MECH 579: Numerical Optimization
Department of Mechanical Engineering, McGill University

Final Project: Due 19th December, 2025

You are being contacted to aid in the design of an integrated CPU to minimize the maximum temperature experienced by the CPU when operating at steady state. The company has provided you with a heat equation solver that they use to develop the CPU's thermal model. This CPU is to be cooled by a fan in the case that moves air, with velocity v, along the top of the CPU at room temperature, $T_\infty = 293\text{K}$. This heat solver has the following assumptions:

1. It assumes that the CPU is a constant material throughout, with physical properties of silicon at $293K$
2. The heat loss of the sides of the CPU can be modelled as a wall with natural convection
3. The heat loss from the top of the CPU is modelled as forced convection and is treated inside the heat source term
4. There is no heat loss from the bottom of the CPU

The mathematical model is shown below,

$$\begin{aligned} \rho c\frac{\partial T}{\partial t} - k\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right) &= \dot{E} \in \Omega[0,0]\times[0.04,0.04] \\ \rho c\frac{\partial T}{\partial t} = \overset{\text{convection}}{\sum} h_s A_s(T_\infty - T) + \overset{\text{conduction}}{\sum} kA\frac{\partial T}{\partial \xi} &+ \dot{E} \in \partial\Omega \\ \dot{E} &= h_T A_T(T_\infty - T) + f(x,y) \\ f(x,y) &= ax + by + c; \end{aligned} \tag{1}$$

where ρ is the density of the CPU, c is the heat capacity of the CPU, k is the thermal conductivity coefficient of the CPU, h_s is convection coefficient along the side of the CPU, A_s is the area along the convection face, A is the area along the conduction face, ξ is an arbitrary direction, h_T is the convection coefficient along the top of the CPU, A_T is the area on the top of the CPU, and a, b, c are variables that describe the CPU's heat generation. The convection coefficient for the forced convection from the fan on the top surface is modelled as a flat plate and is calculated using the following equation,

$$\begin{cases} h_T(v,x) = 0.332\frac{k_a}{x}\text{Re}_x^{0.5}\text{Pr}^{1/3} & \text{Re}_x < 5\times 10^5 \\ \\ h_T(v,x) = 0.0296\frac{k_a}{x}\text{Re}_x^{0.8}\text{Pr}^{1/3} & \text{Re}_x \geq 5\times 10^5 \end{cases}, \tag{2}$$

where Re_x is the Reynolds number at location x, k_a is the conductive heat transfer coefficient of the air, and Pr is the Prandtl number of the air. The fan's efficiency as a function of the velocity can be represented as,

$$\eta(v) = -\frac{2}{1000}v^2 + \frac{8}{100}v. \tag{3}$$

The convection coefficient for the natural convection on the sides can be calculated using the following set of equations,

$$\begin{aligned} \beta(T) &= (\frac{T_\infty + T}{2})^{-1} \\ \mathrm{Ra} &= \frac{g\beta(T - T_\infty)dz^3}{\nu^2}\mathrm{Pr} \\ \mathrm{Nu} &= (0.825 + \frac{0.387\mathrm{Ra}^{1/6}}{(1 + (0.492/\mathrm{Pr})^{9/16})^{8/27}})^2 \\ h_s(T) &= \frac{k_a}{d\xi}\mathrm{Nu}, \end{aligned} \tag{4}$$

where dz is the height of the CPU and g is the acceleration due to gravity. The variables, a, b, c, associated with $f(x, y)$ are adjustable, allowing you to determine where the circuits of the CPU are located. However, they want to ensure that the CPU is rated for 10W. The company wants you to minimize the maximum temperature in the CPU and maximize the efficiency of the fan. This gives the following optimization statement,

$$\begin{aligned} \min_{a,b,c,v} \quad & \omega_1 \max_{x,y} T - \omega_2 \eta \\ \text{with respect to} \quad & a, b, c, v \\ \text{s.t.} \quad & \iiint f(x, y)dV = 10 \end{aligned} \tag{5}$$

Complete all questions below:

(a) Complete the evaluation of the objective function and constraint functions within the given `python` code.

(b) For weights $\omega_1 = 0.2$, solve the given optimization problem. Provide plots of the convergence of the gradient of the Lagrangian function, objective function, the maximum T, the efficiency η, the total power generation constraint, and the design parameter values a, b, c, and v as a function of the number of design iterations.

(c) Evaluate the sensitivity of the objective function with respect to the design parameters using automatic differentiation (AD) through the `jax` library. First, for one of the four design parameters, compare the AD-based derivative against finite-difference-based (FD) value. However, first establish a converged FD-based gradient. Provide a table with all significant digits comparing the derivatives. You may use either a direct or an adjoint approach.

(d) Revisit (b) above, but now provide your own AD-based derivatives.

Reports must be handed in a PDF format (hand-written notes will not be accepted). Provide a single zip (only zip file) that includes the `python` code(s) and the project. The settings for any parameters and initial design point should be identical to those used in the results shown in the report. The `python` code should produce figures that replicate the results that appear in the written report. The code will be executed, and the figures should match those within the report.

1 Appendix

There are a few aspects when it comes to coding that will aid you in the completion of the project. Below you will see all the preprogrammed routines in the HeatEquation2D class.

1.1 HeatEquation2D Member Functions and Variables

- verbose -> output the steady state error after 1000 iterations

- heat_generation_total -> The total integrated energy from the heat generation function. Updated every time self.heat_generation_function is called

- __init__(
    x:float, y:float, height:float, n_x:int, n_y:int,
    k:float=1.0, rho:float=1.0, cp:float=1.0, CFL:float=0.1,
    init_condition:Callable[[np.ndarray,np.ndarray], np.ndarray]=np.tanh
)
  Intializition function for the heat equation

  Parameters

  ---

  x (float): Physical Size of CPU in x-direction [m]

  y (float): Physical Size of CPU in y-direction [m]

  height (float): Physical Height of the CPU [m]‘

  n_x (int): Number of grid points in x-direction [m]

  n_y (int): Number of grid points in y-direction [m]

  k (float): The heat transfer coefficient of the CPU [W/[mK]]

  rho (float): Constant density of CPU [kg/m^3]

  cp (float): Specific heat capacity of CPU [kJ/[kgK]]

  CFL (float): Courant-Friedrichs-Lewy Number

  init_condition (function(x,y)): Initial condition of the CPU

- set_initial_conditions(
    initial_conditions:Callable[[np.ndarray,np.ndarray],np.ndarray]
)

Sets the initial condition

Parameters

---

initial_conditions(function(x,y)): Initial condition of the CPU

- reset()
  Resets the heat equation

- set_heat_generation(
      heat_generation_function: Callable[[np.ndarray,np.ndarray,float,
                                          float,float], np.ndarray],
      a: float, b: float, c: float
)
  Sets the heat generation function and associated variables

  Parameters

  ---

  heat_generation_function (function(x,y,a,b,c)): Function that
  dictates the heat generation by the CPU

  integrated_total (float): Total integrated value

  a, b, c (float): Variables associated with the heat generation function

- set_fan_velocity(
      v: float
)
  Sets the fan velocity

  Parameters

  ---

  v (float): Variable associated with the fan velocity

- step_forward_in_time()
  Steps forward in time 1 timestep

- solve_until_time(
      final_time: float
)
  Solves until time is reached

Parameters

---

final_time (float): Final time of simulation

1.2 Variable definitions

- $L = 0.04$ m
- N = 25
- $k = 149 \frac{\mathrm{W}}{\mathrm{mK}}$
- $\rho = 2323 \frac{\mathrm{kg}}{m^3}$
- $c = 704.611 \frac{\mathrm{J}}{\mathrm{kgK}}$
- $k_a = 0.02772 \frac{\mathrm{W}}{\mathrm{mK}}$
- $\nu = 1.847 \times 10^{-5} \frac{\mathrm{m}^2}{\mathrm{s}}$
- Pr = 0.7215
- $T_\infty = 293\mathrm{K}$
- $\alpha = \frac{k}{\rho c}$
- $\tau = \frac{\alpha dt}{dxdy}$

1.3 Time Stepping

The CPU has been discretized with a first order accurate scheme in space and an explicit first order scheme in time. Below is a description of the discretization on the boundaries and the volume terms. Bottom Left Corner: $i = 0, \quad j = 0$

$$T_{ij}^{n+1} = T_{ij}^n + 2\tau \frac{h_s dy}{k}(T_\infty - T_{ij}) + 2\tau \frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dx \frac{T_{i.j+1} - T_{ij}}{dy} + 2\tau dy \frac{T_{i+1.j} - T_{ij}}{dx} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau \frac{\dot{e}}{k} dxdy \quad (6)$$

Bottom Right Corner: $i = n_x - 1, \quad j = 0$

$$T_{ij}^{n+1} = T_{ij}^n + 2\tau \frac{h_s dy}{k}(T_\infty - T_{ij}) + 2\tau \frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dx \frac{T_{i.j+1} - T_{ij}}{dy} + 2\tau dy \frac{T_{i-1.j} - T_{ij}}{dx} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau \frac{\dot{e}}{k} dxdy \quad (7)$$

Top Left Corner: $i = 0, \quad j = n_y - 1$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dy}{k}(T_\infty - T_{ij}) + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dx\frac{T_{i.j-1} - T_{ij}}{dy} + 2\tau dy\frac{T_{i+1.j} - T_{ij}}{dx} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (8)$$

Top Right Corner: $i = n_x - 1, \quad j = n_y - 1$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dy}{k}(T_\infty - T_{ij}) + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dx\frac{T_{i.j-1} - T_{ij}}{dy} + 2\tau dy\frac{T_{i-1.j} - T_{ij}}{dx} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (9)$$

Left Boundary: $i = 0, \quad j \neq 0, n_y - 1$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dy\frac{T_{i+1} - T_{ij}}{dx} + \tau dx\frac{T_{i.j+1} - T_{ij}}{dy} + \tau dy\frac{T_{i.j-1} - T_{ij}}{dy} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (10)$$

Right Boundary: $i = n_x - 1, \quad j \neq 0, n_y - 1$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + 2\tau dy\frac{T_{i-1} - T_{ij}}{dx} + \tau dx\frac{T_{i.j+1} - T_{ij}}{dy} + \tau dy\frac{T_{i.j-1} - T_{ij}}{dy} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (11)$$

Bottom Boundary: $i \neq 0, n_x - 1, \quad j = 0$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + \tau dy\frac{T_{i-1} - T_{ij}}{dx} + \tau dy\frac{T_{i+1.j} - T_{ij}}{dx} + 2\tau dx\frac{T_{i.j+1} - T_{ij}}{dy} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (12)$$

Top Boundary: $i \neq 0, n_x - 1, \quad j = n_y - 1$

$$T_{ij}^{n+1} = T_{ij}^{n} + 2\tau\frac{h_s dx}{k}(T_\infty - T_{ij}) + \tau dy\frac{T_{i-1} - T_{ij}}{dx} + \tau dy\frac{T_{i+1.j} - T_{ij}}{dx} + 2\tau dx\frac{T_{i.j-1} - T_{ij}}{dy} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (13)$$

Volume

$$T_{ij}^{n+1} = T_{ij}^{n} + \tau dy\frac{T_{i-1} - T_{ij}}{dx} + \tau dy\frac{T_{i+1.j} - T_{ij}}{dx} + \tau dx\frac{T_{i.j-1} - T_{ij}}{dy} + \tau dx\frac{T_{i.j+1} - T_{ij}}{dy} + \frac{h_T}{k}\tau dydx(T_\infty - T_{ij}) + \tau\frac{\dot{e}}{k}dxdy \quad (14)$$