Project 1 - MANE 6710

Unique ID: 5284

Contents

List of Figures

List of Tables

A Introduction

The project in hand is to design a heat exchanger that can transfer heat from Medium 1 (water) which is at a temperature $T = 90°C$ to another Medium 2 (air) which is at a temperature $T = 20°C$. The main motivation behind this project is to find out which design of the heat exchanger is capable of maximizing the heat flux that is transferred from one medium to another. Some of the properties of a preliminary heat exchanger and its constraints are provided: the coefficient of thermal conductivity $K = 20\frac{W}{mK}$, Height range $h_{max} = 5cm$, $h_{min} = 1cm$ and the length of one unit of the heat exchanger $L = 5cm$.

B Choice of Design Variables

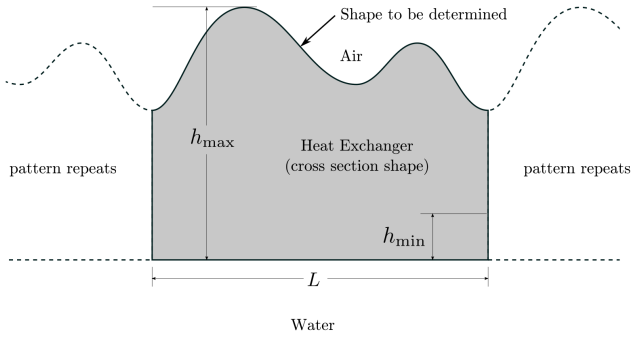


Figure 1: Illustration of Heat Exchanger Design Problem

This is a 2-D cross section of the heat exchanger to be designed and it can be assumed that it extrudes into or out of the paper as a 3-D object with a width. The profile of the heat exchanger can be assumed to be a function along the X-axis which is aligned with the length L of the heat exchanger: $h(x)$. It is mentioned in the problem statement that the this function $h(x)$ shall not be implicit. Hence we can choose to make this a function into something that can be parameterized about the values of x along the length of the heat exchanger. The parameters a_i are the design variables that can help parameterize $h(x; a)$ where $x \in \Omega$ and $a \in \mathbb{R}^n$ Three different types of parameterization to proceed forward have been chosen:

B.1 Linear Parameterization

$$h(x; a) = a_1 + \sum_{j=2}^{j=n} a_j \frac{x}{L} \tag{1}$$

This parameterization is going to generate an initial profile which is linear in nature. For $a \in \mathbb{R}^n$ a linear profile looks like the example in Figure 2a
$h(x(i)) = a_1 + \sum_{j=2}^{j=n} a_j \frac{x(i)}{L}$ as x is discretized along the length L gives the height of the profile at each $x(i)$.

B.2 Sinusoidal Parameterization

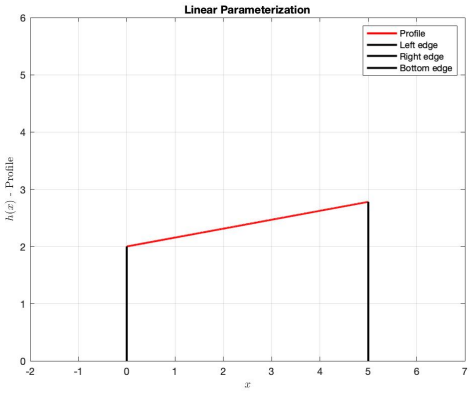
$$h(x; a) = a_1 + \sum_{j=2}^{j=n} a_j sin\left(\frac{2\pi(j-1)x}{L}\right) \tag{2}$$

If this parameterization is used, a sinusoidal profile is formed and an example profile for a design variable of dimension $a \in \mathbb{R}^{21}$ can be seen in Figure 2b. Similar to the linear parameterization case, x is discretized along the length L and the above function provides the height of the profile at each $x(i)$.
This is linear with respect to the design variable a and this profile is the linear combination of $c_i a_i$ where c_i are sine waves. Or it can be thought of as superimposition of sine waves.

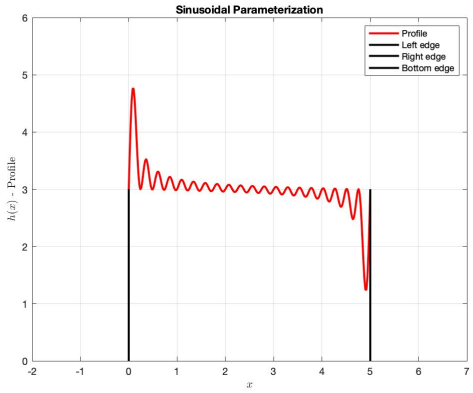
B.3 Triangular Parameterization

$$h(x;a) = a_1 - \frac{a_2}{\pi}\sum_{j=1}^{j=\infty}(-1)^j \frac{sin(\frac{2\pi a_3 jx}{L})}{j} \tag{3}$$

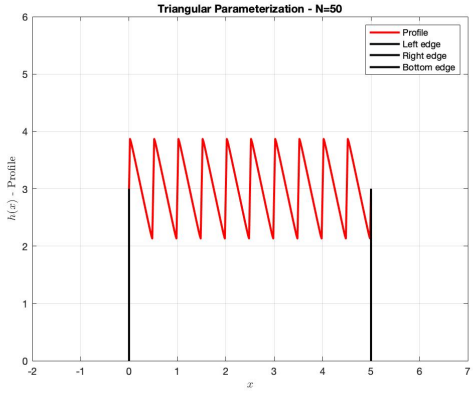
A sawtooth profile that has an offset of a_1 is formed in this case. This parameterization requires the design variable $a \in \mathbb{R}^3$. Since it is not possible to numerically run the summation $j = 1(1)n; n \to \infty$ is not possible, an appropriately large value of n can be chosen. The value chosen for this project is $n = 50$. a_2 is the height of the sawtooth and a_3 is the frequency of sawtooth profile. An example profile is displayed in Figure 2c
One specific detail to note in the case of using a triangular parameterization is that the relationship between $h(x;a)$ and design variable a is nonlinear.



(a) Linear Parameterization



(b) Sinusoidal Parameterization



(c) Triangular Parameterization

Figure 2: Profile based on different Parameterization

C Objective Function and Constraints

The objective function is the calculation of heat flux.

$$q = \int -(\kappa \nabla T) dA \tag{4}$$

This is calculated by the function ***CalcFlux.m*** . This function takes in as inputs : Profile of the heat exchanger $h(x;a)$, $\kappa, L, N_x, N_y, T_{water}, T_{air}$. This function computes the steady state heat transfer using a Finite Volume approach. The geometry of this heat exchanger is made into a finite volume mesh using the function ***BuildMesh.m***. ***BuildMesh.m*** requires the inputs $L, h(x;a), N_x, N_y$ and is called inside ***CalcFlux.m***. Steady state heat transfer is then solved using another function ***BuildSystem.m*** which is called within ***CalcFlux.m*** after building a mesh. ***BuildSystem.m*** requires the inputs $L, h, N_x, N_y, \kappa, T_{water}, T_{air}$. The solver returns a Temperature field over the finite volume elements and this Temperature field can be used to calculate the heat flux using Equation.4

D Setting up the Optimization Problem

The optimization problem is now set up as follows:

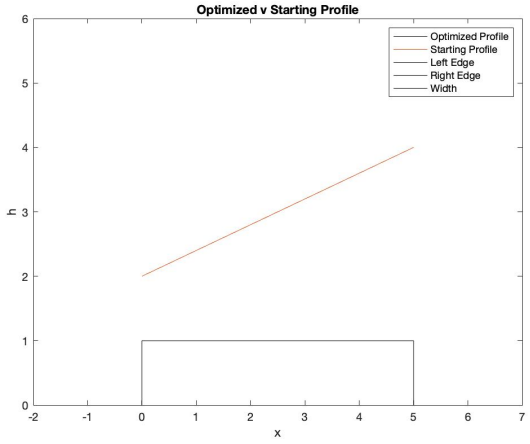
$$\begin{aligned} &\max_{a \in R^n} \; q(a) \\ s.t \;\; &h_{min} \leq h(x;a) \leq \; h_{max} \end{aligned} \tag{5}$$

It is important to note that both linear and sinusoidal parameterization produces $h(x;a)$ where a an $x(i)$ are linearly related each other and hence, the constraint equation can be written as $Ca \leq b$ where C matrix holds the coefficients that when multiplied with the vector(design variable) a produces the constraint thats always $\leq b$. However this is not the case with Triangular parameterization and a function should be specifically written in order to calculate this nonlinear inequality between $h_{min} \leq h(x;a) \leq h_{max}$.
The optimizer that is used in this project is an inbuilt optimization algorithm from $MATLAB$ called $fmincon(...)$. This optimizer has the function arguments: $obj, a_0, C, b, C_{eq}, b_{eq}, Nonlinear\ constraints,\ Options.$ obj is the objective function and a_0 is the initial starting point
For the linear and sinusoidal cases, the function call to the optimizer follows $fmincon(obj, a_0, C, b, [.], [.], [.],\ Options)$ because there are no equality or nonlinear constraints involved. When triangular parameterization is used, the function call is of the form: $fmincon(obj, a_0, [.], [.], [.], [.], nonlinear\ constraint,\ Options)$. ([.] means an empty matrix)
The code for generation of the profile $h(x;a)$, generating the objective function $q = obj(a)$ and setting up the constraints matrix C, b can be found in the Appendix.

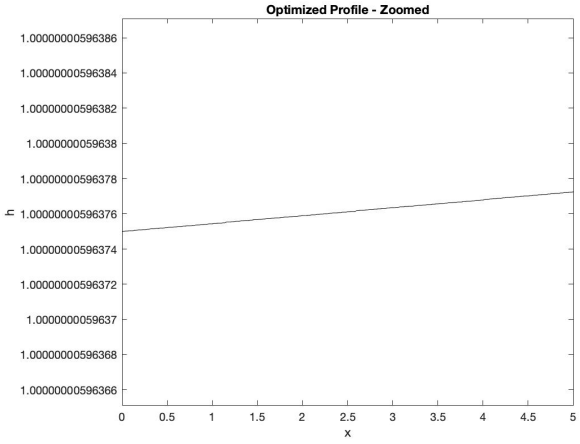
E Comparison of Preliminary Results

Preliminary analysis has been done on a mesh with spacing $N_x = N_y = 100$.

1. Linear Parameterization
   - Initial Condition : $a_0(1) = 2; a_0(2:21) = 0.1; \quad a \in \mathbb{R}^{21}$
   - Optimal Design Variable : Found in Appendix
   - Final Top Profile:



(a) Starting v Final - Linear



(b) Final Zoomed - Linear

Figure 3: Optimized Profile using Linear Parameterization

- Flux through Optimized profile: $6953.64 \frac{W}{m^2}$
- Optimizer convergence:
  (a) Total number of iterations: 13
  (b) The total number of function calls: 287
  (c) Algorithm used: $Interior - Point$

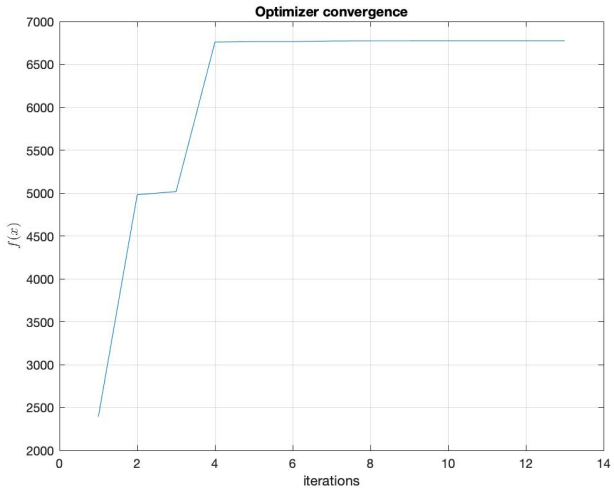
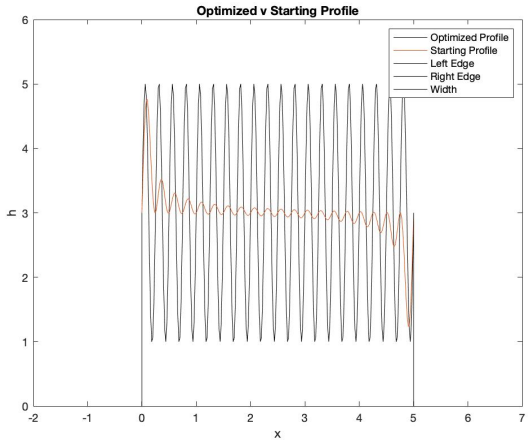


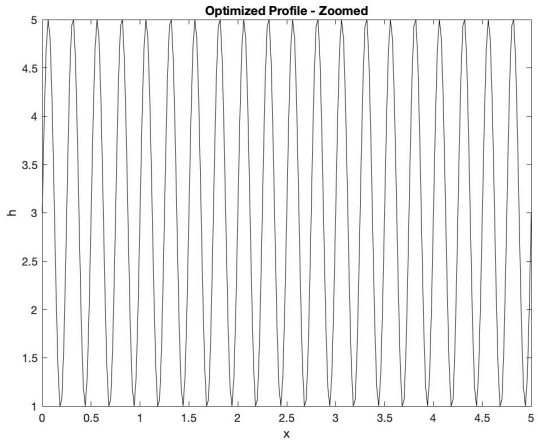
Figure 4: Flux v Iterations - Linear

2. Sinusoidal Parameterization

- Initial Condition : $a_0(1) = 3; a_0(2:21) = 0.12; \quad a \in \mathbb{R}^{21}$
- Optimal Design Variable : Found in Appendix
- Final Top Profile:



(a) Starting v Final - Sinusoidal



(b) Final Zoomed - Sinusoidal

Figure 5: Optimized Profile using Sinusoidal Parameterization

- Flux through Optimized profile: $58064.3129 \frac{W}{m^2}$
- Optimizer convergence:
  (a) Total number of iterations: 10
  (b) The total number of function calls: 242
  (c) Algorithm used: $Interior - Point$

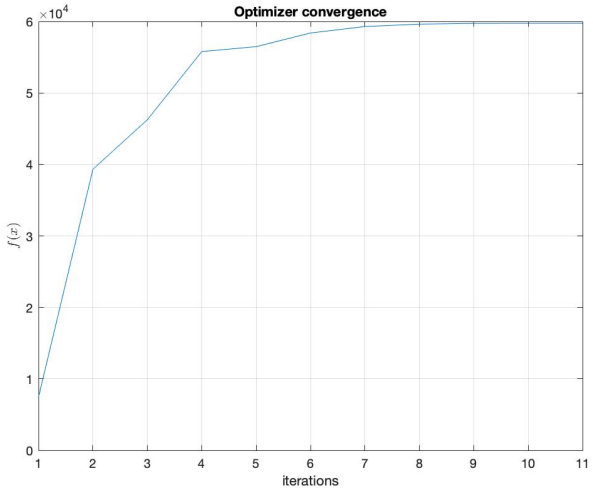
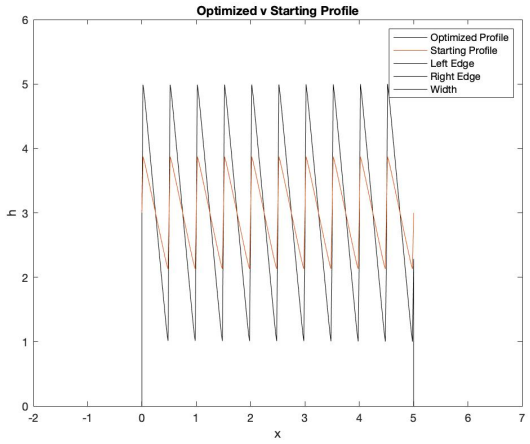


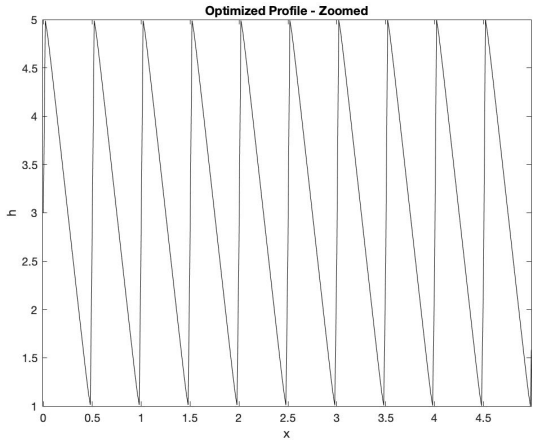
Figure 6: Flux v Iteration - Sinusoidal

3. Triangular Parameterization

- Initial Condition : $a_0 = [3; 2; 10] \quad a \in \mathbb{R}^3$
- Optimal Design Variable : Found in Appendix
- Final Top Profile:



(a) Starting v Final - Triangular



(b) Final Zoomed - Triangular

Figure 7: Optimized Profile using Triangular Parameterization

- Flux through Optimized profile: $40849.37706 \frac{W}{m^2}$
- Optimizer convergence:
  (a) Total number of iterations: 33
  (b) The total number of function calls: 138
  (c) Algorithm used: $Interior - Point$

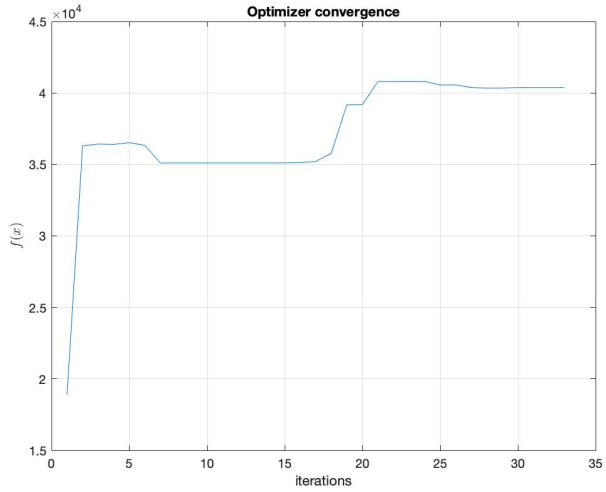
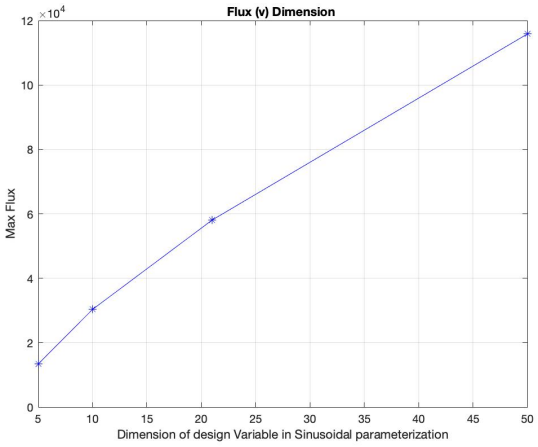
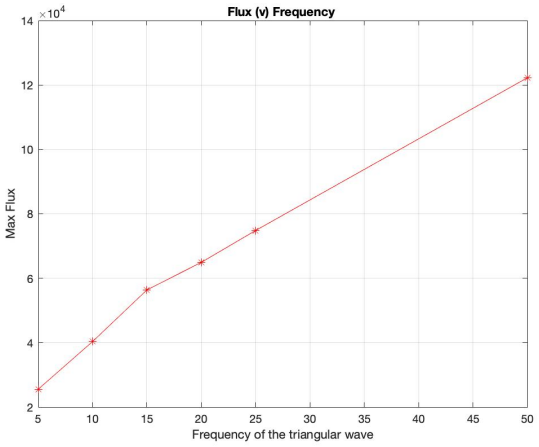


Figure 8: Flux v Iteration - Triangular

Between the three methods and starting points, it is clear that the linear parameterization does not lead to the maximum flux that can possibly be obtained from the conditions and criteria specified because it leads to a profile which is a flat rectangle of dimensions $1cm$ x $5cm$. The maximum flux that can come out of this profile can be analytically calculated and that is $7000\frac{W}{m^2}$. The optimizer led to achieving a profile that can lead to a flux value of $6953.64\frac{W}{m^2}$.
Both the sinusoidal and triangular parameterization leads to a significantly large flux compared to the result a linear parameterization produced (approximately 700% and 500% respectively). It is also observed that as the number of dimensions are increased for the sinusoidal parameterization, the heat flux value keeps increasing. The same phenomenon is observed if the frequency parameter of the triangular parameterization is increased and that leads to more saw tooth profiles. $N_x = N_y = 100$ is used for this study and Figure.9 shows this phenomena.



(a) Flux v Dimension of a - Sinusoidal parameterization



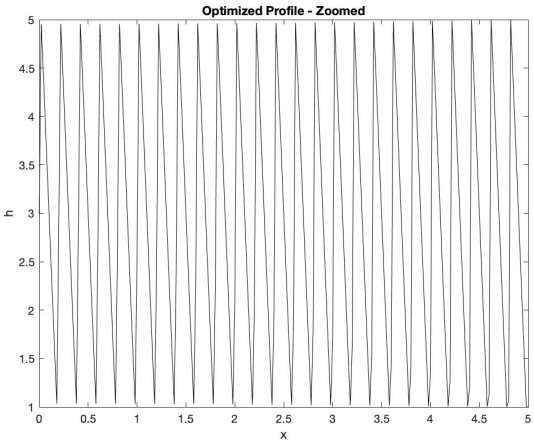
(b) Flux v ν of a - Triangular parameterization

Figure 9: Increase of flux with certain aspects of design variable a

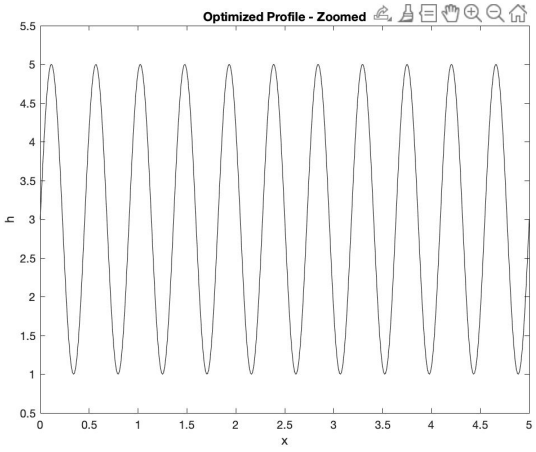
F Convergence Study

The choice of $N_x = N_y = 100$ was entirely arbitrary and that leads to a decent approximation of what the heat flux would be for the optimized profile. But in order to find out what the closest possible to exact heat flux is through the optimized profile, refinement of mesh to have smaller finite volumes is necessary. Hence a convergence study will help figure out the relationship between a chosen optimized profile and the relationship with the size of the finite volumes.
Fig. 10a had an optimized design with a frequency of $\nu = 25$ using triangular parameterization. Fig.10b has a starting point $a_0 \in \mathbb{R}^{12}$ and lastly, the third profile was of sinusoidal parameterization in Fig.10c but with a starting point $a_0(1) = 3; a_0(2:10) = 0$ and $a \in \mathbb{R}^{10}$. The intention for this design is two fold. One is to perform the convergence

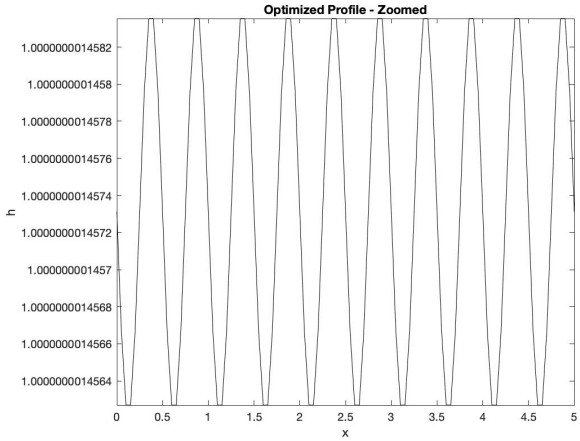
analysis and the other is to show how the choice of the starting point gives an optimal point belonging to different local minima.
Fig.10d shows how with finer and finer meshes, the results for Heat Flux converged to the closest approximation of the actual analytical solution. The problem that comes with finer meshes is the computational cost that comes with it. The same could be said about the increase in dimensionality in the case of a sinusoidal parameterization of the top profile. The biggest advantage of using a triangular parameterization was the fact that even with just 3 dimensions, it enables the optimizer to converge to results that can possibly produce high heat flux values.



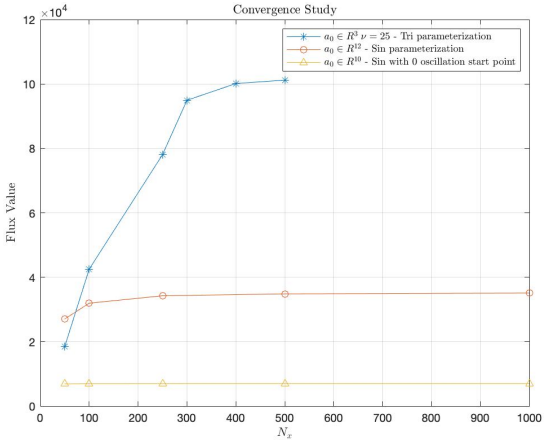
(a) Optimized Profile with Δ Parameterization $\nu = 25$



(b) Optimized Profile with Sin Parameterization $a \in \mathbb{R}^{12}$



(c) Optimized Profile for a flat sin parameterization $a \in \mathbb{R}^{10}$



(d) Convergence Analysis

Figure 10: Convergence of Final Flux value with increasing mesh refinement

G Results and Inferences

1. This project sheds light on the well known fact that in order to maximize heat transfer or heat flux from one medium to another, the most optimal design is to have structures called **Fins**. These are structures with very less thickness in order to increase the surface area of heat transfer. Both sinusoidal and triangular parameterization leads to creation of a **fin-like profile**.

2. From a design stand point, it can be shown that different starting points or initial points lead to different optimal designs that satisfy the constraints or bounds.

3. Another side note is that the number of iterations and function calls increase heavily if the initial point is out of the tolerable bounds. For example, an initial point with $a_0(1) = 3; a_0(2:21) = 0.5; a_0 \in \mathbb{R}^{21}$ gives a profile that is out of bounds as shown in Fig.11. The number of function calls were 1344 and the maximum number of iterations was restricted to 60. The Final Flux attained was $53111.309266 \frac{W}{m^2}$.Comparing this with the solution for a problem with the same dimensionality like in Fig.3a, the clear increase in number of function calls and iterations can be seen.

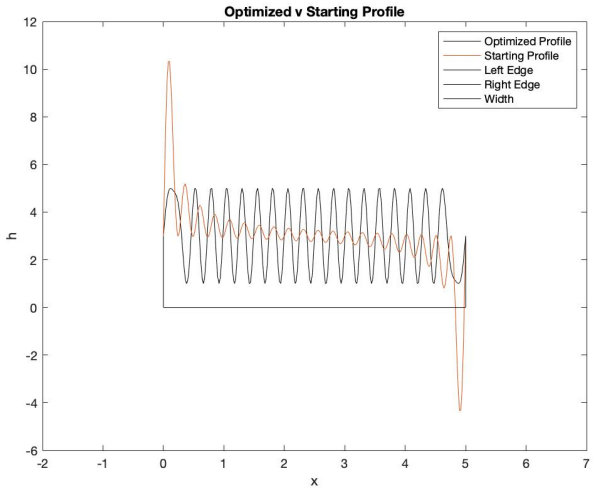


Figure 11: Optimal Profile if starting with an infeasible point

4. Fig. 9 is another clear indication that with more number of fins, the maximum heat flux will increase. That was evident with the rise in heat flux values with the increase in the number of dimensions of the design variable in the sinusoidal case and with the increase of frequency parameter in the triangular parameterization case.

5. From this project, the final flux values being reported are the following:

   - Linear Parameterization of $a \in \mathbb{R}^{21}$ yields - $6993.64 \frac{W}{m^2}$ with $N_x = N_y = 1000$
   - Sinusoidal Parameterization of $a \in \mathbb{R}^{21}$ yields - $59741.29 \frac{W}{m^2}$ with $N_x = N_y = 500$
   - Triangular Parameterization of $a \in \mathbb{R}^3$ and with $\nu_0 = 25$ yields - $101299.8736 \frac{W}{m^2}$ with $N_x = N_y = 500$

H Appendix

.1 Optimal points from Optimizer:

X`linear =
[1.00000000596375; −0.00211736922847506; −0.00211736921230914; −0.00214675654158376; ...
− 0.00214675654158372; −0.00103336585774427; 0.00372561891272210; −0.00104535054108106; ...
0.00250995878394754; −0.00191602678535082; 0.00269913312708719; −0.00116146051702897; ...
− 0.00373671543331022; −0.00370640515815103; −0.00174780407094227; 0.00508637365704515; ...
0.00474105248772209; 0.00172282541372274; −0.00305458551853304; 0.00329743748706565; ...
0.00214756553680345]
X`sinusoidal =
[2.99999996885313; $-9.41854650580605e-10$; $1.01625785627599e-10$; $1.72461166932447e-09$; ...
$2.13650671495380e-09$; $1.38715769824086e-09$; $-2.92563268723410e-10$; $-2.35288492567810e-09$; ...
$-3.03543755880989e-09$; $-9.36670529000131e-10$; −0.0218389278795468; $3.79154438677764e-10$; ...
$1.13461569109465e-09$; $-8.39196431264226e-10$; $-1.20349378889991e-09$; $-3.81241874638362e-11$; ...
$1.94669430010950e-09$; $3.97997270171530e-09$; $4.32586722307759e-09$; $-1.84413351597820e-09$; ...
2.01893364029489]
X`triangular = [3.00062733283020; 4.56749698436269; 9.99841012519896]

.2 Optimizer convergence - Flux value V Iterations

Linear Parameterization Fig.4 Flux value v Iterations:

Iteration	Flux Value
1	2.392837e+03
2	4.980509e+03
3	5.016815e+03
4	6.759283e+03
5	6.764436e+03
6	6.764331e+03
7	6.772434e+03
8	6.774177e+03
9	6.774173e+03
10	6.774193e+03
11	6.774193e+03
12	6.774193e+03
13	6.774194e+03

Table 1: Flux v Iterations - Linear parameterization

Sinusoidal Parameterization Fig.6 Flux value v Iterations:

Iteration	Flux Value
1	7.436319e+03
2	3.932569e+04
3	4.624590e+04
4	5.578685e+04
5	5.647216e+04
6	5.838431e+04
7	5.928845e+04
8	5.962379e+04
9	5.972434e+04
10	5.974116e+04
11	5.974129e+04

Table 2: Flux v Iterations - Sinusoidal parameterization

Triangular Parameterization Fig.8 Flux value v Iterations:

Iteration	Flux Value
1	1.889699e+04
2	3.629552e+04
3	3.642467e+04
4	3.639738e+04
5	3.652243e+04
6	3.633551e+04
7	3.510292e+04
8	3.510715e+04
9	3.510792e+04
10	3.510791e+04
11	3.510792e+04
12	3.510793e+04
13	3.510800e+04
14	3.510833e+04
15	3.511008e+04
16	3.512022e+04
17	3.520962e+04
18	3.577353e+04
19	3.917566e+04
20	3.918646e+04
21	4.080098e+04
22	4.080920e+04
23	4.080888e+04
24	4.080932e+04
25	4.056148e+04
26	4.056181e+04
27	4.037784e+04
28	4.033888e+04
29	4.034618e+04
30	4.037414e+04
31	4.037414e+04
32	4.037414e+04
33	4.037414e+04

Table 3: Flux v Iterations - Triangular parameterization

.3 Convergence Analysis

Triangular Parameterization Optimized profile with frequency 25:

N_x	Flux Value
50	$1.844683713076328e+04$
100	$4.248849784574824e+04$
250	$7.805146638013054e+04$
300	$9.494107046818522e+04$
400	$1.001292884539378e+05$
500	$1.012998736898211e+05$

Table 4: N_x v Flux - Case 1

Sinusoidal parameterization with $a \in \mathbb{R}^{12}$:

N_x	Flux Value
50	$2.702959713192348e+04$
100	$3.194138632807931e+04$
250	$3.424638727249664e+04$
500	$3.483118953557032e+04$
1000	$3.508359374896967e+04$

Table 5: N_x v Flux - Case 2

Sinusoidal Parameterization with $a_0 \in \mathbb{R}^{12}$; $a_0(1) = 3; a_0(2:end) = 0;$

N_x	Flux Value
50	$6.865384605379591e+03$
100	$6.931372538918941e+03$
250	$6.972222212062424e+03$
500	$6.986055766722282e+03$
1000	$6.993013961830300e+03$

Table 6: N_x v Flux - Case 3

.4 MATLAB Code

The $MATLAB$ code that was created to perform this analysis has been attached as a part of the appendix for all three methodologies.

Contents


```
clc
clear all
```

Setting up Initial Starting point

a0 is starting point of DESIGN VARIABLE uses linear parameterization

```
%a0 = [1.5 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1]';% a0 - inital starting point a - DESIGN VARIABLE
%a0 = [3 0 0 0 0 0 0 0 0 0 0]';
a0 = 0.1*ones(21,1); a0(1) = 2;
n = length(a0); % length of Design var
x = (0:0.05:5)'; % descretization of x-axis
```

Setting up Constraints Matrix

Sets up the Constraints Matrix A which holds the coefficients that are linearly added up with design variables to generate the profile on the top

```
%--------------------------------------------------------------------------

hmin = 1; % min height of radiator in cm
hmax = 5; % max height of radiator in cm
L = 5; % Length of Radiator in cm
for i=1:length(x)
    for j=1:length(a0)
        if(j==1)
            A1(i,j) = 1;
        else
            A1(i,j) = x(i)/L;
        end
    end
end

A = [A1;-A1];
b = [ones(length(x),1)*hmax; -ones(length(x),1)*hmin];
```

optimization algorithm - fmincon used

```
options = optimoptions(@fmincon,'Display','iter');
[X,fvalue,exitflag,output] = fmincon(@CalcFlux_obj,a0,A,b,[],[],[],[],[],options);
```

```
[flux_final,T_final,dTdX,XY] = plot_profile(X,a0);
```

Setting up Objective function

It calls the functions - Calc_h to generate the profile which is a funciton of the design variable 'a' It calls upon the function CalcFlux.m to calculate the flux CalcFlux_obj returns the -Flux calculated by CalcFlux.m Objective function only requires input - the design variable 'a'

```
%-------------------------------------------------------------------------

function Flux = CalcFlux_obj(a)
    L = 5;
    Kappa = 20; %
    T_top = 20; % deg cel
    T_btm = 90; % deg cel
    x = (0:0.05:5)';
    % Calculate h
    h = Calc_h(x,a,L);

    % Set Nx and Ny
    nx = length(h)-1;
    ny = 30;

    % Calculate Flux
    [Flux,~,~,~] = CalcFlux(L,h,nx,ny,Kappa,T_top,T_btm);

    % Negate Flux for maximization problem
    Flux = -Flux;
end
```

plot final profile

This function generates a plot comparing the Optimized profile with the initial profile and also another plot containing the zoomed optimal profile This function takes inputs: X- Optimized design, a0 - initial design.

```
%-------------------------------------------------------------------------

function [flux_final,T_final,dTdX,XY] =  plot_profile(X,a0)
x = (0:0.02:5)';
L = 5;
T_top = 20;
T_btm = 90;
kappa = 20;
h = Calc_h(x,X,L);
nx = length(h)-1;
ny = 150;
[flux_final,T_final,dTdX,XY] = CalcFlux(L,h,nx,ny,kappa,T_top,T_btm);
h0 = Calc_h(x,a0,L);
figure(1);
plot(x,h,'k');
hold on;
plot(x,h0);
plot([0,0],[0,h(1)],'k');
plot([5,5],[0,h(end)],'k');
plot([0,0],[0,0],'k');
legend('Optimized Profile','Starting Profile','Left Edge',...
    'Right Edge','Width');
axis([-2 7 0 6]);
title('Optimized v Starting Profile');
%text(1,5.5,'HeatFlux = 7396.6 W/m')
xlabel('x');
ylabel('h');
```

```
figure(2)
plot(x,h,'k');
title('Optimized Profile - Zoomed');
xlabel('x');
ylabel('h');
end
```

Calculate h - Profile height

Function returns the profile height h - h(x;a) Function takes inputs - x: discretization about X-axis, a: the design variable, L: Length of the heat exchanger

```
%-------------------------------------------------------------------------

function h = Calc_h(x,a,L)
% Uses linear parameterization to create profile
% h(x) = a(1) + sigma (a(j)*(x/L)) j:1(1)N
for i=1:length(x)
    S = 0;
    for j=2:length(a)
       S = S + a(j)*x(i)/L;
    end
    h(i,1) = a(1) + S;
end
end
```

Published with MATLAB® R2021a

Contents


```
clc
clear all
```

Setting up Initial Starting point

a0 is starting point of DESIGN VARIABLE uses sinusoidal parameterization

```
%a0 = [3 0.1 0.5 0.3 0.8 0.2 0.9 1 1 0.1 0.1 0.1]';
%a0 = [3 0 0 0 0 0 0 0 0 0 0]';
%a0 = [3 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1]';
a0 = 0.12*ones(21,1); a0(1) = 3;
%a0 = ones(101,1); a0(1) = 3;a0(2:end) = rand(100,1);
%a0 = ones(101,1); a0(1) = 3;a0(2:end) = -1+2*rand(100,1);
%a0 = ones(31,1); a0(1) = 3;a0(2:end) = rand(30,1);
%a0 = 0.5*ones(21,1); a0(1) = 3;
%a0 = 0.11*ones(50,1); a0(1) = 3;
```

Setting up Constraints Matrix

Sets up the Constraints Matrix A which holds the coefficients that are linearly added up with design variables to generate the profile on the top

```
%-------------------------------------------------------------------------

hmin = 1; % min height of radiator in cm
hmax = 5; % max height of radiator in cm
L = 5; % Length of Radiator in cm
n = length(a0); % length of Design var
x = (0:0.02:5)'; % descretization of x-axis

for i=1:length(x)
    for j=1:length(a0)
        if(j==1)
            A1(i,j) = 1;
        else
            A1(i,j) = sin(2*pi*(j-1)*x(i)/L);
        end
    end
end

A = [A1;-A1];
b = [ones(length(x),1)*hmax; -ones(length(x),1)*hmin];
```

optimization algorithm - fmincon used

```
options = optimoptions(@fmincon,'MaxIter',60,'Display','iter');
[X,fvalue,exitflag,output] = fmincon(@CalcFlux_obj,a0,A,b,[],[],...
    [],[],[],options);
```

Plotting

```
[flux_final,T_final,dTdX,XY] = plot_profile(X,a0);
```

Setting up Objective function

It calls the functions - Calc_h to generate the profile which is a funciton of the design variable 'a' It calls upon the function CalcFlux.m to calculate the flux CalcFlux_obj returns the -Flux calculated by CalcFlux.m Objective function only requires input - the design variable 'a'

```
%-------------------------------------------------------------------------

function Flux = CalcFlux_obj(a)
    L = 5; % cm
    Kappa = 20; % W/(m.K)
    T_top = 20; % deg cel
    T_btm = 90; % deg cel
    x = (0:0.02:5)';

    % calculate h
    h = Calc_h(x,a,L);

    % set Nx and Ny
    nx = length(h)-1;
    ny = 250;

    % Calculate Flux
    [Flux,~,~,~] = CalcFlux(L,h,nx,ny,Kappa,T_top,T_btm);

    % Negate Flux for maximization problem
    Flux = -1*Flux;
end
```

plot final profile

This function generates a plot comparing the Optimized profile with the initial profile and also another plot containing the zoomed optimal profile This function takes inputs: X- Optimized design, a0 - initial design.

```
%-------------------------------------------------------------------------

function [flux_final,T_final,dTdX,XY] =  plot_profile(X,a0)
x = (0:0.02:5)';
L = 5;
T_top = 20;
T_btm = 90;
kappa = 20;
h = Calc_h(x,X,L);
nx = length(h)-1;
ny = 150;
[flux_final,T_final,dTdX,XY] = CalcFlux(L,h,nx,ny,kappa,T_top,T_btm);
```

```
h0 = Calc_h(x,a0,L);
figure(1);
plot(x,h,'k');
hold on;
plot(x,h0);
plot([0,0],[0,h(1)],'k');
plot([5,5],[0,h(end)],'k');
plot([0,5],[0,0],'k');
legend('Optimized Profile','Starting Profile','Left Edge',...
    'Right Edge','Width');
axis([-2 7 -6 12]);
title('Optimized v Starting Profile');
xlabel('x');
ylabel('h');

figure(2)
plot(x,h,'k');
title('Optimized Profile - Zoomed');
xlabel('x');
ylabel('h');

end
```

Calculate h - Profile height

Function returns the profile height h - h(x;a) Function takes inputs - x: discretization about X-axis, a: the design variable, L: Length of the heat exchanger

```
%-------------------------------------------------------------------------

function h = Calc_h(x,a,L)
% Uses sinusoidal parameterization to create profile
% h(x) = a(1) + sigma (a(j)*sin(2*pi*(j-1)*x(i)/L)) j:1(1)N
for i=1:length(x)
    S = 0;
    for j=2:length(a)
        S = S + a(j)*sin(2*pi*(j-1)*x(i)/L);
    end
    h(i,1) = a(1) + S;
end
end
```

Published with MATLAB® R2021a

Contents


```
clc
clear all
```

Setting up Initial Starting point

using a triangluar parameterization

```
%-------------------------------------------------------------------------
a0 = [3 4 10]'; %SP1
%a0 = [3 2 16]'; %SP2
hmin = 1; % min height of radiator in cm
hmax = 5; % max height of radiator in cm
L = 5; % Length of Radiator in cm
n = length(a0); % length of Design var
x = (0:0.02:5)'; % descretization of x-axis
```

optimization algorithm used - fmincon

```
A = [];
b = [];
Aeq = [];
beq = [];
lb = [];
ub = [];
nonlcon = @nonlincon; % function call to calculate nonlinear constraints
options = optimoptions(@fmincon,'Display','iter');
[X,fvalue,exitflag,output] = fmincon(@CalcFlux_obj,a0,A,b,Aeq,beq,lb,ub,...
    nonlcon,options);
```

Plotting

```
[flux_final,T_final,dTdX,XY] = plot_profile(X,a0);
```

Setting up Objective function

It calls the functions - Calc_h to generate the profile which is a funciton of the design variable 'a' It calls upon the function CalcFlux.m to calculate the flux CalcFlux_obj returns the -Flux calculated by CalcFlux.m Objective function only requires input - the design variable 'a'

```
%-------------------------------------------------------------------------

function Flux = CalcFlux_obj(a)
    L = 5; % cm
    Kappa = 20; % W/(m.K)
    T_top = 20; % deg cel
    T_btm = 90; % deg cel
    x = (0:0.02:5)'; % cm

    % calculate h
    h = Calc_h(x,a,L);

    % set Nx and Ny
    nx = length(h)-1;
    ny = 150;

    % Calculate Flux
    [Flux,~,~,~] = CalcFlux(L,h,nx,ny,Kappa,T_top,T_btm);

    % Negate Flux for maximization problem
    Flux = -1*Flux;
end
```

Setting up nonlinear Constraints

This function is set up to calculate the nonlinear constraints c(a)<= s; Here c(a) - vector of functions taking the same input 'a' and 's' is the containing the constraint values Input to function : a - Design Variable

```
%-------------------------------------------------------------------------

function [c,ceq] = nonlincon(a)
x = (0:0.02:5)';
L = 5; %cm
hmin = 1; % cm
hmax = 5; % cm
for i=1:length(x)
    S = 0;
    for j=1:50
        S = S + (a(2)/pi)*(-1^j)*sin(2*pi*a(3)*j*x(i)/L)/j;
    end
    c1(i,1) = a(1) - S - hmax;
end
for i=1:length(x)
    S = 0;
    for j=1:50
        S = S + (a(2)/pi)*(-1^j)*sin(2*pi*a(3)*j*x(i)/L)/j;
    end
    c2(i,1) = hmin-(a(1) - S);
end
ceq = [];
c = [c1;c2];
end
```

plot final profile

This function generates a plot comparing the Optimized profile with the initial profile and also another plot containing the zoomed optimal profile This function takes inputs: X- Optimized design, a0 - initial design.

```
%-------------------------------------------------------------------------

function [flux_final,T_final,dTdX,XY] =  plot_profile(X,a0)
x = (0:0.02:5)';
L = 5;
T_top = 20;
T_btm = 90;
kappa = 20;
h = Calc_h(x,X,L);
nx = length(h)-1;
ny = 150;
[flux_final,T_final,dTdX,XY] = CalcFlux(L,h,nx,ny,kappa,T_top,T_btm);
h0 = Calc_h(x,a0,L);
figure(1);
plot(x,h,'k');
hold on;
plot(x,h0);
plot([0,0],[0,h(1)],'k');
plot([5,5],[0,h(end)],'k');
plot([0,0],[0,0],'k');
legend('Optimized Profile','Starting Profile','Left Edge',...
    'Right Edge','Width');
axis([-2 7 0 6]);
title('Optimized v Starting Profile');
xlabel('x');
ylabel('h');

figure(2)
plot(x,h,'k');
title('Optimized Profile - Zoomed');
xlabel('x');
ylabel('h');
end
```

Calculate h - Profile height

Function returns the profile height h - h(x;a) Function takes inputs - x: discretization about X-axis, a: the design variable, L: Length of the heat exchanger

```
%-------------------------------------------------------------------------

function h = Calc_h(x,a,L)
% Uses triangular parameterization to create the profile
% h(x) = a1 + sigma ((a(2)/pi)*(-1^j)*sin(2*pi*a(3)*j*x(i)/L)/j) j:1(1)N

for i=1:length(x)
    S = 0;
    for j=1:50
        S = S + (a(2)/pi)*(-1^j)*sin(2*pi*a(3)*j*x(i)/L)/j;
    end
    h(i,1) = a(1) - S;
end
end
```

Published with MATLAB® R2021a