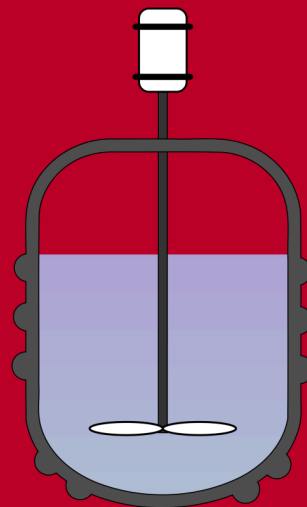


Carnegie Mellon University



Introductory Numerical Methods for Simulating Batch Reactors

Dr. Joshua Pulsipher



Learning Outcomes

1. The relative advantages/disadvantages of using **explicit Euler** methods
2. How to implement explicit Euler to **simulate ODEs** using common computation environments (e.g., Python)
3. How to **simulate batch reactors** using numerical methods
4. [Time Permitting] A familiarity of **other numerical methods/tools** for simulating batch reactors



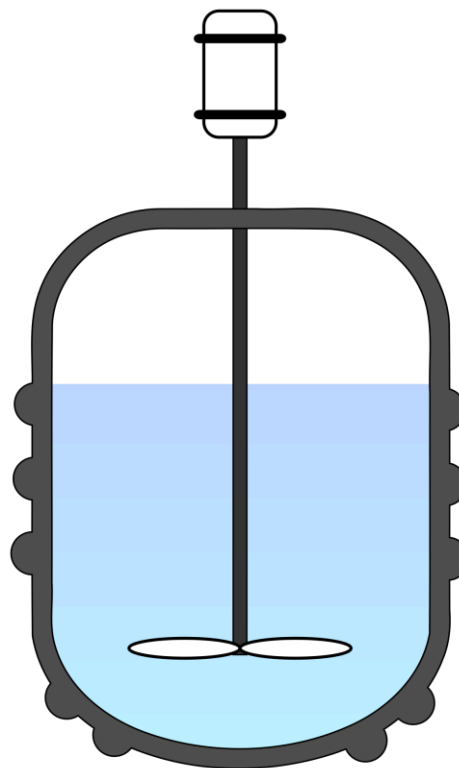
Outline

- Overview
- Explicit Euler
- Batch Reactors
- Other Numerical Methods (If time allows)



Outline

- **Overview**
- Explicit Euler
- Batch Reactors
- Other Numerical Methods





Simulating Dynamic Systems

- Simulating dynamic systems is vital for enabling engineering applications



Curiosity Rover



Batch Reactor

- Simulate using **numerical methods to approximate dynamics** (e.g., differential equations)
- Enables us to **computationally experiment** and implement **automation**



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

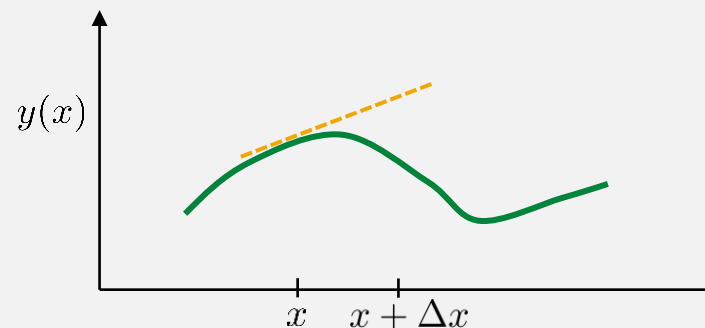
$$\frac{dy}{dx} = \frac{x+y}{x-y} \quad \Rightarrow \quad \frac{1}{2} \log \left(\frac{y^2}{x^2} + 1 \right) - \tan^{-1} \left(\frac{y}{x} \right) = C - \log(x)$$

- Solving general ODEs is often difficult or not possible

Numerical Methods

- Seek to numerically approximate the solution

- **Finite difference methods** are common

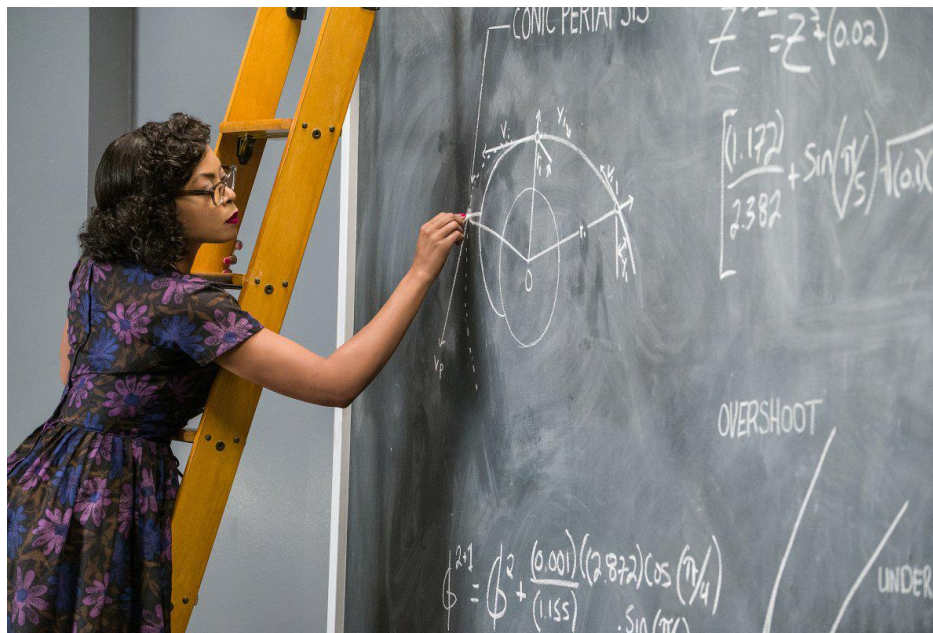


- More advanced methods are available
 - Not the focus of today



Outline

- Overview
- **Explicit Euler**
- Batch Reactors
- Other Numerical Methods



Katherine Goble





The Basics

Methodology

- Consider a 1st order ODE

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

- Define time steps Δt

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

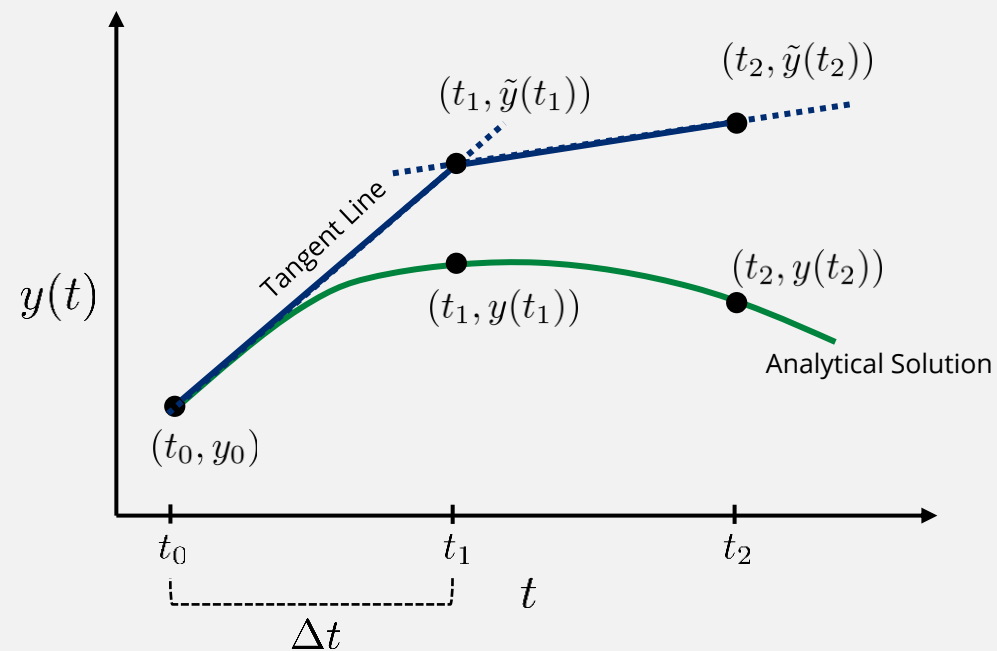
- Approximate derivative as finite difference

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

- Define update rule

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

Illustration





Simulating a System of ODEs

System of 1st order ODEs

- General representation

$$\frac{dy_1(t)}{dt} = f_1(y_1, y_2, \dots, y_n)$$

$$\frac{dy_2(t)}{dt} = f_2(y_1, y_2, \dots, y_n)$$

\vdots

$$\frac{dy_n(t)}{dt} = f_n(y_1, y_2, \dots, y_n)$$

$$y_1(0) = y_{1,0}, y_2(0) = y_{2,0}, \dots, y_n(0) = y_{n,0}$$

- Vectorize

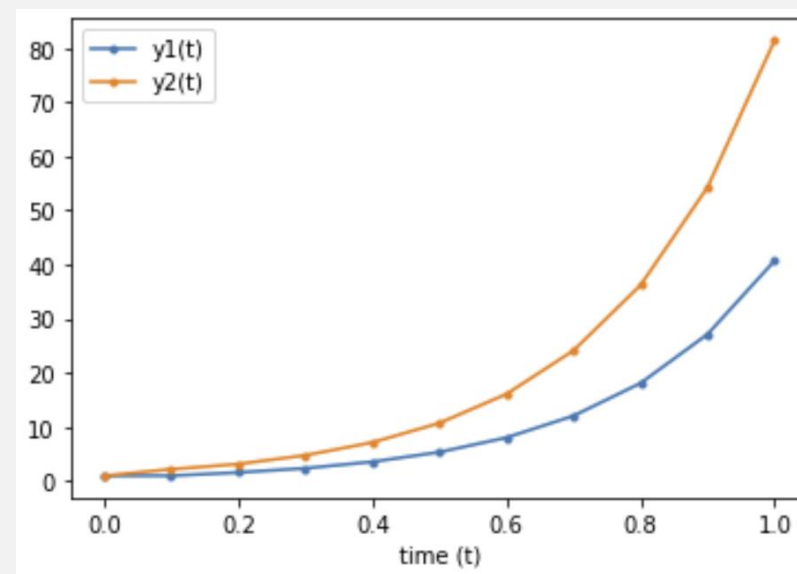
$$\frac{d\mathbf{y}(t)}{dt} = \mathbf{f}(\mathbf{y}(t), t)$$

$$\mathbf{y}(0) = \mathbf{y}_0$$

Vectorized Explicit Euler

- Update rule uses vectorized representation

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





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

$$O(\Delta t^2)$$

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

$$O(\Delta t)$$

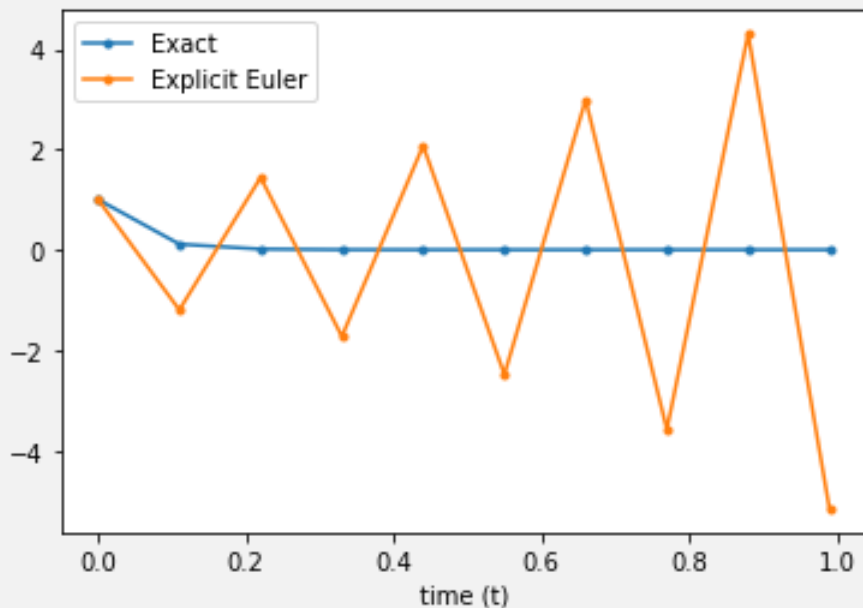
- Hence, explicit Euler is a **first order method**

- Higher order methods are available



Properties: Stability

Example



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? → Use **implicit methods**

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



Tutorials

1. Go to <https://github.com/pulsipher/eulercourse>
2. Click on the “launch binder” button or click on the “Open in Colab” button
3. Open “exercise.ipynb” if you used binder



master 1 branch 0 tags Go to file Add file <> Code

pulsipher minor formatting change 9010cda now 15 commits

.gitignore	Initial commit	3 weeks ago
LICENSE	Initial commit	3 weeks ago
README.md	minor formatting change	now
exercises.ipynb	remove unneeded todo	3 hours ago
exercises_completed.ipynb	remove unneeded todo	3 hours ago
lecture_slides.pdf	Shorten course	3 hours ago
requirements.txt	Shorten course	3 hours ago

README.md

Introduction to Numerical Methods for Simulating Batch Reactor Exercises

Here we have the exercises for Euler methods.

The files are as follows:

- `exercises.ipynb` : The worksheet of all the exercises with helper code.
- `exercises_completed.ipynb` : The worksheet with all the answers. Avoid the temptation to cheat :)
- `lecture_slides.pdf` : The lecture slides (note some figures do not present correctly in PDF form)

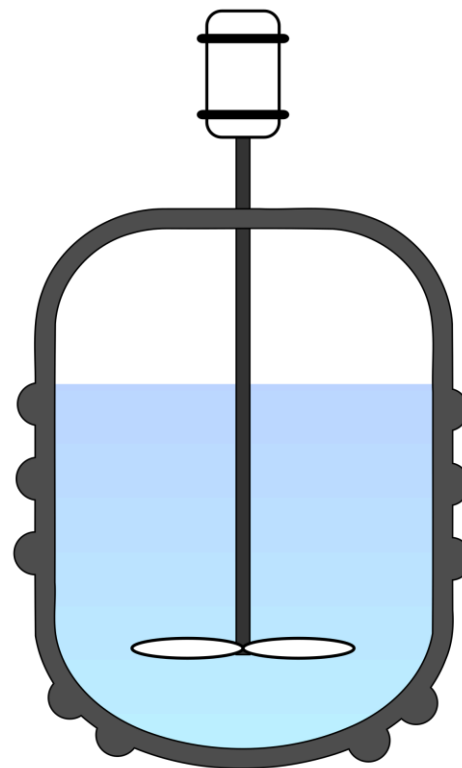
To try these out without having to download anything, please go to [launch binder](#).

Alternatively, you can use [Open in Colab](#) to assess `exercises.ipynb`.



Outline

- Overview
- Explicit Euler
- **Batch Reactors**
- Other Numerical Methods





Batch Reactor 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

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

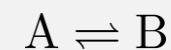
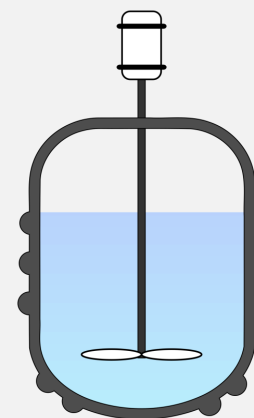
- Species generation rates

$$g_i(\mathbf{c}, t) = \sum_{j \in J} \gamma_{ij} r_j(\mathbf{c}, t)$$

- Species balances

$$\begin{aligned} \frac{dc_i(t)}{dt} &= g_i(\mathbf{c}, t), \quad i \in I \\ c_i(0) &= c_{i,0}, \quad i \in I \end{aligned}$$

The Basics



- Derivation from mole balance

$$\frac{N_i(t)}{dt} = \cancel{F_{i,in}(t)} - \cancel{F_{i,out}(t)} + \int^V g_i(t) dV(t)$$

- Assume perfect mixing and constant volume

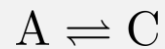
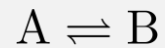
$$\frac{N_i(t)}{dt} = g_i(t)V \quad \rightarrow \quad \frac{c_i(t)}{dt} = g_i(t)$$



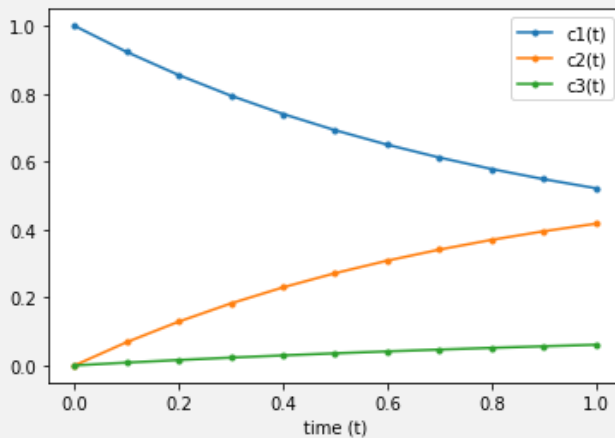
Exercise: Batch Reactor

Exercise 1

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



- Experiment with different choices of Δt



Problem Information

$$R = 1.987$$

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

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

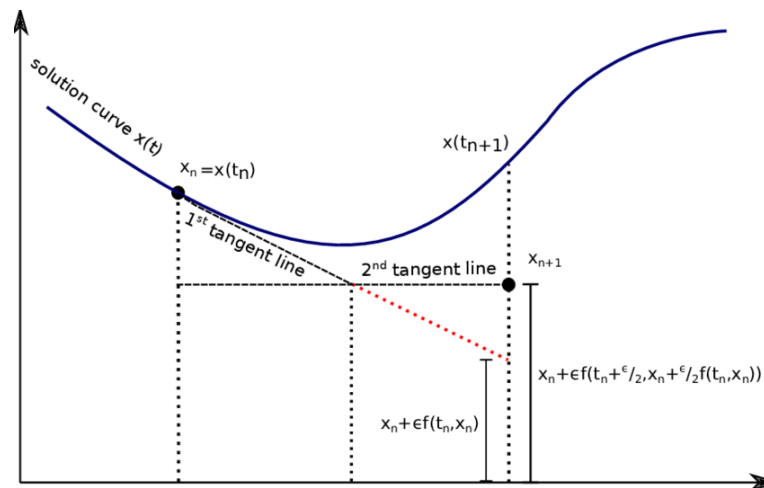
$$T(t) = \begin{cases} 333, & t < 0.5 \\ 325, & t \geq 0.5 \end{cases}$$

$$\mathbf{c}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$



Outline

- Overview
- Explicit Euler
- Batch Reactors
- **Other Numerical Methods**

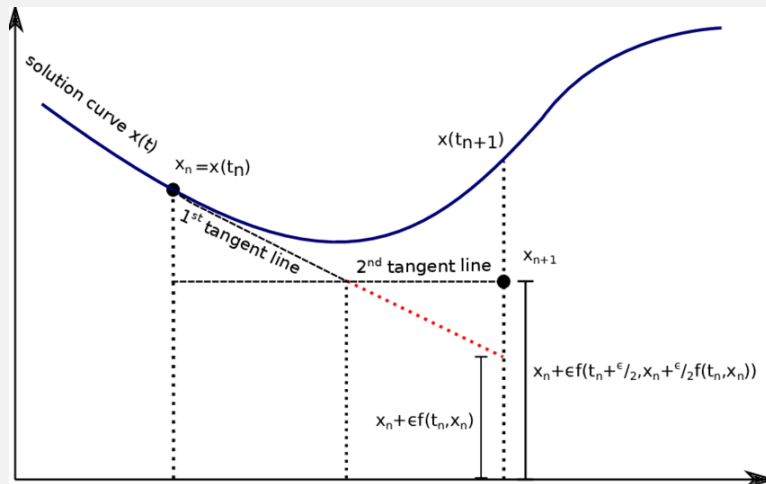




More Advanced Methods

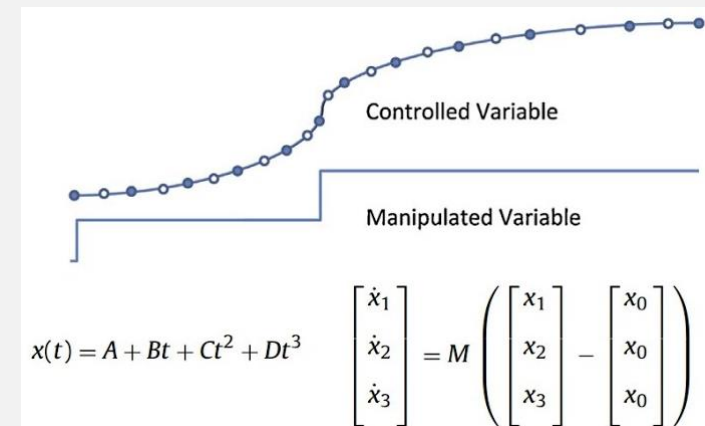
Runge-Kutta

- 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

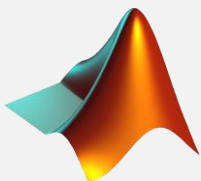




Common Simulation Tools

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

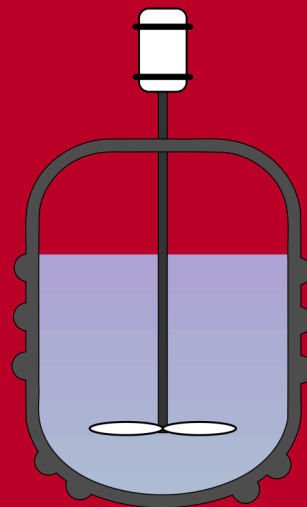


Problem Specific

- Simulate dynamics for particular systems



Carnegie Mellon University



Introductory Numerical Methods for Simulating Batch Reactors

Dr. Joshua Pulsipher