McGill University

Multidisciplinary Design Optimization

MECH 579

Final Project: Shell and tube heat exchanger design optimization

---

Name: Swadesh Suman
McGill ID: 261097252
Email: swadesh.suman@mail.mcgill.ca

April 26, 2024

1 Introduction

Shell and tube heat exchangers are crucial in cooling hydraulic fluids, demanding a balance between heat transfer efficiency and cost-effectiveness. There are several applications of this type of heat exchanger in the industry. Optimization involves various components like shell, tube, and baffle to maximize conductivity while maintaining affordability. This project aims to enhance the heat conductivity of a shell and tube heat exchanger using Multi-disciplinary design Optimization.

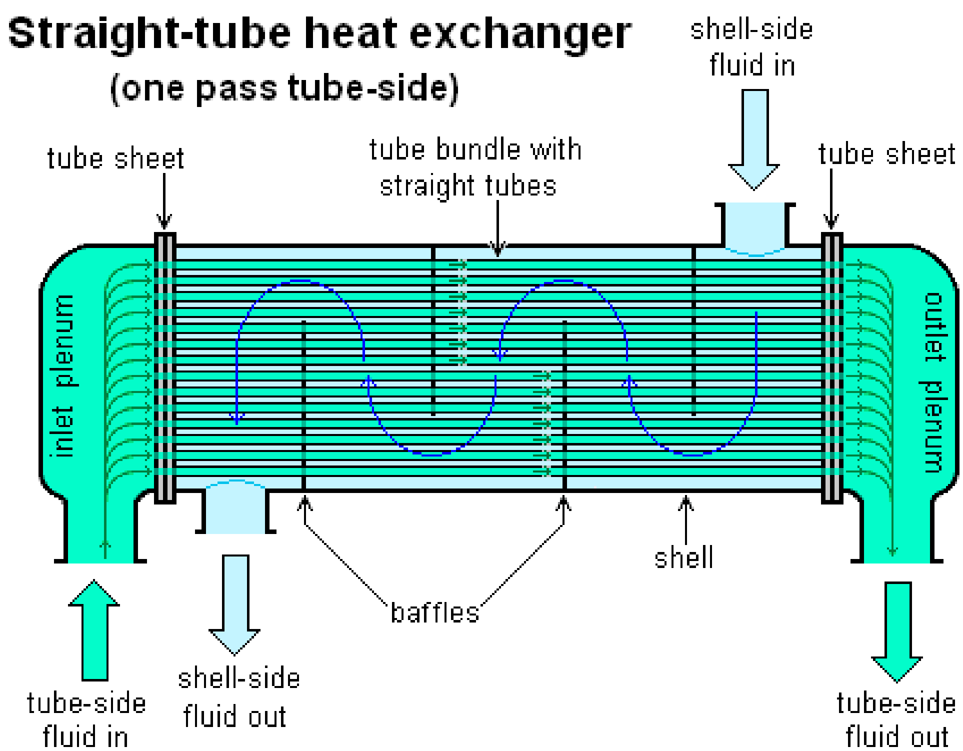

Figure 1: Shell and tube heat exchanger

Fig. 1 shows the diagram of a shell and tube heat exchanger. It is a one-pass tube side. The fluid to be cooled is water flowing in the tube, whereas the shell contains the hydraulic fluid used to cool the water. In this project, I have considered air as the cooling fluid. The entry temperature of the air is lower than the temperature of water.

2 Discipline

The following disciplines are involved in a Shell and Tube Heat Exchanger:

1. Structure Design
2. Thermal Engineering

3 Objective function

3.1 Structure Design Discipline

For efficient water cooling, a larger shell with more cooling fluid seems ideal, but beyond a point, increasing fluid volume doesn't enhance the cooling rate. Thus, determining the minimum shell size essential for required cooling efficiency is crucial for economic viability. This defines the objective function of the shell subsystem, f:

$$\text{minimize } F_s = 2.66175 \times L_t \times (D_{ot})^2 \times N_t$$

3.2 Thermal Engineering Discipline

Tubes play a vital role in preventing fluid contamination and facilitating efficient heat transfer. Increasing tube count directly enhances overall heat transfer. Objective: Minimize f to maximize the overall heat transfer coefficient U.

$$\text{minimize } F_t = \frac{1}{\mathrm{h}_i A_i (T_i - T_a)} + \left(\frac{1}{2\pi k L_t (T_o - T_i)}\right) \ln\left(\frac{D_{ot}}{D_{it}}\right) + \frac{1}{h_o A_o (T_{eW} - T_o)}$$

3.3 Combined objective function

$$\text{minimize } F = w_1(F_s) - (1 - w_1)F_t$$

where, w_1 is the weight associated with the objective function. In this case, it will be varied from 0.1 to 0.9 in the step of 0.2.

4 Optimization problem formulation

$$
\begin{aligned}
\text{minimize} \quad & F = w_1(F_s(\mathbf{X_s}, \mathbf{Z}) - (1-w_1)F_t(\mathbf{X_t}, \mathbf{Z}, T_o(\mathbf{X_t}, \mathbf{Z}, A)) \\
\text{with respect to} \quad & \mathbf{X_s} \in R^3, \mathbf{X_t} \in R^3, \mathbf{Z} \in R^2 \\
\text{subject to Governing equation} \quad & \hat{R}_k(\mathbf{X_t}, A, \mathbf{Z}, T_o(A, \mathbf{X_t}, \mathbf{Z})) = 0 \\
\text{subject to constraint} \quad & \hat{C}_1(\mathbf{X_t}, \hat{R}_k, \mathbf{Z}, \mathrm{HTR}(\mathbf{X_t}, \hat{R}_k, \mathbf{Z})) > 0 \\
\text{subject to constraint} \quad & \hat{C}_2(\mathbf{X_t}, \mathbf{Z}) > 0 \\
\text{subject to constraint} \quad & \hat{C}_3(\mathbf{X_t}, \mathbf{Z}) > 0 \\
\text{subject to constraint} \quad & \hat{C}_4(\mathbf{X_t}) > 0
\end{aligned}
$$

T_o will be solved using the heat conduction governing equation given in the constraint section. HTR will be solved using the formulae given in the Design variable section.

5 Design variables

5.1 Global Variables (Z)

L_t = length of a tube
D_{ot} = Outer tube diameter

5.2 Local Variables

Structure Design Discipline ($\mathbf{X_s}$)
D_s = Diameter of the shell - related to D_{ot} and L_t (discussed in section 5.4)

Thermal Engineering Discipline ($\mathbf{X_t}$)
D_{it} = Inner tube diameter
T_i = Temperature of the inner surface of the tube
T_o = Temperature of the outer surface of the tube
V_w = Velocity of the water A = Thermal constant (present in the governing equation)

5.3 Constant Variables and other parameters

Both disciplines
P_t - Pitch of the tube = 1.5*D_{ot}
N_t = Number of tubes - 5
L_s - Length of the shell = 1.1*L_t
K - Thermal conductivity of the tube

V_a - Velocity of the cooling fluid
T_a - Temperature of the cooling fluid
T_{eW} - Entry temperature of the water
T_{iW} - Exit temperature of the water
h_i - Heat transfer coefficient - inside pipe
h_o - Heat transfer coefficient - outside pipe
Re_i - Reynolds number - inside pipe
Re_o - Reynolds number - outside pipe
$\dot{M}$ - Mass flow rate of water
Q - Energy supplied from the water

5.4 Formulae relating variables

$$D_s = 0.660538 \left(\frac{A_{\mathrm{tube}} \left(\frac{P_t}{D_t}\right)^2 \cdot D_t}{L_t} \right)^{1/2}$$

$$h_i = \frac{(\frac{f}{2})\,(\mathrm{Re}_i - 1000)\,\mathrm{Pr}_i}{1 + 12.7\left((\mathrm{Pr}^{2/3} - 1)(f/2)^{0.5}\right)} \frac{k}{D_{it}}$$

$$h_o = 0.0237 \cdot \mathrm{Re}_o^{0.618} \cdot \mathrm{Pr}_o^{1/3} \cdot \frac{k}{D_{ot}}$$

$$\mathrm{HTR} = \frac{k(T_{ot} - T_{it})2\pi D_{ot} L_t}{D_{ot} - D_{it}}$$

6 Constraints

Constraint	**Definition**
$\hat{R}_k : \frac{d}{dr}(r\frac{dT}{dr}) = 0$	Temperature across the thickness of tube must satisfy the differential heat conduction equation
$\hat{C}_1 : L_t/V_w - \left(\frac{Q(D_{ot}-D_{it})}{K(T_o-T_i)2\pi D_{ot}L_t}\right) > 0$	Energy supplied from water must satisfy required cooling time
$\hat{C}_2 : D_o - D_i - 0.0128 > 0$	Difference between inner and outer tube should be greater than 0.0128
$\hat{C}_3 : D_i - D_o + 0.0215 > 0$	Difference between inner and outer tube should be less than 0.0215
$\hat{C}_4 : T_o - T_i - 1 > 0$	The outer tube temperature should always be greater than the inner temperature by 1 K

Problem d)Write a code to solve the stated optimization problem using a direct or adjoint method to compute the derivatives. Show your derivation of the equations. Use a fixed step length (No line search is required) and select an appropriate point as your initial design point. Explain the choice of the initial design point.

Governing equation : $\frac{d}{dr}(r\frac{dT}{dr}) = 0$
The following analytical equations can be derived from the above governing equation:

Analytical equation: $T_o = A * \ln \frac{D_{ot}}{D_{it}} + T_i$

Method to find derivative
For finding the gradient and hessian of the objective function, automatic differentiation is used and this is calculated using the **Autograd** library. This gradient and hessian are then supplied to the scipy library for optimization.

Initial Point Selection
The design variable vector: $\mathbf{X}$ = [Inner tube dia, Outer tube dia, Inner tube temp, A (constant), Length of tube, Velocity of Water, Outer tube temp].
The initial point: $\mathbf{X}$ = [0.1, 0.21, 245, 30, 11, 0.11, 322].

Parameters	**Bounds**
D_{it}	0.025 to 0.416 ft
D_{ot}	0.0416 to 0.429 ft
T_i	243 to 322 K
A	0 to 41
L_t	5 to 12 ft
V_w	0.1 to 3.0 ft/s
T_o	273 to 322 K

Table 1: Bounds on the parameters for practical applications

For the length, the unit is taken in ft, temperature in K, and Velocity in ft/s. The inner and outer tube diameter is taken around 0.1 ft and 0.21 ft, respectively. The initial points are in the bounds for practical applications. The inner and outer temp is taken around 245 and 322 k, respectively. The initial velocity of water is taken in 0.11 ft/s, this has been selected as keeping in mind mass flow rate.

Weight (W_1) for multiobjective function
For the contour plots and function optimization $W_1 = 0.5$. However, for plotting the Pareto front W_1 is varying from 0.1 to 0.9 in the step of 0.2.

Constant variable values

Parameters	**Values**
Prandtl number (cooling fluid) Pr_l	4.34
Prandtl number (water) Pr_w	6.90
Velocity (cooling fluid)	5 ft/s
Temperature (cooling fluid)	242 K
Thermal conductivity (steel)	0.00257 $Btu/ft/s/F$
Entry temperature (Water)	323 K
Exit temperature (Water)	273 K
Mass flow rate (Water)	2.94 Kg/s

Table 2: Constant value of the parameters

Problem e) Show a contour plot of the function, f(x) and overplot the optimization path.

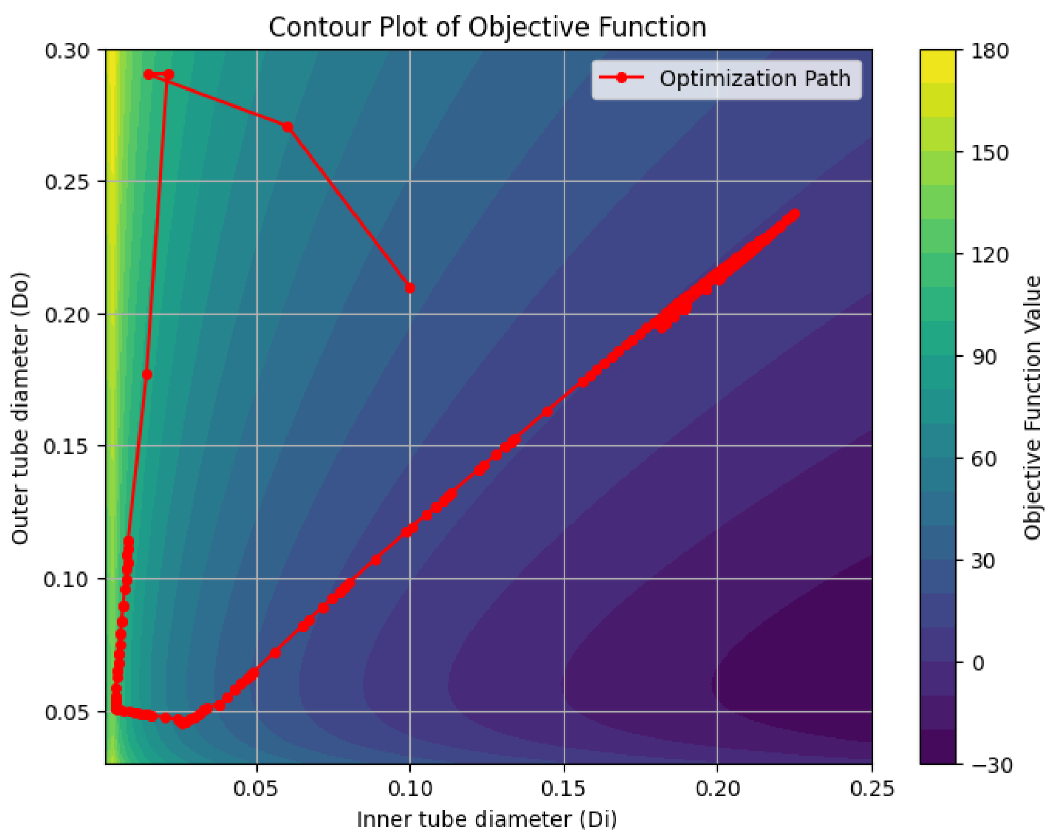

Figure 2: Contour plot and optimization path of algorithm

Fig. 2 shows the contour plot of the objective function based on two parameters inner and outer tube diameter. It also shows the optimization path for the algorithm. For plotting the objective function, these two parameters have been used. For this optimization weight is kept equal for both the objective functions from structure and thermal discipline. So, in this case, $W_1 = 0.5$.

The "**trust-constr**" method is used in the **Scipy** library for optimization. This method used the **finite difference method** to calculate the gradient of the lagrangian function and the constraint. The tolerance value of the convergence is set at the default value of 1E-08.

The final optimized point: $\mathbf{X}$ = [0.188, 0.201, 272, 15.234, 11.999, 0.249, 273].

Problem f) Provide convergence plots of the log of the norm of the gradient and the function value versus the design iterations.

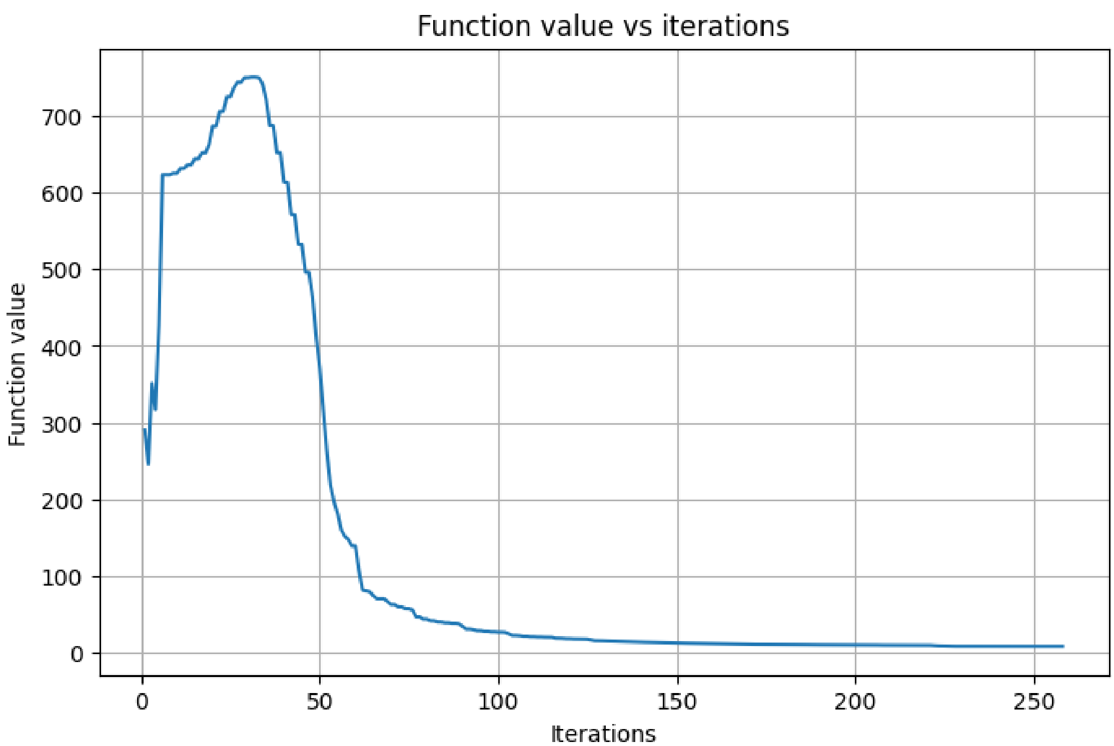

Figure 3: Convergence plot of the objective function vs iterations

Fig. 3 shows the minimization of the function value vs the number of iterations. Fig. 4 shows the convergence of the norm of the gradient vs the number of iterations. For the optimization problem, the gradient norm of the lagrangian function is reduced till 1E-08.

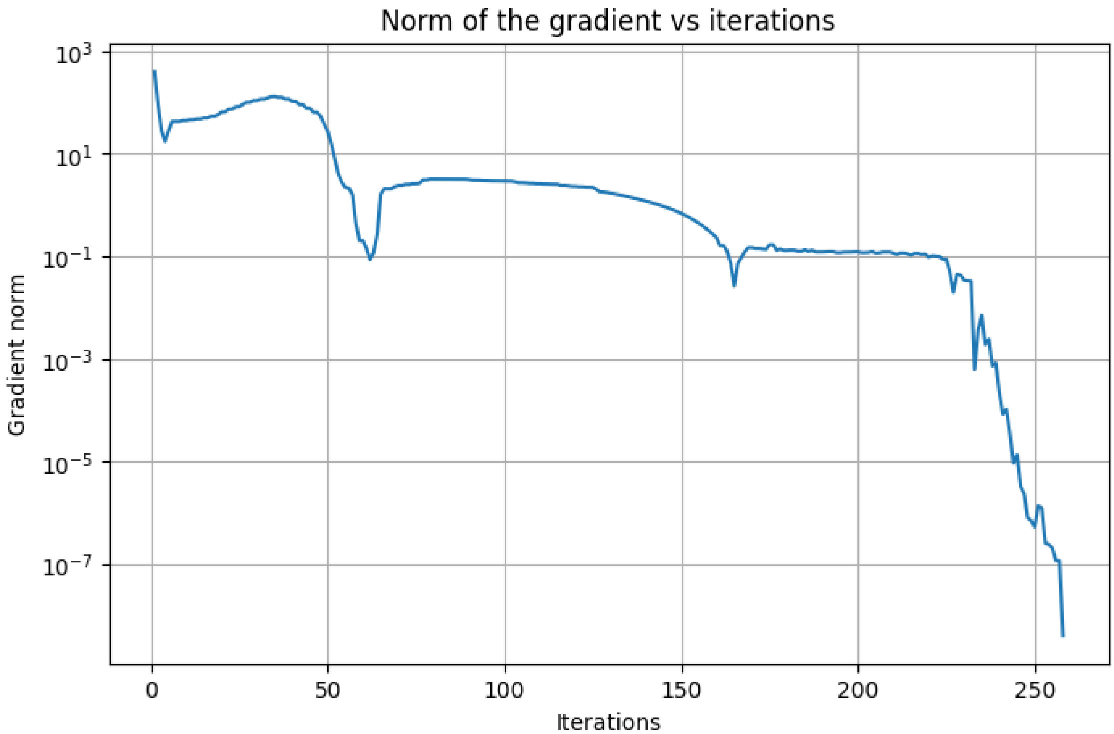

Figure 4: Convergence plot of the gradient norm vs iterations

Problem g) Demonstrate that at every design iteration, the governing equations are fully satisfied.

Governing equation : $\frac{d}{dr}(r\frac{dT}{dr}) = 0$
The following analytical equations can be derived from the above governing equation:

Analytical equation: $T_o = A * \ln \frac{D_{ot}}{D_{it}} + T_i$

The 1D heat conduction equation is used as the equality constraint in the above optimization problem. Fig. 5 shows the optimized values satisfying the governing equations at each iteration. For plotting the graph, the optimized point value at each iteration was put in the governing equation and the obtained governing equation value was plotted at each iteration.

The initial fluctuations in the values are because of the initial random point selection which does not satisfy the governing equation. But even if the initial points do not satisfy the governing equations, the new optimized points obtained in the next iterations satisfy the governing equations.

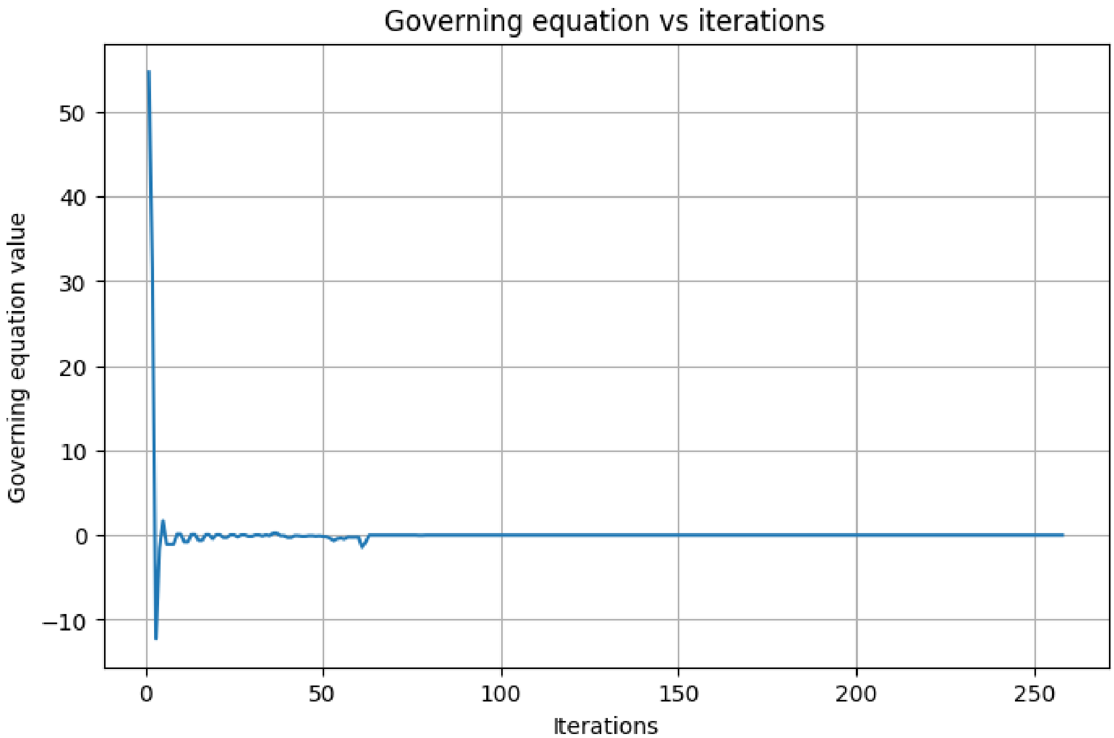

Figure 5: Plot showing the optimized values satisfying the governing equations at each iteration

Pareto front

Fig. 6 shows the Pareto front for the multiobjective function. As mentioned in the beginning, the objective function is the combination of two functions namely: the structure function and the thermal design function. To plot this Pareto front, weight is varied from 0.1 to 0.9 in the step of 0.2, and the function value is obtained at all the optimized points for multiobjective functions at each weight.

Weight	D_i	D_o	T_i	A	L_t	V_w	T_o
0.1	0.325	0.338	272	25.39	11.99	0.419	273
0.3	0.233	0.246	272	18.72	11.99	0.304	273
0.5	0.188	0.201	272	15.23	11.99	0.249	273
0.7	0.151	0.163	271.9	13.06	11.99	0.215	273
0.9	0.0968	0.1096	271.1	14.9	11.99	0.251	273

Table 3: Optimized parameter value at each weight value

Table 3 shows the optimized parameters value at each of the weights.

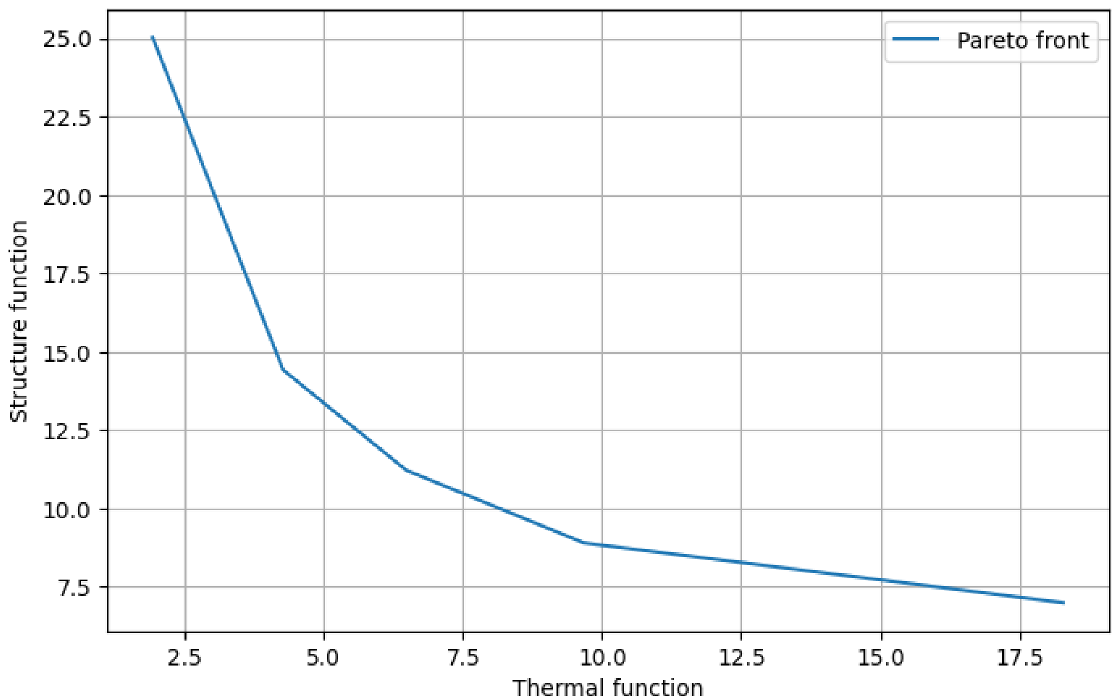

Figure 6: Plot showing the Pareto front for the multiobjective function

+*In[28]:*+
[source, ipython3]

```
import autograd.numpy as np
from autograd import grad, hessian
from scipy.optimize import minimize, approx_fprime
from scipy.optimize import NonlinearConstraint
from scipy.optimize import Bounds
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
```

+*In[29]:*+
[source, ipython3]

```
def obj_func(X):

    x1, x2, x3, x4, x5, x6, x7 = X
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7

    # constants
    Pr_l = 4.34
    Pr_w = 6.90
    Vl = 5  # ft/s
    T_wi = 323
    #T_we = 273
    #T_l = 273
    Ta = 242
    k = 0.00257
    v = 1.05*0.00001 # visocosity of fluid - same for both
    r = 1.4*0.00001  # roughness of the pipe

    # variable
    Ai = (np.pi*Di**2)/4
    Ao = (np.pi*Do**2)/4
    Re_i = (Vl * Di)/v
    Re_o = (Vw * Do)/v

    w1 = 0.5
    #To = A*np.log(Do/Di) + Ti
    S = np.log10(Re_i/1.816*np.log10(1.1*Re_i/np.log10(1+1.1*Re_i)))
    f = -2*np.log10(2.18*S/Re_i + r/(3.71*Di))
    Nu_i = ((f/2)*(Re_i-1000)*Pr_l)/(1+12.7*(Pr_l**(2/3)-1)*(f/2)**0.5)
```

```
    Nu_o = 0.0237*Re_o**0.618*Pr_w**(1/3)
    h_i = Nu_i*k/Di
    h_o = Nu_o*k/Do

    part_1 = 1/(h_i*Ai*(Ti - Ta))
    part_2 = (1/(2*np.pi*k*Lt))*A*np.log(Do/Di)
    part_3 = 1/(h_o*Ao*(T_wi - To))

    therm_val = part_1 + part_2 + part_3

    shell_val = 2.66175 * Lt * Do**2 * 5

    obj_val = w1*shell_val + (1-w1)*therm_val

    return obj_val

def grad_obj_func(X):

    grad_fun = grad(obj_func)
    grad_val_fun = grad_fun(X)

    return grad_val_fun

def hess_obj_func(X):
    H_fun = hessian(obj_func)
    H_matrix_fun = H_fun(X)
    return H_matrix_fun

# Functions for constraint 1
def constraint_1(X):

    x1, x2, x3, x4, x5, x6, x7 = X  # Fix variable name
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7

    T_wi = 323
    T_we = 273
    m = 2.94
    cp = 0.998
    k = 0.00257

    Q = m*cp*(T_wi - T_we)
    #To = A*np.log(Do/Di) + Ti
    const_val = (Lt/Vw)  - (Q*(Do - Di))/(k*(To - Ti)*2*np.pi*Do*Lt)  # Fix formula
```

```
    return const_val


# Functions for constraint 1
def constraint_2(X):

    x1, x2, x3, x4, x5, x6, x7 = X  # Fix variable name
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7

    const_val_2 = To - ((A*np.log(Do/Di)) + Ti)  # Fix formula

    return const_val_2

# Functions for constraint 1
def constraint_3(X):

    x1, x2, x3, x4, x5, x6, x7 = X  # Fix variable name
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7

    const_val_3 = Do - Di - 0.0128 # Fix formula

    return const_val_3

# Functions for constraint 1
def constraint_4(X):

    x1, x2, x3, x4, x5, x6, x7 = X  # Fix variable name
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7

    const_val_4 = Di - Do + 0.0215  # Fix formula

    return const_val_4
```

```
# Functions for constraint 1
def constraint_5(X):

    x1, x2, x3, x4, x5, x6, x7 = X  # Fix variable name
    Di = x1
    Do = x2
    Ti = x3
    A = x4
    Lt = x5
    Vw = x6
    To = x7
    #To = A*np.log(Do/Di) + Ti
    const_val_5 = To - Ti - 1  # Fix formula

    return const_val_5
```

+*In[30]:*+
[source, ipython3]

```
# Define the values of the independent variables
X0 = np.array([0.1, 0.21, 245, 30, 11, 0.11, 322])

# Define bounds for variables
bounds = ((0.025, 0.416),  # Di
          (0.0416, 0.429),  # Do
          (243, 322),  # Ti
          (0, 41),  # A
          (5, 12),  # Lt
          (0.1, 3.0),  # Vw
          (273, 322))  # To

# Define multiple constraints
constraints = [{'type': 'ineq', 'fun': constraint_1},
               {'type': 'eq', 'fun': constraint_2},
               {'type': 'ineq', 'fun': constraint_3},
               {'type': 'ineq', 'fun': constraint_4},
              {'type': 'ineq', 'fun': constraint_5}]

# Initialize lists to store convergence data
itr_list = []
obj_list = []
grad_list = []
value_list = []
grad_lag = []
```

```
def callback_function(xk, state):
    itr_list.append(state.niter)
    obj_list.append(obj_func(xk))
    grad_lag.append(state.optimality)
    value_list.append(xk)
    grad_list.append(np.linalg.norm(grad_obj_func(xk)))
    callback_function.iteration += 1

callback_function.iteration = 0  # Initialize iteration count

# Minimize the objective function
#result = minimize(obj_func, X0, method='SLSQP', bounds=bounds,
constraints=constraints, callback=callback_function)
result = minimize(obj_func, X0, method='trust-constr', jac=grad_obj_func,
hess=hess_obj_func, \
               bounds=bounds, constraints=constraints, \
                  options={'gtol': 1e-8,'xtol':1e-12, 'maxiter': 300, 'verbose': 3},
\
                  callback=callback_function);

# Extract the first and second elements of value_list
Di_val = [value[0] for value in value_list]
Do_val = [value[1] for value in value_list]
A_val = [value[3] for value in value_list]
Ti_val = [value[2] for value in value_list]
To_val = [value[6] for value in value_list]

# Print the result
print("Optimal solution:", result.x)


gov_val = []
for i in range(len(Ti_val)):
    gov = To_val[i] - (A_val[i]*np.log(Do_val[i]/Di_val[i]) + Ti_val[i])
    gov_val.append(gov)

----


+*Out[30]:*+
----
| niter |f evals|CG iter|  obj func   |tr radius |   opt    |  c viol  | penalty
|barrier param|CG stop|
|-------|-------|-------|-------------|----------|----------|----------|----------|-
------------|-------|
|   1   |   1   |   0   | +2.9070e+02 | 1.00e+00 | 4.02e+02 | 5.47e+01 | 1.00e+00 |
1.00e-01   |   0   |
|   2   |   2   |   4   | +2.4636e+02 | 7.00e+00 | 9.19e+01 | 3.16e+01 | 1.00e+00 |
1.00e-01   |   2   |
|   3   |   3   |  10   | +3.5135e+02 | 7.71e+00 | 2.78e+01 | 1.23e+01 | 9.65e+00 |
```

								1.00e-01	1
4	4	16	+3.1716e+02	9.29e+00	1.73e+01	1.69e+00	9.65e+00	1.00e-01	1
5	5	22	+4.2760e+02	9.63e+00	2.79e+01	1.66e+00	1.11e+02	1.00e-01	1
6	6	27	+6.2275e+02	9.63e+00	4.24e+01	1.08e+00	1.11e+02	1.00e-01	4
7	7	32	+6.2275e+02	9.63e-01	4.24e+01	1.08e+00	1.11e+02	1.00e-01	4
8	8	37	+6.2275e+02	9.63e-02	4.24e+01	1.08e+00	1.11e+02	1.00e-01	2
9	9	39	+6.2466e+02	6.74e-01	4.42e+01	1.13e-01	1.11e+02	1.00e-01	2
10	10	44	+6.2466e+02	6.74e-02	4.42e+01	1.13e-01	1.11e+02	1.00e-01	2
11	11	45	+6.3106e+02	4.72e-01	4.58e+01	7.96e-01	1.11e+02	1.00e-01	2
12	12	49	+6.3106e+02	7.94e-02	4.58e+01	7.96e-01	1.11e+02	1.00e-01	2
13	13	52	+6.3564e+02	5.56e-01	4.73e+01	7.63e-02	1.11e+02	1.00e-01	2
14	14	56	+6.3564e+02	8.70e-02	4.73e+01	7.63e-02	1.11e+02	1.00e-01	2
15	15	58	+6.4350e+02	6.09e-01	4.96e+01	6.10e-01	1.11e+02	1.00e-01	2
16	16	61	+6.4350e+02	1.29e-01	4.96e+01	6.10e-01	1.11e+02	1.00e-01	2
17	17	64	+6.5127e+02	2.57e-01	5.35e+01	7.03e-02	1.11e+02	1.00e-01	2
18	18	67	+6.5127e+02	1.29e-01	5.35e+01	7.03e-02	1.11e+02	1.00e-01	2
19	19	70	+6.6140e+02	2.57e-01	5.74e+01	4.27e-01	1.11e+02	1.00e-01	2
20	20	73	+6.8594e+02	5.14e-01	6.50e+01	6.08e-02	1.52e+02	1.00e-01	2
21	21	76	+6.8594e+02	2.35e-01	6.50e+01	6.08e-02	1.52e+02	1.00e-01	2
22	22	79	+7.0517e+02	4.71e-01	7.36e+01	2.76e-01	1.52e+02	1.00e-01	2
23	23	82	+7.0517e+02	2.35e-01	7.36e+01	2.76e-01	1.52e+02	1.00e-01	2
24	24	84	+7.2426e+02	4.71e-01	8.20e+01	5.14e-02	1.52e+02	1.00e-01	2
25	25	85	+7.2426e+02	2.35e-01	8.20e+01	5.14e-02	1.52e+02	1.00e-01	2
26	26	86	+7.3623e+02	2.35e-01	9.12e+01	1.98e-01	1.52e+02	1.00e-01	2
27	27	87	+7.4327e+02	4.71e-01	1.00e+02	4.41e-02	1.52e+02	1.00e-01	2
28	28	88	+7.4327e+02	2.35e-01	1.00e+02	4.41e-02	1.52e+02		

```
1.00e-01    |    2    |
|  29    |  29    |  89    | +7.4915e+02 | 4.71e-01 | 1.08e+02 | 1.38e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  30    |  30    |  90    | +7.4915e+02 | 2.35e-01 | 1.08e+02 | 1.38e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  31    |  31    |  91    | +7.4991e+02 | 4.71e-01 | 1.15e+02 | 3.86e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  32    |  32    |  92    | +7.4991e+02 | 2.35e-01 | 1.15e+02 | 3.86e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  33    |  33    |  93    | +7.4892e+02 | 4.71e-01 | 1.21e+02 | 1.05e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  34    |  34    |  94    | +7.4089e+02 | 4.71e-01 | 1.29e+02 | 3.35e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  35    |  35    |  95    | +7.2059e+02 | 4.71e-01 | 1.31e+02 | 7.14e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  36    |  36    |  96    | +6.8669e+02 | 9.42e-01 | 1.25e+02 | 2.43e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  37    |  37    |  97    | +6.8669e+02 | 4.71e-01 | 1.25e+02 | 2.43e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  38    |  38    |  98    | +6.5147e+02 | 9.42e-01 | 1.16e+02 | 6.92e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  39    |  39    |  99    | +6.5147e+02 | 4.71e-01 | 1.16e+02 | 6.92e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  40    |  40    |  100   | +6.1279e+02 | 9.42e-01 | 1.04e+02 | 2.78e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  41    |  41    |  101   | +6.1279e+02 | 4.71e-01 | 1.04e+02 | 2.78e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  42    |  42    |  102   | +5.7076e+02 | 9.42e-01 | 9.04e+01 | 7.15e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  43    |  43    |  103   | +5.7076e+02 | 4.71e-01 | 9.04e+01 | 7.15e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  44    |  44    |  104   | +5.3225e+02 | 9.42e-01 | 7.67e+01 | 1.42e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  45    |  45    |  105   | +5.3225e+02 | 4.71e-01 | 7.67e+01 | 1.42e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  46    |  46    |  106   | +4.9568e+02 | 9.42e-01 | 6.40e+01 | 7.87e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  47    |  47    |  107   | +4.9568e+02 | 4.71e-01 | 6.40e+01 | 7.87e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  48    |  48    |  108   | +4.6351e+02 | 9.42e-01 | 5.32e+01 | 1.32e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  49    |  49    |  109   | +4.1275e+02 | 9.42e-01 | 3.75e+01 | 9.58e-02 | 1.52e+02 |
1.00e-01    |    2    |
|  50    |  50    |  111   | +3.7181e+02 | 1.88e+00 | 2.73e+01 | 1.57e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  51    |  51    |  113   | +3.1686e+02 | 3.77e+00 | 1.60e+01 | 2.01e-01 | 1.52e+02 |
1.00e-01    |    2    |
|  52    |  52    |  116   | +2.6256e+02 | 6.60e+00 | 7.99e+00 | 3.70e-01 | 1.52e+02 |
1.00e-01    |    4    |
|  53    |  53    |  119   | +2.1915e+02 | 3.64e+01 | 4.01e+00 | 6.46e-01 | 1.52e+02 |
```

								1.00e-01	4
54	54	121	+1.9708e+02	3.64e+01	2.78e+00	4.24e-01	1.52e+02	1.00e-01	4
55	55	123	+1.8196e+02	3.64e+01	2.20e+00	3.32e-01	1.52e+02	1.00e-01	4
56	56	125	+1.6059e+02	4.57e+01	2.12e+00	4.61e-01	1.52e+02	1.00e-01	4
57	57	126	+1.5219e+02	4.57e+01	1.59e+00	2.23e-01	1.52e+02	1.00e-01	4
58	58	127	+1.4791e+02	4.57e+01	4.27e-01	2.41e-01	1.52e+02	1.00e-01	4
59	59	128	+1.3958e+02	4.57e+01	2.02e-01	2.29e-01	1.97e+02	1.00e-01	4
60	60	129	+1.3958e+02	1.07e+01	2.02e-01	2.29e-01	1.97e+02	1.00e-01	4
61	61	130	+1.0673e+02	2.13e+01	1.40e-01	1.44e+00	1.97e+02	1.00e-01	2
62	62	132	+8.2061e+01	2.13e+01	8.57e-02	8.40e-01	1.97e+02	1.00e-01	4
63	63	135	+8.0918e+01	1.49e+02	1.17e-01	1.94e-03	1.97e+02	1.00e-01	2
64	64	138	+7.9889e+01	1.49e+02	2.66e-01	5.24e-04	1.97e+02	1.00e-01	4
65	65	141	+7.4571e+01	1.49e+02	1.65e+00	3.80e-03	1.97e+02	1.00e-01	4
66	67	143	+7.0704e+01	1.49e+02	2.06e+00	7.39e-04	1.97e+02	1.00e-01	4
67	69	145	+7.0704e+01	1.49e+01	2.06e+00	7.39e-04	1.97e+02	1.00e-01	4
68	70	148	+7.0704e+01	1.49e+00	2.06e+00	7.39e-04	1.97e+02	1.00e-01	2
69	72	150	+6.6970e+01	1.05e+01	2.28e+00	1.75e-03	1.97e+02	1.00e-01	2
70	74	152	+6.3144e+01	1.05e+01	2.39e+00	1.78e-03	1.97e+02	1.00e-01	4
71	76	154	+6.3144e+01	1.05e+00	2.39e+00	1.78e-03	1.97e+02	1.00e-01	4
72	78	156	+6.0046e+01	7.32e+00	2.52e+00	2.07e-03	1.97e+02	1.00e-01	2
73	80	158	+6.0046e+01	7.32e-01	2.52e+00	2.07e-03	1.97e+02	1.00e-01	4
74	82	160	+5.7801e+01	5.13e+00	2.60e+00	9.26e-04	1.97e+02	1.00e-01	2
75	84	162	+5.7801e+01	5.13e-01	2.60e+00	9.26e-04	1.97e+02	1.00e-01	2
76	86	164	+5.6146e+01	3.59e+00	2.66e+00	4.68e-04	1.97e+02	1.00e-01	2
77	88	166	+4.7201e+01	3.59e+00	3.06e+00	2.76e-02	1.97e+02	1.00e-01	2
78	89	168	+4.7201e+01	8.38e-01	3.06e+00	2.76e-02	1.97e+02		

								1.00e-01	2
79	91	170	+4.4263e+01	5.87e+00	3.18e+00	3.59e-03	1.97e+02	1.00e-01	2
80	92	172	+4.4263e+01	5.87e-01	3.18e+00	3.59e-03	1.97e+02	1.00e-01	2
81	94	173	+4.2098e+01	4.11e+00	3.20e+00	2.03e-03	1.97e+02	1.00e-01	2
82	95	175	+4.2098e+01	4.11e-01	3.20e+00	2.03e-03	1.97e+02	1.00e-01	2
83	97	176	+4.0563e+01	2.87e+00	3.19e+00	8.53e-04	1.97e+02	1.00e-01	2
84	99	178	+4.0563e+01	2.87e-01	3.19e+00	8.53e-04	1.97e+02	1.00e-01	2
85	101	179	+3.9484e+01	2.01e+00	3.18e+00	3.37e-04	1.97e+02	1.00e-01	2
86	103	180	+3.9484e+01	2.01e-01	3.18e+00	3.37e-04	1.97e+02	1.00e-01	2
87	105	181	+3.8727e+01	1.41e+00	3.17e+00	1.28e-04	1.97e+02	1.00e-01	2
88	107	182	+3.8727e+01	1.41e-01	3.17e+00	1.28e-04	1.97e+02	1.00e-01	2
89	109	183	+3.8197e+01	9.86e-01	3.16e+00	4.70e-05	1.97e+02	1.00e-01	2
90	111	184	+3.4696e+01	9.86e-01	3.10e+00	1.36e-02	1.97e+02	1.00e-01	2
91	113	185	+3.1278e+01	1.97e+00	3.03e+00	1.73e-02	1.97e+02	1.00e-01	2
92	115	187	+3.1278e+01	1.97e-01	3.03e+00	1.73e-02	1.97e+02	1.00e-01	2
93	116	188	+3.0558e+01	3.94e-01	3.01e+00	5.55e-03	1.97e+02	1.00e-01	2
94	118	189	+2.9220e+01	2.76e+00	2.98e+00	1.57e-03	1.97e+02	1.00e-01	2
95	120	191	+2.9220e+01	2.76e-01	2.98e+00	1.57e-03	1.97e+02	1.00e-01	2
96	122	192	+2.8309e+01	1.93e+00	2.96e+00	5.65e-04	1.97e+02	1.00e-01	2
97	124	194	+2.8309e+01	1.93e-01	2.96e+00	5.65e-04	1.97e+02	1.00e-01	2
98	126	195	+2.7681e+01	1.35e+00	2.94e+00	2.01e-04	1.97e+02	1.00e-01	2
99	128	196	+2.7681e+01	1.35e-01	2.94e+00	2.01e-04	1.97e+02	1.00e-01	2
100	130	197	+2.7247e+01	9.47e-01	2.92e+00	7.08e-05	1.97e+02	1.00e-01	2
101	132	198	+2.7247e+01	9.47e-02	2.92e+00	7.08e-05	1.97e+02	1.00e-01	2
102	134	199	+2.6946e+01	6.63e-01	2.91e+00	2.47e-05	1.97e+02	1.00e-01	2
103	136	200	+2.4927e+01	6.63e-01	2.83e+00	7.23e-03	1.97e+02		

								1.00e-01	2
104	138	201	+2.2976e+01	1.33e+00	2.72e+00	7.78e-03	1.97e+02	1.00e-01	2
105	140	202	+2.2976e+01	1.33e-01	2.72e+00	7.78e-03	1.97e+02	1.00e-01	2
106	141	203	+2.2579e+01	2.65e-01	2.69e+00	2.59e-03	1.97e+02	1.00e-01	2
107	143	204	+2.1824e+01	1.86e+00	2.64e+00	6.05e-04	1.97e+02	1.00e-01	2
108	145	205	+2.1824e+01	1.86e-01	2.64e+00	6.05e-04	1.97e+02	1.00e-01	2
109	147	206	+2.1307e+01	1.30e+00	2.60e+00	2.16e-04	1.97e+02	1.00e-01	2
110	149	207	+2.1307e+01	1.30e-01	2.60e+00	2.16e-04	1.97e+02	1.00e-01	2
111	151	208	+2.0951e+01	9.09e-01	2.57e+00	7.69e-05	1.97e+02	1.00e-01	2
112	153	209	+2.0951e+01	9.09e-02	2.57e+00	7.69e-05	1.97e+02	1.00e-01	2
113	155	210	+2.0704e+01	6.37e-01	2.54e+00	2.71e-05	1.97e+02	1.00e-01	2
114	157	211	+2.0704e+01	6.37e-02	2.54e+00	2.71e-05	1.97e+02	1.00e-01	2
115	159	212	+2.0532e+01	4.46e-01	2.53e+00	9.47e-06	1.97e+02	1.00e-01	2
116	161	213	+1.9372e+01	8.91e-01	2.40e+00	2.91e-03	1.97e+02	1.00e-01	2
117	163	214	+1.9372e+01	8.91e-02	2.40e+00	2.91e-03	1.97e+02	1.00e-01	2
118	164	215	+1.9140e+01	1.78e-01	2.37e+00	1.20e-03	1.97e+02	1.00e-01	2
119	166	216	+1.8691e+01	1.25e+00	2.31e+00	2.20e-04	1.97e+02	1.00e-01	2
120	168	217	+1.8691e+01	1.25e-01	2.31e+00	2.20e-04	1.97e+02	1.00e-01	2
121	170	218	+1.8384e+01	8.73e-01	2.26e+00	7.70e-05	1.97e+02	1.00e-01	2
122	172	219	+1.8384e+01	8.73e-02	2.26e+00	7.70e-05	1.97e+02	1.00e-01	2
123	174	220	+1.8173e+01	6.11e-01	2.23e+00	2.70e-05	1.97e+02	1.00e-01	2
124	176	221	+1.8173e+01	6.11e-02	2.23e+00	2.70e-05	1.97e+02	1.00e-01	2
125	178	222	+1.8027e+01	4.28e-01	2.20e+00	9.39e-06	1.97e+02	1.00e-01	2
126	180	223	+1.7051e+01	4.28e-01	2.02e+00	2.86e-03	1.97e+02	1.00e-01	2
127	182	224	+1.6146e+01	8.56e-01	1.79e+00	2.94e-03	1.97e+02	1.00e-01	2
128	184	225	+1.6146e+01	8.56e-02	1.79e+00	2.94e-03	1.97e+02		

								1.00e-01	2
129	185	226	+1.5972e+01	8.56e-02	1.75e+00	1.09e-03	1.97e+02	1.00e-01	2
130	186	227	+1.5804e+01	8.56e-02	1.70e+00	1.03e-03	1.97e+02	1.00e-01	2
131	187	228	+1.5640e+01	8.56e-02	1.65e+00	1.01e-03	1.97e+02	1.00e-01	2
132	188	229	+1.5480e+01	8.56e-02	1.60e+00	9.85e-04	1.97e+02	1.00e-01	2
133	189	230	+1.5323e+01	8.56e-02	1.55e+00	9.59e-04	1.97e+02	1.00e-01	2
134	190	231	+1.5170e+01	8.56e-02	1.50e+00	9.31e-04	1.97e+02	1.00e-01	2
135	191	232	+1.5019e+01	8.56e-02	1.45e+00	8.98e-04	1.97e+02	1.00e-01	2
136	192	233	+1.4873e+01	8.56e-02	1.39e+00	8.62e-04	1.97e+02	1.00e-01	2
137	193	234	+1.4729e+01	8.56e-02	1.34e+00	8.22e-04	1.97e+02	1.00e-01	2
138	194	235	+1.4589e+01	8.56e-02	1.29e+00	7.80e-04	1.97e+02	1.00e-01	2
139	195	236	+1.4451e+01	8.56e-02	1.24e+00	7.35e-04	1.97e+02	1.00e-01	2
140	196	237	+1.4316e+01	8.56e-02	1.19e+00	6.88e-04	1.97e+02	1.00e-01	2
141	197	238	+1.4184e+01	8.56e-02	1.14e+00	6.41e-04	1.97e+02	1.00e-01	2
142	198	239	+1.4055e+01	8.56e-02	1.09e+00	5.95e-04	1.97e+02	1.00e-01	2
143	199	240	+1.3929e+01	8.56e-02	1.04e+00	5.50e-04	1.97e+02	1.00e-01	2
144	200	241	+1.3805e+01	8.56e-02	9.87e-01	5.07e-04	1.97e+02	1.00e-01	2
145	201	242	+1.3684e+01	8.56e-02	9.37e-01	4.67e-04	1.97e+02	1.00e-01	2
146	202	243	+1.3566e+01	8.56e-02	8.88e-01	4.32e-04	1.97e+02	1.00e-01	2
147	203	244	+1.3451e+01	8.56e-02	8.38e-01	4.00e-04	1.97e+02	1.00e-01	2
148	204	245	+1.3338e+01	8.56e-02	7.87e-01	3.71e-04	1.97e+02	1.00e-01	2
149	205	246	+1.3230e+01	8.56e-02	7.37e-01	3.47e-04	1.97e+02	1.00e-01	2
150	206	247	+1.3125e+01	8.56e-02	6.87e-01	3.25e-04	1.97e+02	1.00e-01	2
151	207	248	+1.3023e+01	8.56e-02	6.36e-01	3.06e-04	1.97e+02	1.00e-01	2
152	208	249	+1.2926e+01	8.56e-02	5.86e-01	2.89e-04	1.97e+02	1.00e-01	2
153	209	250	+1.2833e+01	8.56e-02	5.37e-01	2.72e-04	1.97e+02		

								1.00e-01	2
154	210	251	+1.2744e+01	8.56e-02	4.88e-01	2.56e-04	1.97e+02	1.00e-01	2
155	211	252	+1.2660e+01	8.56e-02	4.40e-01	2.40e-04	1.97e+02	1.00e-01	2
156	212	253	+1.2581e+01	8.56e-02	3.94e-01	2.22e-04	1.97e+02	1.00e-01	2
157	213	254	+1.2506e+01	8.56e-02	3.49e-01	2.04e-04	1.97e+02	1.00e-01	2
158	214	255	+1.2435e+01	8.56e-02	3.07e-01	1.85e-04	1.97e+02	1.00e-01	2
159	215	256	+1.2370e+01	8.56e-02	2.67e-01	1.64e-04	1.97e+02	1.00e-01	2
160	216	257	+1.2308e+01	1.71e-01	2.30e-01	1.42e-04	1.97e+02	1.00e-01	2
161	218	258	+1.2196e+01	1.20e+00	1.62e-01	9.26e-06	1.97e+02	1.00e-01	2
162	219	260	+1.2196e+01	1.20e-01	1.62e-01	9.26e-06	1.97e+02	1.00e-01	2
163	221	261	+1.2128e+01	8.39e-01	1.24e-01	1.60e-06	1.97e+02	1.00e-01	2
164	223	262	+1.1725e+01	1.68e+00	7.26e-02	2.90e-04	1.97e+02	1.00e-01	2
165	224	264	+1.1662e+01	1.68e+00	2.64e-02	6.67e-04	1.97e+02	1.00e-01	2
166	225	266	+1.1684e+01	1.68e+00	7.27e-02	1.68e-03	1.97e+02	1.00e-01	2
167	227	268	+1.1689e+01	3.36e+00	9.35e-02	4.77e-06	1.97e+02	1.00e-01	2
168	229	270	+1.1644e+01	3.36e+00	1.24e-01	1.34e-05	1.97e+02	1.00e-01	4
169	231	273	+1.1383e+01	6.08e+00	1.47e-01	9.72e-06	1.97e+02	1.00e-01	4
170	233	275	+1.1383e+01	6.08e-01	1.47e-01	9.72e-06	1.97e+02	1.00e-01	2
171	234	277	+1.1317e+01	1.22e+00	1.42e-01	1.96e-04	1.97e+02	1.00e-01	2
172	236	279	+1.1317e+01	1.22e-01	1.42e-01	1.96e-04	1.97e+02	1.00e-01	2
173	237	281	+1.1306e+01	1.22e-01	1.40e-01	1.42e-04	1.97e+02	1.00e-01	2
174	239	283	+1.1296e+01	8.51e-01	1.37e-01	1.66e-06	1.97e+02	1.00e-01	2
175	241	284	+1.1255e+01	1.70e+00	1.69e-01	1.33e-04	1.97e+02	1.00e-01	2
176	243	286	+1.1255e+01	1.70e-01	1.69e-01	1.33e-04	1.97e+02	1.00e-01	2
177	245	288	+1.1212e+01	3.40e-01	1.31e-01	8.63e-05	1.97e+02	1.00e-01	2
178	247	290	+1.1175e+01	3.40e-01	1.38e-01	7.93e-05	1.97e+02		

							1.00e-01	2
179	249	292	+1.1143e+01	6.80e-01	1.30e-01	2.88e-05	1.97e+02	1.00e-01	2
180	251	294	+1.1143e+01	6.80e-02	1.30e-01	2.88e-05	1.97e+02	1.00e-01	2
181	253	295	+1.1134e+01	4.76e-01	1.32e-01	9.10e-07	1.97e+02	1.00e-01	2
182	255	296	+1.1084e+01	9.53e-01	1.31e-01	1.02e-07	1.97e+02	1.00e-01	2
183	257	297	+1.0985e+01	1.91e+00	1.25e-01	9.15e-07	1.97e+02	1.00e-01	2
184	259	299	+1.0985e+01	1.91e-01	1.25e-01	9.15e-07	1.97e+02	1.00e-01	2
185	261	300	+1.0962e+01	3.81e-01	1.34e-01	6.74e-05	1.97e+02	1.00e-01	2
186	263	302	+1.0921e+01	7.62e-01	1.25e-01	2.65e-05	1.97e+02	1.00e-01	2
187	265	304	+1.0830e+01	7.62e-01	1.31e-01	6.34e-05	1.97e+02	1.00e-01	2
188	267	306	+1.0743e+01	5.33e+00	1.22e-01	1.69e-06	1.97e+02	1.00e-01	2
189	269	308	+1.0743e+01	5.33e-01	1.22e-01	1.69e-06	1.97e+02	1.00e-01	2
190	271	309	+1.0743e+01	5.33e-02	1.22e-01	1.69e-06	1.97e+02	1.00e-01	2
191	273	310	+1.0736e+01	3.73e-01	1.23e-01	2.79e-07	1.97e+02	1.00e-01	2
192	275	311	+1.0692e+01	2.61e+00	1.24e-01	7.41e-07	1.97e+02	1.00e-01	2
193	277	313	+1.0692e+01	2.61e-01	1.24e-01	7.41e-07	1.97e+02	1.00e-01	2
194	279	314	+1.0669e+01	5.23e-01	1.18e-01	4.66e-06	1.97e+02	1.00e-01	2
195	281	316	+1.0669e+01	5.23e-02	1.18e-01	4.66e-06	1.97e+02	1.00e-01	2
196	283	317	+1.0660e+01	3.66e-01	1.21e-01	1.83e-06	1.97e+02	1.00e-01	2
197	285	318	+1.0660e+01	3.66e-02	1.21e-01	1.83e-06	1.97e+02	1.00e-01	2
198	287	319	+1.0655e+01	2.56e-01	1.21e-01	3.12e-08	1.97e+02	1.00e-01	2
199	289	320	+1.0624e+01	1.79e+00	1.23e-01	1.05e-06	1.97e+02	1.00e-01	2
200	291	322	+1.0624e+01	1.79e-01	1.23e-01	1.05e-06	1.97e+02	1.00e-01	2
201	293	323	+1.0608e+01	3.59e-01	1.18e-01	2.76e-06	1.97e+02	1.00e-01	2
202	295	325	+1.0608e+01	3.59e-02	1.18e-01	2.76e-06	1.97e+02	1.00e-01	2
203	297	326	+1.0603e+01	2.51e-01	1.19e-01	2.87e-07	1.97e+02		

								1.00e-01	2
204	299	327	+1.0572e+01	5.02e-01	1.24e-01	1.40e-05	1.97e+02	1.00e-01	2
205	301	329	+1.0519e+01	1.00e+00	1.16e-01	6.76e-06	1.97e+02	1.00e-01	2
206	303	331	+1.0396e+01	1.00e+00	1.17e-01	1.05e-05	1.97e+02	1.00e-01	2
207	305	333	+1.0248e+01	7.03e+00	1.22e-01	2.05e-05	1.97e+02	1.00e-01	2
208	307	336	+1.0248e+01	7.03e-01	1.22e-01	2.05e-05	1.97e+02	1.00e-01	4
209	309	338	+1.0248e+01	7.03e-02	1.22e-01	2.05e-05	1.97e+02	1.00e-01	2
210	311	339	+1.0243e+01	1.41e-01	1.15e-01	6.91e-06	1.97e+02	1.00e-01	2
211	313	340	+1.0237e+01	2.81e-01	1.09e-01	4.76e-06	1.97e+02	1.00e-01	2
212	315	342	+1.0210e+01	5.62e-01	1.16e-01	2.41e-05	1.97e+02	1.00e-01	2
213	317	344	+1.0210e+01	5.62e-02	1.16e-01	2.41e-05	1.97e+02	1.00e-01	2
214	319	345	+1.0206e+01	3.94e-01	1.12e-01	9.86e-07	1.97e+02	1.00e-01	2
215	321	346	+1.0179e+01	3.94e-01	1.04e-01	1.07e-06	1.97e+02	1.00e-01	2
216	323	348	+1.0138e+01	7.87e-01	1.14e-01	5.63e-05	1.97e+02	1.00e-01	2
217	325	350	+1.0138e+01	7.87e-02	1.14e-01	5.63e-05	1.97e+02	1.00e-01	2
218	327	351	+1.0133e+01	5.51e-01	1.08e-01	3.29e-06	1.97e+02	1.00e-01	2
219	329	352	+1.0080e+01	5.51e-01	1.10e-01	7.37e-06	1.97e+02	1.00e-01	2
220	331	354	+1.0036e+01	3.86e+00	9.63e-02	7.97e-06	1.97e+02	1.00e-01	2
221	331	354	+1.0036e+01	1.93e+01	1.02e-01	7.97e-06	1.00e+00	2.00e-02	0
222	332	358	+9.6431e+00	1.93e+01	1.00e-01	1.93e-03	1.00e+00	2.00e-02	4
223	333	361	+9.3957e+00	1.93e+01	9.89e-02	1.10e-01	1.00e+00	2.00e-02	4
224	334	364	+9.2505e+00	1.93e+01	8.72e-02	7.98e-04	1.00e+00	2.00e-02	4
225	335	368	+9.2505e+00	2.44e+00	8.72e-02	7.98e-04	1.00e+00	2.00e-02	2
226	336	370	+9.0487e+00	4.88e+00	4.97e-02	3.25e-03	1.00e+00	2.00e-02	2
227	337	372	+8.9017e+00	4.98e+00	1.97e-02	3.41e-03	1.00e+00	2.00e-02	4
228	338	374	+8.8248e+00	1.32e+01	4.52e-02	8.36e-04	1.00e+00		

								2.00e-02	4
229	339	377	+8.8101e+00	1.32e+01	4.31e-02	9.80e-06	1.00e+00	2.00e-02	2
230	340	380	+8.8463e+00	1.32e+01	3.41e-02	8.61e-06	1.00e+00	2.00e-02	4
231	341	381	+8.8459e+00	1.32e+01	3.35e-02	3.10e-08	1.00e+00	2.00e-02	4
232	342	382	+8.8456e+00	1.32e+01	3.30e-02	2.72e-08	1.00e+00	2.00e-02	4
233	344	384	+8.8115e+00	1.32e+01	6.24e-04	7.07e-06	1.00e+00	2.00e-02	4
234	344	384	+8.8115e+00	6.61e+01	3.67e-03	7.07e-06	1.00e+00	4.00e-03	0
235	345	386	+8.7543e+00	6.61e+01	7.04e-03	1.11e-04	1.00e+00	4.00e-03	4
236	346	388	+8.7413e+00	6.61e+01	1.91e-03	3.14e-06	1.00e+00	4.00e-03	4
237	346	388	+8.7413e+00	3.31e+02	2.44e-03	3.14e-06	1.00e+00	8.00e-04	0
238	347	390	+8.7252e+00	3.31e+02	7.32e-04	5.64e-07	1.00e+00	8.00e-04	4
239	347	390	+8.7252e+00	1.65e+03	8.38e-04	5.64e-07	1.00e+00	1.60e-04	0
240	348	392	+8.7220e+00	1.65e+03	2.19e-04	3.29e-08	1.00e+00	1.60e-04	4
241	349	397	+8.7218e+00	1.65e+03	8.30e-05	2.98e-09	1.00e+00	1.60e-04	4
242	349	397	+8.7218e+00	8.27e+03	1.04e-04	2.98e-09	1.00e+00	3.20e-05	0
243	350	399	+8.7213e+00	8.27e+03	3.43e-05	9.54e-10	1.00e+00	3.20e-05	4
244	351	404	+8.7212e+00	8.27e+03	9.31e-06	1.04e-09	1.00e+00	3.20e-05	4
245	351	404	+8.7212e+00	4.13e+04	1.36e-05	1.04e-09	1.00e+00	6.40e-06	0
246	352	406	+8.7211e+00	4.13e+04	3.23e-06	6.08e-11	1.00e+00	6.40e-06	4
247	352	406	+8.7211e+00	2.07e+05	2.38e-06	6.08e-11	1.00e+00	1.28e-06	0
248	353	408	+8.7210e+00	2.07e+05	8.02e-07	2.56e-12	1.00e+00	1.28e-06	4
249	354	413	+8.7210e+00	2.07e+05	6.99e-07	7.54e-10	1.00e+00	1.28e-06	4
250	354	413	+8.7210e+00	1.03e+06	5.29e-07	7.54e-10	1.00e+00	2.56e-07	0
251	355	415	+8.7210e+00	1.03e+06	1.36e-06	1.06e-11	1.00e+00	2.56e-07	4
252	356	416	+8.7210e+00	1.03e+06	1.23e-06	5.68e-14	1.00e+00	2.56e-07	4
253	358	417	+8.7210e+00	1.03e+06	2.56e-07	0.00e+00	1.00e+00		

```
2.56e-07   |   4   |
|  254  |  359  |  423  | +8.7210e+00 | 1.03e+06 | 2.40e-07 | 1.33e-09 | 1.00e+00 |
2.56e-07   |   1   |
|  255  |  359  |  423  | +8.7210e+00 | 5.17e+06 | 2.05e-07 | 1.33e-09 | 1.00e+00 |
5.12e-08   |   0   |
|  256  |  360  |  425  | +8.7210e+00 | 5.17e+06 | 1.16e-07 | 6.20e-12 | 1.00e+00 |
5.12e-08   |   4   |
|  257  |  361  |  431  | +8.7210e+00 | 5.17e+06 | 1.17e-07 | 2.01e-10 | 1.00e+00 |
5.12e-08   |   1   |
|  258  |  362  |  432  | +8.7210e+00 | 5.17e+06 | 4.06e-09 | 0.00e+00 | 1.00e+00 |
5.12e-08   |   4   |

`gtol` termination condition is satisfied.
Number of iterations: 258, function evaluations: 362, CG iterations: 432,
optimality: 4.06e-09, constraint violation: 0.00e+00, execution time:  5.5 s.
Optimal solution: [1.88675408e-01 2.01475408e-01 2.72000001e+02 1.52347973e+01
 1.19999998e+01 2.49482263e-01 2.73000001e+02]
----
```

+*In[31]:*+
[source, ipython3]

```
# Define the ranges for Do and Lt
Do_values = np.linspace(0.03, 0.3, 100)  # Range for Do
Di_values = np.linspace(0.001, 0.25, 100)  # Range for Di

# Create a grid of Do and Lt values
Do_grid, Di_grid = np.meshgrid(Do_values, Di_values)

# Compute the objective function values for each combination of Do and Lt
obj_values = obj_func([0.05, 0.1, 274, 40, 0.58, 0.5, 322])  # Assuming initial
values for other variables
Z = np.array([[obj_func([Di, Do, 272, 13.7, 12, 0.226, 273]) for Do in Do_values]
for Di in Di_values])

# Plot the contour
plt.figure(figsize=(8, 6))
contour = plt.contourf(Di_grid, Do_grid, Z, cmap='viridis', levels=20)
cbar = plt.colorbar(contour)
cbar.set_label('Objective Function Value')
plt.xlabel('Inner tube diameter (Di)')
plt.ylabel('Outer tube diameter (Do)')
plt.title('Contour Plot of Objective Function')

# Plot the optimization path
plt.plot(Di_val, Do_val, color='red', marker='o', linestyle='-', label='Optimization
Path',markersize=4)

# Show the legend
```

```
plt.legend()
#plt.savefig('Optimization_path.png',dpi = 300)
# Show the plot
plt.grid(True)
plt.show()
----
```

```
+*Out[31]:*+
----
![png](output_3_0.png)
----
```

```
+*In[32]:*+
[source, ipython3]
----
# Create a 3D plot
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')

# Plot the surface
surf = ax.plot_surface(Di_grid, Do_grid, Z, cmap='viridis', edgecolor='none')
cbar = fig.colorbar(surf)
cbar.set_label('Objective Function Value')
ax.set_xlabel('Inner tube diameter (Di)')
ax.set_ylabel('Outer tube diameter (Do)')
ax.set_zlabel('Objective Function Value')
ax.set_title('Surface Plot of Objective Function')
#plt.savefig('Surface_plot.png',dpi = 300)
plt.show()
----
```

```
+*Out[32]:*+
----
![png](output_4_0.png)
----
```

```
+*In[33]:*+
[source, ipython3]
----
fig = plt.figure(figsize=(8, 5))
plt.plot(itr_list,obj_list)
plt.xlabel('Iterations')
plt.ylabel('Function value')
plt.title('Function value vs iterations')
#plt.savefig('fun_value.png',dpi = 300)
# Show the plot
```

```
plt.grid(True)
plt.show()
----
```

+*Out[33]:*+

```
----
![png](output_5_0.png)
----
```

+*In[34]:*+
[source, ipython3]

```
----
fig = plt.figure(figsize=(8, 5))
plt.plot(itr_list,grad_lag)
plt.xlabel('Iterations')
plt.ylabel('Gradient norm')
plt.yscale('log')
plt.title('Norm of the gradient vs iterations')
#plt.savefig('grad_value.png',dpi = 300)
# Show the plot
plt.grid(True)
plt.show()
----
```

+*Out[34]:*+

```
----
![png](output_6_0.png)
----
```

+*In[35]:*+
[source, ipython3]

```
----
fig = plt.figure(figsize=(8, 5))
plt.plot(itr_list,gov_val)
plt.xlabel('Iterations')
plt.ylabel('Governing equation value')
plt.title('Governing equation vs iterations')
#plt.savefig('gov_value.png',dpi = 300)
# Show the plot
plt.grid(True)
plt.show()
----
```

+*Out[35]:*+

```
----
```

```
![png](output_7_0.png)
----


+*In[36]:*+
[source, ipython3]
----
X01 = [0.3256, 0.33844, 272, 25.938, 11.999, 0.419, 273]
X03 = [0.2332, 0.24608, 272, 18.7211, 11.999, 0.3047, 273]
X05 = [0.188, 0.2014, 272, 15.23, 11.999, 0.249, 273]
X07 = [0.1506, 0.1634, 271.93, 13.06, 11.999, 0.215, 273]
X09 = [0.0968, 0.1096, 271.1, 14.901, 11.999, 0.251, 273]

X_value = np.array([X01,X03,X05,X07,X09])

shell_value = []
therm_value = []

for i in range(len(X_value)):

    Di = X_value[i][0]
    Do = X_value[i][1]
    Ti = X_value[i][2]
    A = X_value[i][3]
    Lt = X_value[i][4]
    Vw = X_value[i][5]
    To = X_value[i][6]

    # constants
    Pr_l = 4.34
    Pr_w = 6.90
    Vl = 5  # ft/s
    T_wi = 323
    #T_we = 273
    #T_l = 273
    Ta = 242
    k = 0.00257
    v = 1.05*0.00001 # visocosity of fluid - same for both
    r = 1.4*0.00001  # roughness of the pipe

    # variable
    Ai = (np.pi*Di**2)/4
    Ao = (np.pi*Do**2)/4
    Re_i = (Vl * Di)/v
    Re_o = (Vw * Do)/v

    S = np.log10(Re_i/1.816*np.log10(1.1*Re_i/np.log10(1+1.1*Re_i)))
    f = -2*np.log10(2.18*S/Re_i + r/(3.71*Di))
    Nu_i = ((f/2)*(Re_i-1000)*Pr_l)/(1+12.7*(Pr_l**(2/3)-1)*(f/2)**0.5)
    Nu_o = 0.0237*Re_o**0.618*Pr_w**(1/3)
```

```
    h_i = Nu_i*k/Di
    h_o = Nu_o*k/Do

    part_1 = 1/(h_i*Ai*(Ti - Ta))
    part_2 = (1/(2*np.pi*k*Lt))*A*np.log(Do/Di)
    part_3 = 1/(h_o*Ao*(T_wi - To))

    therm_val = part_1 + part_2 + part_3

    shell_val = 2.66175 * Lt * Do**2 * 5

    shell_value.append(shell_val)
    therm_value.append(therm_val)

fig = plt.figure(figsize=(8, 5))
plt.plot(shell_value,therm_value, label = 'Pareto front')
plt.xlabel('Thermal function')
plt.ylabel('Structure function')
plt.legend()
#plt.savefig('pareto_value.png',dpi = 300)
# Show the plot
plt.grid(True)
plt.show()
----


+*Out[36]:*+
----
![png](output_8_0.png)
----


+*In[ ]:*+
[source, ipython3]
----

----
```