Ordinary Differential Equation (ODE) Solvers

James Rosado

Mathematics Department
Temple University

Applied Mathematics Fall 2017
Outline

1. ODE Basics
   - What is an ODE?
   - Example ODE
   - Types of ODEs
   - Stability and Stiffness

2. ODE Solvers
   - What is a solver?
   - An example
   - Which solver and when?
Outline

1. ODE Basics
   - What is an ODE?
   - Example ODE
   - Types of ODEs
   - Stability and Stiffness

2. ODE Solvers
   - What is a solver?
   - An example
   - Which solver and when?
What is an ODE? Example ODE Types of ODEs Stability and

A general $n$-th order ODE with constant coefficients is of the form:

$$a_n \frac{d^n y}{dx^n} + a_{n-1} \frac{d^{n-1} y}{dx^{n-1}} + \cdots + a_2 \frac{d^2 y}{dx^2} + a_1 \frac{dy}{dx} = f(x, y), \quad (1)$$

and will have $n$ initial conditions

$$y(x_0) = y_0, y'(x_0) = y_1, \cdots, y^{(n)}(x_0) = y_n. \quad$$

We consider first-order ODEs [4] of the form

$$y' = \frac{dy}{dx} = f(x, y) \text{ with } y(x_0) = y_0, \quad (2)$$

where $y$ is a vector of dimension $d$. 

James Rosado
Ordinary Differential Equation (ODE) Solvers
Since $y = \{y_1(x), y_2(x), \ldots, y_d(x)\}$ is a vector then we have a set of first-order equations

$$
\begin{align*}
\begin{pmatrix}
 y_1'(x) \\
 y_2'(x) \\
 \vdots \\
 y_d'(x)
\end{pmatrix} &= 
\begin{pmatrix}
 f_1(x, y_1, y_2, \ldots, y_d) \\
 f_2(x, y_1, y_2, \ldots, y_d) \\
 \vdots \\
 f_d(x, y_1, y_2, \ldots, y_d)
\end{pmatrix}.
\end{align*}
$$

High order ODE problems like (1) can always be reduced to a set of first-order ODEs.
What? & Why?

- Arise in physics, astronomy to model celestial mechanics and motions of particles.
- Used in chemistry to model reaction rates.
- In biology ODEs are needed to determine the rate of spread for various infectious diseases.
- ODEs are also found in economics to measure interest rates and predict stock trends.
- Integrate & Fire model, Hodgkin-Huxley model,...for describing the action potential across a neuron membrane.

ODEs model many natural phenomena to predict changes, but it is very difficult to solve (if not impossible) these ODEs analytically.
Let $y_1(t) =$ population of predator at time $t$ and $y_2(t) =$ population of prey at time $t$.

Population growth rates are related (Lotka-Volterra [7]) as follows

$$y_1'(t) = \alpha y_1 + \beta y_1 y_2,$$

$$y_2'(t) = \delta y_1 y_2 - \gamma y_2;$$

with initial populations $P_1 = y_1(t_0)$ and $P_2 = y_2(t_0)$.

This set of equations can be written in the form (2)

$$y'(t) = \begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} f_1(t, y_1, y_2) \\ f_2(t, y_1, y_2) \end{pmatrix} = f(t, y).$$
Predator-Prey

The graph illustrates the predator-prey model. The green line represents the prey (baboons) population, and the black line represents the predators (cheetahs) population. The graph shows the fluctuations in the number of prey and predators over time.

The x-axis represents time (dimensionless), and the y-axis represents the number of prey and predators. The peak and troughs indicate the dynamic interaction between the two species.
Neuronal Cell Membrane Potential

Hodgkin-Huxley [6] model:

\[ C_m \frac{dV_m}{dt} = \sum_j I_j(t, V_m), \]

where \( V_m(t) \) = the membrane potential and \( I_j(t, V_m) \) = is the current induced by the flow of ions traversing the membrane.

Each ion induces a current:

\[ I_{Na^+}(t, V_m) = g_{Na^+}(V_m(t) - V_{eq}), \]
\[ I_{K^+}(t, V_m) = g_{K^+}(V_m(t) - V_{eq}), \]

and \( g_X \) = conductance for particular ion, \( V_{eq} \) = resting potential of the membrane.
(IVP) Initial value problems, ODE with initial values like (2).
Types of ODEs

- (IVP) Initial value problems, ODE with initial values like (2).
- (BVP) Boundary value problems, ODE with values of derivatives and function given at multiple points

Example (BVP)

ODE: \( y''(x) + y(x) = 0 \) for \( x \in [0, \pi/2] \) with \( y(0) = y(\pi/2) = 2 \)
Types of ODEs

- **(IVP)** Initial value problems, ODE with initial values like (2).
- **(BVP)** Boundary value problems, ODE with values of derivatives and function given at multiple points

**Example (BVP)**

ODE: \( y''(x) + y(x) = 0 \) for \( x \in [0, \pi/2] \) with \( y(0) = y(\pi/2) = 2 \)

- **Stiff** ODEs are numerically *unstable* for certain numerical algorithms...
- **Non-stiff** ODEs have numerical solutions that are easily obtained using *classical* methods.
An ODE has what are called *equilibrium* points.

**Figure:** Asymp. Stable: 
\[
\frac{dy}{dx} = y^2 - y - 6.
\]
An ODE has what are called *equilibrium* points.

This is where the rates of change are 0.

**Figure:** Asymp. Stable: \[ \frac{dy}{dx} = y^2 - y - 6. \]
Stability

- An ODE has what are called *equilibrium* points.
- This is where the rates of change are 0.
- These are constant solutions of the ODE.

**Figure:** Asymp. Stable:

$$\frac{dy}{dx} = y^2 - y - 6.$$
Stability

- An ODE has what are called *equilibrium* points.
- This is where the rates of change are 0.
- These are constant solutions of the ODE.
- ODE is unstable if small moves away from equilibrium points yields significant changes in the rate of change.

**Figure:** Asympt. Stable:
\[ \frac{dy}{dx} = y^2 - y - 6. \]
Recall (2) for first order ODEs: \( y' = f(x, y) \).

- To understand ODE behavior examine regions near equilibrium points.
Stability: Jacobian

Recall (2) for first order ODEs: \( y' = f(x, y) \).

- To understand ODE behavior examine regions near equilibrium points.
- If \( y' = f(x_0, y_0) = 0 \), then \((x_0, y_0)\) is an equilibrium point.
Stability: Jacobian

Recall (2) for first order ODEs: \( y' = f(x, y) \).

- To understand ODE behavior examine regions near equilibrium points.
- If \( y' = f(x_0, y_0) = 0 \), then \((x_0, y_0)\) is an equilibrium point.
- Re-write the system as a linearization near \((x_0, y_0)\):

\[
\begin{pmatrix} y'_1 \\ y'_2 \end{pmatrix} = \begin{pmatrix} f_1(x, y_1, y_2) \\ f_2(x, y_1, y_2) \end{pmatrix} \approx \begin{pmatrix} u'_1 \\ u'_2 \end{pmatrix} = J_0 \begin{pmatrix} u_1 \\ u_2 \end{pmatrix},
\]

where \( J_0 \) is the Jacobian [1]:

\[
\begin{pmatrix} \frac{\partial f_1}{\partial x}(x_0, y_0) & \frac{\partial f_1}{\partial y}(x_0, y_0) \\ \frac{\partial f_2}{\partial x}(x_0, y_0) & \frac{\partial f_2}{\partial y}(x_0, y_0) \end{pmatrix}.
\]
Recall (2) for first order ODEs: \( y' = f(x, y) \).

- To understand ODE behavior examine regions near equilibrium points.
- If \( y' = f(x_0, y_0) = 0 \), then \((x_0, y_0)\) is an equilibrium point.
- Re-write the system as a linearization near \((x_0, y_0)\):

\[
\begin{pmatrix}
y'_1 \\
y'_2
\end{pmatrix} = \begin{pmatrix}
f_1(x, y_1, y_2) \\
f_2(x, y_1, y_2)
\end{pmatrix} \approx \begin{pmatrix}
u'_1 \\
u'_2
\end{pmatrix} = J_0 \begin{pmatrix}
u_1 \\
u_2
\end{pmatrix},
\]

where \( J_0 \) is the Jacobian [1]:

\[
\begin{pmatrix}
\frac{\partial f_1}{\partial x}(x_0, y_0) & \frac{\partial f_1}{\partial y}(x_0, y_0) \\
\frac{\partial f_2}{\partial x}(x_0, y_0) & \frac{\partial f_2}{\partial y}(x_0, y_0)
\end{pmatrix}.
\]

- If all spectral values of \( J_0 \) have negative real part, then stable near equilibrium point [8].
Recall (2) for first order ODEs: \( y' = f(x, y) \).

- To understand ODE behavior examine regions near equilibrium points.
- If \( y' = f(x_0, y_0) = 0 \), then \((x_0, y_0)\) is an equilibrium point.
- Re-write the system as a linearization near \((x_0, y_0)\):

\[
\begin{pmatrix}
y'_1 \\
y'_2
\end{pmatrix} =
\begin{pmatrix}
f_1(x, y_1, y_2) \\
f_2(x, y_1, y_2)
\end{pmatrix} \approx
\begin{pmatrix}
u'_1 \\
u'_2
\end{pmatrix} = J_0 \begin{pmatrix} u_1 \\ u_2 \end{pmatrix},
\]

where \( J_0 \) is the Jacobian [1]:

\[
\begin{pmatrix}
\frac{\partial f_1}{\partial x}(x_0, y_0) & \frac{\partial f_1}{\partial y}(x_0, y_0) \\
\frac{\partial f_2}{\partial x}(x_0, y_0) & \frac{\partial f_2}{\partial y}(x_0, y_0)
\end{pmatrix}.
\]

- If all spectral values of \( J_0 \) have negative real part, then stable near equilibrium point [8].
- If one of spectral value has positive real part, then equilibrium point is unstable [8].
There is no concise definition of *stiffness*. 
ODE Stiffness

- There is no concise definition of \textit{stiffness}.
- Depends on the \textit{stability} of an ODE with respect to the numerical solver.
ODE Stiffness

- There is no concise definition of *stiffness*.
- Depends on the *stability* of an ODE with respect to the numerical solver.
- If step size greatly affects stability for a given solver, then ODE is stiff [4, 5].

Rapidly changing terms in the ODE.
The Jacobian of an ODE can give insight into the stiffness of the ODE.

Definition (Curtiss & Hirschfelder [3])
Initial value problems are stiff if they are (exceedingly) difficult to solve by ordinary, explicit step-by-step methods, whereas certain implicit methods perform quite well.
ODE Stiffness

- There is no concise definition of *stiffness*.
- Depends on the *stability* of an ODE with respect to the numerical solver.
- If step size greatly affects stability for a given solver, then ODE is stiff [4, 5].
- Rapidly changing terms in the ODE.
ODE Stiffness

- There is no concise definition of *stiffness*.
- Depends on the *stability* of an ODE with respect to the numerical solver.
- If step size greatly affects stability for a given solver, then ODE is stiff [4, 5].
- Rapidly changing terms in the ODE.
- The Jacobian of an ODE can give insight into the stiffness of the ODE.
ODE Stiffness

- There is no concise definition of *stiffness*.
- Depends on the *stability* of an ODE with respect to the numerical solver.
- If step size greatly affects stability for a given solver, then ODE is stiff [4, 5].
- Rapidly changing terms in the ODE.
- The Jacobian of an ODE can give insight into the stiffness of the ODE.

**Definition (Curtiss & Hirschfelder [3])**

Initial value problems are stiff if they are (exceedingly) difficult to solve by ordinary, explicit step-by-step methods, whereas certain implicit methods perform quite well.
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \textit{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.$$

- $\Delta x$ too big leads to large variation in slopes [3].

\begin{figure}
\centering
\includegraphics[width=\textwidth]{stiff_ode.png}
\caption{A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].}
\end{figure}
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- \( \Delta x = \text{resolution}, \text{ step size, mesh size for numerical approx.} \)
- From (2) let \( f(x, y) = \frac{y - G(x)}{a(x, y)} \)
  \[
  \left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}
  \]
- \( \Delta x \) too big leads to large variation in slopes [3].

**Figure:** A stiff ODE \( \frac{dy}{dx} = 5(y - x^2) \) from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution, step size, mesh size for numerical approx.}$
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$
  \[ \left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}. \]
- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y-G(x)}{a(x,y)}$

$$\left| \frac{a(x,y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}$$

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.$$ 

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \textit{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$.
  \[
  \left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.
  \]
- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y-G(x)}{a(x,y)}$

$$\left|\frac{a(x,y)}{\Delta x}\right| \ll 1 \Rightarrow \text{stiff.}$$

- $\Delta x$ too big leads to large variation in slopes [3].

**Figure:** A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \textit{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.$$

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}$$

- $\Delta x$ too big leads to large variation in slopes [3].

**Figure:** A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \implies \text{stiff}.$$

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y-G(x)}{a(x,y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.$$  

- $\Delta x$ too big leads to large variation in slopes [3].

**Figure:** A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
**ODE Stiffness**

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}$$

- $\Delta x$ too big leads to large variation in slopes [3].

*Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].*
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

\[
\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}
\]

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y-G(x)}{a(x,y)}$
  \[
  \left| \frac{a(x,y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.
  \]
- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = resolution$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$

$$\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}$$

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y-G(x)}{a(x,y)}$

$$\left| \frac{a(x,y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.$$

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \textit{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x,y)}$

\[
\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.
\]

- $\Delta x$ too big leads to large variation in slopes [3].

Figure: A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Basics  ODE Solvers  References

ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- $\Delta x = \text{resolution}$, step size, mesh size for numerical approx.
- From (2) let $f(x, y) = \frac{y - G(x)}{a(x, y)}$
  \[
  \left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff}.
  \]
- $\Delta x$ too big leads to large variation in slopes [3].

**Figure:** A stiff ODE $\frac{dy}{dx} = 5(y - x^2)$ from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- \( \Delta x = resolution \), step size, mesh size for numerical approx.
- From (2) let \( f(x, y) = \frac{y - G(x)}{a(x, y)} \)

\[
\left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \Rightarrow \text{stiff.}
\]

- \( \Delta x \) too big leads to large variation in slopes [3].

**Figure:** A stiff ODE \( \frac{dy}{dx} = 5(y - x^2) \) from [3].
ODE Stiffness

- Rapidly changing slopes positive/negative away from stable solution.
- \( \Delta x = \text{resolution}, \) step size, mesh size for numerical approx.
- From (2) let \( f(x, y) = \frac{y - G(x)}{a(x, y)} \)
  \[
  \left| \frac{a(x, y)}{\Delta x} \right| \ll 1 \implies \text{stiff.}
  \]
- \( \Delta x \) too big leads to large variation in slopes [3].

**Figure:** A stiff ODE \( \frac{dy}{dx} = 5(y - x^2) \) from [3].
Outline

1. ODE Basics
   - What is an ODE?
   - Example ODE
   - Types of ODEs
   - Stability and Stiffness

2. ODE Solvers
   - What is a solver?
   - An example
   - Which solver and when?
An ODE solver...

- Also called *numerical integration*. 
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots\} \) of numerical solutions *converge*.
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots\} \) of numerical solutions *converge*.
- You need a *step size*, \( \Delta x \), (also called resolution).
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots\} \) of numerical solutions *converge*.
- You need a *step size*, \( \Delta x \), (also called resolution).
- Begin with a start point \( x_0 \) *near* the solution endpoint, then \( x_1 = x_0 + \Delta x \).
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots\} \) of numerical solutions *converge*.
- You need a *step size*, \( \Delta x \), (also called resolution).
- Begin with a start point \( x_0 \) near the solution endpoint, then \( x_1 = x_0 + \Delta x \).
- Algorithm uses \( x_0, x_1, y_0 \) to generate \( y_1 \), to get \( y_2 \) repeat algorithm with \( x_1, x_2, y_1 \) and so on...
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots \} \) of numerical solutions *converge*.
- You need a *step size*, \( \Delta x \), (also called resolution).
- Begin with a start point \( x_0 \) *near* the solution endpoint, then \( x_1 = x_0 + \Delta x \).
- Algorithm uses \( x_0, x_1, y_0 \) to generate \( y_1 \), to get \( y_2 \) repeat algorithm with \( x_1, x_2, y_1 \) and so on...
- Each \( y_j \) required an iteration of algorithm.
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \( \{y_0, y_1, \ldots\} \) of numerical solutions *converge*.
- You need a *step size*, \( \Delta x \), (also called resolution).
- Begin with a start point \( x_0 \) *near* the solution endpoint, then \( x_1 = x_0 + \Delta x \).
- Algorithm uses \( x_0, x_1, y_0 \) to generate \( y_1 \), to get \( y_2 \) repeat algorithm with \( x_1, x_2, y_1 \) and so on...
- Each \( y_j \) required an iteration of algorithm.
- Problem: each iteration has a “cost”, the number of computations!
An ODE solver...

- Also called *numerical integration*.
- Algorithm that computes a sequence of approximate solutions.
- Hopefully the sequence \(\{y_0, y_1, \ldots\}\) of numerical solutions converge.
- You need a step size, \(\Delta x\), (also called resolution).
- Begin with a start point \(x_0\) near the solution endpoint, then \(x_1 = x_0 + \Delta x\).
- Algorithm uses \(x_0, x_1, y_0\) to generate \(y_1\), to get \(y_2\) repeat algorithm with \(x_1, x_2, y_1\) and so on...
- Each \(y_j\) required an iteration of algorithm.
- Problem: each iteration has a “cost”, the number of computations!
- Smaller step size = better resolution, but more computations per iteration = more time.
## Common ODE Solvers

### Stiff ODEs
- Implicit Runge-Kutta Methods (RKF)
- Linear Multistep Methods (LMF)
- Backward Differentiation Formulas (BDF)
- Extrapolation Methods, trapezoidal rule
- Second Derivative Multistep Formulas
- Blended Linear Multistep Methods

### Non-Stiff ODEs
- Classical RKFs: Euler, Backward Euler
- Runge-Kutta-Fehlberg
- LMF: Adam’s formulas
- Richardson Extrapolation Method
- Taylor Series
- Multi-rate Methods
The ODE: \( y'(x) = -15y(x) \) with \( y(0) = 1 \)

\[
\frac{dy}{dx} = -15y \Rightarrow \frac{dy}{y} = -15 \, dx \Rightarrow \int \frac{dy}{y} = \int -15 \, dx \Rightarrow \ln(y) = -15x + C
\]

\( \Rightarrow y(x) = y_0 e^{-15x} \)

Figure: Solution curve: \( y(x) = e^{-15x} \)
The ODE: \( y'(x) = -15y(x) \) with \( y(0) = 1 \)

Figure: Euler with \( \Delta x = 0.25 \)

With step size \( \Delta x = 0.25 \) over \([0, 1]\) we have points \( \{0, 0.25, 0.5, 0.75, 1\} \).

\[
y_{n+1} = y_n + \Delta x \cdot f(x_n, y_n) = y_n + \Delta x \cdot (-15y_n)
\]

There is extreme variation among approximations.
The ODE: \( y'(x) = -15y(x) \) with \( y(0) = 1 \)

**Figure:** Euler with \( \Delta x = 0.125 \)

With step size \( \Delta x = 0.125 \) over \([0, 1]\) we have points \( \{0, 0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875, 1\} \).

We have a closer approximation, with less variation, because of smaller (half) step size = twice as many iterations.
The ODE: $y'(x) = -15y(x)$ with $y(0) = 1$

Figure: Adam’s Method

Here, we let $\Delta x = 0.25$, with points $\{0, 0.25, 0.5, 0.75, 1\}$:

$$y_{n+1} = y_n + \frac{1}{2} \Delta x \cdot (f(x_n, y_n) + f(x_{n+1}, y_{n+1})) .$$

We achieve an even closer approximation with only 4 iterations, instead of 8 with Euler.
The ODE: \( y'(x) = -15y(x) \) with \( y(0) = 1 \)

This example was an example of a stiff ODE.

Notice for Euler method:

\[
y_{n+1} = y_n(1 - 15 \Delta x) = y_{n-1}(1 - 15 \Delta x)^2 = \cdots = (1 - 15 \Delta x)^k y_0,
\]

where \((1 - 15 \Delta x)^k\) grows geometrically if \(\Delta x\) is too large.
Best solvers to use.

Runge-Kutta Formulas (RKF): Euler, Backward Euler...
- Explicit ODEs, no implicitly defined equations.
- Pair up RKF (known as RK-Fahlberg).
- Cost of iteration $< 5\sqrt{\text{no. computations in equation}}$.

Linear Multistep Formulas (LMF)
- Based on Adam’s formula, like trapezoid rule and Simpson’s rule.
- Well suited for applications with many output points.
- Most efficient when cost of iteration $> 50\sqrt{\text{no. computations in equation}}$
- Closer approximations in fewer iterations.
Implicit Methods

Definition (Adams-Moulton Four-Step [2])

Given: $y_0, y_1, y_2, y_3$ then

$$y_{n+1} = y_n + \frac{\Delta x}{720} \left( f(x_{n+1}, y_{n+1}) + 646f(x_n, y_n) - 264f(x_{n-1}, y_{n-1}) + 106f(x_{n-2}, y_{n-2}) - 19f(x_{n-3}, y_{n-3}) \right).$$

Yields better approximations for implicitly defined ODEs, but it may not be possible to explicitly obtain $y_{n+1}$ at each iteration.

With any method there are local and global errors that occur.

Local Errors: truncation error after one iteration of the algorithm.

Global Error: truncation error after a complete run of algorithm.
Considerations for choosing a solver...

- Number of calculations per iteration, degree of ODE.
- Type of stability of the ODE.
- Stiff vs. Non-stiff.
- Step size $\Delta x$, resolution.
- Explicitly or Implicitly defined ODEs.
- We have yet to consider ODEs with \textit{non-constant} coefficients.
References


