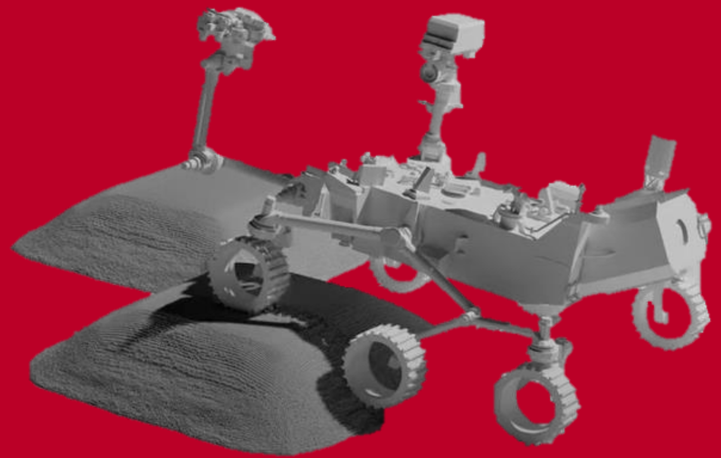
Carnegie Mellon University



Introductory Numerical Methods for Simulating Dynamic Systems

Dr. Joshua Pulsipher

Learning Outcomes

1. The relative advantages/disadvantages of using **explicit and implicit Euler methods**

2. How to implement explicit/implicit Euler to **simulate ODEs** using common computation environments (e.g., Julia)

3. How to simulate **reaction networks** using numerical methods

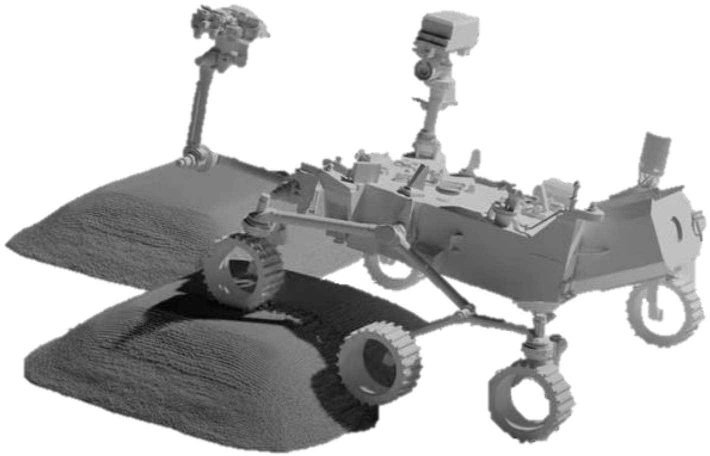
4. A familiarity of other **numerical methods/tools** for simulating ODEs

Outline

- Motivation
- Explicit Euler
- Implicit Euler
- Reaction Networks
- Other Methods

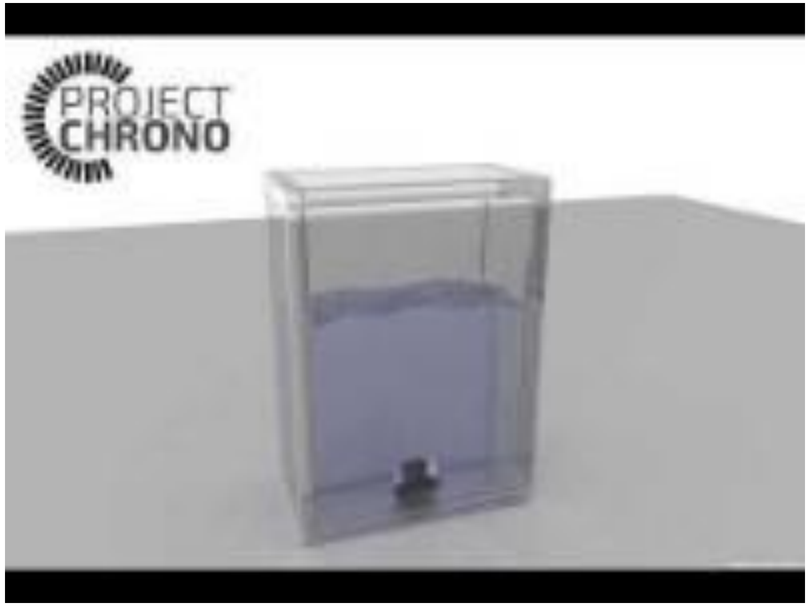
Outline

- **Motivation**
- Explicit Euler
- Implicit Euler
- Reaction Networks
- Other Methods



Simulating Dynamic Systems

- Simulating dynamic systems is vital for enabling engineering applications



- Simulate using **numerical methods to approximate dynamics** (e.g., differential equations)

- Enables us to **computationally experiment** and implement **automation**

Differential Equations

Types

- **Ordinary differential equations** (ODEs)
  - Today's focus

$$\frac{dy(t)}{dt} = f(y(t), t)$$
$$y(0) = y_0$$

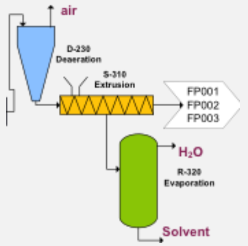
- Partial differential equations (PDEs)

$$\frac{\partial y_c(t,x)}{\partial t} = \xi(x)\left(\frac{\partial^2 y_c(t,x)}{\partial x_1^2} + \frac{\partial^2 y_c(t,x)}{\partial x_2^2}\right) + y_g(t,x)$$
$$y_c(0,x), y_c(t, \text{boundary}) = 0$$

Applications

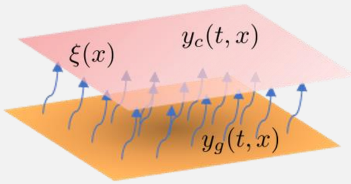
- Transient flow balance

$$\frac{df(t)}{dt} = f_{in}(t) - f_{out}(t) + f_{gen}(t)$$



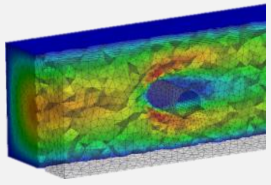
- Heat/mass transfer

$$\frac{\partial T(t)}{\partial t} - \alpha\frac{\partial^2 T(t)}{\partial x^2} = 0$$



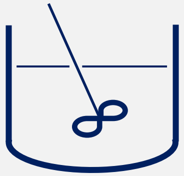
- Fluid flow

$$\rho\frac{DV(t,x)}{Dt} = -\nabla p + \rho g(x) + \mu\nabla^2 V(t,x)$$



- Kinetics

$$\frac{dc(t)}{dt} = kc(t)^{\alpha}$$



Analytical vs. Numerical Methods

Analytical Methods

- Separate and integrate

$$g(y)\frac{dy}{dx} = h(x) \quad \Rightarrow \quad \int g(y)dy = \int h(x)dx + C$$

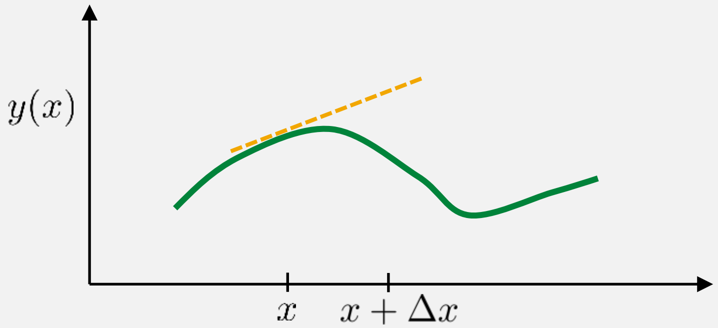
- ODEs of special forms

$$\frac{dy}{dx} = \frac{x+y}{x-y} \quad \Rightarrow \quad \frac{1}{2}\log\left(\frac{y^2}{x^2}+1\right) - \tan^{-1}\left(\frac{y}{x}\right) = C - \log(x)$$

- Solving general ODEs is often difficult or not possible

Numerical Methods

- Seek to numerically approximate the solution
- **Finite difference methods** are common



- More advanced methods are available
  - Not the focus of today

My Teaching Philosophy

Idea: Promote a **tutorial-like** format that encourages active engagement.

Active Learning

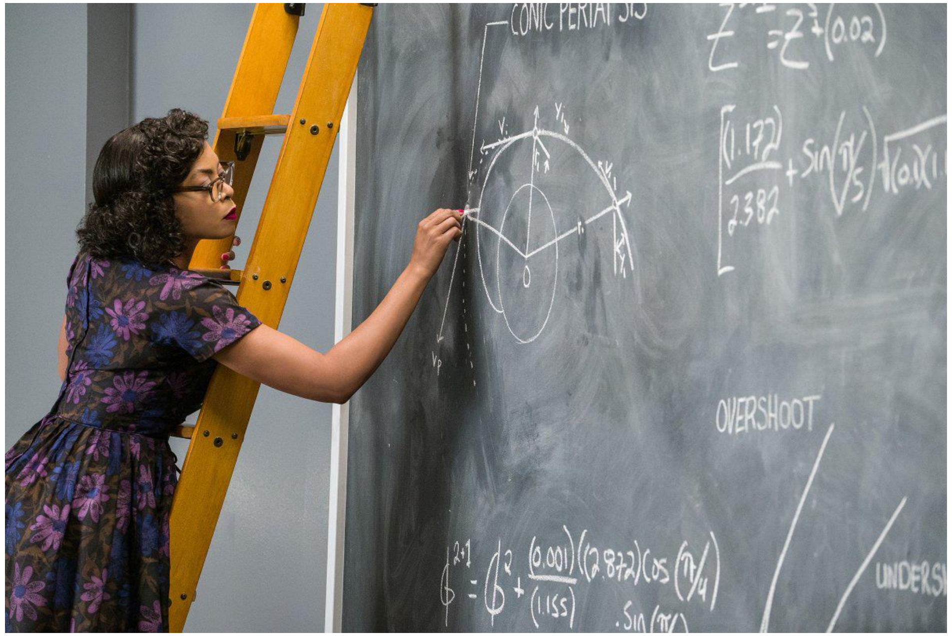
- Mastery comes through **deliberate practice**
- Magnify class time to gain **guided hands-on experiences**

In-Class Exercises

- We will be using Jupyter notebooks with a Julia and/or a Python kernel today
- No downloads/installation are/is needed
- https://pulsipher.info/teaching/courses.html

Outline

- Motivation
- **Explicit Euler**
- Implicit Euler
- Reaction Networks
- Other Methods



The Basics

Methodology

- Consider a 1st order ODE

$$\frac{dy(t)}{dt} = f(y(t), t)$$
$$y(0) = y_0$$

- Define time steps Δt

$$t \in [t_0, t_f] \qquad t_k = t_0 + k\Delta t$$

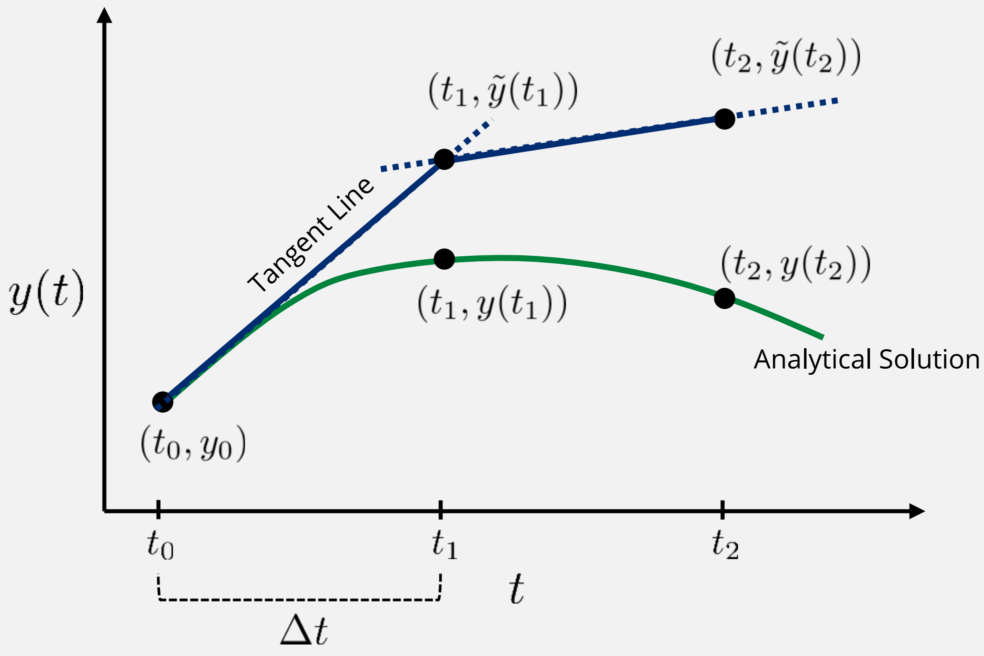
- Approximate derivative as finite difference

$$\left.\frac{dy(t)}{dt}\right|_{t_k} \approx \frac{\tilde{y}(t_{k+1}) - \tilde{y}(t_k)}{\Delta t}$$

- Define update rule

$$\tilde{y}(t_{k+1}) = \tilde{y}(t_k) + f(y(t_k), t_k)\Delta t$$

Illustration



Exercise 1

Problem Setup

- Solve the ODE

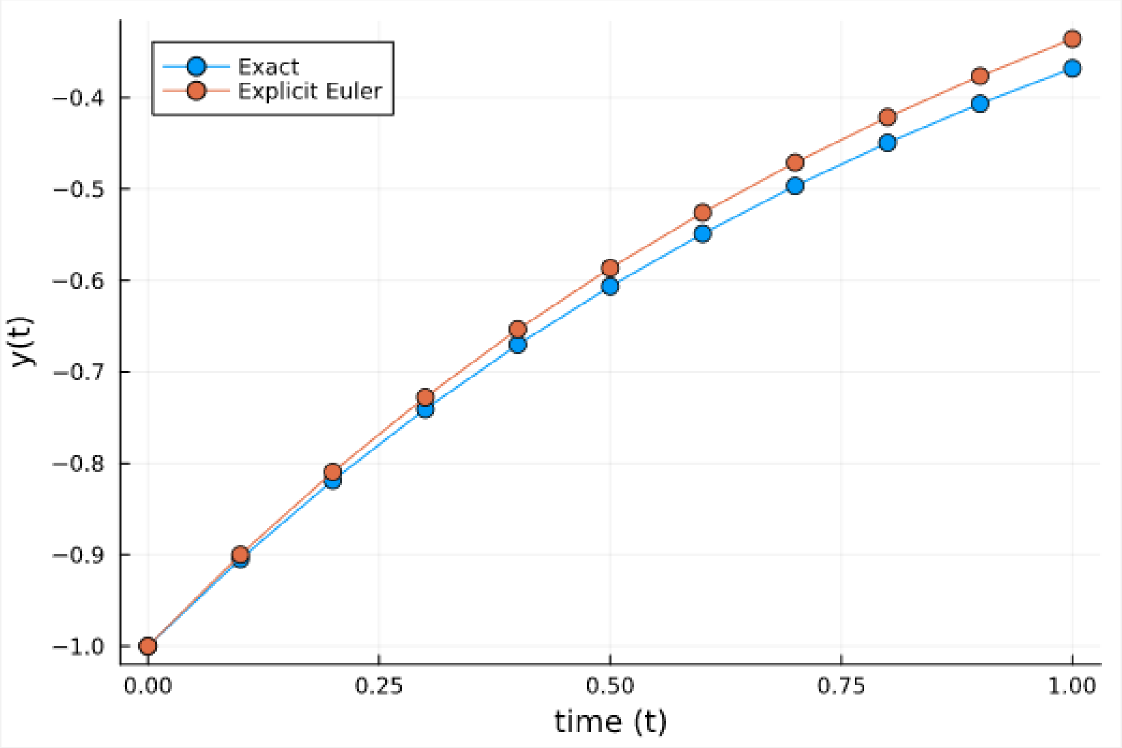
$$\frac{dy(t)}{dt} = e^{-t}$$
$$y(0) = -1$$

- Specifications
  - $t \in [0, 1]$
  - $\Delta t = 0.1$

- Plot the result against the analytical answer

- **Bonus:** Experiment with varied Δt

Expected Answer



Simulating a System of ODEs

System of 1st order ODEs

- General representation

$$\frac{dy_1(t)}{dt} = f_1(y_1, y_2, \ldots, y_n)$$
$$\frac{dy_2(t)}{dt} = f_2(y_1, y_2, \ldots, y_n)$$
$$\vdots$$
$$\frac{dy_n(t)}{dt} = f_n(y_1, y_2, \ldots, y_n)$$
$$y_1(0) = y_{1,0}, y_2(0) = y_{2,0}, \ldots, y_n(0) = y_{n,0}$$

- Vectorize

$$\frac{d\mathbf{y}(t)}{dt} = \mathbf{f}(\mathbf{y}(t), t)$$
$$\mathbf{y}(0) = \boldsymbol{y}_0$$

We can represent a higher order ODE as a 1st order system

Vectorized Explicit Euler

- Update rule uses vectorized representation

$$\tilde{\mathbf{y}}(t_{k+1}) = \tilde{\mathbf{y}}(t_k) + \mathbf{f}(\mathbf{y}(t_k), t_k)\Delta t$$

- **Exercise 2:** Simulate coupled ODEs w/ $\Delta t = 0.01$ and $t \in [0, 1]$

$$\frac{dx(t)}{dt} = -5x(t) + 5y(t)$$
$$\frac{dy(t)}{dt} = 14x(t) - 2y(t)$$
$$x(0) = y(0) = 1$$

Properties: Error

Local Truncation Error (LTE)

- Recall update rule

$$\tilde{y}(t_{k+1}) = \tilde{y}(t_k) + f(y(t_k), t_k)\Delta t$$

- Taylor series expansion of analytic solution

$$y(t_k + \Delta t) = y(t_k) + \Delta t \left.\frac{dy(t)}{dt}\right|_{t_k} + O(\Delta t^2)$$

- Difference w/ explicit Euler

$$y(t_k + \Delta t) - \tilde{y}(t_{k+1}) = O(\Delta t^2)$$

- Hence, the error incurred after one step is

$$\boxed{O(\Delta t^2)}$$

Global Truncation Error (GTE)

- The number of steps

$$\frac{t - t_0}{\Delta t} \propto \frac{1}{\Delta t}$$

- Multiplying this with the LTE, we get GTE that is

$$\boxed{O(\Delta t)}$$

- Hence, explicit Euler is a **first order method**

- Higher order methods are available

Properties: Stability

Exercise 3

- Simulate ODE w/ $\Delta t = 0.1$ in $t \in [0, 1]$

$$\frac{dy(t)}{dt} = -20y(t)$$
$$y(0) = 1$$

- Compare w/ analytic answer

Linear Stability

- Consider the linear ODE

$$\frac{dy(t)}{dt} = \lambda y(t)$$

- For a stable solution we must have

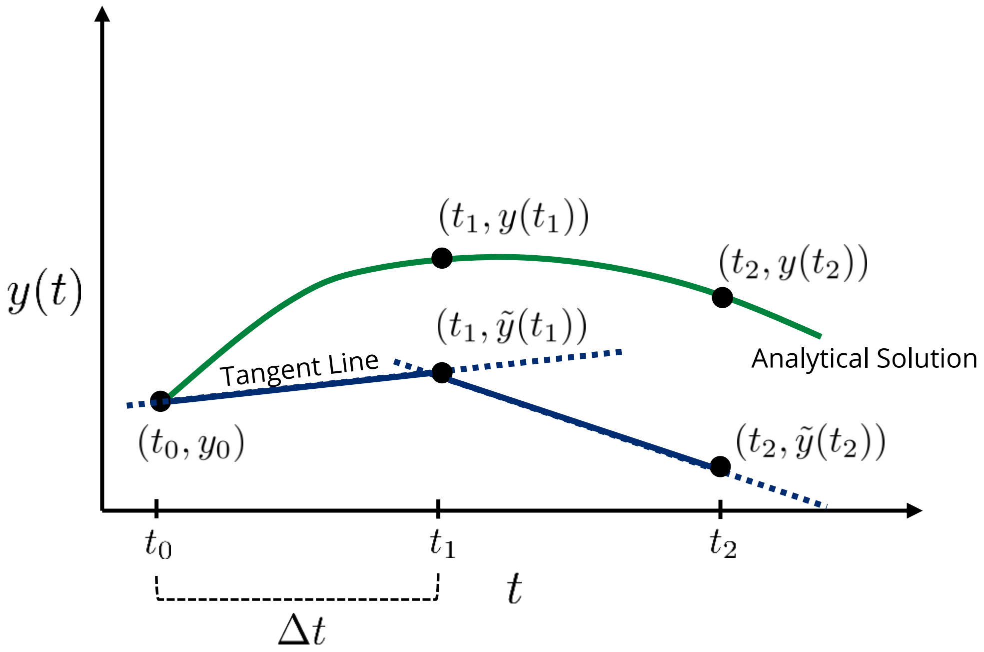
$$|1 + \lambda \Delta t| < 1$$

Stiff ODEs

- Systems that exhibit **numerical instability**
- Precise mathematical definition is nontrivial
- Common with **reaction systems**
  - Coexistence of small and large rate constants
- So, what can we do?

Outline

- Motivation
- Explicit Euler
- **Implicit Euler**
- Reaction Networks
- Other Methods



The Basics

Methodology

- Consider a 1st order ODE

$$\frac{dy(t)}{dt} = f(y(t), t), \quad y(0) = y_0$$

- Define time steps Δt

$$t \in [t_0, t_f] \qquad t_k = t_0 + k\Delta t$$

- Approximate derivative as finite difference

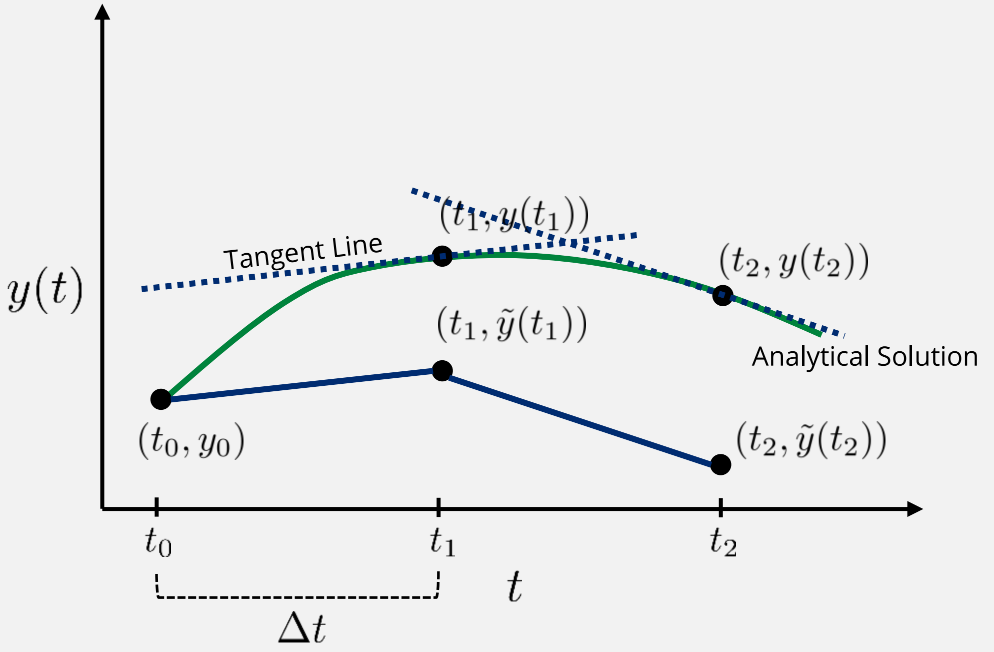
$$\left.\frac{dy(t)}{dt}\right|_{t_{k+1}} \approx \frac{\tilde{y}(t_{k+1}) - \tilde{y}(t_k)}{\Delta t}$$

- Define the update rule

$$\tilde{y}(t_{k+1}) = \tilde{y}(t_k) + f(y(t_{k+1}), t_{k+1})\Delta t$$

- Implicit equation → need to solve nonlinear eq.

Illustration



Exercise 4

Let's revisit w/

$\Delta t = 0.1$ in $t \in [0, 1]$

$$\frac{dy(t)}{dt} = -20y(t)$$

$$y(0) = 1$$

System of ODEs

Generalize Methodology

- Generalize implicit Euler for a system of ODEs

$$\frac{dy_1(t)}{dt} = f_1(y_1, y_2, \dots, y_n)$$
$$\frac{dy_2(t)}{dt} = f_2(y_1, y_2, \dots, y_n)$$
$$\vdots$$
$$\frac{dy_n(t)}{dt} = f_n(y_1, y_2, \dots, y_n)$$
$$y_1(0) = y_{1,0}, y_2(0) = y_{2,0}, \dots, y_n(0) = y_{n,0}$$

- We vectorize

$$\tilde{\mathbf{y}}(t_{k+1}) = \tilde{\mathbf{y}}(t_k) + \mathbf{f}(\mathbf{y}(t_{k+1}), t_{k+1})\Delta t$$

- Now we must solve a system of nonlinear equations

Exercise 5

- Simulate coupled ODEs w/ $\Delta t = 0.01$ and $t \in [0, 1]$

$$\frac{dx(t)}{dt} = -5x(t) + 5y(t)$$
$$\frac{dy(t)}{dt} = 14x(t) - 2y(t)$$
$$x(0) = y(0) = 1$$

Properties

Error

- Local truncation error
  - Taylor series expansion

$$y(t_{k+1} - \Delta t) = y(t_{k+1}) - \Delta t \frac{dy(t)}{dt}\bigg|_{t_{k+1}} + O(\Delta t^2)$$

  - Difference with rule gives

$$y(t_{k+1}) - \tilde{y}(t_{k+1}) = \boxed{-O(\Delta t^2)}$$

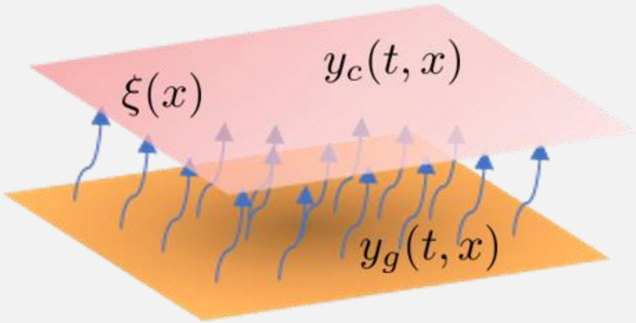
- Global truncation error
  - Multiply LTE with $\frac{1}{\Delta t}$ → $\boxed{O(\Delta t)}$
  - This is a **1st order method**

Stability

- Linear stability

$$\frac{dy(t)}{dt} = \lambda y(t) \qquad \frac{1}{1 - \lambda \Delta t} < 1$$

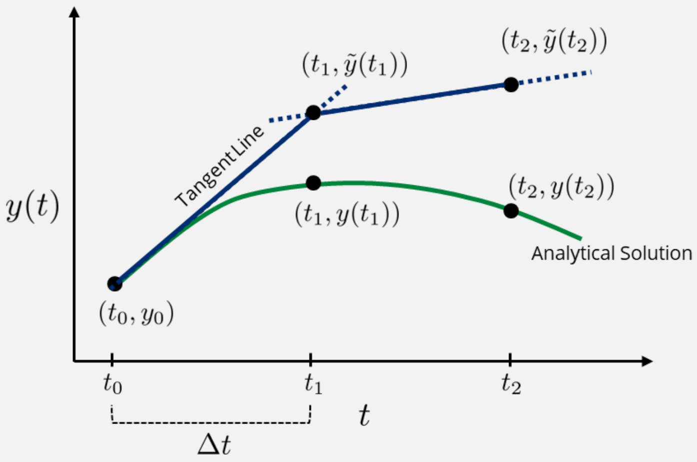
- Typically, stable for stiff systems
- Some exceptions (usually in certain PDEs)
  - e.g., diffusion with nonlinear diffusivity



Explicit vs. Implicit Euler

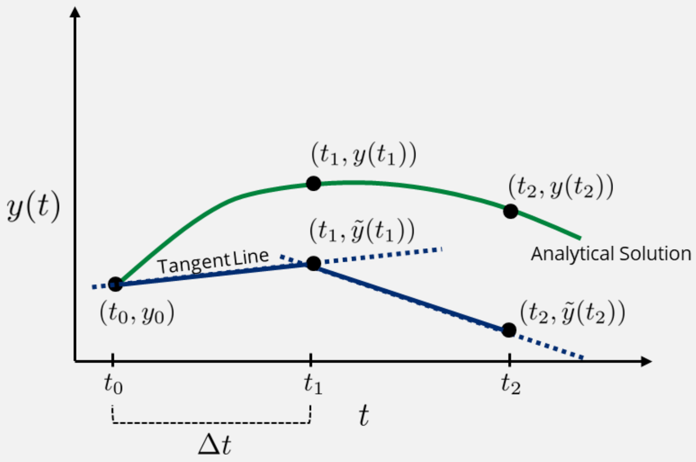
Explicit Euler

- Simple and low computational cost
- Often unstable for stiff systems
- 1st order numerical method



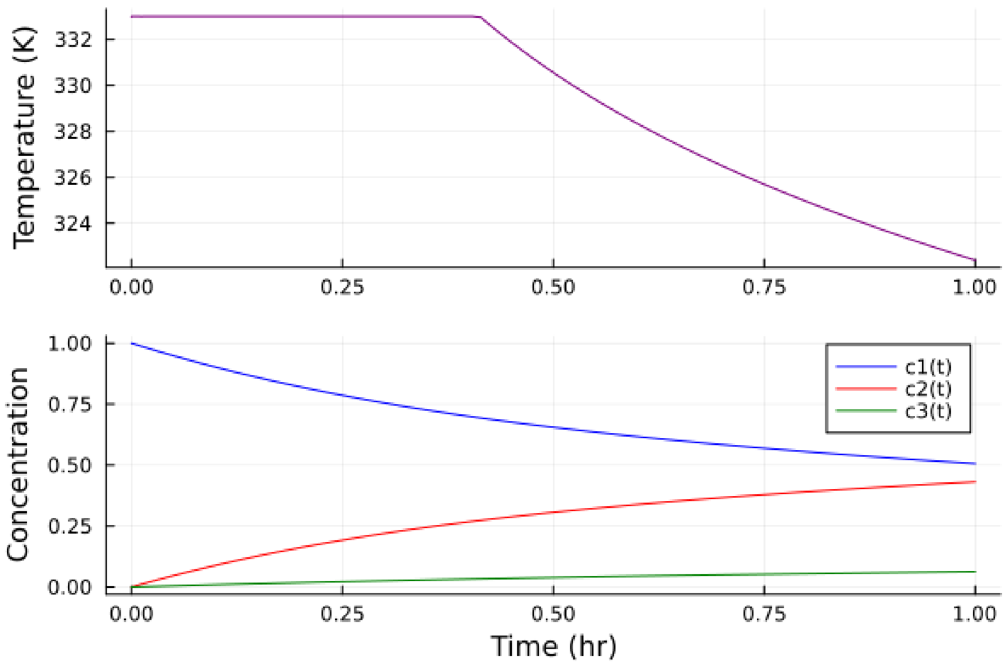
Implicit Euler

- Increased computational cost due to solving nonlinear equation at each step
- Stable for stiff systems
- 1st order numerical method



Outline

- Motivation
- Explicit Euler
- Implicit Euler
- **Reaction Networks**
- Other Methods



Reaction Network Modeling

Simple ODE Model

- Arrhenius equation for species i and reaction j

$$k_{ij}(t) = A_{ij} \exp\left(\frac{-E_{a,ij}}{RT(t)}\right)$$

- Reaction rates

$$r_j(\mathbf{c}, t) = \sum_{i \in I} k_{ij}(t) c_i^{\beta_{ij}}(t)$$

- Species balances

$$\frac{dc_i(t)}{dt} = \sum_{j \in J} \gamma_{ij} r_j(\mathbf{c}, t), \quad i \in I$$

$$c_i(0) = c_{i,0}, \qquad i \in I$$

Example

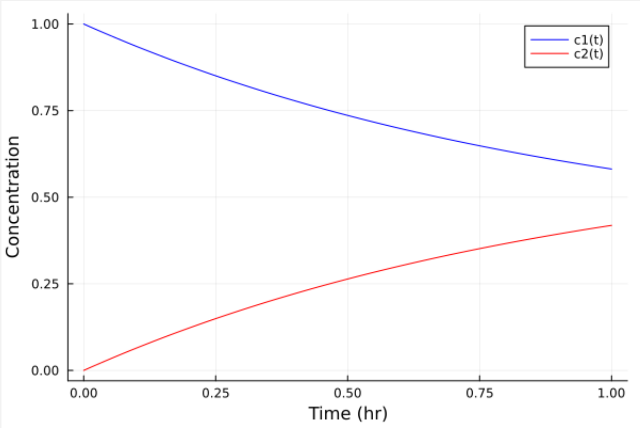
- A ↔ B

$$\frac{dc_1(t)}{dt} = c_2(t)k_2(t) - c_1(t)k_1(t))$$

$$\frac{dc_2(t)}{dt} = c_1(t)k_1(t) - c_2(t)k_2(t)$$

$$c_1(0) = 1, \;\; c_2(0) = 0$$

- Simulate with $t \in [0, 1]$ at $T = 325$



Exercise: Batch Reactor

Exercise 6

- Simulate the following reaction system using explicit Euler for $t \in [0, 1]$

$$\mathrm{A} \rightleftharpoons \mathrm{B}$$
$$\mathrm{A} \rightleftharpoons \mathrm{C}$$

- Experiment with different choices of Δt

Problem Information

$$R = 1.987$$

$$A = \begin{bmatrix} 3.6362e6 & 190.6879 \\ -2.5212e16 & 0 \\ 0 & -8.7409e24 \end{bmatrix}$$

$$E_a = \begin{bmatrix} 10000 & 5000 \\ 25000 & 0 \\ 0 & 40000 \end{bmatrix} \qquad \beta = 1 \qquad \gamma = \begin{bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$T(t) = \begin{cases} 333, & t < 0.5 \\ 325, & t \geq 0.5 \end{cases} \qquad \mathbf{c}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

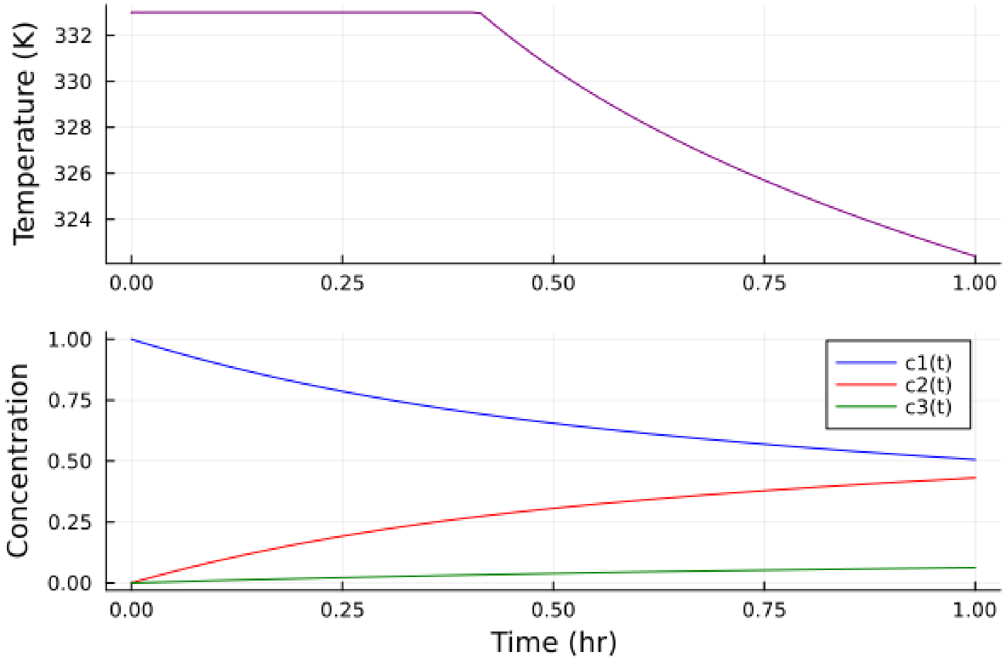
Optimizing Reaction Networks

- We can optimize certain outcomes using numerical approximations of ODEs and optimization software

- For instance, let's maximize the final concentration of B by controlling $T(t)$

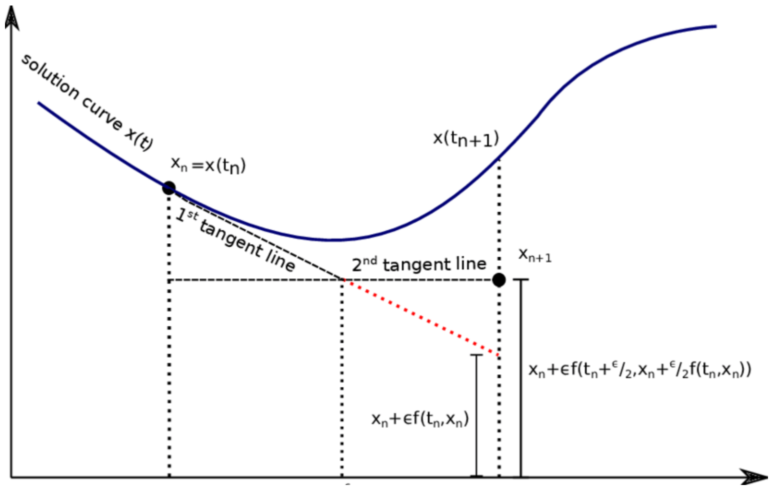
$$
\begin{aligned}
\max \quad & c_2(t_f) \\
\text{s.t.} \quad & \frac{dc_i(t)}{dt} = \sum_{j \in J} \gamma_{ij} r_j(\mathbf{c}, t), && t \in [0,1] \\
& 0 \leq \mathbf{c}(t) \leq 1, && t \in [0,1] \\
& \underline{T} \leq T(t) \leq \overline{T}, && t \in [0,1] \\
& \mathbf{c}(0) = \mathbf{c}_0
\end{aligned}
$$

InfiniteOpt



Outline

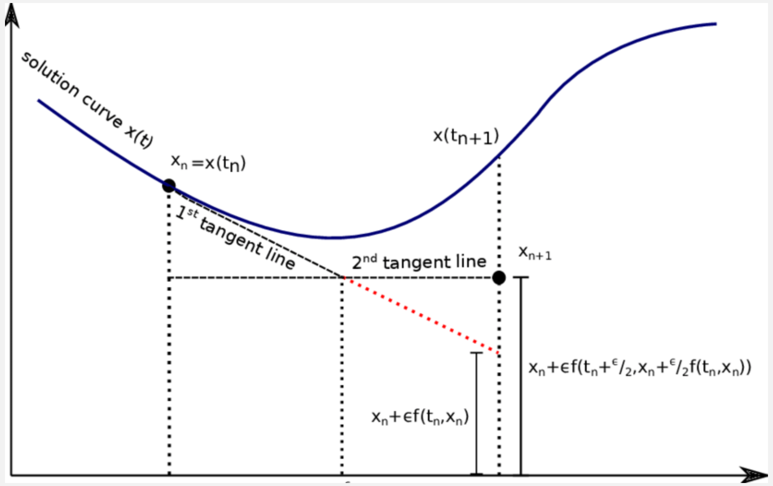
- Motivation
- Explicit Euler
- Implicit Euler
- Reaction Networks
- **Other Methods**



More Advanced Methods

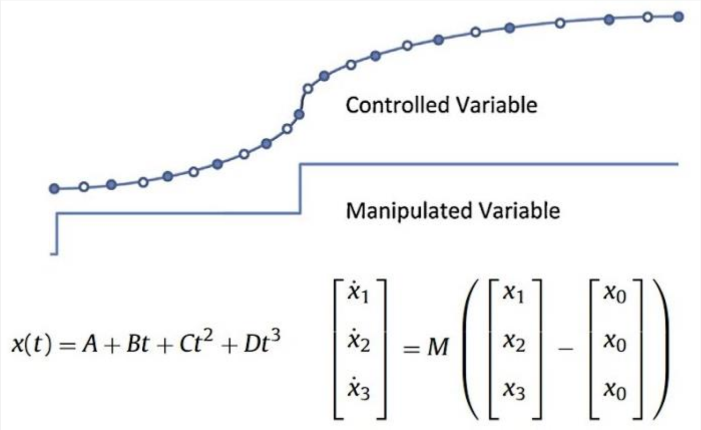
Runge-Kutta

- Family of explicit and implicit iterative methods
- Various orders based on GTE $O(h^p)$
  - 1st order methods are the Euler methods
  - 4th order methods are popular



Orthogonal Collocation over Finite Elements

- The discretization uses finite elements
- We approximate the solution in each element as a polynomial function
- End up solving a system of linear equations



Common Simulation Tools

ODE Integrators

- Common in scripting languages
- Provide numerical solutions to ODE systems

Symbolic Solvers

- Can provide analytic solutions when possible
- Typically, not used for large problems

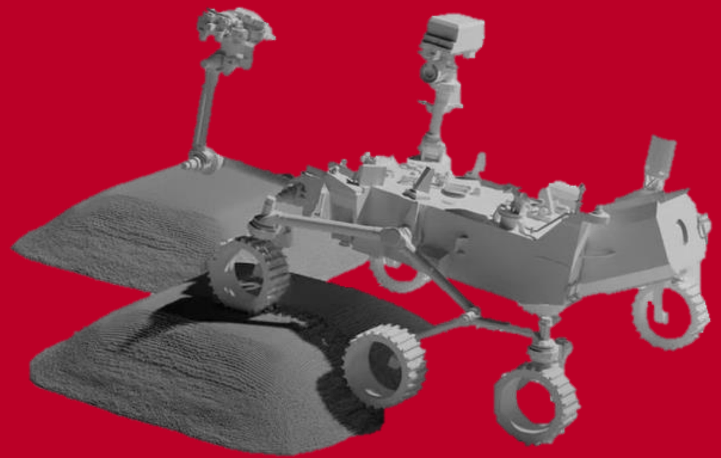
Optimization Tools

- Can incorporate differential equations when solving optimization problems

Problem Specific

- Simulate dynamics for particular systems

Carnegie Mellon University



Introductory Numerical Methods for Simulating Dynamic Systems

Dr. Joshua Pulsipher