Numerical Problem Solving across the Curriculum with Python and MATLAB Using Interactive Coding Templates

Paper ID: 38671

Austin N. Johns, University of Buffalo (anjohns@buffalo.edu)

Robert P. Hesketh, Rowan University (hesketh@rowan.edu)

Matthew D. Stuber, University of Connecticut (matthew.stuber@uconn.edu)

Ashlee N. Ford Versypt, University at Buffalo (ashleefv@buffalo.edu)

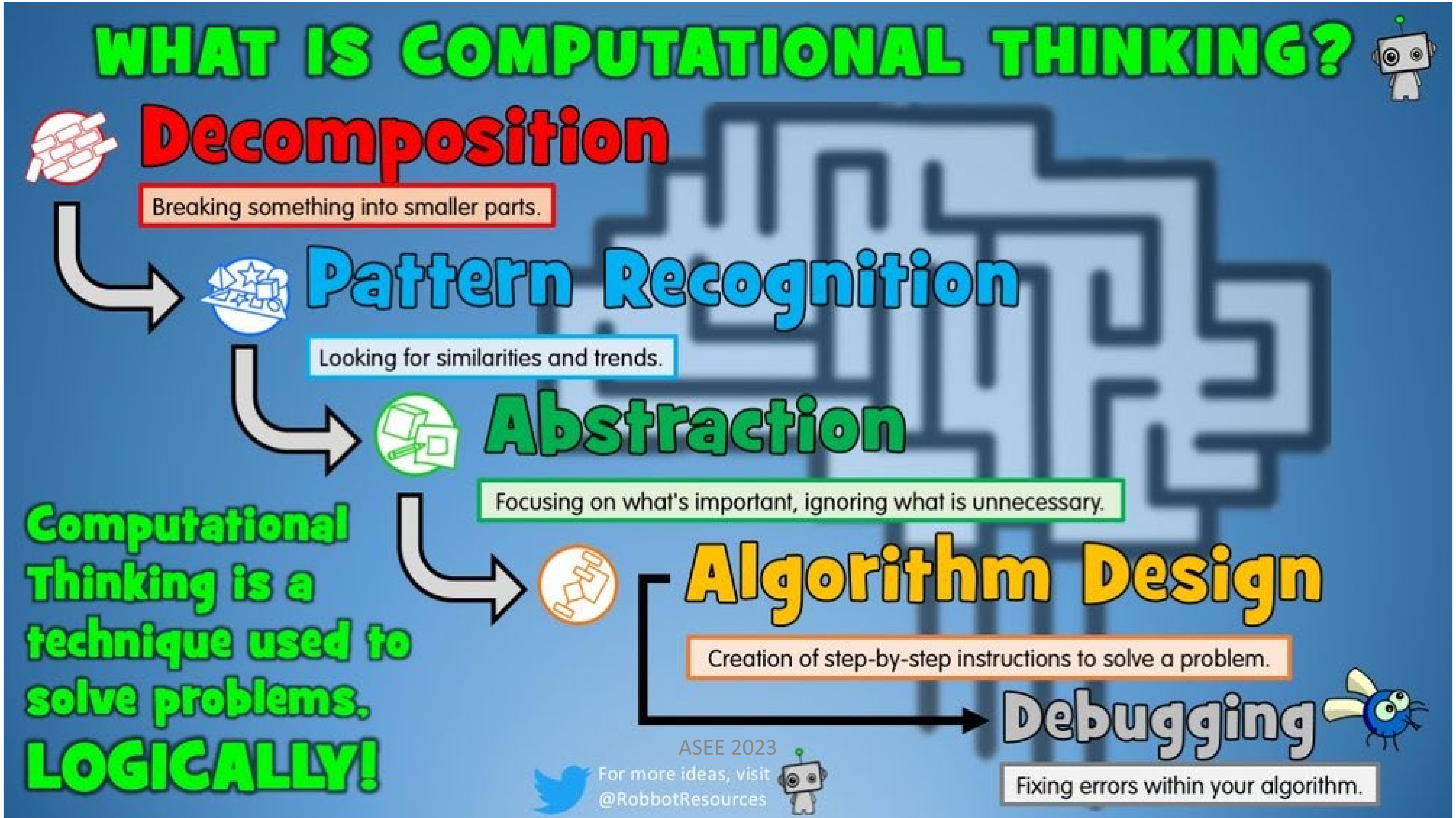
Wednesday, June 28, 2023

Chemical Engineering Division – Innovative Pedagogy

Outline

1. Computational Thinking and Literate Programming
2. Interactive Coding Notebooks/Templates
3. Use Cases and Workshop
4. Summary and Conclusion

Educational goal: computational thinking for problem solving



Literate programming
making code human readable

- Focus on natural language
- Text and code blocks
- Formatting for clear documentation
  - Alternative to extensive commenting
- Educational applications
- Markup language vs programming language
- “Interactive coding notebooks/templates”

```
LaTeX is also an important tool for writing mathematical symbols and
formulas. You can use LaTeX in text cells by bracketing your LaTeX equations
with dollar signs ($). The following links contain tutorials on how to format
text and equations using [Markdown](https://colab.research.google.com/
notebooks/markdown_guide.ipynb) and [LaTeX](https://colab.research.google.com/
github/johnpharmd/DS-Sprint-03-Creating-Professional-Portfolios/blob/master/
LaTeX.ipynb). Many similar tutorials exist and it is easy to find cheat
sheets for each language online.

The following is an example from our solveODEs case study of an equation
written in LaTeX:

<p>$\frac{dT}{dV}=\frac{U_a(T_a-T)\;+\;(-r_{1A})(\Delta H_{Rx1A})\;+\;(-r_{2A}
)(\Delta H_{Rx2A})}{F_A C_{P_A}\;+\;F_B C_{P_B}\;+\;F_C C_{P_C}}$<br>

The source code for this equation can be found by double-clicking this cell.
LaTeX source code can also be represented using the "Format as Code" option
on the text cell toolbar, the fourth option from the left.

```
$\frac{dT}{dV}=\frac{U_a(T_a-T)\;+\;(-r_{1A})(\Delta H_{Rx1A})\;+\;(-r_{2A})
(\Delta H_{Rx2A})}{F_A C_{P_A}\;+\;F_B C_{P_B}\;+\;F_C C_{P_C}}$
```
```

Markup language (markdown+LaTeX)

Compiled text block

13. Toggle Reposition Markdown Preview Location
14. Move Selected Cell Up
15. Move Selected Cell Down
16. Hyperlink to Selected Cell
17. Add Comment to Selected Cell
18. Open Editor Settings
19. Toggle Markdown Editor
20. Mirror Cell in Tab
21. Delete Cell
22. Select/Copy/Cut Cell

LaTeX is also an important tool for writing mathematical symbols and formulas. You can use LaTeX in text cells by bracketing your LaTeX equations with dollar signs ($). The following links contain tutorials on how to format text and equations using Markdown and LaTeX. Many similar tutorials exist and it is easy to find cheat sheets for each language online.

The following is an example from our solveODEs case study of an equation written in LaTeX:

$$\frac{dT}{dV}=\frac{U_a(T_a-T)\;+\;(-r_{1A})(\Delta H_{Rx1A})\;+\;(-r_{2A})(\Delta H_{Rx2A})}{F_A C_{P_A}\;+\;F_B C_{P_B}\;+\;F_C C_{P_C}}$$

The source code for this equation can be found by double-clicking this cell. LaTeX source code can also be represented using the "Format as Code" option on the text cell toolbar, the fourth option from the left.

```
$\frac{dT}{dV}=\frac{U_a(T_a-T)\;+\;(-r_{1A})(\Delta H_{Rx1A})\;+\;(-r_{2A})(\Delta H_{
```

```
[ ] from math import sin
    x = 3
    y = x**2
    z = sin(y)
    print('The answer is',z)

    The answer is 0.4121184852417566
```

Programming language in code block

Ways to use these interactive coding templates in ChE classrooms across the curriculum

- Reinforce computational thinking
- Introduce/enhance familiarity with MATLAB and/or Python
- Student use cases
  - Lecture notes with in-class activities
  - Pre-class readings with embedded activities
  - Worked example case studies
  - In-class problems
  - Homework or project problems

Example interactive coding templates

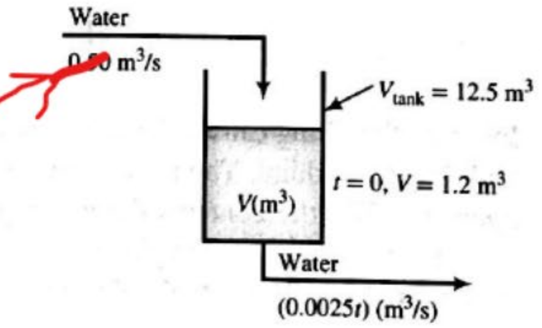
- Ex. 10.2-1 from MEB book by Felder, Rousseau, & Bullard
- Jupyter Notebook (Python) version: J6_TankDrainage.ipynb
- MATLAB version: M6_TankDrainage.mlx

ODEfun(t, y_i's)

$$\frac{dV}{dt} = \dot{v}_{in} - \dot{v}_{out}$$

Example 10.2-1 **Mass Balance on a Water Storage Tank**

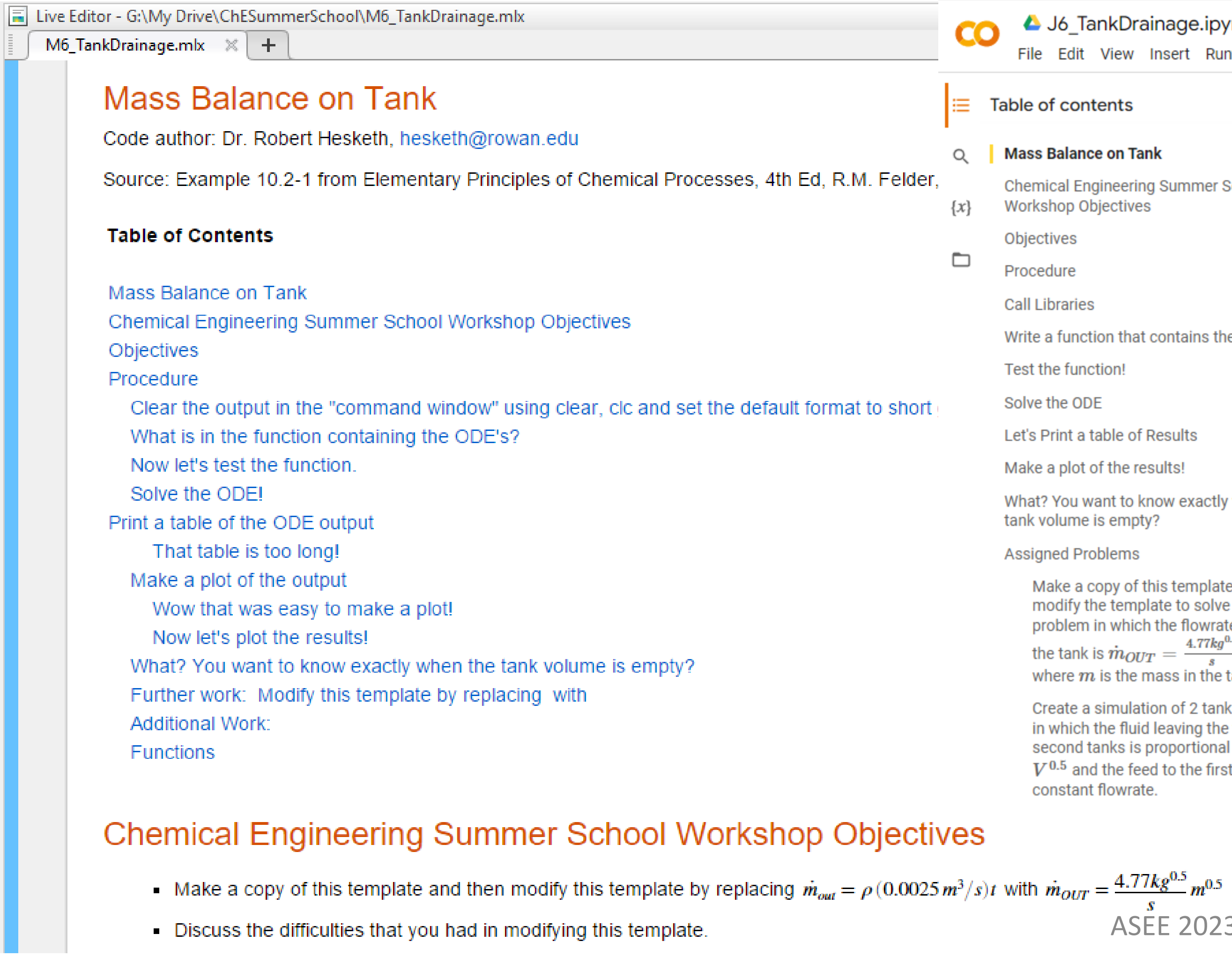
A 12.5-m^3 tank is being filled with water at a rate of 0.050 m^3/s. At a moment when the tank contains 1.20 m^3 of water, a leak develops in the bottom of the tank and gets progressively worse with time. The rate of leakage (m^3/s) can be approximated as 0.0025t, where t(s) is the time from the moment the leak begins.

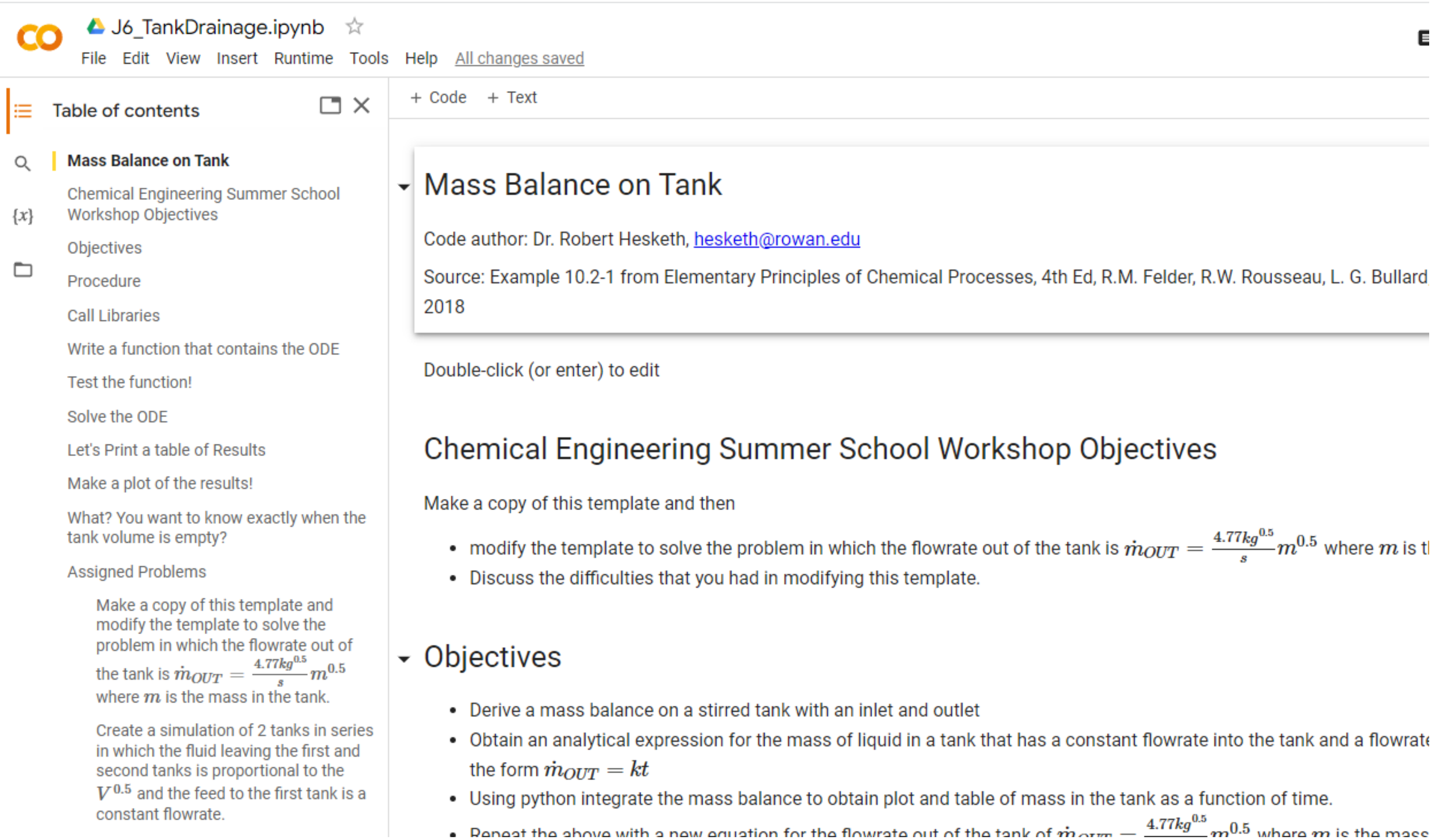


1. Write a mass balance on the tank and use it to obtain an expression for dV/dt, where V is the volume of water in the tank at any time. Provide an initial condition for the differential equation.
2. Solve the balance equation to obtain an expression for $V(t)$ and draw a plot of V versus t.

Mass Balance on Tank: Table of Contents

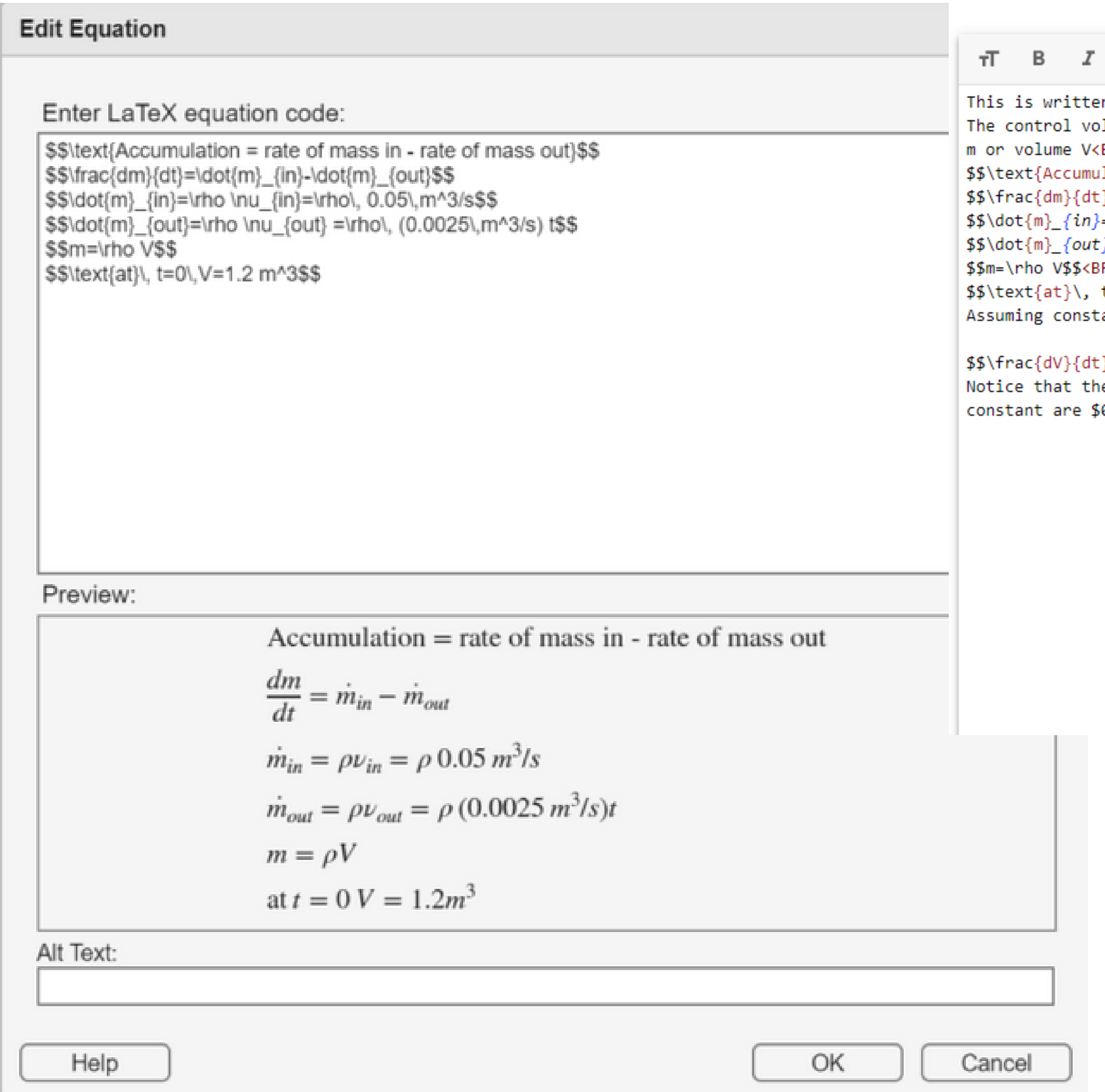
Matlab vs Python

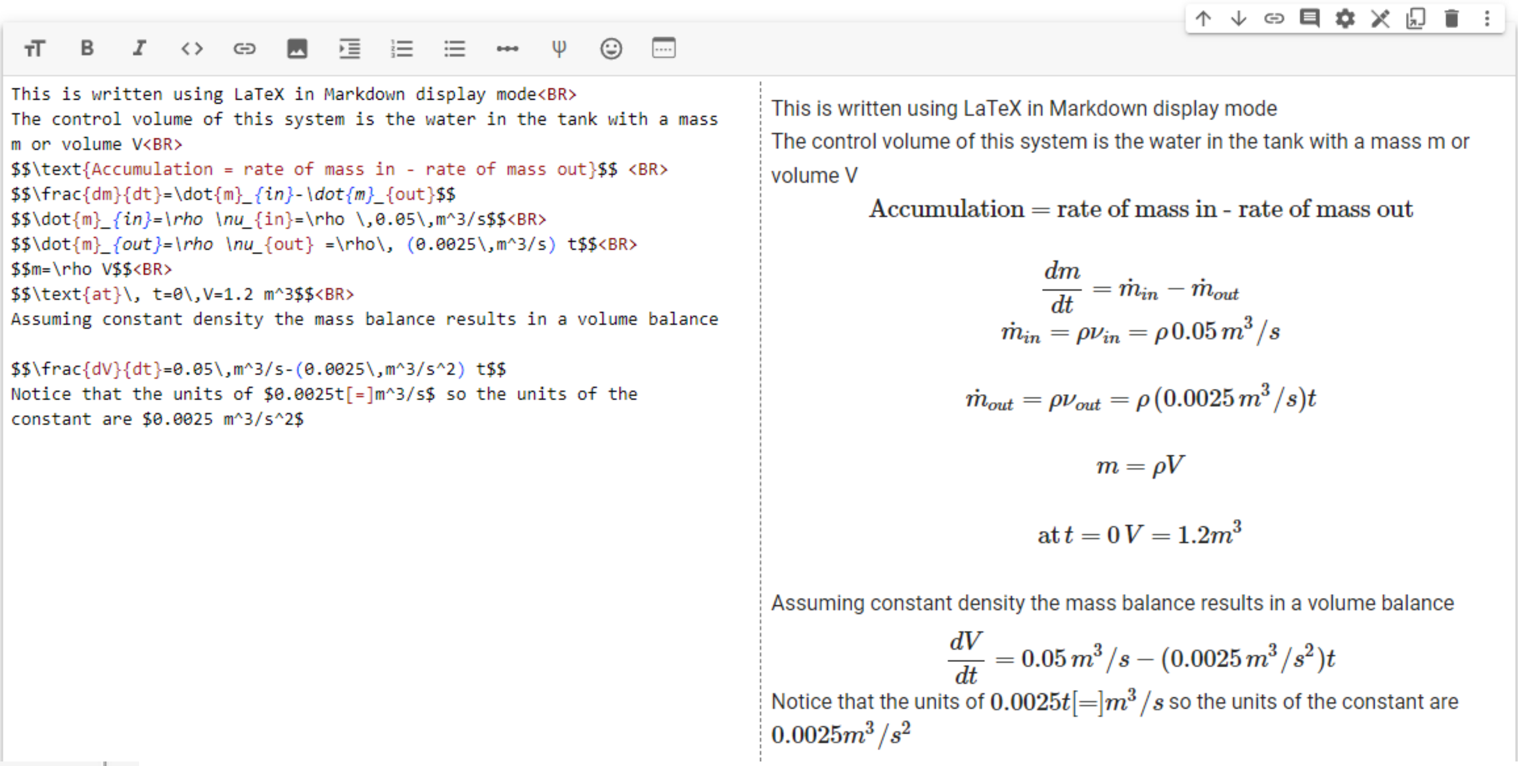




Mass Balance on Tank: Type set equations using LaTeX

Matlab vs Python



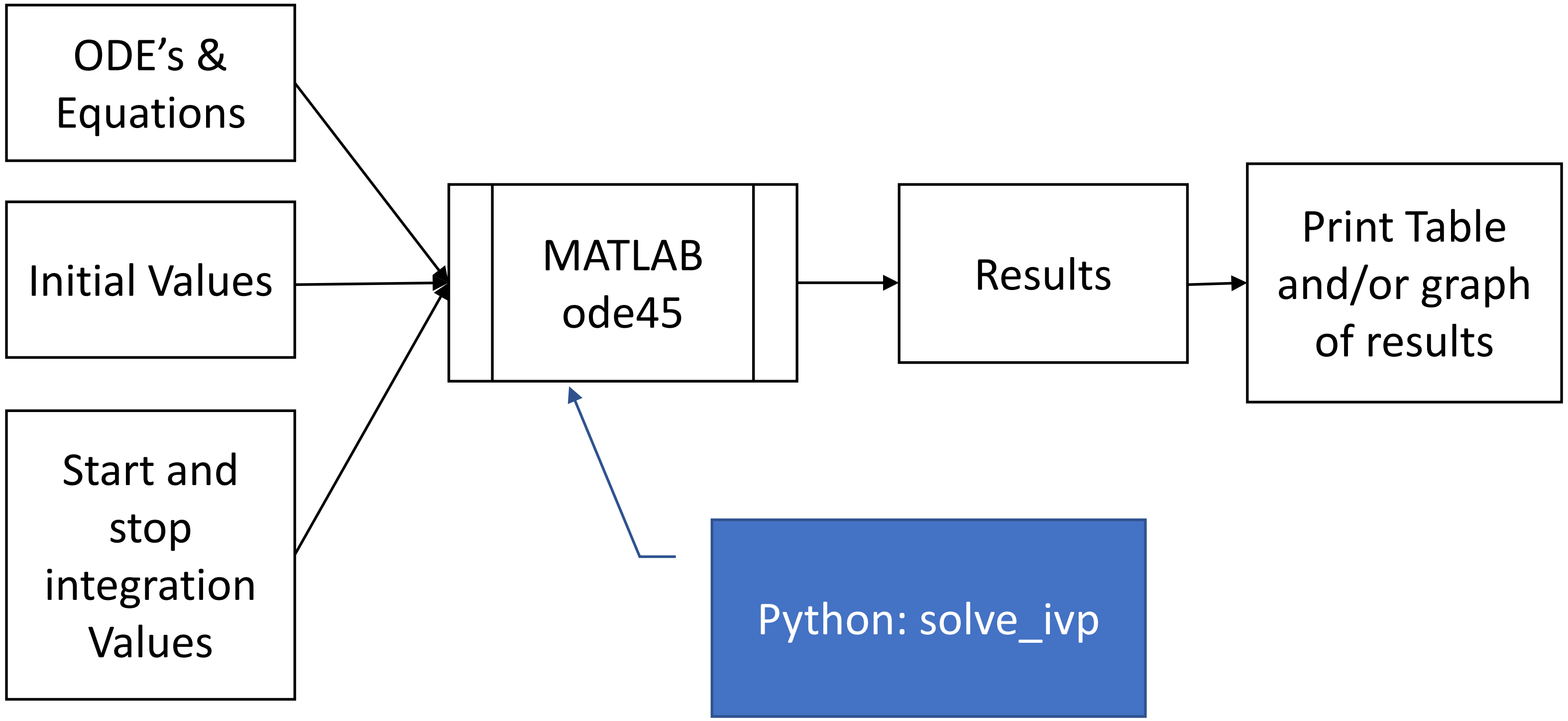


Mass Balance on Tank: Solve ODE

Python: call libraries at start

```
import numpy as np
from scipy.integrate import solve_ivp
import math
import matplotlib.pyplot as plt
```

ODE Solver Flowchart



Matlab vs Python

```matlab
function dYfuncvecdt = ODEfun(t,Yfuncvec)
V = Yfuncvec(1);
dVdt = 0.05 - (0.0025 * t);
dYfuncvecdt = dVdt;
end
```

ODE’s & Equations

```python
def  ODEfun(t,Yfuncvec):
    V = Yfuncvec[0]
    dVdt = 0.05-0.0025*t
    dYfuncvecdt = [dVdt]
    return dYfuncvecdt
```

```matlab
y0 = 1.2; % Initial values
tspan = [0 80];
```

Initial Values and Integration interval

```python
y0 = [1.2] # Initial Value
tspan = (0,80)
```

```matlab
[t,y]=ode45(@ODEfun,tspan, y0)
```

Runge Kutta Solver

```python
sol=solve_ivp(ODEfun,tspan,y0)
```

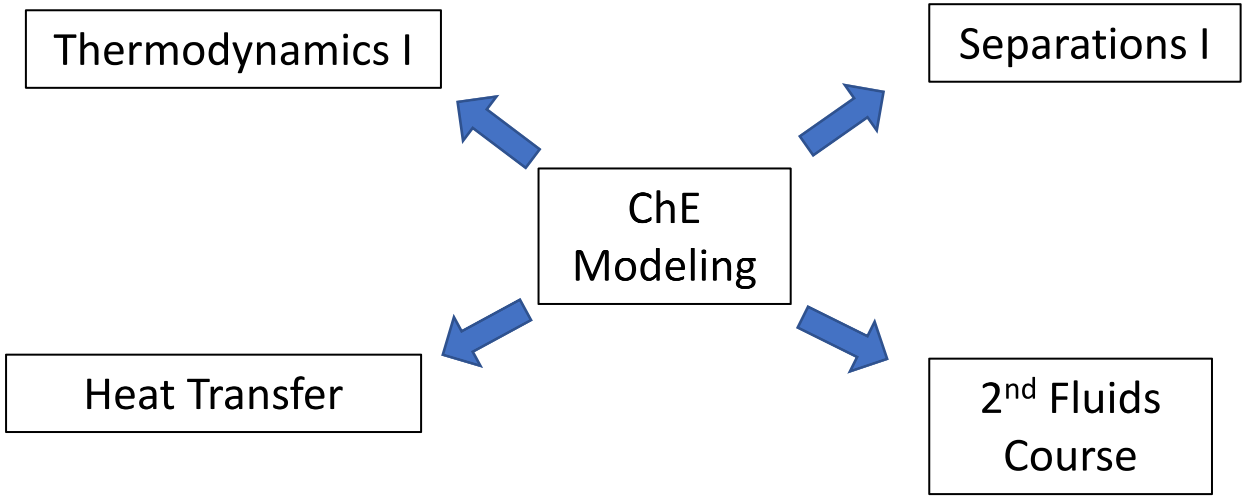
```matlab
plot (t,y);
xlabel('t (s)');
ylabel('V (m^3) ');
legend('V (m^3)');
title('Example 10.2-1');
```

Plot

```python
plt.plot(sol.t,sol.y[0])
plt.xlabel('Time ($s$)')
plt.ylabel('Volume ($m^3$)')
plt.legend()
plt.title(' Example 10.2-1')
```

Rowan Fall Semester

- Students learn python & practical numerical methods in ChE Modeling
- Use & modify python templates in the concurrent ChE classes
- Use python in Spring classes: Separations II, Reactor Design, Thermodynamcs II.



- Equations of State

- Diffusion in gas
- Unsteady-State Diffusion
- Mass Transfer solid to liquid
- Gas Absorption Column

- Conduction with generation term

- Complex pipe network
- Microscopic momentum balance-Newtonian fluid
- Non-Newtonian fluid

Workshop for Chemical Engineering Faculty: 2022 ASEE/AIChE Summer School

- Created and curated interactive notebook materials
  - 9 coding templates creates (7 in both Python and MATLAB)
  - Covered topics: linear/nonlinear algebraic systems, ODE-IVPs, ODE-BVPs, PDEs
- Workshop used interactive learning activities
  - Exercise 1: “how to create” using prepared tutorial notebooks with gaps
  - Exercise 2: “create your own” notebook from scratch
  - Exercise 3: explore the provided templates and adapt them

Conclusion

- Interactive coding templates enhance student learning and create interactive learning opportunities in/out of the classroom
- Workshop given to ChemE faculty to (hopefully) broaden adoption and enhance computational thinking exercises/practices
- Workshop participants without prior experience identified that they'd still need knowledge of the underlying software language
- Workshop materials were made available:

More computational educational materials from us

- Video highlighting key features of MATLAB Live Scripts and Jupyter Notebooks (Austin Johns, Ford Versypt Lab):
https://youtu.be/u5YkzFl6FbE

- Ford Versypt: https://github.com/ashleefv/ApplNumComp

- Hesketh: https://github.com/heskethrp

- Stuber: https://github.com/PSORLab/Chemical_Engineering_Analysis_Notebooks

More related computational educational materials from others

Source	Programming Language	Description	Software Lic
https://github.com/ashleefv/ApplNumComp	MATLAB, Python	Ashlee N. Ford Versypt, Duncan H. Mullins: Introduction to MATLAB and Python. Includes a section on converting MATLAB code to Python.	https://github.com/ashleefv/ApplNumC
https://github.com/heskethrp	Python, Jupyter Notebook, MATLAB	Robert Hesketh: Jupyter Notebooks for four of his chemical engineering courses. Note each is in a seperate repository.	BSD 3-Clause "New" or "Revised" Licens
https://github.com/PSORLab/Chemical_Engineering_Analysis_Notebooks	Julia, MATLAB, Jupyter Notebook	Matthew Wilhelm, Chenyu Wang, Matthew Stuber: Repository of supplemental Jupyter Notebooks for use in courses centered around application of numerical methods with a chemical engineering context	https://github.com/PSORLab/Chemical_ ks/blob/master/LICENSE
http://websites.umich.edu/~elements/5e/live/index.html	MATLAB, Python, Polymath, Wolfram, AspenPlus	Elements of Chemical Reaction Engineering, 5th Edition. Living example problems each solved in numerous programming languages. Additional computer simulation problem statements.	
https://cache.org/	MATLAB, Wolfram, Python	Computer Aids for Chemical Engineering; Categorized by subject including Material/Energy Balances, Fluid Mechanics, Heat Transfer, Thermodynamics, Kinetics, etc; Includes recommended textbooks, interactive simulations, and software sections that link to material that could be used for the summer session	Contains links to code from many differe information is provided on a case by cas
https://www.routledge.com/Introduction-to-Modeling-and-Simulation-with-MATLAB-and-Python/Gordon-Guilfoos/p/book/9780367573362	MATLAB, Python	Steven I. Gordon and Brian Guilfoos, Introduction to Modeling and Simulation with MATLAB and Python, CRC Press, 2017; ISBN: 0367573369; Associated code available at http://www.intromodeling.com	
https://www.cambridge.org/us/academic/subjects/engineering/chemical-engineering/numerical-methods-chemical-engineering-applications?format=HB	MATLAB	Numerical Methods with Chemical Engineering Applications by Kevin D. Dorfman and Prodromos Daoutidis, University of Minnesota; Undergraduate chemical engineering textbook with MATLAB example problems	
https://github.com/jckantor/CBE20255	Python, Jupyter Notebook	CBE20255 Introduction to Chemical Engineering Analysis. The Jupyter Notebooks demonstrate these basic chemical engineering calculations using Python.	https://github.com/jckantor/CBE20255/
https://apmonitor.com/che263/	MATLAB, Python, Excel, VBA, MATHCAD	Website hosted by John Hedengren, leader of the BYU PRISM group. Supports Chemical Engineering 263, Problem Solving with Programming for Engineers. Focuses on teaching programming languages to engineers rather than examples of how that code might be used outside of a selection of case studies.	
https://github.com/numerical-mooc/numerical-mooc	Python, Jupyter Notebook	Jupyter Notebook modules with relevant chemical engineering problems including: finite-difference solutions of PDEs, convection problems, diffusion problems, and elliptic problems. Problems are worked step-by-step with an explaination for each step. Material supports a Massive Open Online Course (MOOC).	https://github.com/numerical-mooc/nu NSE
https://github.com/jupyter/jupyter/wiki	Python, Jupyter Notebook	This page is a curated collection of Jupyter/IPython notebooks that are notable. Includes sections on engineering education, mathematics, physics, chemistry, and biology. Links are all rendered using nbviewer. Useful organic chemistry notebooks and more.	Contains links to code from many different authors; Licensing information is provided on a case by case basis
https://www.mathworks.com/help/examples.html	MATLAB	MathWorks Examples: Single Hydraulic Cylinder Simulation	
https://www.mathworks.com/academia/courseware.html	MATLAB	MathWorks Courseware; Subjects include Intro to Engineering, Chemistry, and Controls; Similar to cache.org but catering only to MathWorks software	
https://www.mathworks.com/products/matlab-grader.html	MATLAB	MathWorks Grader; Contains prebuilt problem sets for System Dynamics and Control, Statistics, Numerical Methods, Electrical Circuits, Calculus, etc	
https://www.mathworks.com/matlabcentral/fileexchange/	MATLAB	MATLAB file exhange: 290 MathWorks, 42890 Community files, 89 Tagged Chemical Engineering; Can sync with Github for distribution; https://www.mathworks.com/matlabcentral/content/fx/about.html	https://www.mathworks.com/matlabcentral/content/fx/fx-transition-faq.html
https://www.mathworks.com/products/matlab/live-script-gallery.html	MATLAB	MATLAB Live Script Gallery; Includes Tune PID Controller from Measured Plant Data, Chemical Kinetics, Heat Transfer in Pipes, etc; Can interact in browser or download from file exchange	https://www.mathworks.com/matlabcentral/content/fx/fx-transition-faq.html
https://matlabacademy.mathworks.com/	MATLAB	MATLAB Self-Paced Online Courses: Possible supplimental resource for students after tutorials	
https://github.com/CalebBell	Python	Thermo, Fluids, Heat Exchange, and chemical Github repositories; active development; Each repository contains a Python library and extensive documentation	All repositories appear to fall under the MIT License
https://github.com/chemics	Python	The Chemics package is a collection of Python functions for performing calculations in the field of chemical and fluidization engineering. Includes heat capacity, dimensionless numbers., molecular weight, etc. Includes source of reference data, extensive documentation, and some simple example problems	All repositories appear to fall under the MIT License
https://github.com/CAChemE/learn#chemical-and-process-engineering-interactive-simulations	Python, Jupyter Notebook	Interactive IPython/Jupyter notebooks with simulations for chemical and process engineering courses and posting them in this repository. These simulations will allow the user to change equation parameters —using just sliders and buttons— in order to obtain a better understanding of the system being modeled. Spanish and English sections.	https://github.com/CAChemE/learn/blob/master/LICENSE
https://cantera.org/	MATLAB, Python, C++, Fortran	Cantera is an open-source suite of tools for problems involving chemical kinetics, thermodynamics, and transport processes. Cantera can be used from Python and Matlab, or in applications written in C/C++ and Fortran 90.	https://github.com/Cantera/cantera/blob/main/License.txt
https://github.com/jkitchin/pycse	Python	Notes on using python in scientific and engineering calculations. The aim is to collect examples that span the types of computation/calculations scientists and engineers typically do to demonstrate the utility of python as a computational platform in engineering education. Includes links to similar MATLAB codes.	https://github.com/jkitchin/pycse/blob/master/LICENSE

Thank you! Any Questions?

Paper ID: 38671

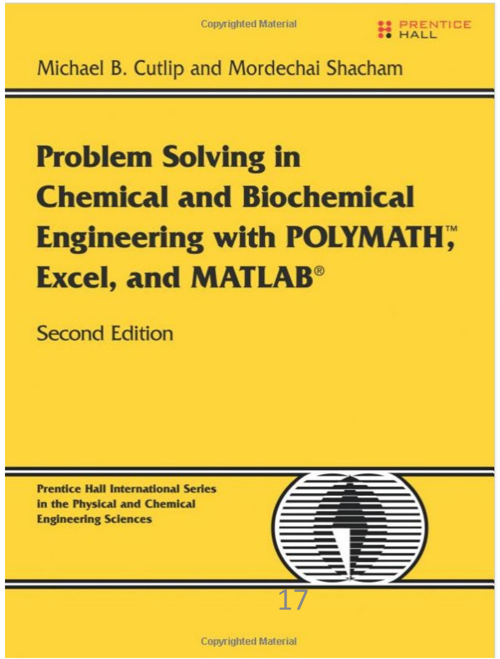
This material is based upon work supported by the National Science Foundation under Grant No. 1932723 (Stuber) and No. 2133411 (Ford-Versypt). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

This material is based upon work supported by the CACHE Organization Mini Grants awarded to Stuber and Ford-Versypt

Computer Aids for Chemical Engineering

Robert’s Use of Python in the Classroom

- Mass & Energy Balance (Felder, Rousseau, Bullard)
- ChE Fluid Mechanics (fluids 1)
- Process Fluid Transport (fluids 2)
- Separations I
- Chemical Reaction Engineering
- Transport Phenomena
- Grad Classes



Use Cases: Numerical Methods/Analysis

- Developed and deployed Live Scripts for UG chemical engineering numerical methods
  - Lectures, practice materials, and homework problem sets

Lecture #4 Linear Algebraic Systems, GE w/PP

Author: Matthew D. Stuber

Consider the following linear algebraic system $\mathbf{Ax} = \mathbf{b}$ with

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 2 & 3 & 1 & 5 \\ -1 & 1 & -5 & 3 \\ 3 & 1 & 7 & -2 \end{pmatrix}, \quad \mathbf{b} = (10, 31, -2, 18).$$

Now, let's put them into MATLAB arrays and create the augmented matrix that we'll perform Gauss elimination on.

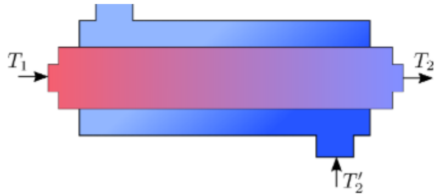
```
A = [1 1 1 1;2 3 1 5;-1 1 -5 3;3 1 7 -2];% A matrix
b = [10; 31; -2; 18];% b vector
n = length(b);% size of system
D = [A b];% Step 0, Augmented matrix
```

I've written code below to perform a single forward elimination step. Given the augmented matrix D, the iteration number k (pivot column), and the row number i that we wish to eliminate, a single forward elimination step is performed.

Forward Elimination

Now, let's manually loop and perform Step 1 of Gauss elimination.

```
k = 1 ;% Loop over pivot columns (k=1:n-1)
i = 2 ;% Loop over rows (i=k+1:n)
l = -D(i,k)/D(k,k);
D(i,:) = -D(i,k)/D(k,k)*D(k,:)+D(i,:);
fprintf('The multiplier is %3.1f\n',l)
fprintf('Row %i is D(%i,:)=[%3.1f %3.1f %3.1f %3.1f %3.1f]\n',i,i,D(i,:))
```



In this unit operation, we are cooling the (hot) inner stream with the (cold) outer stream. The prime symbols indicate the outer stream properties. The design equation for this operation is

$$Q = UA\Delta T_{lm},$$

where Q is the heat/energy transferred from the hot to the cold stream, U is the overall heat transfer coefficient, A is the area over which the heat transfer occurs, and

$$\Delta T_{lm} = \frac{(T_2 - T_2') - (T_1 - T_1')}{\log(T_2 - T_2') - \log(T_1 - T_1')}$$

is called the log-mean temperature difference, used to define the effective driving force for heat transfer in the system. The First Law of Thermodynamics requires the following energy balance

$$Q = \dot{m}C_p(T_1 - T_2) = \dot{m}'C_p'(T_1' - T_2')$$

where $\dot{m}$ and $\dot{m}'$ are the mass flowrates of the hot (inner) and cold (outer) streams, respectively, and the heat capacities or the corresponding fluids are given by C_p and C_p', respectively. Note that since $T_1 > T_2$ and $T_1' > T_2'$, we have defined Q to be a positive quantity.

Part (a)

We'll start with a simple analysis of the system. The hot fluid has a flowrate of 1.20 kg/s and a heat capacity of 2.236 kJ/kg-K. The cold fluid has a flowrate of 3.80 kg/s and a heat capacity of 4.184 kJ/kg-K. The heat exchanger cools the hot stream from 100°C to 50°C and the cooling stream enters at 15°C. The overall heat transfer coefficient is 1.25 kW/m^2-K. Write a MATLAB subroutine that calculates and prints out the outlet temperature of the cooling fluid, the log-mean temperature difference, and the heat transfer area.

```
% Model parameters common to all parts:
% counter-current heat exchanger notation
% m, Cp = center fluid
% mp, Cpp = cooling fluid
```