Finite Element Simulation of 1-D Heat Transport in a Pipe

1 Recap of the Problem

A tubular reactor used for chemical reactions under high-temperature conditions has an inner diameter of 5 cm and a wall thickness of 2.5 cm, made from ceramic material. To ensure reaction efficiency at high temperatures and minimize heat loss, the reactor is externally wrapped with an insulation layer that is 5 cm thick.

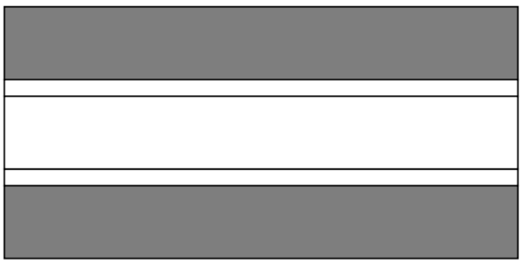

Fig. 1. Illustration

Material Properties

Ceramic Material:

- Density $\rho_1 = 2600\, kg/m^3$
- Specific Heat Capacity $c_1 = 1150\, J/(kgK)$
- Thermal Conductivity $k_1 = 3.0\, W/(mK)$

Insulation Material:

- Density $\rho_2 = 600\, kg/m^3$
- Specific Heat Capacity $c_2 = 200\, J/(kgK)$
- Thermal Conductivity $k_2 = 0.2\, W/(mK)$

Initial and Boundary Conditions

- The initial temperature of the reactor is 300 K.
- The inner surface is suddenly exposed to a reactant flow at 1500 K, with a convective heat transfer coefficient between the inner surface and the reactant flow of 500 W/(m^2·K)
- The outer surface of the insulation layer is in contact with an air flow at 298 K, with a convective heat transfer coefficient between the air flow and the insulation layer of 10 W/(m^2·K).

Questions Please provide the following: 1. The variation over time of the inner surface temperature of the insulation layer. 2. The temperature distribution within the insulation layer. 3. Discuss the impact of the insulation layer thickness on the above results.

Methodological Requirements

- Use numerical methods for solving the problem.
- Clearly document the calculation process.
- Provide a runnable calculation program (e.g., an m-file for MATLAB).
- Offer necessary analysis of the obtained results.

2 Problem Analysis

2.1 The Simplification of Enengy Equation

The general heat transfer equation is:

$$\rho \frac{DH}{Dt} = \frac{Dp}{Dt} + k\nabla^2 T + \phi + \dot{q}$$

Where ϕ is the **Dissipation factor**, it is usually very small. Exceptions include high viscosity fluid or high speed flow.

$$\phi := \mu[2(\frac{\partial u_x}{\partial x})^2 + 2(\frac{\partial u_y}{\partial y})^2 + 2(\frac{\partial u_z}{\partial z})^2 + (\frac{\partial u_x}{\partial y} + \frac{\partial u_y}{\partial x})^2) + (\frac{\partial u_x}{\partial z} + \frac{\partial u_z}{\partial x})^2) + (\frac{\partial u_z}{\partial y} + \frac{\partial u_y}{\partial z})^2) - \frac{2}{3}\mu(\nabla \cdot \mathbf{u})^2]$$

For uncompressible fluid(soild), $\nabla \mathbf{u} = 0, c_v \approx c_p, H = U + p/\rho$, and we assume the k, c_p are constant.

$$\rho \frac{DH}{Dt} - \frac{Dp}{Dt} = \rho \frac{DU}{Dt} \approx \rho c_p \frac{DT}{Dt} = \rho c_p \frac{\partial T}{\partial t}$$

$$\rho c_p \frac{\partial T}{\partial t} = k\nabla^2 T + \dot{q}$$

$$\frac{\partial T}{\partial t} = \alpha\nabla^2 T + \frac{\dot{q}}{\rho c_p},\ \alpha = \frac{k}{\rho c_p}$$

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \nabla^2 T + \frac{\dot{q}}{k},\ \alpha = \frac{k}{\rho c_p}$$

In the aforementioned problem, there is no heat generation $\dot{q} = 0$, thus the final equation is:

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \nabla^2 T,\ \alpha = \frac{k}{\rho c_p}$$

Cylindrical coordinates can be applied to the pipe:

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}(r\frac{\partial T}{\partial r}) + \frac{1}{r^2}\frac{\partial^2 T}{\partial \theta^2} + \frac{\partial^2 T}{\partial z^2}$$

As symmetry indicates $\frac{\partial}{\partial \theta} = \frac{\partial}{\partial z} = 0$:

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \frac{1}{r}\frac{\partial}{\partial r}(r\frac{\partial T}{\partial r})$$

Now we have the final equation：

$$\frac{1}{\alpha}\frac{\partial T}{\partial t} = \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial r^2}$$

2.2 Initial and Boundary Condition

- $T(t = 0, r) = 300K$
- $-k\frac{\partial T}{\partial r} = h_{in}(T - T_{\infty,in}), r = r_{in} = 0.025m$
- $-k\frac{\partial T}{\partial r} = h_{out}(T - T_{\infty,out}), r = r_{out} = 0.1m$
- On the interface between ceramic and insulation layer $-k_1\frac{\partial T}{\partial r} = -k_2\frac{\partial T}{\partial r}, r = r_m = 0.05m$

2.3 Finite Difference Method

Using the forward difference for the first derivatives:

$$\frac{dT}{dx}|_i = \frac{T_{i+1} - T_i}{\Delta x} + O(\Delta x)$$

Using the central difference for the second derivatives:

$$\frac{d^2T}{dx^2}|_i = \frac{T_{i+1} + T_{i-1} - 2T_i}{\Delta x^2} + O(\Delta x^2)$$

2.4 Combine 2.1, 2.2, 2.3

Energy Equation:

$$\frac{1}{\alpha}\frac{T_{i,s+1} - T_{i,s}}{\Delta t} = \frac{1}{r}\frac{T_{i+1,s} - T_{i,s}}{\Delta r} + \frac{T_{i+1,s} + T_{i-1,s} - 2T_{i,s}}{\Delta r^2},\ i, s = 0, 1, 2, \ldots, N$$

Note: i is the index for differential r, and s is the index for differential t.
The only difference of insulation layer and the ceramic layer is the α.

$$T_{i,s+1} = T_{i,s} + \frac{\Delta r}{r}(T_{i+1,s} - T_{i,s})\frac{\alpha\Delta t}{\Delta r^2} + (T_{i+1,s} + T_{i-1,s} - 2T_{i,s})\frac{\alpha\Delta t}{\Delta r^2}$$

Assume $M = \frac{\Delta r^2}{\alpha\Delta t}$

$$T_{i,s+1} = T_{i,s} + \frac{1}{M}\frac{\Delta r}{r}(T_{i+1,s} - T_{i,s}) + \frac{1}{M}(T_{i+1,s} + T_{i-1,s} - 2T_{i,s})$$

For ceramic (Θ), and insulation layer(Γ) respectively,

$$T_{i,s+1,\Theta} = T_{i,s,\Theta} + \frac{1}{M_\Theta}\frac{\Delta r}{r}(T_{i+1,s,\Theta} - T_{i,s,\Theta}) + \frac{1}{M_\Theta}(T_{i+1,s,\Theta} + T_{i-1,s,\Theta} - 2T_{i,s,\Theta})$$

$$T_{i,s+1,\Theta} = (1 - \frac{1}{M_\Theta}\frac{\Delta r}{r} - 2\frac{1}{M_\Theta})T_{i,s,\Theta} + (\frac{1}{M_\Theta}\frac{\Delta r}{r} + \frac{1}{M_\Theta})T_{i+1,s,\Theta} + \frac{1}{M_\Theta}T_{i-1,s,\Theta}$$

$$T_{i,s+1,\Gamma} = T_{i,s,\Gamma} + \frac{1}{M_\Gamma}\frac{\Delta r}{r}(T_{i+1,s,\Gamma} - T_{i,s,\Gamma}) + \frac{1}{M_\Gamma}(T_{i+1,s,\Gamma} + T_{i-1,s,\Gamma} - 2T_{i,s,\Gamma})$$

$$T_{i,s+1,\Gamma} = (1 - \frac{1}{M_\Gamma}\frac{\Delta r}{r} - 2\frac{1}{M_\Gamma})T_{i,s,\Gamma} + (\frac{1}{M_\Gamma}\frac{\Delta r}{r} + \frac{1}{M_\Gamma})T_{i+1,s,\Gamma} + \frac{1}{M_\Gamma}T_{i-1,s,\Gamma}$$

Boundary conditions:

We can randomly choose the start point of i. Let's say the inner surface is where $T_{0,s}$ located.

On the 2 boundries where convective heat transport occurs,

$$-k_\Theta \frac{T_{1,s,\Theta} - T_{0,s,\Theta}}{\Delta r} = h_{in}(T_{\infty,in} - T_{0,s,\Theta}), r = r_{in} = 0.025m$$

$$-k_\Gamma \frac{T_{N,s,\Gamma} - T_{N-1,s,\Gamma}}{\Delta r} = h_{out}(T_{N,s,\Gamma} - T_{\infty,out}), r = r_{out} = 0.1m$$

As a result, the boundary temperature can be renewed as:

$$T_{0,s,\Theta} = \frac{h_{in} T_{\infty,in} + \frac{k_\Theta}{\Delta r} T_{1,s,\Theta}}{h_{in} + \frac{k_\Theta}{\Delta r}}$$

$$T_{N,s,\Gamma} = \frac{h_{out} T_{\infty,out} + \frac{k_\Gamma}{\Delta r} T_{N-1,s,\Gamma}}{h_{out} + \frac{k_\Gamma}{\Delta r}}$$

On the interface between ceramic and insulation layer,

$$-k_\Theta \frac{T_{N,s,\Theta} - T_{N-1,s,\Theta}}{\Delta r} = -k_\Gamma \frac{T_{1,s,\Gamma} - T_{0,s,\Gamma}}{\Delta r}, r = r_m = 0.05m$$

$$T_{N,s,\Theta} = T_{0,s,\Gamma} = \frac{k_\Theta T_{N-1,s,\Theta} + k_\Gamma T_{1,s,\Gamma}}{k_\Theta + k_\Gamma}$$

With the 4 equations of boundary conditions, we can figure out the temperature on the 3 boundries: $T_{0,s,\Theta}$, $T_{N,s,\Theta} = T_{0,s,\Gamma}$, $T_{N,s,\Gamma}$.

2.5 Hyperparameters and Osciallation problem

A stable solution requires $1 - \frac{1}{M}\frac{\Delta r}{r} - 2\frac{1}{M}$ to be larger than 0. as $\frac{\Delta r}{r}$ is small, we approximately have: $1 - 2\frac{1}{M} > 0$.

This gives:

$$M > 2$$

Actually, my algorithm has very serious osillation problems. Osillation usually happens at the boundary. It starts from a very small perturbation(10^{-12}) of temperature and goes exponentially, finally causing Runtime Overflow(10^{308}). This is crazy. This figure below shows how the temperature profile looks like when osillation happens:

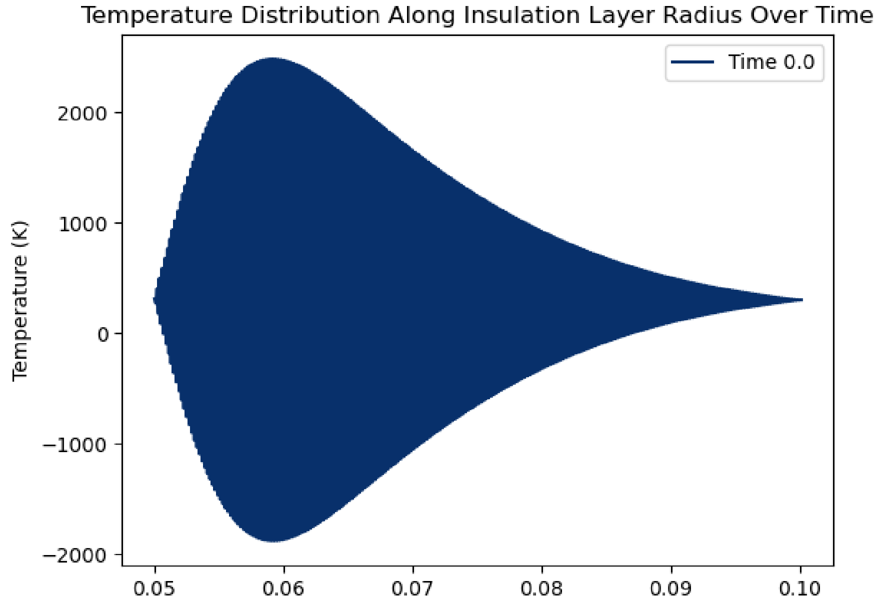

Fig. 2. Osillation happens at the boundary and propagate along the radius, and finally forms a surface instead a sinlge line. This is a profile before the Runtime Overflow

How does the osillation happened? This is interesting. One may think of factor M as a cause, but this every calculation strictly satisfies the condition $M > 2$. I guess the reason can only be the unbalanced heat flow on the boundary.

Reasonable hyperparameters, in this case, the dr and dt are tested multiple times. Small dr and dt result in stable and accurate compuatation, while on the other hand increase the computation time. I choose multiple combinations of dr and dt, and decided to use the simplest version of $dr = 0.01m$ and $dt = 0.001s$ and run $500s$ in total.

3 Results and discussion

3.1 The temperature distribution within the insulation layer

The temperature distribution are demonstrated in the Figure 3(c),(d).

3.2 Inner surface temperature of the insulation layer

The inner surface of the insulation layer (interface) is demonstrated in the Figure 4 red line. The temperature keeps unchanged at the begining and starts to grow rapidly 50s later.

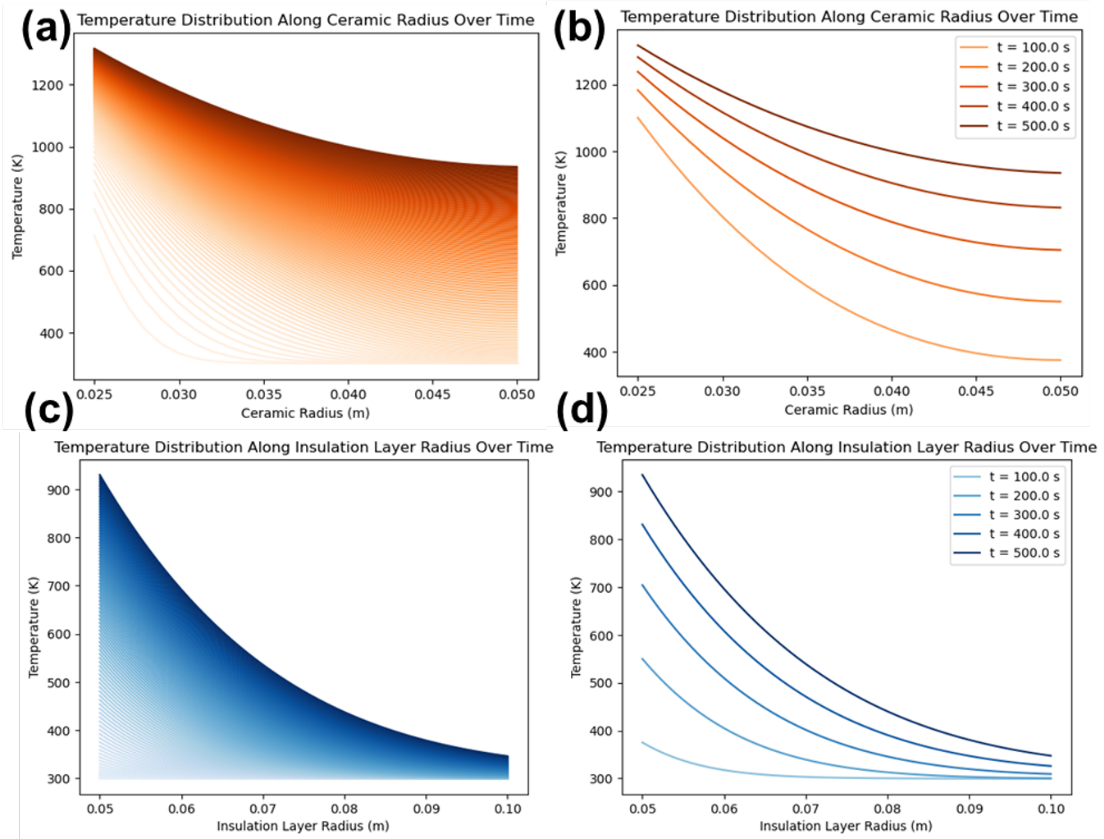

Fig. 3. Temperature distribution with the time propagation. (a) and (b) are the temperature distributions of ceramic reactor along the radius. (c) and (d) are the temperature distributions of insulation layer along the radius. In (a) and (c), deeper color means longer time. Characteristic temperature distribution are shown in (b) and (d).

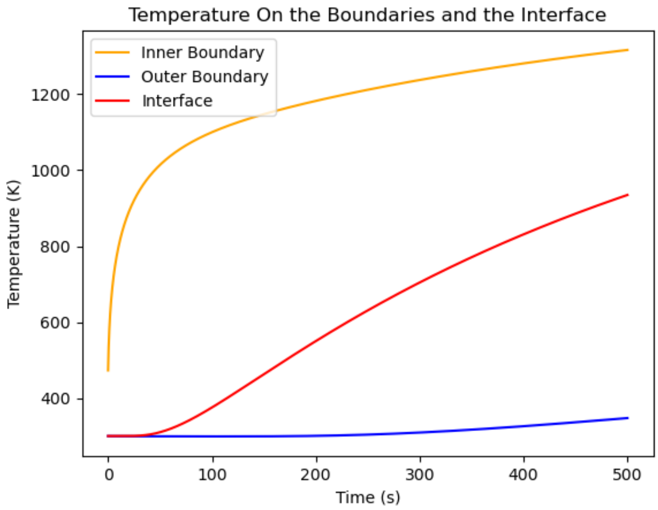

Fig. 4. Temperature propagation along time on the inner boundary(orange) of ceramic reactor, interface(red), and outer boundary(blue) of insulation layer

3.3 The impact of the insulation layer

From the lectures, the worst thickness for a insulation layer can be approximately calculated by $r_c = k/h = 0.2/10 = 0.02m$ in this case. However, the $r_{ceramic} = 0.025m > r_c$. As a result, the insulation layer can always hold the heat. The results are well proofed by the computation results. The heat flow rate with different thickness of insulation layer shows no obvious local maximum. This means the thicker the better if we measure the heat emission on the outer surface of insulation layer Figure 5. However, a thicker insulation layer means more energy are required to heat the insulation layer as demonstrated by the head absorption on the inner surface of ceramic reactor(Figure5(a)and(b)). As the Figure 5 (a) shows, there is no signifcant, but still a little variations(Figrue 5(b)), heat absorption on the inner boundary.

4 Appendix1: Python Scripts

The whole program is available in the file *column_HT_1d.ipynb*. Here is only a simplified version.

Listing 1: the pipe class

```
import numpy as np
import pandas as pd
```

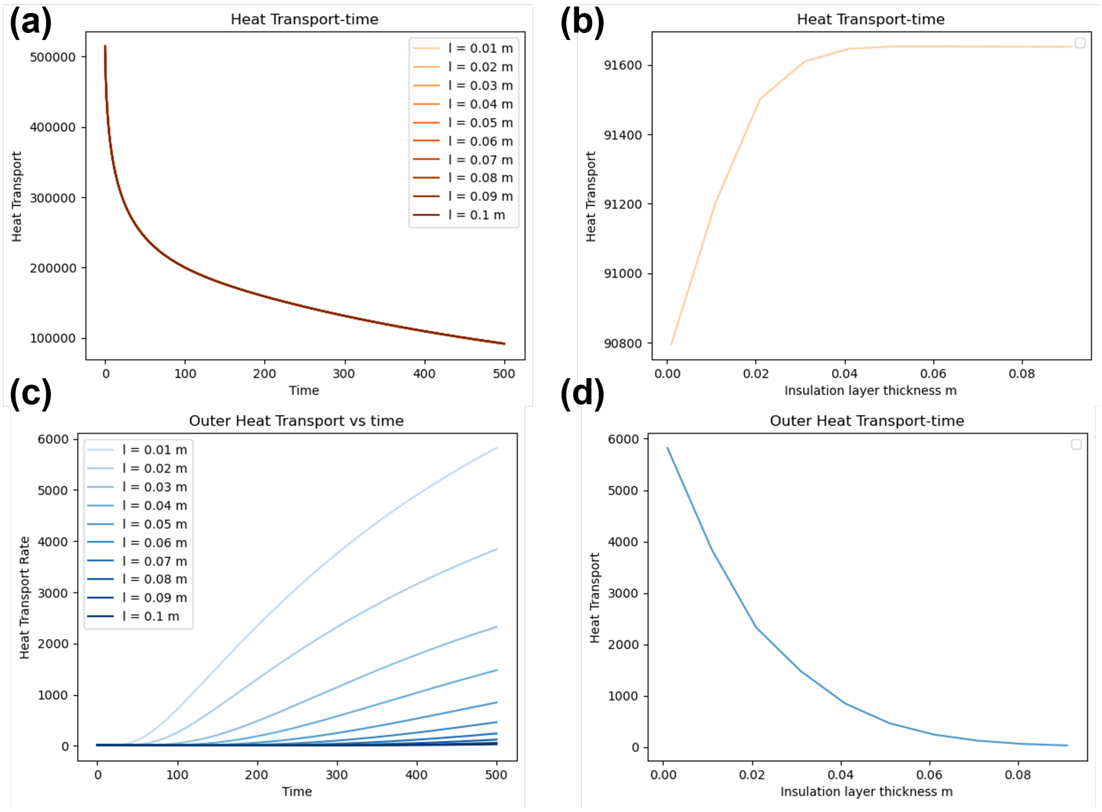

Fig. 5. (a),(c) illustrate the heat transformation rate of the inner ceramic reactor and the outer insulation layer boundary with the propagation of time respectively. (b) and (d) show after 500s, how the heat flow rate of inner and outer boundary varies with insulation layer thickness.

```
def step_1_D_cylindrical_bulk(dr, dt, k, rho, cp, radius, temp_arr
    ):

    # Do not deal with the elements in the first and the last
    temp_arr_cp = temp_arr.copy()
    alpha = k / (rho * cp)
    m_factor = dr**2/(alpha*dt)
    # You have to be really careful that the np.roll(temp_arr_cp,
        -1) is T_i+1
    first_derivative_term = 1/m_factor * dr/radius[1:-1] *(np.roll
        (temp_arr_cp, -1)[1:-1]-np.roll(temp_arr_cp, 0)[1:-1])

    second_derivative_term = 1/m_factor * (np.roll(temp_arr_cp, 1)
        [1:-1]+np.roll(temp_arr_cp, -1)[1:-1]-2*np.roll(temp_arr_cp
        , 0)[1:-1])
    temp_arr_cp[1:-1] += first_derivative_term +
        second_derivative_term

    return temp_arr_cp

# Refresh the temperature in the interface
def step_1_D_cylindrical_boundary(dr, h, k, T_inf, T_next):
    factor = h*dr/k
    nominator = factor*T_inf + T_next
    # print(nominator)
    denominator = factor + 1
    T_result = nominator/denominator

    return T_result

def step_1_D_cylindrical_interface(k1,k2,T1,T2):
    nominator = k1*T1 + k2*T2
    denominator = k1 + k2
    T_result = nominator/denominator
    print(f'Interface␣temperature␣=␣{T_result}')

class cylindrical_pipe(object):
    def __init__(self, dt, t_final, dr, T_initial, insu_thickness)
```

```
:

    self.dr = dr
    self.dt = dt
    self.insu_thickness = insu_thickness
    self.t_final = t_final
    self.T_initial = T_initial  # K

    self.k_cera = 3.0
    self.k_insu = 0.2
    self.rho_cera, self.c_cera = 2600, 1150
    self.rho_insu, self.c_insu = 600, 200


    self.alpha_cera = self.k_cera / (self.rho_cera * self.
        c_cera)
    self.alpha_insu = self.k_insu / (self.rho_insu * self.
        c_insu)
    # print(f'alpha_cera = {alpha_cera}')
    # print(f'alpha_insu = {alpha_insu}')

    self.r_i = 0.025
    self.r_m = self.r_i + 0.025
    self.r_o= self.r_m + self.insu_thickness

    self.r_cera = np.arange(self.r_i, self.r_m + dr, dr)
    self.r_insu = np.arange(self.r_m, self.r_o + dr, dr)


    self.T_cera_0 = np.ones_like(self.r_cera) * self.T_initial
    self.T_insu_0 = np.ones_like(self.r_insu) * self.T_initial


    self.T_inner = 1500
    self.T_outer = 298

    self.h_outer = 10
    self.h_inner = 500
```

```python
        self.heat_inner = []
        self.heat_outer = []
        self.heat_interface = []

    def run(self):


        T_cera = self.T_cera_0.copy()
        T_insu = self.T_insu_0.copy()

        self.heat_inner = []
        self.heat_outer = []
        self.heat_interface = []

        for t in range(int(self.t_final/self.dt)):
            # Ceramic zone
            T_cera[0] = step_1_D_cylindrical_boundary(self.dr,
                self.h_inner, self.k_cera,self.T_inner, T_cera[1])
            print(f'T_cera[0] = {T_cera[0]}')
            T_cera = step_1_D_cylindrical_bulk(self.dr, self.dt,
                self.k_cera, self.rho_cera, self.c_cera, self.
                r_cera, T_cera)
            # print(T_cera[0])
            #insulation layer zone
            T_insu[-1] = step_1_D_cylindrical_boundary(self.dr,
                self.h_outer, self.k_insu, self.T_outer, T_insu
                [-2])
            T_insu = step_1_D_cylindrical_bulk(self.dr, self.dt,
                self.k_insu, self.rho_insu,self.c_insu, self.r_insu
                , T_insu)
            print(f'T_insu[-1] = {T_insu[-1]}')
            # interface
            print(f"Temperature around interface \n inner = {
                T_cera[-2]} \n outer = {T_insu[1]}")
            T_cera[-1] = T_insu[0] =
                step_1_D_cylindrical_interface(self.k_cera,self.
                k_insu,T_cera[-2],T_insu[1])
```

```
            # print(f'T_cera[-1] = {T_cera[-1]}')
            if np.isnan(T_cera).any() or np.isnan(T_insu).any() :
                raise ValueError('There in nan in your calculation
                    , automatically stoppd')


            # Calculate heat transfer rate
            self.heat_inner.append(self.h_inner*(self.T_inner-
                T_cera[0]))
            self.heat_outer.append(self.h_outer*(T_insu[-1] - self
                .T_outer))
            self.heat_interface.append(-1*(T_insu[1] - T_insu[0])*
                self.k_insu/self.dr)

            # Refreash log
            T_cera_pd = pd.DataFrame([T_cera])
            T_insu_pd = pd.DataFrame([T_insu])
            T_cera_pd.to_csv(f'T_cera_{self.dt}_{self.dr}_{self.
                insu_thickness}.csv', mode='a', header=False)
            T_insu_pd.to_csv(f'T_insu_{self.dt}_{self.dr}_{self.
                insu_thickness}.csv', mode='a', header=False)
```

Listing 2: the code for experiment

```
t_final = 500
T_initial = 300
dt = 0.01
dr = 0.001
insu_thickness_ls = np.arange(dr, 0.1 + dr, dr*10)
pipe_ls = []
heat_inner_ls = []

for insu_thickness in insu_thickness_ls:
    pipe = cylindrical_pipe(dt, t_final, dr, T_initial,
        insu_thickness)
    pipe.run()
    heat_inner_ls.append(pipe.heat_inner)
    pipe_ls.append(pipe)
```

Listing 3: the code for plotting

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import pandas as pd
t_ls = np.arange(0, t_final, dt)

fig, ax = plt.subplots()

# print(len(T_cera_log_selected))
cmap = cm.get_cmap('Oranges', len(heat_inner_ls)+3)



for i in range(len(heat_inner_ls)):
    # print(i)
    ax.plot(t_ls, heat_inner_ls[i], label=f'l␣=␣{i*0.01}␣m', color
        =cmap(i+3))

ax.legend()

ax.set_title('Heat␣Transport-time')
ax.set_xlabel('Time')
ax.set_ylabel('Heat␣Transport')

plt.show()
```