Chemical Engineering 512

RELAP5-3D

Lecture 12

Batch Files and Python Interfacing

BYU

Spiritual Thought

They **always return home** – Female sea turtles return to the exact beach **where they were born** to lay their eggs, even after traveling thousands of miles in the ocean.

My dear brothers and sisters, my dear friends, I testify that God sees us as we truly are—and He sees us worthy of rescue.

You may feel that your life is in ruins. You may have sinned. You may be afraid, angry, grieving, or tortured by doubt. But just as the Good Shepherd finds His lost sheep, if you will only lift up your heart to the Savior of the world, He will find you.

He will rescue you.

He will lift you up and place you on His shoulders.

He will carry you home.

- Dieter F. Uchtdorf, April 2016

Objectives

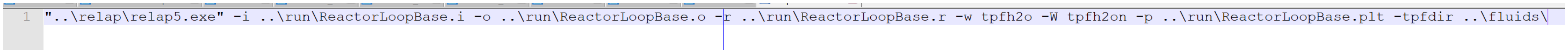
- Batch Files
- Practice Problem
- Python Interfacing

Batch Files – What They Are

- A Microsoft Windows script file
- File extension - .bat
- Can be written with Notepad ++ (or any other text editor)
- Command line script can be used

Batch Files and RELAP5-3D

- The RGUIStationJar is an interface to allow you to input file names and then runs a batch file
- RELAP/r3d434ie/relap/relapcommand.bat
- Batch files can be used to run multiple RELAP runs much more quickly



```
"..\relap\relap5.exe" -i ..\run\ReactorLoopBase.i -o ..\run\ReactorLoopBase.o -r ..\run\ReactorLoopBase.r -w tpfh2o -W tpfh2on -p ..\run\ReactorLoopBase.plt -tpfdir ..\fluids\
```

Create a batch file to run RELAP

- RELAP command (if preformed in the relap folder)
- "relap5.exe" -i file.i -o file.o -r file.r -w tpfh2o –W tpfh2on –p file.plt -tpfdir ../fluids/

- Delete a file in windows command line
- del /f file.o

How does Python come into all of this

- We can rapidly create multiple RELAP input decks, run them, and pull in results data all in one python script
- Why is this valuable?

Python Integration Practice

- Write a python script to do the following:
  - Delete output file
  - Execute RELAP command

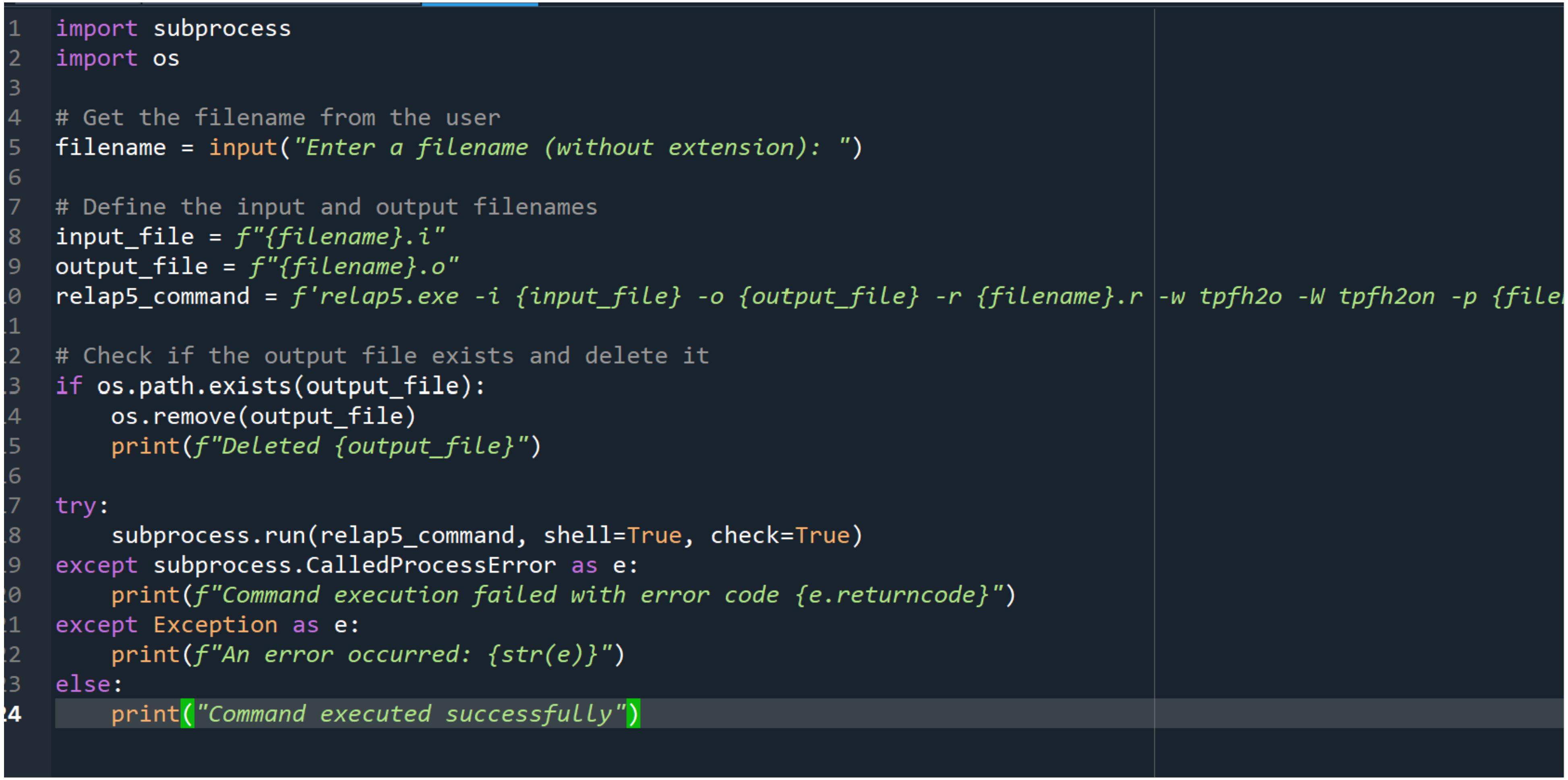
RELAP command (if preformed in the relap folder)

```
"relap5.exe" -i file.i -o file.o -r file.r -w tpfh2o –W tpfh2on –p
file.plt -tpfdir ../fluids/
```

Delete a file in windows command line

```
del /f file.o
```

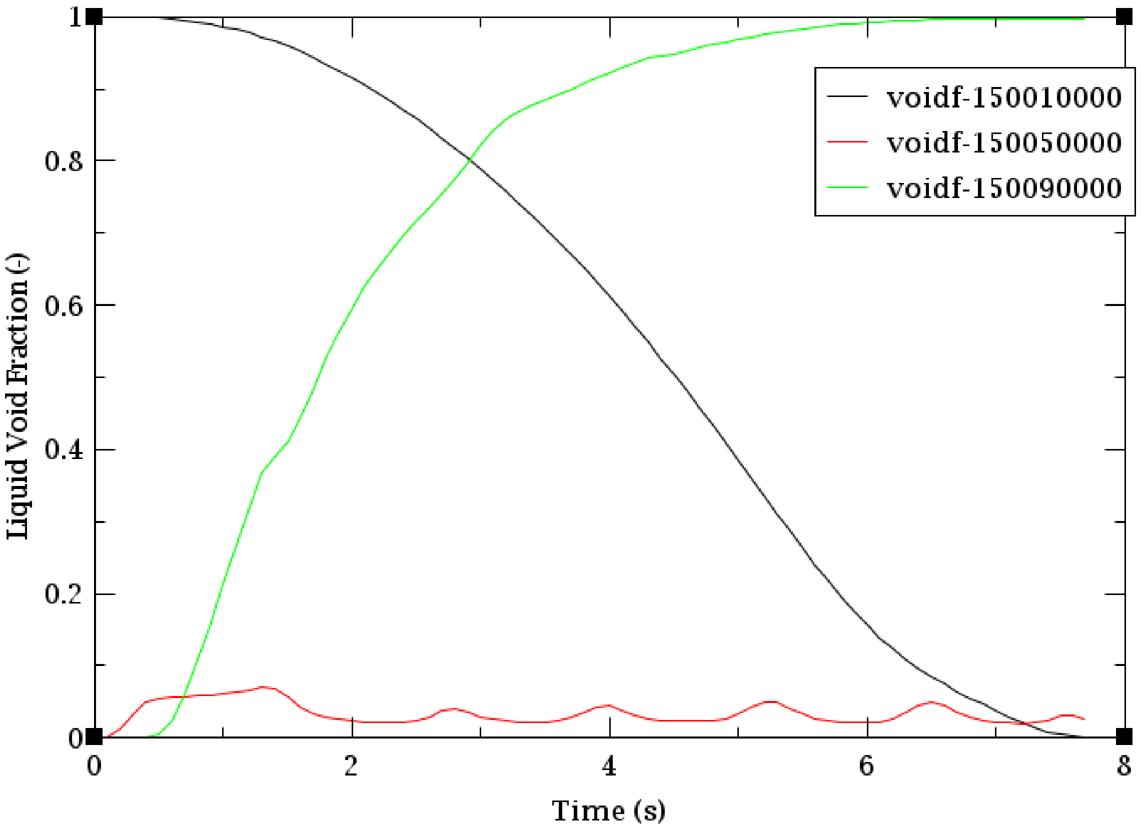
Python Integration Practice

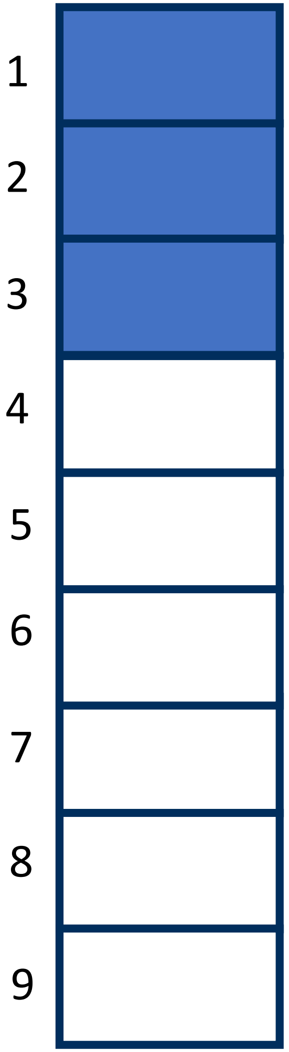


```python
import subprocess
import os

# Get the filename from the user
filename = input("Enter a filename (without extension): ")

# Define the input and output filenames
input_file = f"{filename}.i"
output_file = f"{filename}.o"
relap5_command = f'relap5.exe -i {input_file} -o {output_file} -r {filename}.r -w tpfh2o -W tpfh2on -p {file

# Check if the output file exists and delete it
if os.path.exists(output_file):
    os.remove(output_file)
    print(f"Deleted {output_file}")

try:
    subprocess.run(relap5_command, shell=True, check=True)
except subprocess.CalledProcessError as e:
    print(f"Command execution failed with error code {e.returncode}")
except Exception as e:
    print(f"An error occurred: {str(e)}")
else:
    print("Command executed successfully")
```

Practice: Nodalization Study

- Remember HW 2?



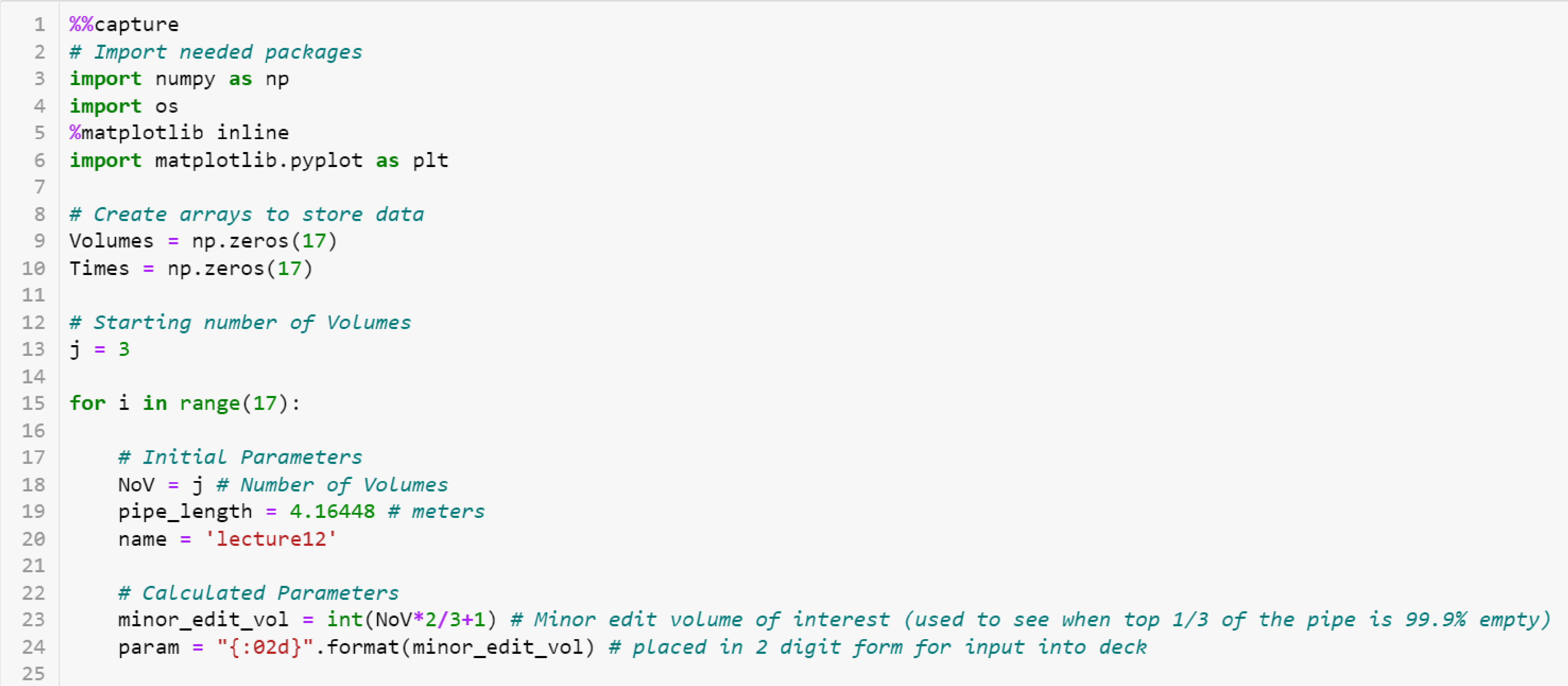


Practice: Nodalization Study

- What is the optimum number of volumes?
- Time step has already been set to be less than Courant limit
- To compare results we will plot the time it takes for all of the water to leave the top 1/3 of the pipe

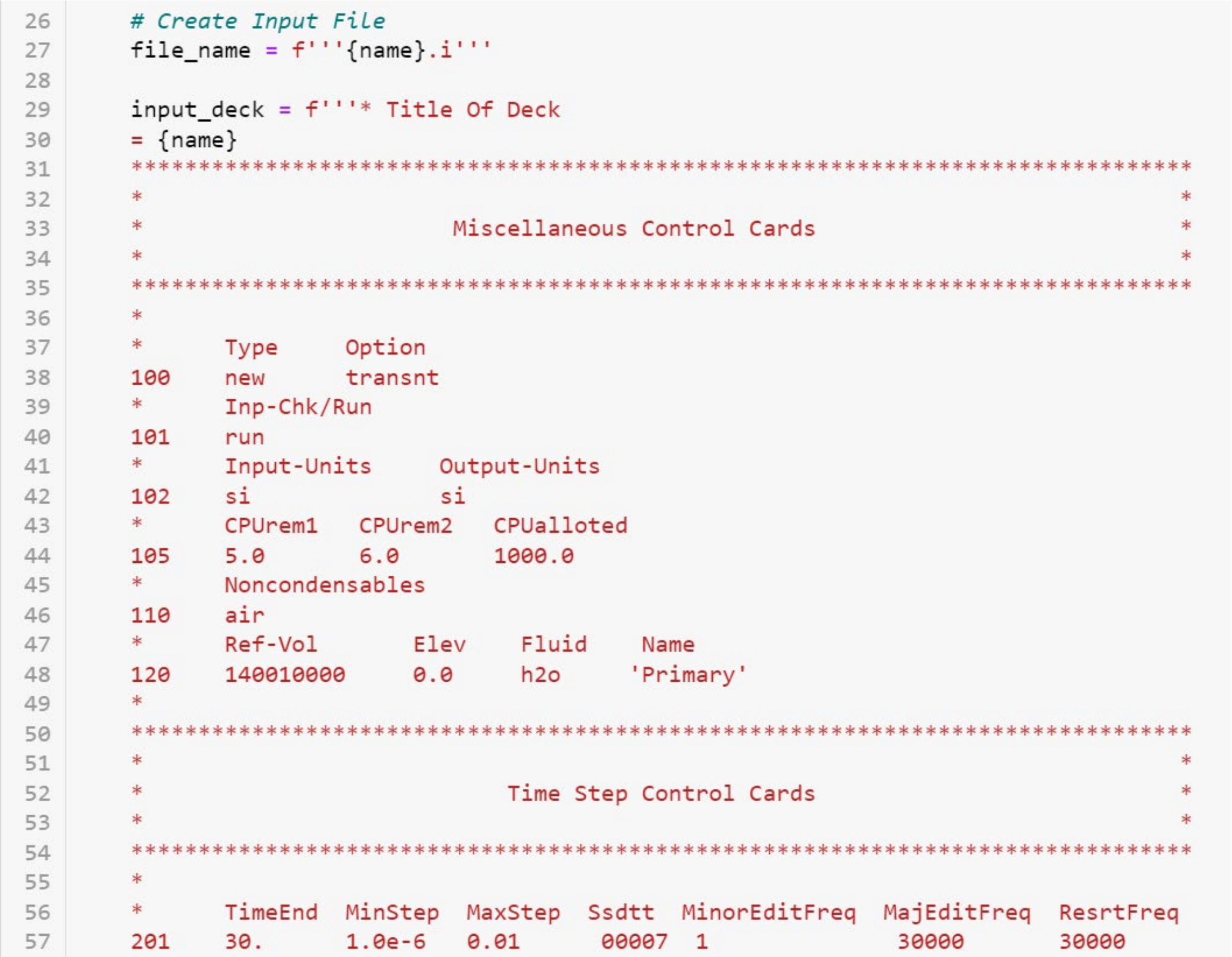
- (19 RELAP experts)*(10 min) = 3.17 hours of manpower
- (1 python script)*(18 sec) = 0.005 hours of machinepower

Let's take a look at the code



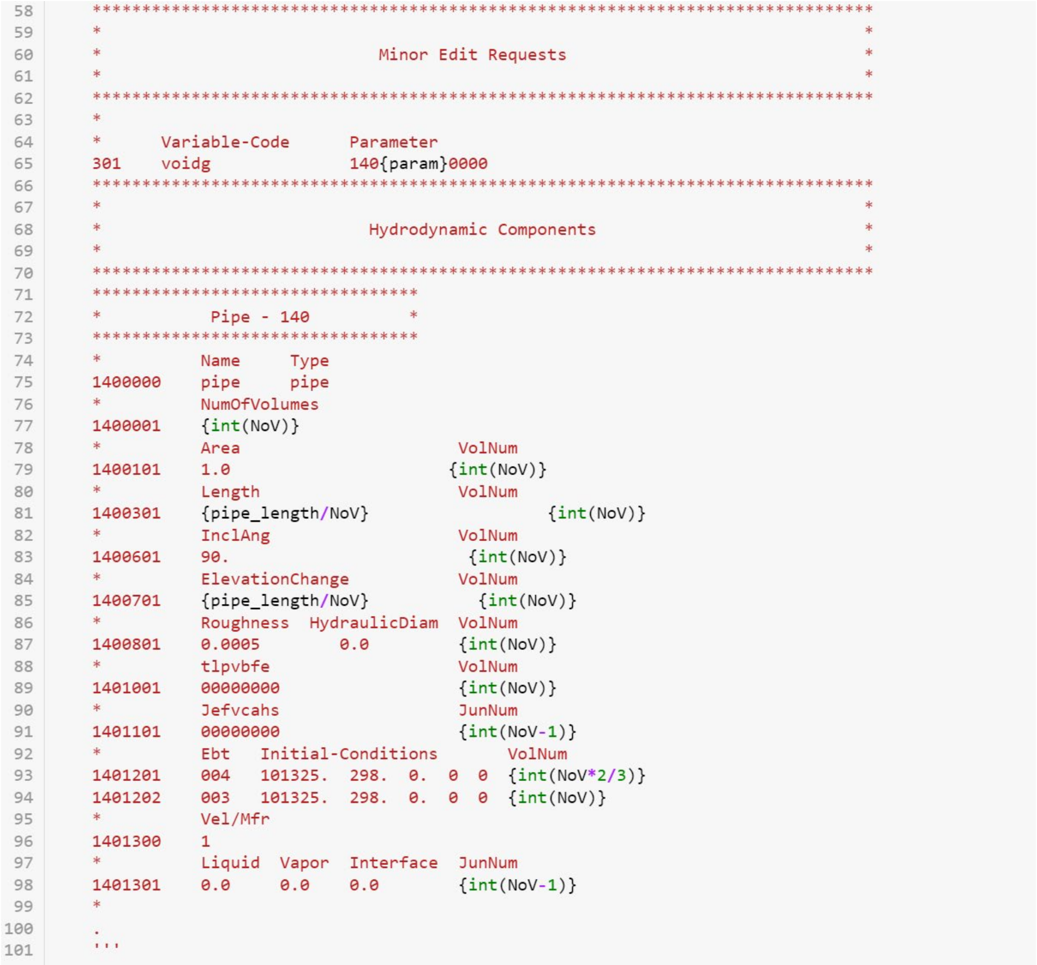
```
%%capture
# Import needed packages
import numpy as np
import os
%matplotlib inline
import matplotlib.pyplot as plt

# Create arrays to store data
Volumes = np.zeros(17)
Times = np.zeros(17)

# Starting number of Volumes
j = 3

for i in range(17):

    # Initial Parameters
    NoV = j # Number of Volumes
    pipe_length = 4.16448 # meters
    name = 'lecture12'

    # Calculated Parameters
    minor_edit_vol = int(NoV*2/3+1) # Minor edit volume of interest (used to see when top 1/3 of the pipe is 99.9% empty)
    param = "{:02d}".format(minor_edit_vol) # placed in 2 digit form for input into deck

```

Creating an input deck



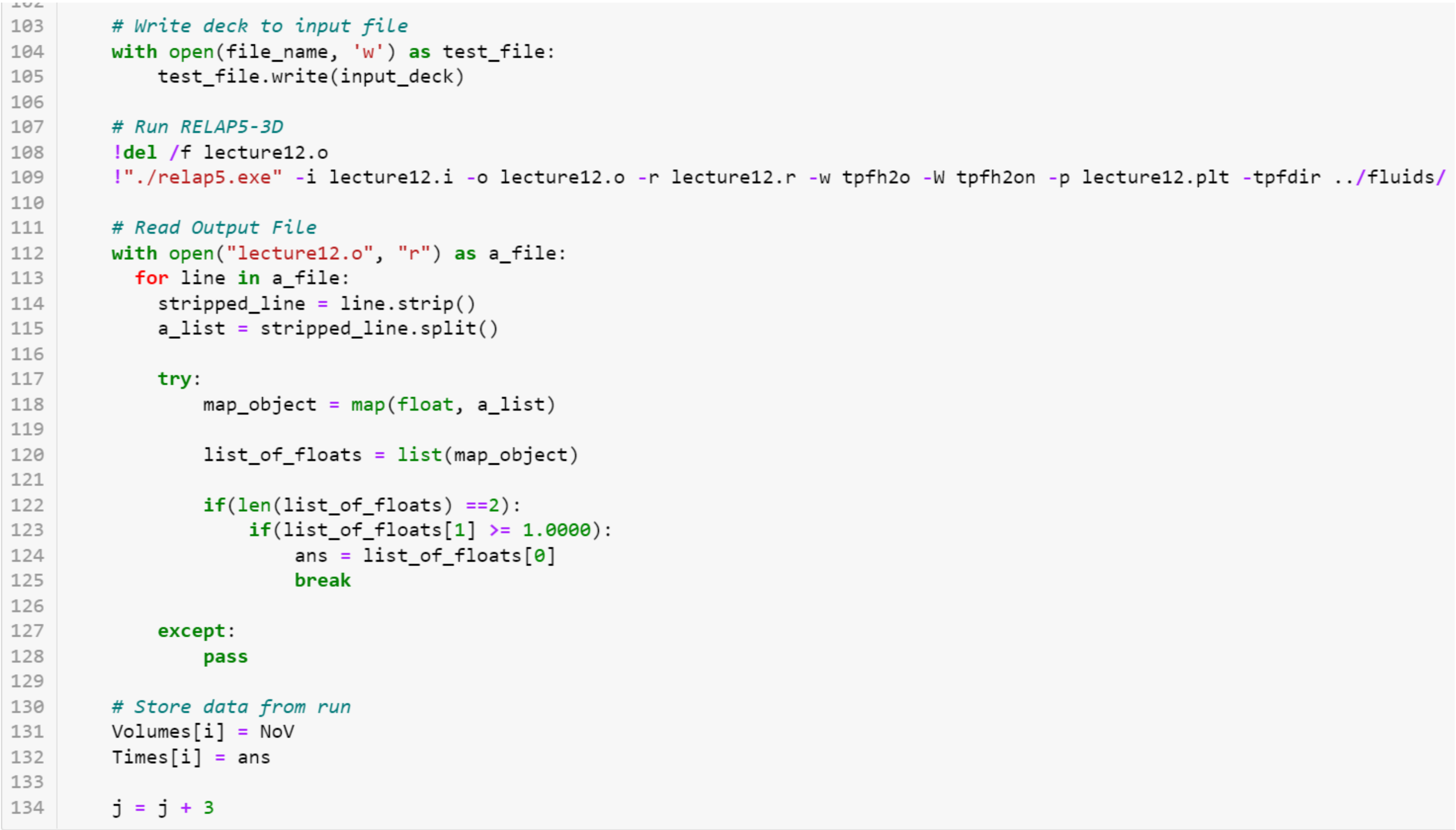
```
# Create Input File
file_name = f'''{name}.i'''

input_deck = f'''* Title Of Deck
= {name}
*******************************************************************************
*                                                                             *
*                       Miscellaneous Control Cards                           *
*                                                                             *
*******************************************************************************
*
*      Type     Option
100    new      transnt
*      Inp-Chk/Run
101    run
*      Input-Units     Output-Units
102    si              si
*      CPUrem1   CPUrem2   CPUalloted
105    5.0       6.0       1000.0
*      Noncondensables
110    air
*      Ref-Vol       Elev    Fluid    Name
120    140010000     0.0     h2o     'Primary'
*
*******************************************************************************
*                                                                             *
*                           Time Step Control Cards                           *
*                                                                             *
*******************************************************************************
*
*      TimeEnd  MinStep  MaxStep  Ssdtt  MinorEditFreq  MajEditFreq  ResrtFreq
201    30.      1.0e-6   0.01      00007  1              30000       30000
```

Changing values in the input deck



```
*******************************************************************************
*                                                                             *
*                            Minor Edit Requests                              *
*                                                                             *
*******************************************************************************
*
*      Variable-Code      Parameter
301    voidg              140{param}0000
*******************************************************************************
*                                                                             *
*                           Hydrodynamic Components                           *
*                                                                             *
*******************************************************************************
*********************************
*           Pipe - 140          *
*********************************
*          Name     Type
1400000    pipe     pipe
*          NumOfVolumes
1400001    {int(NoV)}
*          Area                      VolNum
1400101    1.0                      {int(NoV)}
*          Length                    VolNum
1400301    {pipe_length/NoV}                  {int(NoV)}
*          InclAng                   VolNum
1400601    90.                        {int(NoV)}
*          ElevationChange           VolNum
1400701    {pipe_length/NoV}           {int(NoV)}
*          Roughness  HydraulicDiam  VolNum
1400801    0.0005        0.0         {int(NoV)}
*          tlpvbfe                   VolNum
1401001    00000000                  {int(NoV)}
*          Jefvcahs                  JunNum
1401101    00000000                  {int(NoV-1)}
*          Ebt   Initial-Conditions       VolNum
1401201    004   101325.  298.  0.  0  0  {int(NoV*2/3)}
1401202    003   101325.  298.  0.  0  0  {int(NoV)}
*          Vel/Mfr
1401300    1
*          Liquid  Vapor  Interface  JunNum
1401301    0.0     0.0    0.0        {int(NoV-1)}
*
.
'''
```

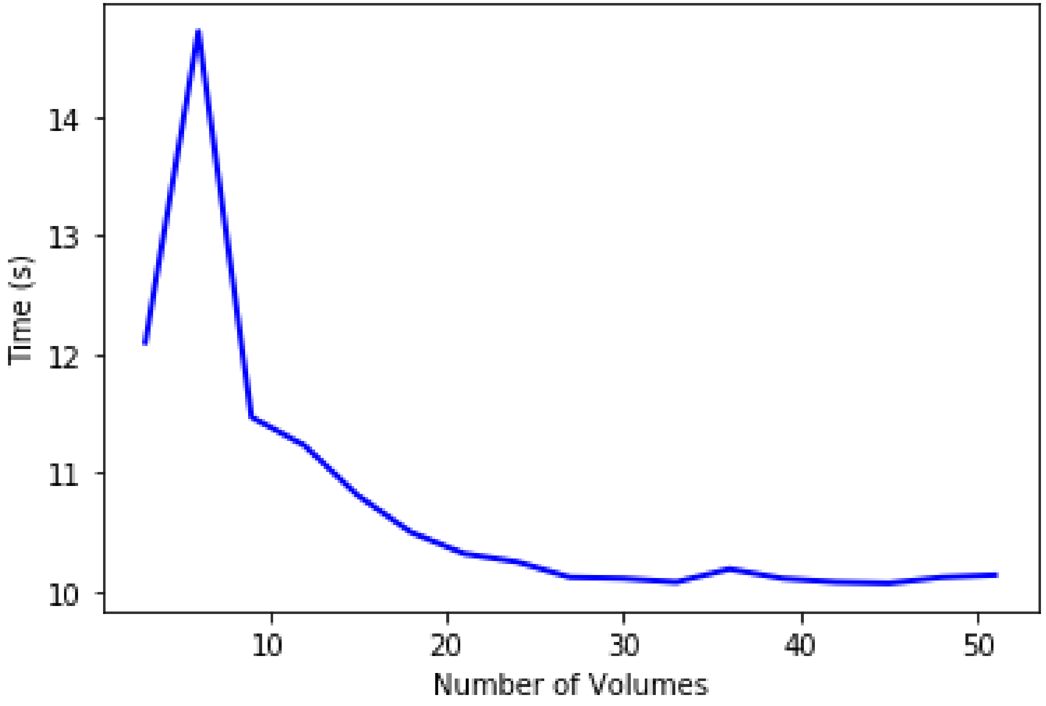
Write deck, run deck, read results



```
    # Write deck to input file
    with open(file_name, 'w') as test_file:
        test_file.write(input_deck)

    # Run RELAP5-3D
    !del /f lecture12.o
    !"./relap5.exe" -i lecture12.i -o lecture12.o -r lecture12.r -w tpfh2o -W tpfh2on -p lecture12.plt -tpfdir ../fluids/

    # Read Output File
    with open("lecture12.o", "r") as a_file:
      for line in a_file:
        stripped_line = line.strip()
        a_list = stripped_line.split()

        try:
            map_object = map(float, a_list)

            list_of_floats = list(map_object)

            if(len(list_of_floats) ==2):
                if(list_of_floats[1] >= 1.0000):
                    ans = list_of_floats[0]
                    break

        except:
            pass

    # Store data from run
    Volumes[i] = NoV
    Times[i] = ans

    j = j + 3
```

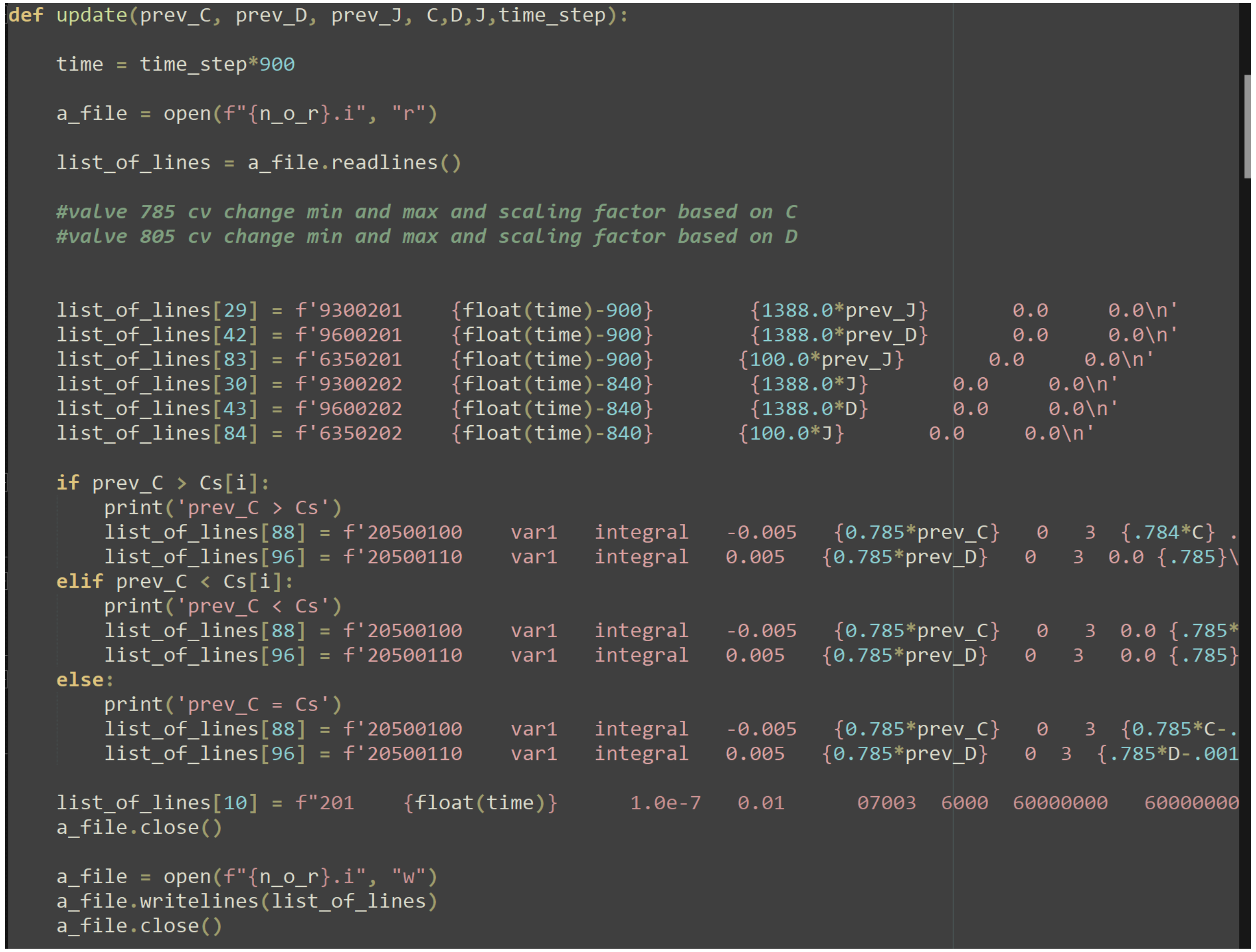
BYU

Results



Brainstorming

- If this is possible, what else is possible?

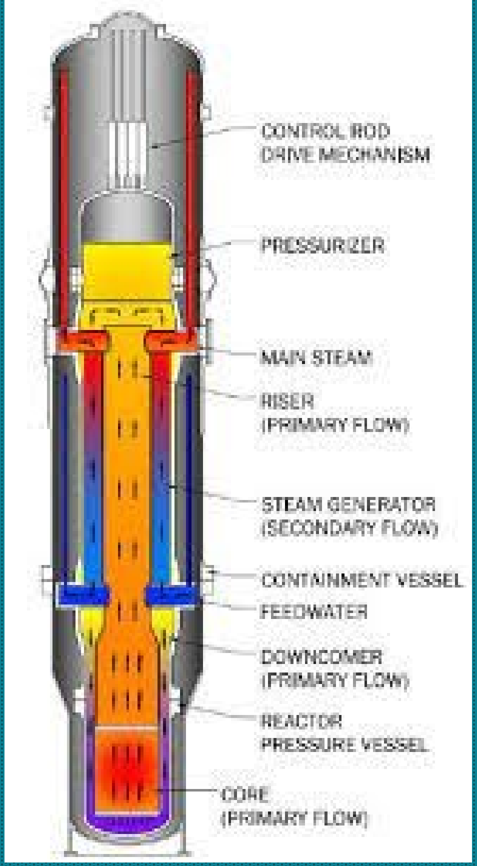


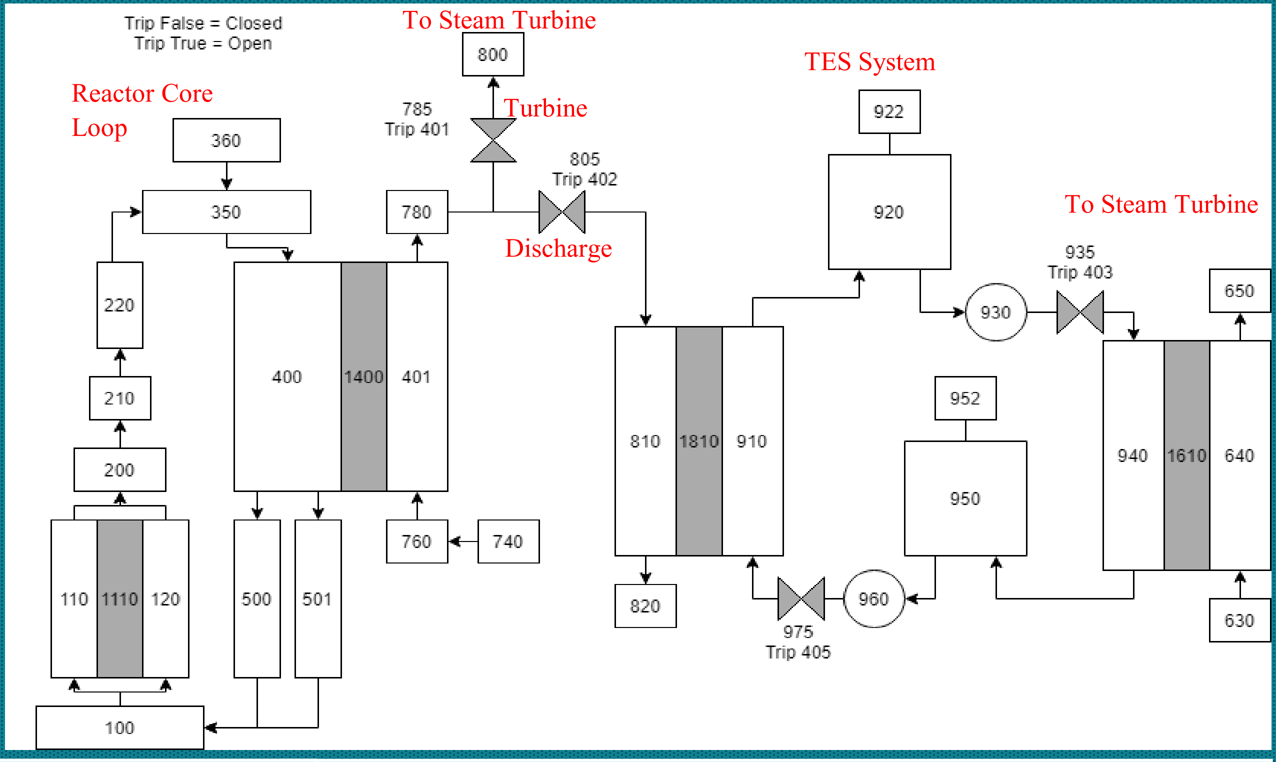
```
def update(prev_C, prev_D, prev_J, C,D,J,time_step):

    time = time_step*900

    a_file = open(f"{n_o_r}.i", "r")

    list_of_lines = a_file.readlines()

    #valve 785 cv change min and max and scaling factor based on C
    #valve 805 cv change min and max and scaling factor based on D


    list_of_lines[29] = f'9300201    {float(time)-900}        {1388.0*prev_J}       0.0     0.0\n'
    list_of_lines[42] = f'9600201    {float(time)-900}        {1388.0*prev_D}       0.0     0.0\n'
    list_of_lines[83] = f'6350201    {float(time)-900}       {100.0*prev_J}       0.0     0.0\n'
    list_of_lines[30] = f'9300202    {float(time)-840}        {1388.0*J}       0.0     0.0\n'
    list_of_lines[43] = f'9600202    {float(time)-840}        {1388.0*D}       0.0     0.0\n'
    list_of_lines[84] = f'6350202    {float(time)-840}       {100.0*J}       0.0     0.0\n'

    if prev_C > Cs[i]:
        print('prev_C > Cs')
        list_of_lines[88] = f'20500100    var1   integral   -0.005   {0.785*prev_C}   0   3  {.784*C} .
        list_of_lines[96] = f'20500110    var1   integral   0.005   {0.785*prev_D}   0   3  0.0 {.785}\
    elif prev_C < Cs[i]:
        print('prev_C < Cs')
        list_of_lines[88] = f'20500100    var1   integral   -0.005   {0.785*prev_C}   0   3  0.0 {.785*
        list_of_lines[96] = f'20500110    var1   integral   0.005   {0.785*prev_D}   0   3   0.0 {.785}
    else:
        print('prev_C = Cs')
        list_of_lines[88] = f'20500100    var1   integral   -0.005   {0.785*prev_C}   0   3  {0.785*C-.
        list_of_lines[96] = f'20500110    var1   integral   0.005   {0.785*prev_D}   0  3  {.785*D-.001

    list_of_lines[10] = f"201    {float(time)}      1.0e-7   0.01      07003  6000  60000000   60000000

    a_file.close()

    a_file = open(f"{n_o_r}.i", "w")
    a_file.writelines(list_of_lines)
    a_file.close()
```

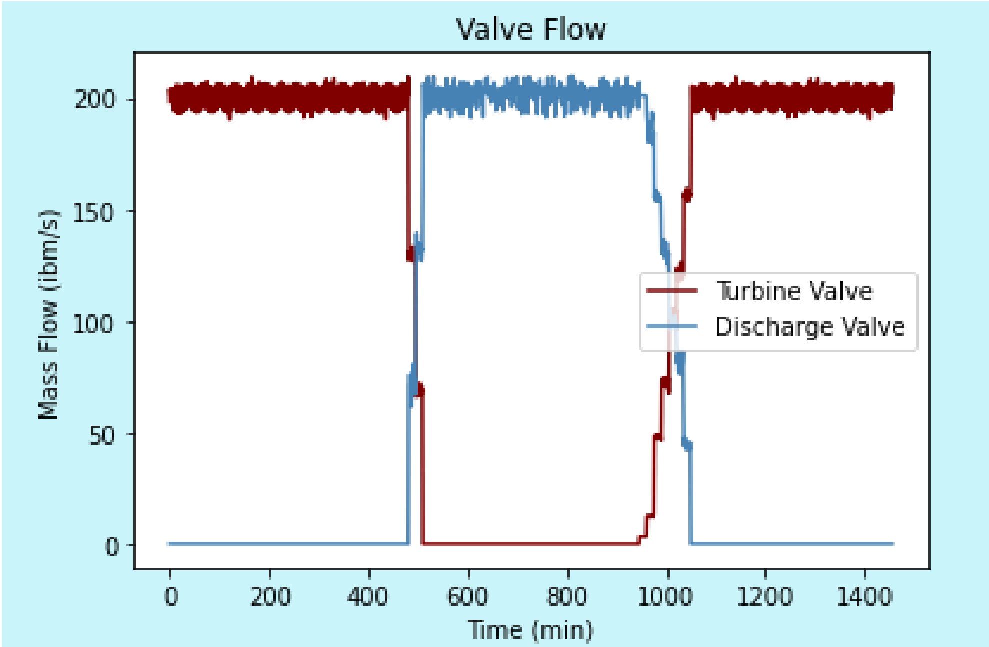
Thermal Energy Storage

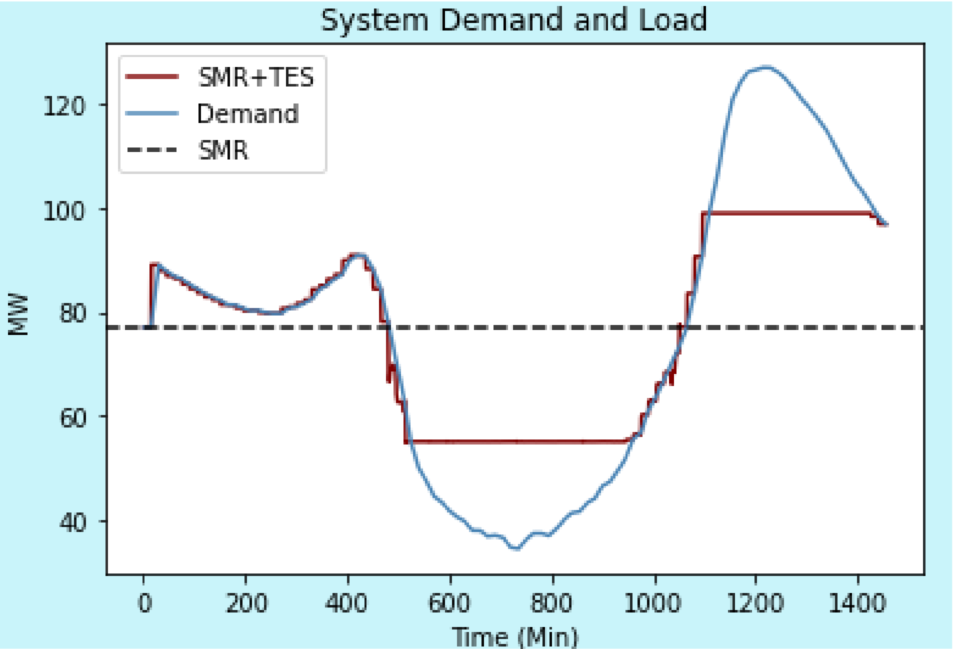
Being a baseload power, keeping up with high energy demand is a problem for nuclear reactors. The thermal energy storage systems are a proposed solution to better meet demand at all times of the day and night. The control of these systems is an important feature that requires extensive study for these reactors to be licensed with a storage system. In this work, a model predictive controller matches the load of the entire system with the demand of the power grid over a 24-hour period.

NuScale SMR



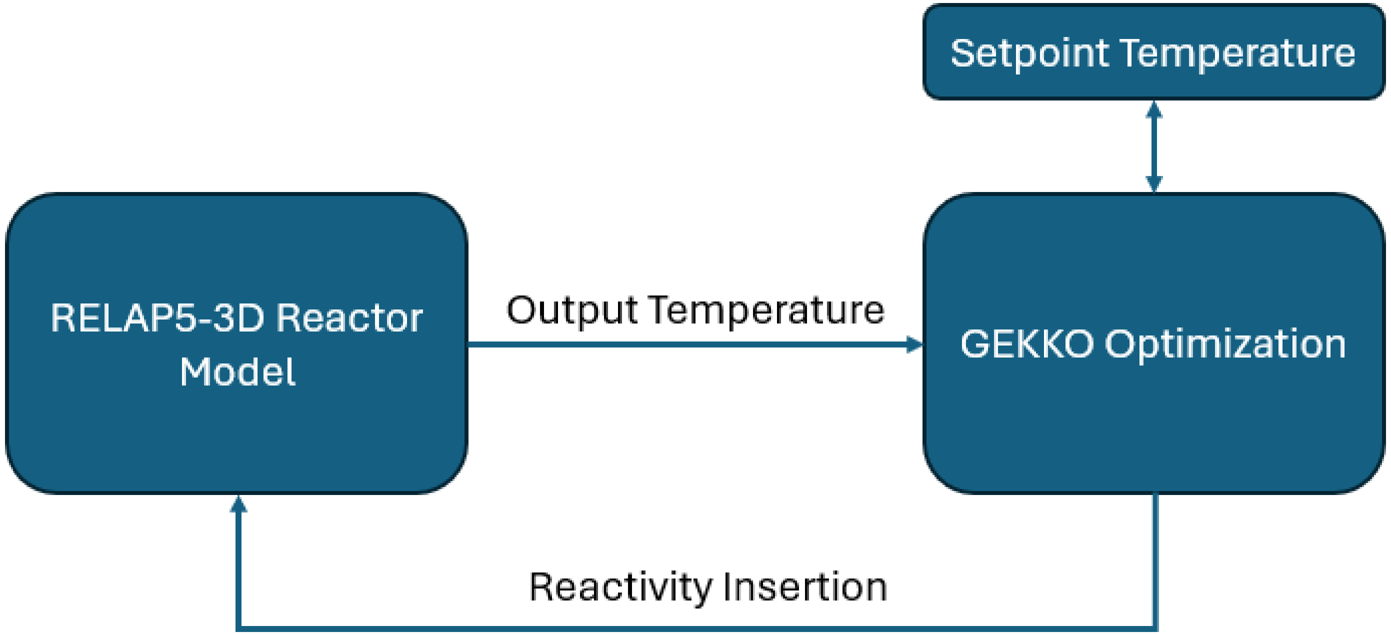


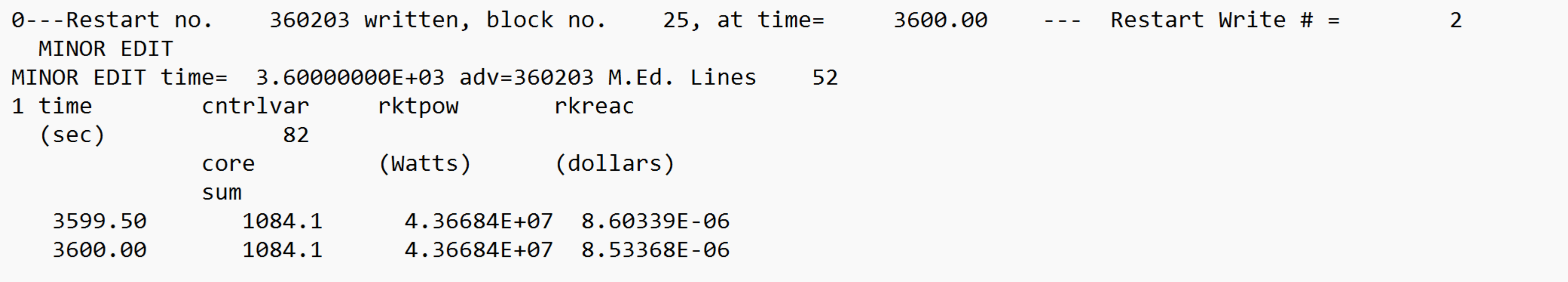




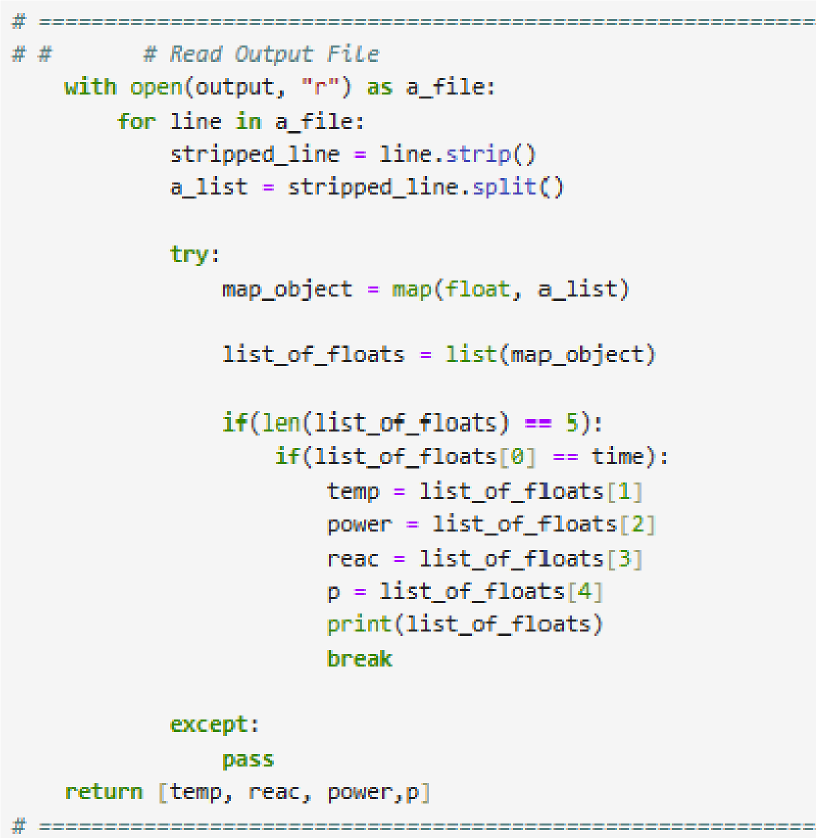
Brainstorming

- If this is possible, what else is possible?

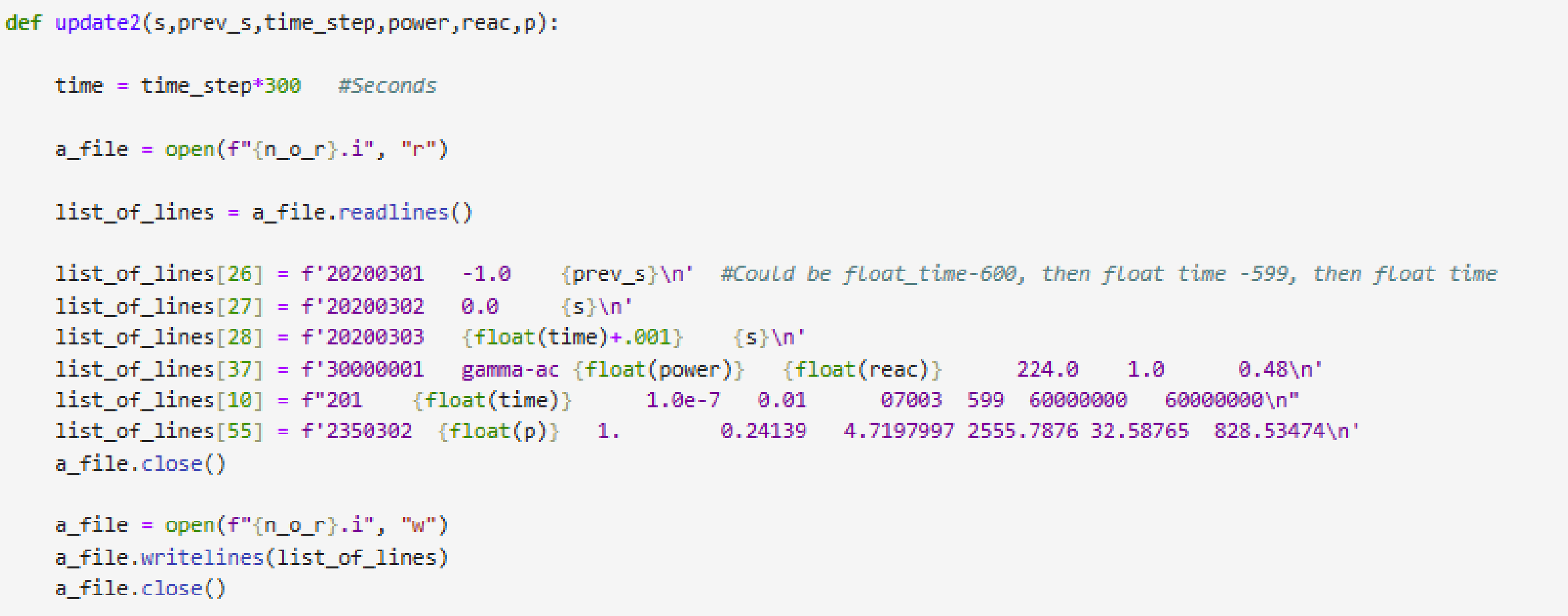




```
0---Restart no.    360203 written, block no.    25, at time=     3600.00    ---  Restart Write # =        2
  MINOR EDIT
MINOR EDIT time=  3.60000000E+03 adv=360203 M.Ed. Lines    52
1 time          cntrlvar     rktpow       rkreac
  (sec)               82
              core         (Watts)      (dollars)
              sum
   3599.50       1084.1      4.36684E+07   8.60339E-06
   3600.00       1084.1      4.36684E+07   8.53368E-06
```

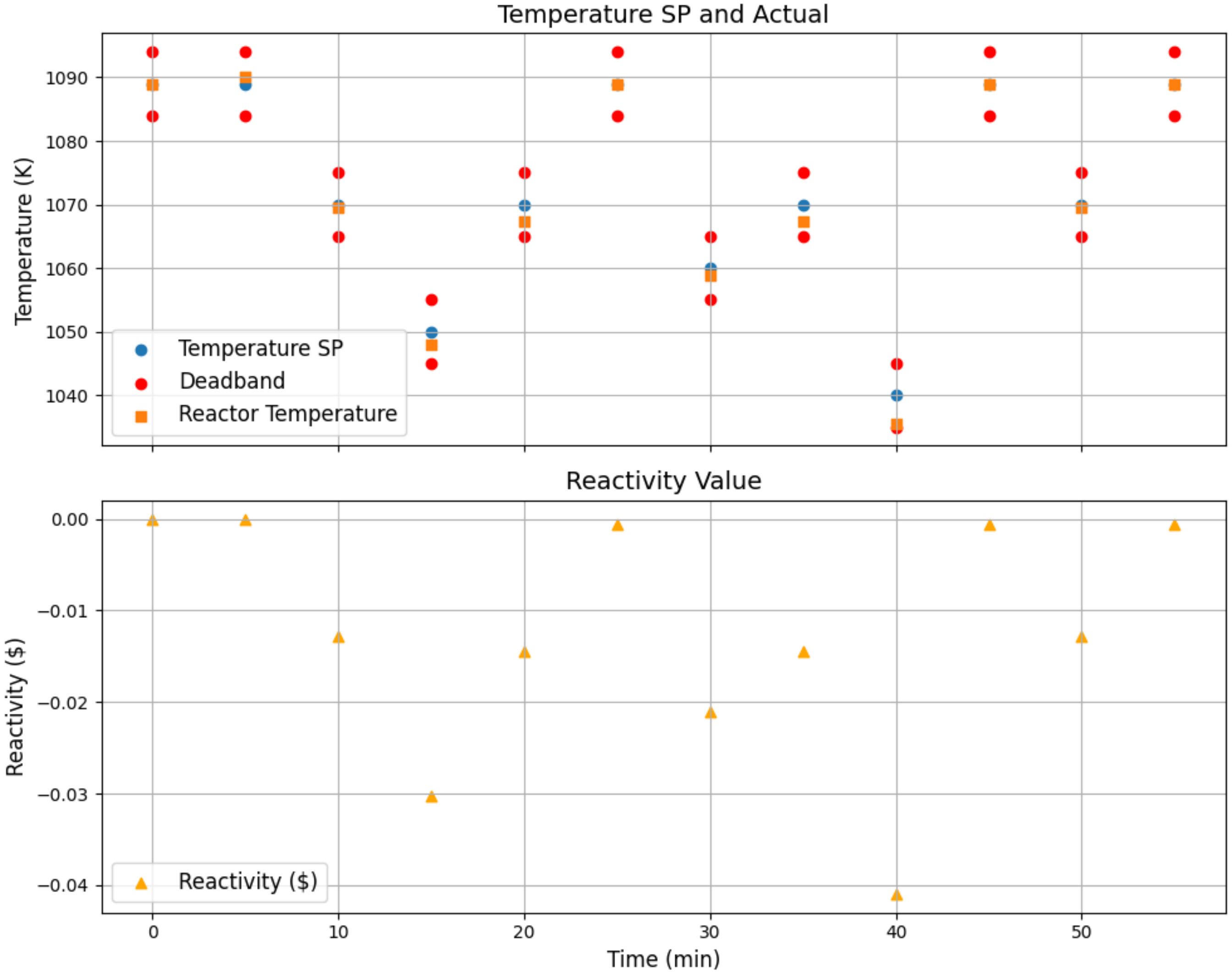


```
# ==============================================================================
# #       # Read Output File
    with open(output, "r") as a_file:
        for line in a_file:
            stripped_line = line.strip()
            a_list = stripped_line.split()

            try:
                map_object = map(float, a_list)

                list_of_floats = list(map_object)

                if(len(list_of_floats) == 5):
                    if(list_of_floats[0] == time):
                        temp = list_of_floats[1]
                        power = list_of_floats[2]
                        reac = list_of_floats[3]
                        p = list_of_floats[4]
                        print(list_of_floats)
                        break

            except:
                pass
    return [temp, reac, power,p]
# ==============================================================================
```

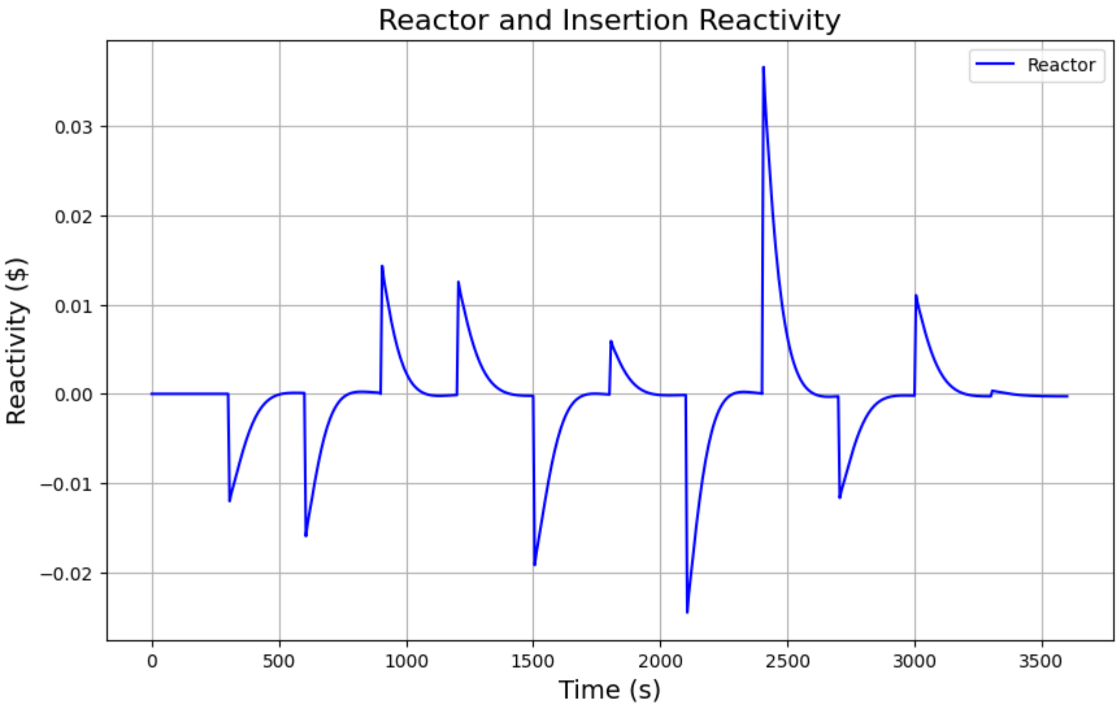


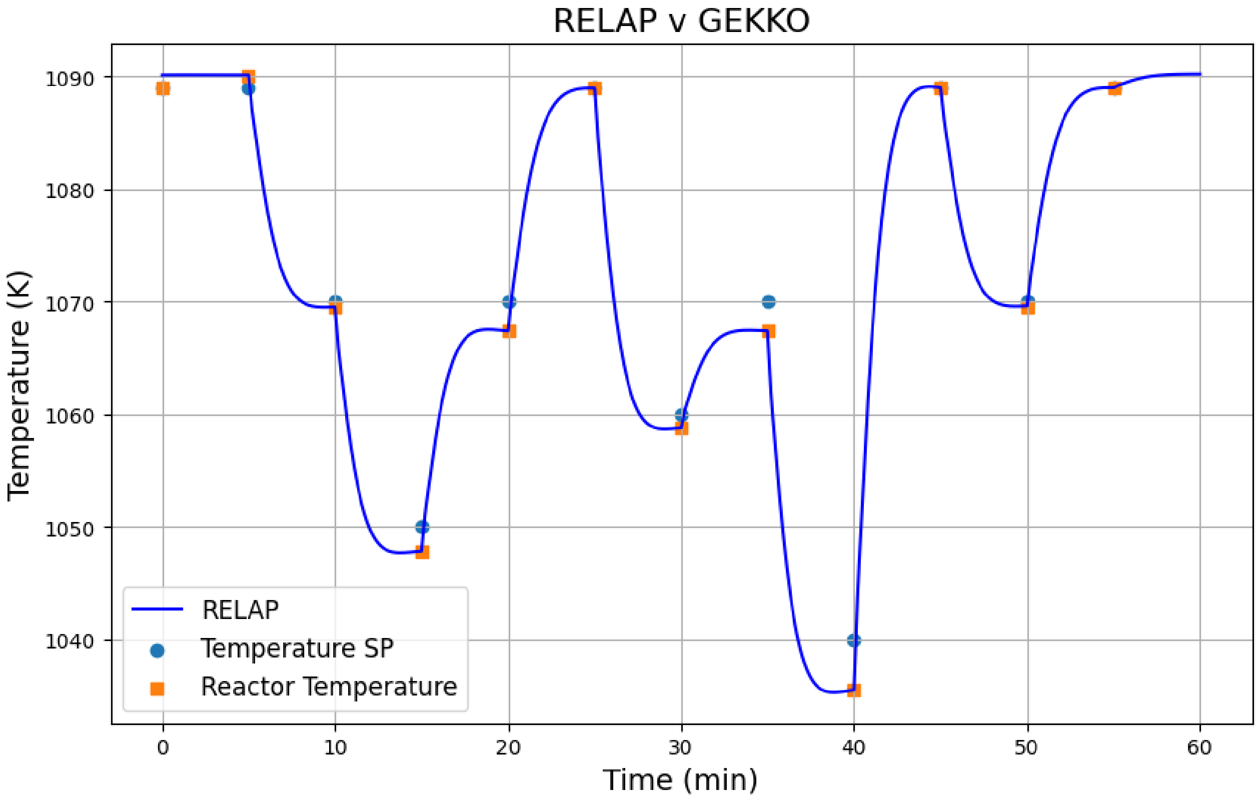
```
def update2(s,prev_s,time_step,power,reac,p):

    time = time_step*300   #Seconds

    a_file = open(f"{n_o_r}.i", "r")

    list_of_lines = a_file.readlines()

    list_of_lines[26] = f'20200301   -1.0    {prev_s}\n'  #Could be float_time-600, then float time -599, then float time
    list_of_lines[27] = f'20200302   0.0     {s}\n'
    list_of_lines[28] = f'20200303   {float(time)+.001}    {s}\n'
    list_of_lines[37] = f'30000001   gamma-ac {float(power)}   {float(reac)}      224.0    1.0      0.48\n'
    list_of_lines[10] = f"201     {float(time)}      1.0e-7    0.01      07003  599  60000000   60000000\n"
    list_of_lines[55] = f'2350302  {float(p)}   1.       0.24139    4.7197997 2555.7876 32.58765  828.53474\n'
    a_file.close()

    a_file = open(f"{n_o_r}.i", "w")
    a_file.writelines(list_of_lines)
    a_file.close()
```

Brainstorming



Brainstorming





Assignment

- Watch DVD sections 64-71 before next class
- HW 6 due Friday (10/17)