CS 450: Numerical Analysis
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).
An **ordinary differential equation (ODE)** usually describes time-varying system by a function $y(t)$ that satisfies a set of equations in its derivatives. The general implicit form is

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

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

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

$$u' = \begin{bmatrix} u_1' \\ \vdots \\ u_{k-1}' \\ u_k' \end{bmatrix} = \begin{bmatrix} u_2 \\ \vdots \\ u_k \end{bmatrix} = f(t, u_1, \ldots, u_k) = \text{where } u_i(t) = y^{(i-1)}(t).$$

Consequently we restrict our focus to systems of first-order ODEs. Of particular importance are **linear ODEs**, which have the form $y' = A(t)y$, whose **coefficients are said to be constant** if $A(t) = A$ for all $t$. 
Example: Newton’s Second Law

\[ F = ma \] corresponds to a second order ODE

\[ F(t, y(t), y'(t)) = my''(t) \]
\[ y''(t) = f(t, y, y') = F(t, y(t), y'(t))/m \]

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

\[ u = \begin{bmatrix} y(t) \\ y'(t) \end{bmatrix} \]
\[ \begin{bmatrix} u'_1 \\ u'_2 \end{bmatrix} = u' = f(t, u) = \begin{bmatrix} u_2 \\ F(t, u)/m \end{bmatrix} \]
Generally, a first order ODE specifies only the derivative, so the solutions are non-unique, an \textit{initial condition} addresses this:

\[ y(t_0) = y_0 \]

\textit{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 \( t \):

\[ y(t) = y_0 + \int_{t_0}^{t} f(s, y(s)) \, ds \]

\textit{In the special case that} \( y' = f(t) \), \textit{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*:
  \[
  \forall y, \hat{y} \in D, \quad ||f(t, \hat{y}) - f(t, y)|| \leq L||\hat{y} - y||,
  \]
  i.e. the rate of change of the ODE should itself change continuously. Any differentiable function $f$ is Lipschitz continuous with
  \[
  L = \max_{(t,y) \in D} ||J_f(t, y)||.
  \]

- 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 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 $\lambda$:

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

- A linear ODE generally has the form $y' = Ay$, with stability dependent on the spectral radius (largest eigenvalue) of $A$:

For general ODEs, stability can be ascertained locally by considering a linear approximation $f(t, y) \approx y + J_f(t)^{-1} y$ and measuring the spectral radius of $J_f(t)^{-1}$. 
Numerical Solutions to ODEs

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

- 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 $(t_k, y_k)$ with slope $f(t_k, y_k)$,
  $$y_{k+1} = y_k + h_k f(t_k, y_k).$$

  It is instructive to observe that this approximation arises as the first order form of various models (Taylor series, finite differences, interpolation, quadrature, undetermined coefficients).
Truncation error is typically the main quantity of interest, which can be defined *globally* or *locally*

Global error is measured at all points

\[ e_k = \hat{y}_k - y(t_k), \]

which local error measures the deviation from the exact solution \( u_{k-1}(t) \) passing through the previous point \( (t_{k-1}, \hat{y}_{k-1}) \),

\[ l_k = \hat{y}_k - u_{k-1}(t_k). \]

The *order of accuracy* of Euler’s method is one less than the order of the leading order term in \( l_k \)

Accuracy is of order \( p \) if \( l_k = O(h_{k}^{p+1}) \). Euler’s method is first order accurate by Taylor expansion analysis.
Stability of Numerical Methods for ODEs

- Stability can be defined for numerical methods similarly to ODEs themselves. A method for an ODE with stable solutions can be unstable. Usually, we will seek to establish a stability region for a method, which describes the step-size conditions necessary for stability in terms of the step size $h$ and (the largest) eigenvalue $\lambda$, 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

$$y_{k+1} = y_k + h\lambda y_k = (1 + h\lambda)y_k.$$  

Euler’s method requires $|1 + h\lambda| \leq 1$ to be stable, which implies $-2 \leq h\lambda \leq 0$ for real $\lambda$. For a general ODE, the eigenvalues of $J_f$ must lie within a circle on the complex plane centered at $-1$ with radius 1.
Implicit Methods

- 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**

  \[ y_{k+1} = y_k + h f(t_{k+1}, y_{k+1}), \]

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

- The backward Euler method for a simple linear scalar ODE stability region 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 \( h \lambda \leq 0 \).
Initial Value Problems for ODEs

- We restrict attention to first-order systems of ODEs and pay special attention to linear and constant-coefficient systems:

  An IVP for an ODE usually has the form $y' = f(t, y)$ with initial value $(t_0, y(t_0))$. A linear ODE $f$ has the form $f(t, y) = A(t)y(t)$, and is said to have constant coefficients if $A(t)$ does not vary with $t$.

- Existence and uniqueness of a solution to an IVP is guaranteed over any domain $D$ on which $f$ is Lipschitz continuous:

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

  which is stronger than continuity of $f$ (differentiability of $y$), but weaker than differentiability of $f$, i.e. $f$ can suddenly begin to change at a different rate. The constant $L$ bounds the rate at which similar solutions $y$ and $\hat{y}$ can diverge or converge.
Convergence and Stability

- Generally, we seek to approximate $y(t_k)$ for a set of points $t_k = t_0 + kh$ by $\hat{y}_k$: Would like to converge with decreasing step-size, i.e. $\lim_{h \to 0} \hat{y}_k = y(t_k)$. Can measure global error (deviation from true solution) or local error (deviation of $\hat{y}_k$ from $u_{k-1}(t_k)$ where $u_{k-1}(t_{k-1}) = \hat{y}_{k-1}$ and $u'_{k-1} = y'$).

- Stability ascertains behavior as $t \to \infty$ either of the ODE itself or of a numerical method:
  - For an ODE, stability identifies convergence/divergence of perturbed solutions.
  - For a numerical method, it identifies step-size bounds necessary to ensure the method converges along with the ODE.
  - If a method is stable, the local error provides a trustworthy metric of global error.
  - A stiff ODE is a convergent ODE with rapidly varying small components, which poses a challenge for stability of methods.
Accuracy and Taylor Series Methods

- By taking a degree-$r$ Taylor expansion of the ODE in $t$, at each consecutive $(t_k, \hat{y}_k)$, we achieve $k$th order accuracy.
  
  We can bound the local approximation error as the error the Taylor expansion,

  $$ y(t_k + h) = y(t_k) + y'(t_k)h + \cdots + y^{(r)}(t_k)h^{r-1}/r! $$

  which is $O(h^{r+1})$, which leads to $O(h^r)$ accuracy in the approximation to $f(t, y)$. 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 $(t_k, \hat{y}_k)$ (multi-stage methods) or simply using previous points, e.g. $(t_{k-1}, \hat{y}_{k-1})$ (multi-step methods).
Multi-Stage Methods

Multi-stage methods construct $\hat{y}_{k+1}$ by approximating $y$ between $t_k$ and $t_{k+1}$:

Runge-Kutta methods are the most well-known family of these, simple example is Heun’s method,

$$\hat{y}_{k+1} = \hat{y}_{k} + h \left[ \frac{f(t_k, y_k)}{2} + f(t_k + h, y_k + h \frac{f(t_k, y_k)}{v_1}) \right]/2 \right].$$

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 $f(t_k + h, y_k + hv_1)$ rather than working with the implicit value of $f(t_k + h, y_{k+1})$.

The 4th order Runge-Kutta scheme is particularly popular:

This scheme uses Simpson’s rule

$$\hat{y}_{k+1} = \hat{y}_k + (h/6)(v_1 + 2v_2 + 2v_3 + v_4)$$

$$v_1 = f(t_k, y_k), \quad v_2 = f(t_k + h/2, y_k + (h/2)v_1),$$

$$v_3 = f(t_k + h/2, y_k + (h/2)v_2), \quad v_4 = f(t_k + h, y_k + hv_3)$$
Runge-Kutta Methods

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

A Runge-Kutta method can be derived as a quadrature rule

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

where $\{(c_i, w_i)\}_{i=0}^r$ are quadrature node, weight pairs, but we still have flexibility in choosing $\hat{y}_{k,i}$. One good choice is to successively construct

\[
\hat{y}_{k,i} = \hat{y}_k + h \sum_j a_{ij} f(t_k + c_i h, \hat{y}_{k,j}) = \hat{y}_k + h c_i f(t_k + c_i h, \hat{y}_{k,i-1}).
\]

More general choices for $a_{ij}$ are often represented by a Butcher tableau,

<table>
<thead>
<tr>
<th>(c)</th>
<th>A</th>
<th>(w^T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1/2</td>
<td>0</td>
</tr>
<tr>
<td>1/2</td>
<td>1/6</td>
<td>1/3</td>
</tr>
</tbody>
</table>

\[
\text{e.g. for RK4,}
\]

If $a_{ij} = 0$ for $j \geq i$, the scheme is explicit, if for $j > i$ then diagonally implicit, and otherwise implicit.
Properties of Runge-Kutta and Extrapolation Methods

- Runge-Kutta methods are self-starting, but are harder to use to obtain error estimates. 

  Self-starting means that we only need $\hat{y}_k$ to form $\hat{y}_{k+1}$. Embedded Runge-Kutta schemes provides 4th + 5th order results, yielding an error estimate.

- Extrapolation methods achieve high accuracy by successively reducing step-size.

  Use single-step method with step sizes $h, h/2, h/4, ...$ to approximate solution at $t_k + h$. 
Multistep Methods

- Multistep methods employ \( \{ \hat{y}_k \}_{i=0}^k \) to compute \( \hat{y}_{k+1} \):
  
  *Linear multistep methods have the form,*

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

  *Interpolation is used to determine each \( \alpha_i \) and \( \beta_i \), method is explicit if \( \beta_0 = 0 \).*

- Multistep methods are not self-starting

  *However, they require few function evaluations. Multivalue methods generalize multistep methods to non-uniformly-spaced points.*