EG2501 MATLAB Pressure Vessel Optimization Report

Contents

1. Optimization Problem Formulation

Design variables:
As per definition, design variables are required to define the product and are decided by the designer. With the given problem of optimizing a pressure tank, we as a designer are allowed to change the material, the thickness (t) and the internal radius (ri) which defines both the height and the length.

Design variables: Material, t, ri

Design qualities:
The measurable features of this product are the cost (CI), height (H), total length (Lt), von mises stress (σ') and the volume of the material (V_{mat}), as we can choose any of these features and optimize based on them.

Design qualities: CI, Lt, H, σ', V_{mat}

Optimization objective
In this design problem, we aim to minimize the cost (CI) of the pressure vessel while all the given constraints are true.

Optimization objective: minimize CI

Design parameters
The parameters that are required to define the product but that we as a designer cannot change, ie. the values given to the program by the user. The user can specify the internal pressure (P), capacity (C), the factor of safety (nd) and the step size of the internal radius (ri_step).

Design parameters: P, C, nd, ri_step

Constraints
The given constraints for the problem are maximum total Length (Ltc), maximum height (Hc), von Mises stress (σ') has to be less than the allowable stress (σall) and internal volume (Vi) has to be equal to desired capacity (C).

- Lt <= Ltc
- H <= Hc
- σ' < σall
- Vi = C

The equality constraint will allow us to create a simpler optimization program, as we can obtain the design variable Lt simply by rearranging this equation. We have the equality constraint Vi = C, but we have to remember that Vi is in m3 and C is in litre, since 1m3 is equivalent to 1000L we are going to divide C by 1000, so Vi = C/1000.

$$V_i = \pi {r_i}^2 \, \mathrm{L} + \frac{4\pi {r_i}^3}{3}$$

$$V_i - \frac{4\pi {r_i}^3}{3} = \pi {r_i}^2 \, \mathrm{L}$$

$$L = \frac{\left(V_i - \frac{4\pi {r_i}^3}{3}\right)}{\pi {r_i}^2}$$

We don't know the value of Vi, but we know that Vi = C/1000 and we know that C is a design parameter, i.e given, so the equation for L becomes:

$$L = \frac{\left(\frac{C}{1000} - \frac{4\pi {r_i}^3}{3}\right)}{\pi {r_i}^2}$$

Standard form

Cost index: CI = $Vmat * CF$

$V_{mat} = 2\pi r_i tL + 4\pi {r_i}^2 t$

CF: cost factor, given for each material

min CI

subject to:

$$L = \frac{\left(\frac{C}{1000} - \frac{4\pi {r_i}^3}{3}\right)}{\pi {r_i}^2}$$

$Lt - Ltc \leq 0$

$H - Hc \leq 0$

$\sigma' - \sigma all < 0$

Material ∈ {1, 2, 3, 4, 5, 6, 7, 8, 9 ,10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23}

t ∈ t_{Mi} {where t_{Mi} ∈ the set of available thicknesses (t) for the material M_i ∈ Materials}

Standard form to MATLAB

To convert this design formulation into a working MATLAB program we must consider what design variable is needed to calculate each value. From the standard from formulation above, it is easy to see that every available thickness depends on the material, as different materials have different thicknesses available. Therefore, we are going to have a for loop going through each material, a nested for loop going through each thickness available for the material, and another nested loop going through every possible internal radius:

```
For material in materials
        For thickness(t) in material
                For ri from min to max with ri_step
                        Do calculations and add to result
                End
        End
End
```

2. Example output

My number = 132/10 = 13.2

Problem 1:

P = 80/13.2 bar
Vi = 140*13.2 Litres (L)
nd = 2
Lt ≤ 10m
H ≤ 3m

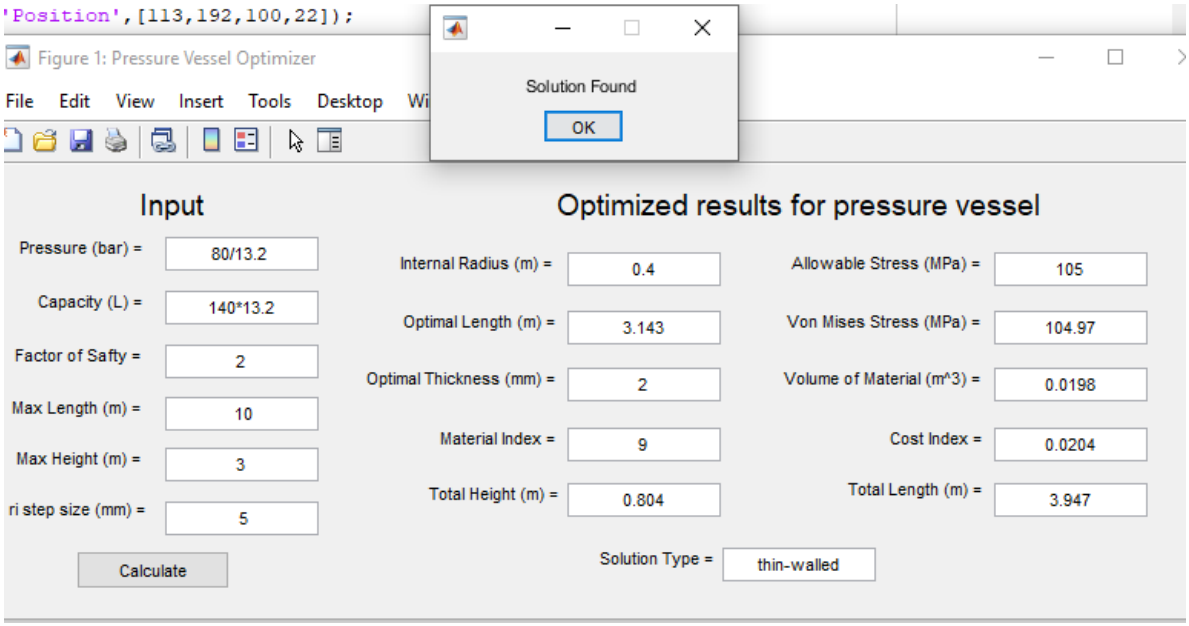

Figure 1: GUI for Problem 1 with a pop-up message informing the user of calculation status

Problem 2:

P = 8*13.2 bar
Vi = 800/13.2 Litres (L)
nd = 3
Lt ≤ 7m
H ≤ 2m

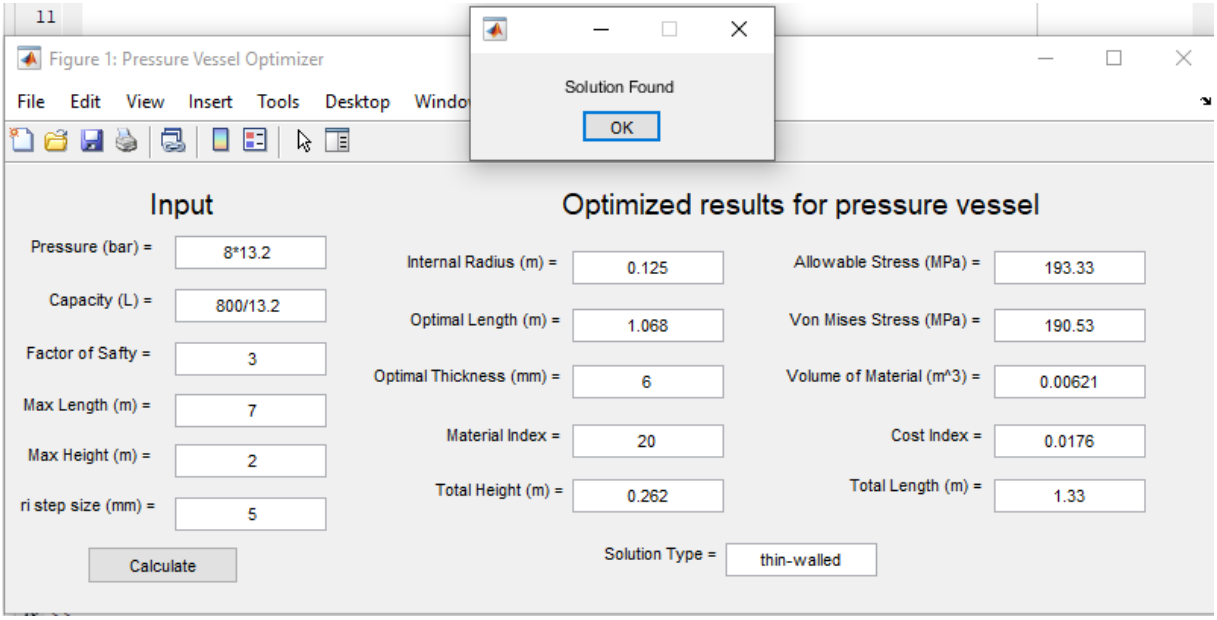

Figure 2: GUI for Problem 2 with a pop-up message informing the user of calculation status

3. MATLAB Code

3.1 Reading excel file

Before we can perform the optimization calculations, we need to be able to access the data in the excel file. We could access the data in the main program, however, this would add unnecessary extra time to the program, as we can load the data once and then use it repeatedly. This can be done by reading the excel file and then saving the data in a .mat file which is faster to access. The code for this only needs to be run when the excel file is changed. (**excel_data_to_mat_file.m**):

```
%Initialisation
clc
clear
format compact
format short g

%loading excel data
table = readcell('Assignment1-Table1-Materials.xlsx');
table(1,:)=[]; %remove the text component

%getting an array for Sy and CF
Index_table = table(:,1);
Sy_table = table(:,2);
Sy_table = cell2mat(Sy_table);
t_table = table(:,3);
CF_table = table(:,4);
CF_table = cell2mat(CF_table);

for i=1:length(Index_table)
    material(i).Sy = Sy_table(i)*1e6;
    %convert arrray of t from string to numbers
    t_row = t_table(i);
    t = str2num(t_row{1});
    material(i).t = t*1e-3;
    material(i).CF = CF_table(i);
end

save('Materials.mat','material');
```

3.2 GUI

The GUI is created using the following code (**GUI_pressure_tank.m**):

```
%Initialisation
clc
clear
close all % closes previously generated GUIs
format short g
format compact
%-----------------------------------------------------------
%Main figure containing the uicontrols
my_figure=figure('Position', [50,50,800,295],'Name','Pressure Vessel
Optimizer'); %position, size

%Text box for Inputs
uicontrol('Parent',my_figure,'Style','text','String', 'Inputs: ',...
 'Position',[95,260,47,22],'HorizontalAlignment','Left', 'FontSize',14);

%Text and Edit boxes for pressure
uicontrol('Parent',my_figure,'Style','text','String', 'Pressure (bar)
=',...
```

```
 'Position',[17,227,82,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','pressure','Style','edit','String', '
',...
  'Position',[113,227,100,22]);

%Text and Edit boxes for Capacity
uicontrol('Parent',my_figure,'Style','text','String', 'Capacity (L) =',...
 'Position',[28,192,71,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','capacity','Style','edit','String', '
',...
  'Position',[113,192,100,22]);

%Text and Edit boxes for Factor of safty
uicontrol('Parent',my_figure,'Style','text','String', 'Factor of Safty
=',...
 'Position',[7,157,91,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','safty_factor','Style','edit','String',
' ',...
  'Position',[113,157,100,22]);

%Text and Edit boxes for Max length
uicontrol('Parent',my_figure,'Style','text','String', 'Max Length (m)
=',...
 'Position',[9,123,89,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','max_length','Style','edit','String', '
',...
  'Position',[113,123,100,22]);

%Text and Edit boxes for Max height
uicontrol('Parent',my_figure,'Style','text','String', 'Max Height (m)
=',...
 'Position',[9,90,89,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','max_height','Style','edit','String', '
',...
  'Position',[113,90,100,22]);

%Text and Edit boxes for step size
uicontrol('Parent',my_figure,'Style','text','String', 'ri step size (mm)
=',... %move
 'Position',[9,55,92,24],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','ri_step','Style','edit','String', '
',...
  'Position',[113,55,100,22]);%30

%%results
%Text box for outputs
uicontrol('Parent',my_figure,'Style','text','String', 'Optimized results
for pressure vessel ',...
 'Position',[367,260,330,22],'HorizontalAlignment','Left', 'FontSize',14);

%Text and Edit boxes for Internal Radius
uicontrol('Parent',my_figure,'Style','text','String', 'Internal Radius (m)
=',...
 'Position',[255,217,111,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','internal_radius','Style','edit','String
', ' ',...
  'Position',[375,217,100,22]);

%Text and Edit boxes for Optimal length
uicontrol('Parent',my_figure,'Style','text','String', 'Optimal Length (m)
=',...
```

```
 'Position',[257,179,111,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','length','Style','edit','String', '
',...
  'Position',[375,179,100,22]);

%Text and Edit boxes for Optimal thickness
uicontrol('Parent',my_figure,'Style','text','String', 'Optimal Thickness
(mm) =',...
 'Position',[221,142,147,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','thickness','Style','edit','String', '
',...
  'Position',[375,142,100,22]);

%Text and Edit boxes for Material index
uicontrol('Parent',my_figure,'Style','text','String', 'Material Index
=',...
 'Position',[257,103,111,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','material_index','Style','edit','String'
, ' ',...
  'Position',[375,103,100,22]);

%Text and Edit boxes for Total Height
uicontrol('Parent',my_figure,'Style','text','String', 'Total Height (m)
=',...
 'Position',[280,67,90,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','total_height','Style','edit','String',
' ',...
  'Position',[375,67,100,22]);

%Text and Edit boxes for Allowable Stree
uicontrol('Parent',my_figure,'Style','text','String', 'Allowable Stress
(MPa) =',...
 'Position',[506,217,139,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','allowable_stress','Style','edit','Strin
g', ' ',...
  'Position',[653,217,100,22]);

%Text and Edit boxes for Von Mises Strees
uicontrol('Parent',my_figure,'Style','text','String', 'Von Mises Stress
(MPa) =',...
 'Position',[506,179,139,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','von_mises_stress','Style','edit','Strin
g', ' ',...
  'Position',[653,179,100,22]);

%Text and Edit boxes for Volume of Material
uicontrol('Parent',my_figure,'Style','text','String', 'Volume of Material
(m^3) =',...
 'Position',[506,142,139,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','material_volume','Style','edit','String
', ' ',...
  'Position',[653,142,100,22]);

%Text and Edit boxes for Cost index
uicontrol('Parent',my_figure,'Style','text','String', 'Cost Index =',...
 'Position',[578,103,68,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','cost_index','Style','edit','String', '
',...
  'Position',[653,103,100,22]);

%Text and Edit boxes for Total Length
```

```
uicontrol('Parent',my_figure,'Style','text','String', 'Total Length (m)
=',...
 'Position',[555,70,91,22],'HorizontalAlignment','Right'); %[555,67,90,22]
uicontrol('Parent',my_figure,'tag','total_length','Style','edit','String',
' ',...
  'Position',[653,67,100,22]);

%Text and Edit boxes for Solution Type
uicontrol('Parent',my_figure,'Style','text','String', 'Solution Type =',...
 'Position',[360,25,111,22],'HorizontalAlignment','Right');
uicontrol('Parent',my_figure,'tag','solution_type','Style','edit','String',
' ',...
  'Position',[476,25,100,22]);


%Push button (calculate)
%calling calculations script
uicontrol('Parent',my_figure,'tag','run','Style','pushbutton','string','Cal
culate',...
'Position',[55,20,100,25],'callback','optimization_search')
```

3.3 Optimization calculations

From the GUI we call the optimization script. This code gets the input from the GUI, checks the input, finds the optimal solution and displays the result with an appropriate message in the GUI (**optimization_search.m**):

```
%adding clear so ensure that the variables are always updated
clear

%---------Get and check inputs:
%getting pressure
P = str2num(get(findobj('tag','pressure'),'string'));
%pressure cannot be negative and cannot be empty
if isempty (P) || P <= 0
 errordlg('Enter valid value for Pressure')
 return
end

%getting capacity
C = str2num(get(findobj('tag','capacity'),'string'));
%capacity cannot be negative and cannot be empty
if isempty (C) || C <= 0
 errordlg('Enter valid value for Capacity')
 return
end

%getting factor of safty
nd = str2num(get(findobj('tag','safty_factor'),'string'));
%capacity cannot be negative and cannot be empty
if isempty (nd) || nd <= 0
 errordlg('Enter valid value for Factor of Safty')
 return
end

%getting Max length
Ltc = str2num(get(findobj('tag','max_length'),'string'));
%length cannot be negative and cannot be empty
if isempty (Ltc) || Ltc <= 0
 errordlg('Enter valid value for Max Length')
 return
```

```
end

%getting Max height
Hc = str2num(get(findobj('tag','max_height'),'string'));
%height cannot be negative and cannot be empty
if isempty (Hc) || Hc <= 0
 errordlg('Enter valid value for Max Height')
 return
end

%getting ri step size
ri_step = str2num(get(findobj('tag','ri_step'),'string'));
%ri_step cannot be negative and cannot be empty
if isempty (ri_step) || ri_step <= 0
 errordlg('Enter valid value for ri step size')
 return
end


%calculations----------------------------------------
%getting the materials and their values from Materials.mat
load('Materials.mat');
%converting P from bar to Pa (1bar = 100000Pa)
P = P *100000;
ri_step = ri_step*10^-3; %convert ri_step from mm to m
ri_min = ri_step; %cannot be 0 so must be size of step
%storing the result
result = [];
for i=1:length(material) %for every material
    Sy = material(i).Sy;
    CF = material(i).CF;
    all_stress = Sy/nd; %allowable stress (Pa)
    available_t = material(i).t;
    for j=1:length(available_t) % for every thickness
        t = available_t(j);
        ri_max = (Hc/2)-t; %maximum possible ri depending on given Hc

        for ri=ri_min:ri_step:ri_max %for every ri, exhaustive Search
            %calculating r0, H, L and Lt
            r0 = ri+t;
            H = 2*r0;
            Vi = C/1000; %vessel internal volume (m3) = capacity (L)/1000
            L = (Vi - (4/3*pi*ri^3))/(pi*ri^2); %length (m)
            Lt = L + 2*r0;
            %calculating material volume and cost index
            Vmat = 2*pi*ri*t*L + 4*pi*(ri)^2*t; %vessel material volume m3)
            CI = Vmat*CF; %vessel cost index

            %calculating von mises stress
            % for type: 0 == 'thin-walled' ,1 == thick-walled
            if t/ri < 0.05
                type = 0;
                stress_t = P*ri/t;
                stress_r = 0;
                stress_l = P*ri/(2*t);
            else
                type = 1;
                stress_t = ((ri^2*P)/(r0^2-ri^2))*(1+((r0)^2)/((ri)^2));
                stress_r = ((ri^2*P)/(r0^2-ri^2))*(1-((r0)^2)/((ri)^2));
                stress_l = (ri^2*P)/(r0^2-ri^2);
            end
```

```
            von_mises_stress = sqrt(((stress_t - stress_r)^2 + (stress_t -
stress_l)^2 + (stress_l - stress_r)^2)/2);

            %ensuring that both height and length are above 0
            %and are below their given constraint values
            %and the condition for strees is fulfilled
            %Lt and H must be less that or equal to Ltc and Hc
            if (0 < H) && (H<= Hc) && (0 < L) &&(0 < Lt) && (Lt <= Ltc) &&
(von_mises_stress < all_stress)
                result = [result; ri, L, t, i, type, all_stress,
von_mises_stress, Vmat, CI, Lt, H];
            end
        end
    end
end

%if we cannot find a solution, popup-message and end calculations
if isempty (result)
    set([findobj('tag','internal_radius'),findobj('tag','length'),
findobj('tag','thickness'), ...
        findobj('tag','material_index'),findobj('tag','total_height'),
findobj('tag','solution_type'),...
        findobj('tag','allowable_stress'),
findobj('tag','von_mises_stress'), ...
        findobj('tag','material_volume'),
findobj('tag','cost_index'),findobj('tag','total_length'), ...
        ], 'string', '');
    errordlg('No solution for this problem')
    return
end


%sort array by cost
sorted_result = sortrows(result, 9);
%selecting the cheapest option
solution = sorted_result(1,:);

%-----getting the values to display, round to appropriate engineering
precision
internal_radius = round(solution(1),4,'significant');
length = round(solution(2),4,'significant');
thickness = solution(3)*10^3;
material_index = solution(4);
%type of solution
if solution(5) == 0
    solution_type = 'thin-walled';
else
    solution_type = 'thick-walled';
end

allowable_stress = round(solution(6)/10^6, 5,'significant');
von_mises_stress = round(solution(7)/10^6, 5,'significant');
material_volume = round(solution(8),3,'significant');
cost_index = round(solution(9),3,'significant');
total_length = round(solution(10),4,'significant');
total_height = round(solution(11),4,'significant');

%-------------Displaying outputs--------------------------
%displaying internal radius
set(findobj('tag','internal_radius'),'string', num2str(internal_radius))
set(findobj('tag','length'),'string', num2str(length))
```

```
set(findobj('tag','thickness'),'string', num2str(thickness))
set(findobj('tag','material_index'),'string', num2str(material_index))
set(findobj('tag','total_height'),'string', num2str(total_height))
set(findobj('tag','solution_type'),'string', num2str(solution_type))
set(findobj('tag','allowable_stress'),'string', num2str(allowable_stress))
set(findobj('tag','von_mises_stress'),'string', num2str(von_mises_stress))
set(findobj('tag','material_volume'),'string', num2str(material_volume))
set(findobj('tag','cost_index'),'string', num2str(cost_index))
set(findobj('tag','total_length'),'string', num2str(total_length))

%message for successful calculation
f = msgbox('                  Solution Found');
```