PROJECT: Optimizing Heat Loss and Boundary Layer Thickness in Laminar Airflow Over a Heated Plate

ES604: Engineering Optimization

Gaurav Budhwani
2211085
Chemical Engineering

Md Sibtain Raza
22110148
Chemical Engineering

Professor

Hari Ganesh

November 12, 2024

Contents

1 Introduction

Heat transfer optimization is a critical area of study in thermal engineering, especially for systems requiring efficient heat management and conservation. Our project explores the optimization of laminar airflow over a heated flat plate—a fundamental setup in convective heat transfer applications. This study specifically addresses the thermal boundary layer thickness, an essential factor influencing heat dissipation rates, and aims to minimize heat loss from the plate to the surrounding air.

The optimization employs Pyomo for formulation in Python and compares results with MATLAB-based solutions, which are both practical and illustrative in academic and applied engineering contexts. We aim to assess and refine boundary conditions, with a focus on optimizing parameters like boundary layer thickness to achieve minimal heat loss. This investigation serves as a basis for selecting the most effective computational tools and methods for laminar heat transfer studies.

By exploring multiple optimization techniques, we expect to offer insights into the efficiency and accuracy of the Pyomo model in comparison to MATLAB results, ultimately guiding future research in selecting robust methods for similar heat transfer optimization problems.

2 Methods

In this study, we employed computational optimization techniques to analyze and minimize heat loss in laminar airflow over a heated flat plate. The primary goal was to model, optimize, and compare key parameters affecting convective heat transfer, specifically targeting thermal boundary layer thickness and overall heat dissipation. The methods were implemented in both *Pyomo*, a Python-based optimization framework, and *MATLAB*, enabling a comparative analysis of solution accuracy and computational efficiency between the two platforms. This section details the formulation of the problem, modeling approach, constraints, and solution procedures for each software tool.

2.1 Problem Setup and Objective

This study examines laminar airflow over a heated flat plate to minimize heat loss while maintaining optimal thermal boundary layer thickness. The thermal boundary layer thickness, influenced by the airflow characteristics and plate temperature, significantly affects the rate of convective heat transfer.

We used *Pyomo* and *MATLAB* to develop optimization models, with the objective of comparing the efficacy of the two software platforms. The optimization problem aimed to achieve minimal heat loss and control the boundary layer thickness under given constraints, such as air velocity, plate temperature, and material properties.

2.2 Constraints and Boundary Conditions

The models in both Pyomo and MATLAB were subjected to the following constraints:

- **Reynolds number constraint**: to ensure laminar flow, with a maximum value of 50,000.
- **Temperature bounds**: plate temperature was constrained between 80°C and 130°C.
- **Boundary layer thickness**: targeted values were defined for maintaining effective heat transfer characteristics.

Step 1: Key Equations and Definitions

- **Reynolds Number** (Re):

$$\mathrm{Re} = \frac{\rho u_\infty x}{\mu}$$

  where:

  - ρ is the fluid density,
  - u_∞ is the free stream velocity,
  - x is the distance from the leading edge of the plate,
  - μ is the dynamic viscosity.

- **Prandtl Number** (Pr):

$$\mathrm{Pr} = \frac{c_p \mu}{k}$$

  where:

  - c_p is the specific heat at constant pressure,
  - μ is the dynamic viscosity,
  - k is the thermal conductivity.

- **Thermal Boundary Layer Thickness** (δ_t):

$$\delta_t = \frac{0.977}{\mathrm{Pr}^{1/3}} \delta$$

  where δ is the hydraulic boundary layer thickness.

- **Hydraulic Boundary Layer Thickness** (δ):

$$\delta = \frac{4.64}{\sqrt{\mathrm{Re}_x}}\, x$$

  where Re_x is the Reynolds number at a distance x from the leading edge.

- **Average Heat Transfer Coefficient** ($\bar{h}$):

$$\bar{h} = \frac{N_u k}{x}$$

where N_u is the Nusselt number. For laminar flow over a flat plate ($\mathrm{Re} < 5 \times 10^5$), N_u is given by:

$$N_u = 0.332\,\mathrm{Re}^{1/2}\,\mathrm{Pr}^{1/3}$$

- **Heat Flow** (Q):

$$Q = \bar{h} A (T_w - T_\infty)$$

where:

  - A is the surface area of the plate,
  - T_w is the plate temperature,
  - T_∞ is the ambient temperature.

Step 2: Formulating the Objective Functions
Formulating the Objective function from the equations obtained above:

2.3 Case 1: Minimizing Thermal Boundary Layer Thickness (δ_t)

- From the definition of thermal boundary layer thickness:

$$\delta_t = \frac{0.977}{\mathrm{Pr}^{1/3}}\delta$$

- Substitute the expression for δ (hydraulic boundary layer thickness):

$$\delta = \frac{4.64}{\sqrt{\mathrm{Re}_x}}\,x$$

Therefore:

$$\delta_t = \frac{0.977}{\mathrm{Pr}^{1/3}} \cdot \frac{4.64}{\sqrt{\mathrm{Re}_x}}\,x$$

- The objective function for Case 1 is:

$$\min\ \delta_t(x, \mathrm{Re})$$

subject to:

$$0 < \mathrm{Re} < 50000, \quad Q^{\mathrm{lo}} < Q < Q^{\mathrm{up}}$$
$$x^{\mathrm{lo}} < x < x^{\mathrm{up}}, \quad T^{\mathrm{lo}} < T < T^{\mathrm{up}}$$

2.4 Case 2: Minimizing Heat Flow (Q)

- From the heat flow equation:

$$Q = \bar{h} A (T_w - T_\infty)$$

- Substitute the expression for the average heat transfer coefficient $\bar{h}$:

$$\bar{h} = \frac{N_u k}{x}$$

where:

$$N_u = 0.332 \, \mathrm{Re}^{1/2} \, \mathrm{Pr}^{1/3}$$

Therefore:

$$Q = \left(\frac{0.332 \, \mathrm{Re}^{1/2} \, \mathrm{Pr}^{1/3} \, k}{x} \right) A (T_w - T_\infty)$$

- The objective function for Case 2 is:

$$\min \, Q(x, T, \delta)$$

subject to:

$$0 < \mathrm{Re} < 50000, \quad \delta^{\mathrm{lo}} < \delta < \delta^{\mathrm{up}}$$
$$x^{\mathrm{lo}} < x < x^{\mathrm{up}}, \quad T^{\mathrm{lo}} < T < T^{\mathrm{up}}$$

2.5 Relationship Between the Two Objectives

In heat transfer problems, there is often a trade-off between the thickness of the boundary layer and the total heat transfer. Minimizing the boundary layer thickness δ_t tends to increase heat transfer rates, potentially increasing Q. However, in this problem, the objective is to explore both scenarios:

2.6 Model Formulation in Pyomo

Pyomo, a Python-based optimization library, was chosen for its flexibility in defining nonlinear optimization problems. We formulated the heat transfer equations in Pyomo based on the governing heat transfer principles, including:

- **Thermal boundary layer thickness** (δ_t) estimated using convective heat transfer correlations.
- **Heat transfer coefficient** (h) derived from Nusselt number correlations, which were then used to compute the heat flux from the plate to the air.
- **Objective Function**: Minimize the heat loss by optimizing the plate temperature and boundary layer thickness while meeting airflow and material constraints.

2.7 Implementation using Pyomo

1. **Define Model and Constants:**
   - We start by initializing a Pyomo ConcreteModel, which will contain all variables, constraints, and objectives.
   - Physical properties of air, such as thermal conductivity k_{air}, dynamic viscosity μ_{air}, Prandtl number Pr, density ρ_{air}, and specific heat capacity c_p, are defined based on known values.
   - The ambient temperature $T_\infty$and free stream velocity u_∞ are set as constants for the model.
2. **Define Design Variables:**
   - Variables to optimize are defined, including:
     - x: Characteristic length of the plate.
     - T: Wall temperature.
     - δ_t: Thermal boundary layer thickness.
   - These variables are constrained to be non-negative to maintain physical validity.
3. **Define Intermediate Variables and Constraints:**
   - Additional variables and constraints related to heat transfer are defined to capture relationships in fluid dynamics and thermodynamics:
     - **Reynolds Number (Re):** Defined as $\text{Re} = \frac{\rho_{\text{air}} u_\infty x}{\mu_{\text{air}}}$.
     - **Nusselt Number (Nu):** Approximated for laminar flow as $\text{Nu} = 0.332\,\text{Re}^{1/2}\,\text{Pr}^{1/3}$.
     - **Heat Transfer Coefficient (h):** Calculated from $h = \frac{\text{Nu} \cdot k_{\text{air}}}{x}$.
     - **Thermal Boundary Layer Thickness (δ_t):** Expressed as $\delta_t = 0.977 \times \frac{4.64 \cdot x}{\text{Re}^{1/2} \cdot \text{Pr}^{1/3}}$.
     - **Heat Flow (Q):** Heat flow across the plate area is defined as $Q = 2 \cdot h \cdot x \cdot w \cdot (T - T_\infty)$, where $w = 1\,\text{m}$.
4. **Objective Function (Minimize δ_t):**
   - The model's objective function is to minimize δ_t to reduce the thermal boundary layer thickness, which potentially increases heat transfer efficiency.
5. **Constraints:**
   - Additional constraints on variables ensure feasible and physically realistic solutions (values taken for Illustrative purposes):
     - $0 < \text{Re} < 50000$ to maintain the laminar flow assumption.
     - Heat flow constraints $140\,\text{W} < Q < 190\,\text{W}$.

  - Characteristic length x is restricted to $0.2 < x < 1\,\text{m}$.
  - Wall temperature T is constrained to the range $80 < T < 130°\text{C}$.

6. **Solve the Model:**

   - Using the IPOPT solver, we solve for the design variables to find optimal values of x and T that minimize δ_t.

```python
from pyomo.environ import ConcreteModel, Var, Objective, Constraint,
    SolverFactory, minimize, NonNegativeReals, ConstraintList
import math

model = ConcreteModel()

T_inf = 65.6
k_air = 0.026
mu_air = 1.85e-5
Pr = 0.7189
rho_air = 1.177
u_inf = 0.8

model.x = Var(domain=NonNegativeReals, initialize=0.1)
model.T = Var(domain=NonNegativeReals, initialize=100)
model.delta_t = Var(domain=NonNegativeReals, initialize=0.016)

model.Re = Var(bounds=(1, 1000000), domain = NonNegativeReals)
model.Nu = Var()
model.h = Var()
model.Q = Var()

model.re_constraint = Constraint(expr=model.Re == (rho_air * u_inf * model.x)
     / mu_air)

model.nu_constraint = Constraint(expr=model.Nu == 0.332 * (((rho_air * u_inf
    * model.x) / mu_air)**0.5) * (Pr**(1/3)))

model.h_constraint = Constraint(expr=model.h == model.Nu * k_air / model.x)

model.delta_t_constraint = Constraint(expr=model.delta_t == 0.977 * 4.64 *
    model.x / (((rho_air * u_inf * model.x) / mu_air))**(1/2) * Pr**(1/3))

w = 1
model.q_constraint = Constraint(expr=model.Q == 2 * model.h * model.x * w * (
    model.T - T_inf))

model.obj_case1 = Objective(expr=model.delta_t, sense=minimize)

model.constraints = ConstraintList()
model.constraints.add(expr= (rho_air * u_inf * model.x) / mu_air >= 0)
model.constraints.add(expr= (rho_air * u_inf * model.x) / mu_air <= 50000)
model.constraints.add(expr=model.Q >= 140)
model.constraints.add(expr=model.Q <= 190)
model.constraints.add(expr=model.x >= 0.2)
model.constraints.add(expr=model.x <= 1)
model.constraints.add(expr=model.T >= 80)
model.constraints.add(expr=model.T <= 130)

solver = SolverFactory('cyipopt')

result_case1 = solver.solve(model, tee=True)
print("Case 1: Minimize delta_t")
print(f"Optimal x: {model.x()}")
print(f"Optimal T: {model.T()}")
print(f"Optimal Re: {(rho_air * u_inf * model.x()) / mu_air}")
print(f"Optimal delta_t: {model.delta_t()}")
print(f"Optimal Q: {model.Q()}")
```

Listing 1: Pyomo Code for Optimization of Case 1

```python
from pyomo.environ import ConcreteModel, Var, Objective, Constraint,
    SolverFactory, minimize, NonNegativeReals, ConstraintList
import math

model = ConcreteModel()

T_inf = 65.6
k_air = 0.026
mu_air = 1.85e-5
Pr = 0.7189
rho_air = 1.177
u_inf = 4

model.x = Var(domain=NonNegativeReals, initialize=0.1)
model.T = Var(domain=NonNegativeReals, initialize=100)
model.delta_t = Var(domain=NonNegativeReals, initialize=0.016)

model.Re = Var(bounds=(1, 1000000), domain = NonNegativeReals)
model.Nu = Var()
model.h = Var()
model.Q = Var()


model.re_constraint = Constraint(expr=model.Re == (rho_air * u_inf * model.x)
     / mu_air)

model.nu_constraint = Constraint(expr=model.Nu == 0.332 * (((rho_air * u_inf
    * model.x) / mu_air)**0.5) * (Pr**(1/3)))


model.h_constraint = Constraint(expr=model.h == model.Nu * k_air / model.x)

model.delta_t_constraint = Constraint(expr=model.delta_t == 0.977 * 4.64 *
    model.x / (((rho_air * u_inf * model.x) / mu_air))**(1/2) * Pr**(1/3))

w = 1
model.q_constraint = Constraint(expr=model.Q == model.h * model.x * w * (
    model.T - T_inf))

model.obj_case2 = Objective(expr=model.Q, sense=minimize)

model.constraints2 = ConstraintList()
model.constraints2.add(expr=(rho_air * u_inf * model.x) / mu_air >= 0)
model.constraints2.add(expr=(rho_air * u_inf * model.x) / mu_air <= 50000)
model.constraints2.add(expr=model.x >= 0.2)
model.constraints2.add(expr=model.x <= 1)
model.constraints2.add(expr=model.delta_t >= 0.00001)
model.constraints2.add(expr=model.delta_t <= 0.02)
model.constraints2.add(expr=model.T >= 80)
model.constraints2.add(expr=model.T <= 130)

solver = SolverFactory('cyipopt')

result_case2 = solver.solve(model, tee=True)
print("\nCase 2: Minimize Q")
print(f"Optimal x: {model.x()}")
print(f"Temperature (T): {model.T()}")
print(f"Optimal Q: {model.Q()}")
```

Listing 2: Pyomo Code for Optimization of Case 2

3 Model Implementation in MATLAB

In MATLAB, the optimization problem was structured with the same parameters and boundary conditions as in Pyomo to ensure comparability. MATLAB's optimization toolbox was used to implement nonlinear programming techniques for calculating heat loss and boundary layer thickness, allowing a direct comparison with Pyomo results.

- **In Case 1:** We are focused on enhancing heat transfer by minimizing δ_t.
- **In Case 2:** We are focused on reducing heat loss by minimizing Q.

These objectives are chosen to study two different aspects of the same heat transfer problem. The results can help identify the best design parameters to achieve efficient heat transfer under different priorities

```
clc;clear;close all

T_inf = 65.6;
k_air = 0.026;
mu_air = 1.85e-5;
Pr = 0.7189;
rho_air = 1.177;
u_inf = 0.8;
w = 1;

x0 = [0.2, 100];

objective = @(vars) 0.977 * 4.64 * vars(1) / sqrt((rho_air * u_inf * vars(1)) / mu_air) * Pr^(1/3);

options = optimoptions('fmincon', 'Display', 'iter', 'Algorithm', 'sqp');

[opt_vars, opt_delta_t] = fmincon(objective, x0, [], [], [], [], [0.2, 80], [1, 130], @constraints, options)
      ;

opt_x = opt_vars(1);
opt_T = opt_vars(2);
opt_Re = (rho_air * u_inf * opt_x) / mu_air;
opt_Nu = 0.332 * sqrt(opt_Re) * Pr^(1/3);
opt_h = opt_Nu * k_air / opt_x;
opt_Q = 2 * opt_h * opt_x * w * (opt_T - T_inf);

fprintf('Optimal x: %.4f\n', opt_x);
fprintf('Optimal T: %.4f\n', opt_T);
fprintf('Optimal Re: %.4f\n', opt_Re);
fprintf('Optimal Q: %.4f\n', opt_Q);
fprintf('Optimal delta_t: %.4f\n', opt_delta_t);

function [c, ceq] = constraints(vars)
    x = vars(1);
    T = vars(2);
    T_inf = 65.6;
    k_air = 0.026;
    mu_air = 1.85e-5;
    Pr = 0.7189;
    rho_air = 1.177;
    u_inf = 0.8;
    w = 1;

    Re = (rho_air * u_inf * x) / mu_air;
    Nu = 0.332 * sqrt(Re) * Pr^(1/3);
    h = Nu * k_air / x;
    Q = 2 * h * x * w * (T - T_inf);

    c = [
        -Re;                % Re >= 1
        Re - 50000;         % Re <= 50000
        140 - Q;            % Q >= 140
        Q - 190;            % Q <= 190
        0.2 - x;            % x >= 0.2
        x - 1;              % x <= 1
        80 - T;             % T >= 80
        T - 130;            % T <= 130
    ];

    ceq = [];
end
```

Listing 3: MATLAB Code for Optimization of Case 1

```
clc;clear;close all;

rho = 1.177;
mu = 1.85e-5;
u_infinity = 4;
Pr = 0.7189;
k = 0.026;
T_infinity = 65.6;

options = optimoptions('fmincon', 'Algorithm', 'interior-point', ...
                       'Display', 'iter', 'EnableFeasibilityMode', true, ...
                       'ConstraintTolerance', 1e-5);

objective2 = @(vars) (k * (0.332 * sqrt((rho * u_infinity * vars(1)) / mu) *
    Pr^(1/3)) / vars(1)) * (vars(2) - T_infinity) * vars(1);

nonlcon2 = @(vars) deal([], (rho * u_infinity * vars(1)) / mu - 50000);

lb2 = [0.02, 80];
ub2 = [1.0, 130];

x0_2 = [0.3, 100];

[opt_vars2, Q_min] = fmincon(objective2, x0_2, [], [], [], [], lb2, ub2,
    nonlcon2, options);

fprintf('Optimal x for minimizing Q (Case 2): %.4f m\n', opt_vars2(1));
fprintf('Optimal T_w for minimizing Q (Case 2): %.2f  C \n', opt_vars2(2));
fprintf('Minimum Heat Transfer (Q): %.4f W\n', Q_min);
```

Listing 4: MATLAB Code for Optimization of Case 2

4 Results and Discussions

The following are the optimal values of the design and constraint variables obtained from Pyomo and MATLAB:

```
EXIT: Optimal Solution Found.
Case 1: Minimize delta_t
Optimal x: 0.3882053057028124
Optimal T: 129.99998691094484
Optimal Re: 19758.600856744226
Optimal Q: 140.00002988057022
Optimal delta_t: 0.01121550271365076
```

Figure 1: Results for Case 1 using Pyomo

```
EXIT: Optimal Solution Found.

Case 2: Minimize Q
Optimal x: 0.19999990008915183
Temperature (T): 79.99999920321237
Optimal Re:  10179.45437426732
Optimal Q: 11.234635269316996
```

Figure 2: Results for Case 2 using Pyomo

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
Optimal x: 0.3882
Optimal T: 130.0000
Optimal Re: 19758.5844
Optimal Q: 140.0000
Optimal delta_t: 0.0112
```

Figure 3: Results for Case 1 using MATLAB

```
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
Optimal x for minimizing Q (Case 2): 0.1965 m
Optimal T_w for minimizing Q (Case 2): 80.00 °C
Optimal Re for minimizing Q (Case 2): 11218.61
Minimum Heat Transfer (Q): 24.8990 W
```

Figure 4: Results for Case 2 using MATLAB

5 Conclusion

6 Acknowledgment

We extend our heartfelt appreciation to everyone who contributed to the successful completion of this project. This accomplishment would not have been possible without the combined efforts, expertise, and support of numerous individuals, including our team members and the lab staff.

First and foremost, we express our deepest gratitude to Prof. Hari Ganesh, our project advisor. His invaluable guidance, insightful suggestions, and unwavering encouragement have been instrumental in shaping the direction of this project. His commitment to fostering our learning and research has been truly inspiring. We would also like to extend our thanks to all the Teaching Assistants for their support.

We extend our appreciation to our fellow peers and colleagues, whose valuable insights, stimulating discussions, and diverse perspectives have enriched our understanding throughout the project. Your collaboration has been invaluable in broadening our perspectives.

In essence, this project is a testament to the collective efforts of numerous individuals and entities, each contributing their expertise, time, and enthusiasm. We are deeply grateful for the opportunity to work on this project and for the support we have received throughout its journey.

Bibliography

[1] "Y. A. Çengel, M. A. Boles, and M. Kanoglu, Thermodynamics 2018."

[2] “Rankine cycle,” Wikipedia, Aug. 20, 2023." https://en.wikipedia.org/wiki/Rankine_cycle