y}_k + (h/2)\mathbf{v}_1),$$

$$\mathbf{v}_3 = \mathbf{f}(t_k + h/2, \mathbf{y}_k + (h/2)\mathbf{v}_2),$$

$$\mathbf{v}_4 = \mathbf{f}(t_k + h, \mathbf{y}_k + h\mathbf{v}_3).$$

Runge-Kutta Methods

- ▶ Runge-Kutta methods evaluate f at $t_k + c_i h$ for $c_0, \dots, c_r \in [0, 1]$,

$$\mathbf{u}_k(t_{k+1}) = \mathbf{y}_k + \int_{t_k}^{t_k+h} \mathbf{f}(s, \mathbf{y}(s)) ds \approx \mathbf{y}_k + h \sum_{i=0}^{r-1} w_i \mathbf{f}(t_k + c_i h, \hat{\mathbf{y}}_{ki}),$$

where $\{(c_i, w_i)\}_{i=0}^r$ are quadrature (node, weight) pairs.

- ▶ A general family of Runge Kutta methods can be defined by

$$\hat{\mathbf{y}}_{ki} = \mathbf{y}_k + h \sum_j a_{ij} \mathbf{f}(t_k + c_i h, \hat{\mathbf{y}}_{kj}).$$

Runge Kutta methods can then be represented by a **Butcher tableau**,

\mathbf{c}	\mathbf{A}	e.g. for RK4 \mathbf{A} has a single subdiagonal,	0				
	\mathbf{w}^T		1/2	1/2			
		1/2	0	1/2			
		1	0	0	1		
			1/6	1/3	1/3	1/6	

If \mathbf{A} is strictly lower triangular ($a_{ij} = 0$ for $j \geq i$), the scheme is explicit, if \mathbf{A} is lower-triangular then it is diagonally implicit, and otherwise implicit.

Multistep Methods

- ▶ *Multistep methods* employ $\{\mathbf{y}_i\}_{i=0}^k$ to compute \mathbf{y}_{k+1} :
Linear multistep methods have the form,

$$\mathbf{y}_{k+1} = \sum_{i=1}^m \alpha_i \mathbf{y}_{k+1-i} + h \sum_{i=0}^m \beta_i \mathbf{f}(t_{k+1-i}, \mathbf{y}_{k+1-i}).$$

Interpolation is used to determine each α_i and β_i , method is explicit if $\beta_0 = 0$.

- ▶ Multistep methods are not self-starting, but have practical advantages:
 - ▶ *Can be initiated by Runge-Kutta methods.*
 - ▶ *They require few function evaluations.*
 - ▶ *Generalize to non-uniformly-spaced points (*multistep methods*).*