

CS 450: Numerical Analysis¹

Initial Value Problems for Ordinary Differential Equations

↳ BVP

$t \rightarrow \text{time}$

University of Illinois at Urbana-Champaign

PDEs many partial derivatives in diff. dimensions

¹These 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 $y(t)$ that satisfies a set of equations in its derivatives.

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

explicit ODE $y^{(k)} = f(t, y, y', \dots, y^{(k-1)})$
 \uparrow
 k th derivative

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

$$u'(t) = f(t, u) \quad \left[\begin{array}{c} u_1' \\ u_2' \\ \vdots \\ u_k' \end{array} \right] = \left[\begin{array}{c} y' \\ y'' \\ \vdots \\ y^{(k-1)} \\ f(t, y, y', \dots, y^{(k-1)}) \end{array} \right] = \left[\begin{array}{c} u_2 \\ u_3 \\ \vdots \\ u_k \\ f(t, y, y', \dots, y^{(k-1)}) \end{array} \right]$$

$u_i = y^{(i-1)}$

Example: Newton's Second Law

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

$$\uparrow$$
$$y''(t)$$

$$F = m y''(t)$$

$$y''(t) = F/m$$

$$u'(t) = \begin{bmatrix} u_2 \\ F/m \end{bmatrix}$$

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

$$u'(t) = \begin{bmatrix} u_1' \\ u_2' \end{bmatrix} = \begin{bmatrix} y'(t) \\ F/m \end{bmatrix} = \begin{bmatrix} u_2 \\ F/m \end{bmatrix}$$

Initial Value Problems

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

IVP

y_0

$t \rightarrow$

$$y(0) = y_0$$

$$y'(0) = f(0, y_0)$$

f be linear, so $y'(t) = A(t)y$
↳ constant-coefficient

$$A(t) = A \text{ (constant)}$$

$$y'(t) = A y$$

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

$$y(t) = y_0 + \int_0^t f(s, y(s)) ds$$

$$y(t) = y_0 + \int_0^t A(s) y(s) ds$$

unknown

discretize s
↳ linear equations

$$y^{(t)} \approx y_0 + \sum_{i=1}^n \omega_i A(s_i) y(s_i)$$

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:

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

true for any differentiable f ↑
Lipschitz constant

$$L = \max_{t, y} \|J_f(t, y)\|$$

- The solutions of an ODE can be stable, unstable, or asymptotically stable:

↑ ↑ ↑
stay the same diverge converge
as $t \rightarrow \infty$

Stability of 1D ODEs

model problem

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

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

- ▶ A constant-coefficient linear ODE has the form $y' = Ay$, with stability dependent on the real parts of the eigenvalues of A :

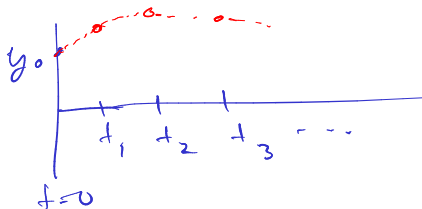
$\forall \operatorname{re}(\lambda_i) < 0 \mid \lambda_i = \text{eigenvalues of } A$

locally stability of a general ODE, eigenvalues

$$y'(t+h, y) \approx y(t, y) + J_f^T(t, y)h \quad \text{of} \quad J_f^T(t, y)$$

Numerical Solutions to ODEs

- Methods for numerical ODEs seek to approximate $\mathbf{y}(t)$ at $\{t_k\}_{k=1}^m$.



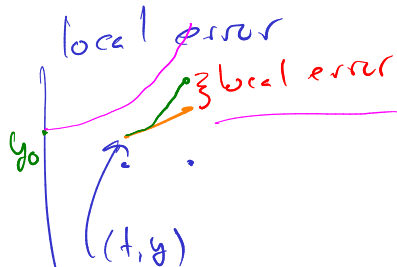
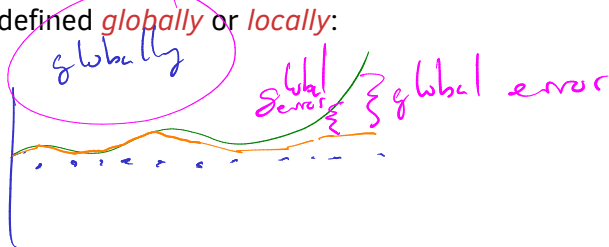
- Euler's method provides the simplest method (attempt) for obtaining a numerical solution:

Forward Euler

$$y_{k+1} \approx y(t_{k+1}) \approx y_k + h \underbrace{f(t_k, y_k)}_{y'(t_k, y_k)}$$

Error in Numerical Methods for ODEs

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



- The *order of accuracy* of a given method is one less than the order of the leading order term in l_k :

$$e_k(t_k, y_k) = \left[y_k + \int_{t_k}^{t_{k+1}} f(t, y) dt \right] - y_{k+1}$$

Forward Euler is a *first order accurate* method *locally*


e.g. Forward Euler

$$y_{k+1} = y_k + (t_{k+1} - t_k) f(t_k, y_k)$$

Convergence and Stability

- ▶ Generally, we seek to approximate $\mathbf{y}(t_k)$ for a set of points $t_k = t_0 + kh$ by $\hat{\mathbf{y}}_k$:
- ▶ Stability ascertains behavior as $t \rightarrow \infty$ either of the ODE itself or of a numerical method:

stiff: rapid oscillations locally that are small on a global scale



Stability of Numerical Methods for ODEs

- Stability can be defined for numerical methods similarly to ODEs themselves.

As $t \rightarrow \infty$ does $y_k \rightarrow \infty$?
 0 ?
 $\approx y_0$.

- Basic stability properties follow from analysis of linear scalar ODE, which serves as a local approximation to more complex ODEs.

$y' = \lambda y$
 $y_{k+1} = y_k + h \lambda y_k \Rightarrow y_k (1 + h \lambda)$
 $\lim_{k \rightarrow \infty} y_k = \begin{cases} 0 & \text{if } |1 + h \lambda| < 1 \\ y_0 & \text{if } |1 + h \lambda| = 1 \\ \infty & \text{if } |1 + h \lambda| > 1 \end{cases}$

in general
 all eigenvalues
 of eq. $y' = Ay$
 A in $\left| \frac{1}{h} \right|$
 to satisfy

asymptotic

Implicit Methods

Demo: Backward Euler stability

Demo: Stiffness

- ▶ Implicit methods for ODEs form a sequence of solutions that satisfy conditions on a local approximation to the solution:
- ▶ The *backward Euler method* for a simple linear scalar ODE stability region is the left half of the complex plane:

Accuracy and Taylor Series Methods

- ▶ By taking a degree- r Taylor expansion of the ODE in t , at each consecutive $(t_k, \hat{\mathbf{y}}_k)$, we achieve k th order accuracy.
- ▶ Taylor series methods require high-order derivatives at each step:

Multi-Stage Methods

- ▶ *Multi-stage methods* construct $\hat{\mathbf{y}}_{k+1}$ by approximating \mathbf{y} between t_k and t_{k+1} :
- ▶ The 4th order Runge-Kutta scheme is particularly popular:

Runge-Kutta Methods

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

Properties of Runge-Kutta and Extrapolation Methods

- ▶ Runge-Kutta methods are *self-starting*, but are harder to use to obtain error estimates.
- ▶ *Extrapolation methods* achieve high accuracy by successively reducing step-size.

Multistep Methods

- ▶ *Multistep methods* employ $\{\hat{\mathbf{y}}_k\}_{i=0}^k$ to compute $\hat{\mathbf{y}}_{k+1}$:
- ▶ Multistep methods are not self-starting, but have practical advantages: