Contents

Problem : Ten-bar Truss Optimization

十桿桁架 (ten-bar truss) 是典型的桁架結構之一，如圖 1 所示。請利用此十桿桁架範例，演示如何使用有限元素法解決桁架的問題，並最佳化桿件截面半徑。

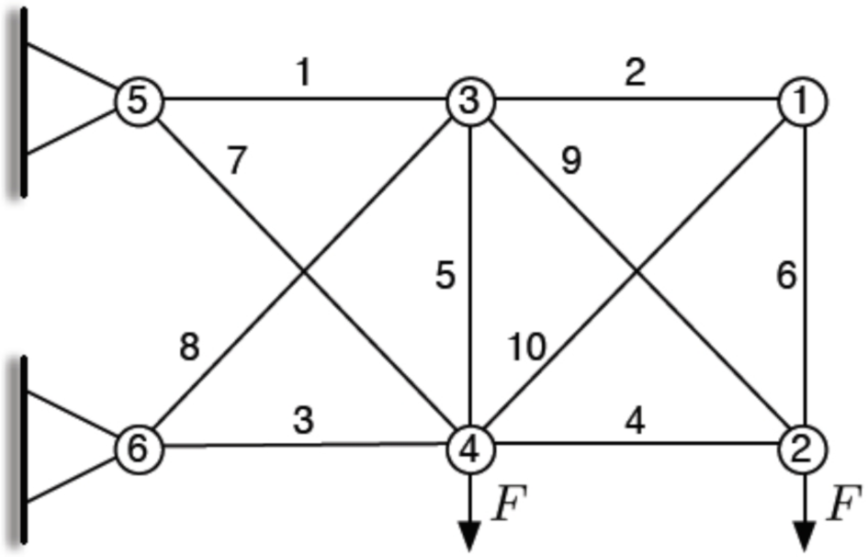

圖 1: Ten-bar truss 結構示意圖

Problem Definition

在以下的已知條件下，給定桿件截面半徑，試求各桿件的位移、應力與反作用力：

- 整體架構處在靜力平衡的情況下
- 所有桿件截面皆爲圓形
- 材料爲鋼，楊氏係數 E = 200 GPa，密度 ρ = 7860 kg/m^3，降伏強度 σ_y = 250 MPa
- 平行桿件與鉛直桿件（桿件 1 至桿件 6）長度皆爲 9.14 m
- 桿件 1 至桿件 6 截面半徑相同爲 r_1，桿件 7 至桿件 10 截面半徑相同爲 r_2
- 所有桿件半徑的最佳化範圍爲 0.001 至 0.5 m 之間
- 在節點 2 和節點 4 上的負載 F 皆爲 1.0×10^7 N 向下

Solution

1. 最佳化數學表示式：

$$
\begin{aligned}
\min_{r_1,r_2} \quad & f(r_1,r_2)=\sum_{i=1}^{6} m_i(r_1)+\sum_{i=7}^{10} m_i(r_2) \\
\text{subject to} \quad & |\boldsymbol{\sigma_i}| \le \sigma_y \\
& \Delta s_2 \le 0.02 \\
\text{where} \quad & f\text{：所有桿件的質量} \\
& \Delta s_2\text{：node 2 的位移} \\
& \sigma_y\text{：降伏應力} \\
& \boldsymbol{\sigma_i}\text{：所有桿件的應力}
\end{aligned} \tag{1}
$$

2. 利用有限元素法分析 10-bar truss problem：

   將有限元素法應用在桁架，各桿件視爲元素 (element)、元素間的連接爲節點 (node)，求解流程如圖 2。首先建立元素表格 (element table)，利用表格中的資料計算出剛性矩陣 (stiffness matrix)，再以力、剛性和位移三者之間的關係找出所求的位移，最後應力和反作用力也能透過位移計算而得。以下使用有限元素法原理計算，運算的過程將利用 Matlab 軟體輔助，撰寫成程式執行。

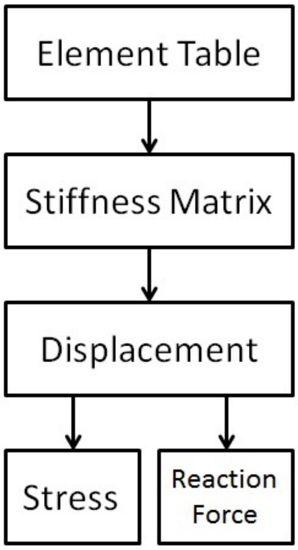

圖 2: 計算流程圖

Step 1 Element Table

根據圖 1 與題目資訊，利用 Excel 製作表 1、2、3，提供 Matlab 在程式中讀取，如程式碼 1。其中桿長與對應的三角函數計算方式如公式 2、3、4 所示：

$$L=\sqrt{(x_{node\,j}-x_{node\,i})^2+(y_{node\,j}-y_{node\,i})^2} \tag{2}$$

$$\cos\theta_e=\frac{(x_{node\,j}-x_{node\,i})}{L} \tag{3}$$

$$\sin\theta_e=\frac{(y_{node\,j}-y_{node\,i})}{L} \tag{4}$$

表 1: 節點座標 (node coordinate)

node	x	y
1	18.28	9.14
2	18.28	0
3	9.14	9.14
4	9.14	0
5	0	9.14
6	0	0

表 2: 元素表格 (element connectivity)

Element	node i	node j
1	3	5
2	1	3
3	4	6
4	2	4
5	3	4
6	1	2
7	4	5
8	3	6
9	2	3
10	1	4

表 3: 節點負載 (load)

node	F_x	F_y
2	0	-1.0×10^7
4	0	-1.0×10^7

```
%%% Parameter
% xlsread(filename,sheet,range)
ncoord   =xlsread('inputdata','node coordinate','A2:C7');
elenode  =xlsread('inputdata','element connectivity','A2:C11');
loadpoint=xlsread('inputdata','load','A2:C3');
% fix parameters
EE=200*10^9;        % Pa
r1=x(1,:);          % m
r2=x(2,:);          % m
Ystress=250*10^6;   % Pa
d2_max=0.02;        % m
```

程式碼 1: 資料讀取與參數設定

Step 2 Stiffness Matrix

每個元素末端連接兩個節點，節點又分別有 x 和 y 兩個方向的自由度 (degree of freedom，DOF)，以 4×4 的剛性矩陣 (stiffness matrix，數學式中以 **k** 表示) 表示元素中 4 個自由度上位移和受力之間的相互關係：

$$\boldsymbol{k}^e = \frac{EA_e}{L_e}\begin{bmatrix} c^2 & cs & -c^2 & -cs \\ cs & s^2 & -cs & -s^2 \\ -c^2 & -cs & c^2 & cs \\ -cs & -s^2 & cs & s^2 \end{bmatrix} \tag{5}$$

其中 c 表示 $\cos\theta_e$；s 表示 $\sin\theta_e$。

桁架整體結構中有 6 個節點，總共有 $2 \times 6 = 12$ 個自由度，以 node1 在 x 方向爲 DOF1、node1 在 y 方向爲 DOF2、node2 的 x 方向爲 DOF3...... 依 node1 到 node6 的順序和 x、y 的順序編號 12 個自由度。透過元素表格揭露的數據代入式 5，下方程式碼 2 將展示 local stiffness matrix 的計算：

```
%%% Calculate the local stiffness matrix
[Ne,~]    =size(elenode); % number of element
[Nnode,~]=size(ncoord);   % number of node
for i=1:Ne
    if i<7                 % for elemetn 1 ~ 6
        A(1,i)=pi*r1^2; % area
    else                   % for elemetn 7 ~ 10
        A(1,i)=pi*r2^2; % area
    end
    % calculate the length and corresponding trigonometric functions
    LL(1,i)=((ncoord(elenode(i,3),2)-ncoord(elenode(i,2),2))^2+...
             (ncoord(elenode(i,3),3)-ncoord(elenode(i,2),3))^2)^0.5;
    theta_c(1,i)=(ncoord(elenode(i,3),2)-ncoord(elenode(i,2),2))/LL(1,i);
    theta_s(1,i)=(ncoord(elenode(i,3),3)-ncoord(elenode(i,2),3))/LL(1,i);
    % local stiffness matrix
    Klocal(1,1,i)= theta_c(1,i)*theta_c(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(1,2,i)= theta_c(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(1,3,i)=-theta_c(1,i)*theta_c(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(1,4,i)=-theta_c(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(2,2,i)= theta_s(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(2,3,i)=-theta_c(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(2,4,i)=-theta_s(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(3,3,i)= theta_c(1,i)*theta_c(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(3,4,i)= theta_c(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
    Klocal(4,4,i)= theta_s(1,i)*theta_s(1,i)*EE*A(1,i)/LL(1,i);
end
% symmetry of the local stiffness matrix
for i=1:Ne
    for j=1:4
        for k=1:4
            if(j>k)
                Klocal(j,k,i)=Klocal(k,j,i);
            end
        end
    end
end
```

程式碼 2: Local Stiffness Matrix

10 個元素皆參考元素表格代入式 5 推導出剛性矩陣後，按照所對應的自由度，整合成 12 × 12 的整體剛性矩陣 **K**：

$$\boldsymbol{K} \leftarrow \sum_{e} \boldsymbol{k}^{e} \tag{6}$$

下方程式碼 3 將展示 local 轉換成 global stiffness matrix 的方法，並一併計算節點負載的 force vector：

```
%%% Calculate the global stiffness matrix and global force vector
% create zero matrix for global stiffness and global force vector
% size = degree of freedom = 6 node * 2 direction = 12
Kglobal(:,:)=zeros(2*Nnode,2*Nnode);
Fglobal     =zeros(2*Nnode,1);
% merge temporary storage for each local stiffness matrix into the global stiffness matrix
for i=1:Ne
    % create zero matrix for temporary storage
    Kglobaltemp(:,:,i)=zeros(2*Nnode,2*Nnode);
    for j=1:2
        for k=1:2
            Kglobaltemp(2*elenode(i,j+1)-1,2*elenode(i,k+1)-1,i)=Klocal(2*j-1,2*k-1,i);
            Kglobaltemp(2*elenode(i,j+1)  ,2*elenode(i,k+1)-1,i)=Klocal(2*j  ,2*k-1,i);
            Kglobaltemp(2*elenode(i,j+1)-1,2*elenode(i,k+1)  ,i)=Klocal(2*j-1,2*k  ,i);
            Kglobaltemp(2*elenode(i,j+1)  ,2*elenode(i,k+1)  ,i)=Klocal(2*j  ,2*k  ,i);
        end
    end
    Kglobal(:,:)=Kglobaltemp(:,:,i)+Kglobal(:,:);
end
% calculate the global forced vector due to point load
[numRows_loadpoint,~]=size(loadpoint); % number of point loads
for n=1:numRows_loadpoint
    Fglobal(2*loadpoint(n,1)-1)=loadpoint(n,2); % Fx
    Fglobal(2*loadpoint(n,1)  )=loadpoint(n,3); % Fy
end
```

程式碼 3: Global Stiffness Matrix and Force Vector

Step 3 Displacement

利用 **Step 2** 中求得的力與剛性，將可求得位移，其三者關係如式 7：

$$\begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_{12} \end{bmatrix} = \begin{bmatrix} K_{1,1} & K_{1,2} & \dots & K_{1,12} \\ K_{2,1} & K_{2,2} & \dots & K_{2,12} \\ \vdots & \vdots & \ddots & \vdots \\ K_{12,1} & K_{12,2} & \dots & K_{12,12} \end{bmatrix} \cdot \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{12} \end{bmatrix} \tag{7}$$

其中必須考慮到邊界條件，由於 node5 和 node6 爲固定端，因此位移爲零，即 d_9 至 d_{12} 等於零，而在運算時也只需考慮 **K** 矩陣的前八行與列、**F** 向量的前八列，化簡矩陣程式碼如 4 所示，而利用式 7 將可求得各節點位移，並最後補上邊界條件的位移，其程式碼如 5 所式：

```
%%% Consider the boundary condition: delete the fix nodes, node5 and node6
% partition the global stiffness matrix and global force vector
Kpart(:,:)=zeros(2*(Nnode-2),2*(Nnode-2));
Fpart     =zeros(2*(Nnode-2),1);
for i=1:2*(Nnode-2)
    for j=1:2*(Nnode-2)
        Kpart(i,j)=Kglobal(i,j);
    end
    Fpart(i)=Fglobal(i);
end
```

程式碼 4: 矩陣去除邊界條件

```
%%% displacement = force/stiffness
dpart=Kpart\Fpart;
% add the displacement in the boundary
displacement=[dpart;0;0;0;0];
```

程式碼 5: 節點位移計算

Step 4 Stress and Reaction Force

應力與應變的關係如下：

$$\sigma = E \times \varepsilon \tag{8}$$

其中，σ 為應力；ε 為應變；E 為楊氏係數。

應變的定義為：

$$\varepsilon = \frac{\delta}{L} \tag{9}$$

其中，δ 為長度變量。

元素的長度變量可以透過節點在各方向的位移作三角函數的轉換得到：

$$\boldsymbol{\sigma} = \frac{E_e}{l_e}\begin{bmatrix} -c & -s & c & s \end{bmatrix}\boldsymbol{d} \tag{10}$$

而利用式 10，將可求得各應力，而反作用力亦可從式 7 求得，其程式碼如 6 所式：

```
%%% Calculate the stress and reaction force
% stress = Young's Modulus * strain = E * (delta length / length)
stress=zeros(Ne,1);
for i=1:Ne
    % calculate the local displacement
    for j=1:2 % local displacement is 4 by 1 matrix
        dlocal(2*j-1,i)=displacement(2*elenode(i,j+1)-1,1);
        dlocal(2*j  ,i)=displacement(2*elenode(i,j+1)  ,1);
    end
    stress(i)=EE/LL(1,i)*[-theta_c(1,i) -theta_s(1,i) theta_c(1,i) theta_s(1,i)]*dlocal(:,i);
end
% force = reacion force + force due to point load = stiffness * displacement
reaction=Kglobal*displacement-Fglobal;
```

程式碼 6: 元素應力與節點反作用力計算

3. 使用 fmincon 進行最佳化：

使用 fmincon 函式進行最佳化時，程式檔案包括 Main 、Objection 、Constraints 三個檔案，本專案分別命名爲 main.m 、FEMobj.m 、FEMcon.m，其中爲符合 fmincon 的輸入型式，須先將最佳化數學式轉換爲 negative null form，如式 11 所示：

$$\begin{aligned} \min_{r_1,r_2} \quad & f(r_1,r_2) = \sum_{i=1}^{6} m_i(r_1) + \sum_{i=7}^{10} m_i(r_2) \\ \text{subject to} \quad & |\boldsymbol{\sigma_i}| - \sigma_y \leq 0 \\ & \Delta s_2 - 0.02 \leq 0 \end{aligned} \tag{11}$$

由於 constrains 式會使用到 FEM 的計算結果，因此上方提及的程式碼 1 至 6 皆是撰寫在 FEMcon.m 中，而 constrains 撰寫與整體 FEMcon.m 檔案 function 格式如程式碼 7 所示：

```
%%% Constraints function
function [g,geq]=FEMcon(x)
```

```
%%% Constraints
for i=1:Ne
    g(i)=abs(stress(i))-Ystress;
end
% the displacement of the node 2 in y direction: index = 4
g(Ne+1)=abs(displacement(4))-d2_max;
geq=[];
```

程式碼 7: Constraints function

該題目目標爲桿件最小總質量，利用截面積、桿長與材料密度來計算桿件質量，再根據目標函數公式 11 撰寫 FEMobj.m，如程式碼 8 所示：

```
%%% Objection function
function f=FEMobj(x)

%%% Parameter
% xlsread(filename,sheet,range)
ncoord   =xlsread('inputdata','node coordinate','A2:C7');
elenode  =xlsread('inputdata','element connectivity','A2:C11');
% fix parameters
DD=7860;        % kg/m3
r1=x(1,:);      % m
r2=x(2,:);      % m
mass_total=0;   % kg

%%% Calculate the mass of truss
[Ne,~]=size(elenode); % number of element
for i=1:Ne
    if i<7                 % for elemetn 1 ~ 6
        A(1,i)=pi*r1^2;  % area
    else                   % for elemetn 7 ~ 10
        A(1,i)=pi*r2^2;  % area
    end
    % calculate the length and mass
    LL(1,i)=((ncoord(elenode(i,3),2)-ncoord(elenode(i,2),2))^2+...
             (ncoord(elenode(i,3),3)-ncoord(elenode(i,2),3))^2)^0.5;
    mass(1,i)=A(1,i)*LL(1,i)*DD;
    mass_total=mass_total+mass(1,i);
end
f=mass_total;
```

程式碼 8: Objection function

最後給定主程式桿件長度範圍與起始點，利用 fmincon 函式並呼叫撰寫完成的 FEMcon、FEMobj 來運算。main.m 撰寫如程式碼 9 所示：

```
%%% main function
clc;clear all;
x0=[0.5;0.5];      % initial points
lb=[0.001;0.001]; % lower bounds
ub=[0.5;0.5];      % upper bounds
options=optimoptions('fmincon','Display','iter','Algorithm','sqp','PlotFcn'
    ,'optimplotfvalconstr');
[x,fval,exitflag]=fmincon(@FEMobj,x0,[],[],[],[],lb,ub,@FEMcon,options);
```

程式碼 9: Main function

根據程式執行結果，可得其最佳值與最佳解：

$$(r_1, r_2) = (0.2937, 0.2665) \quad f = 207470$$

此結果代表在桿件 1 至桿件 6 截面半徑爲 0.2937 m，桿件 7 至桿件 10 截面半徑爲 0.2665 m 時，得到其最小總質量爲 207470 kg，且此設計符合安全規範。其目標函數值與疊帶次數關係如下圖 3：

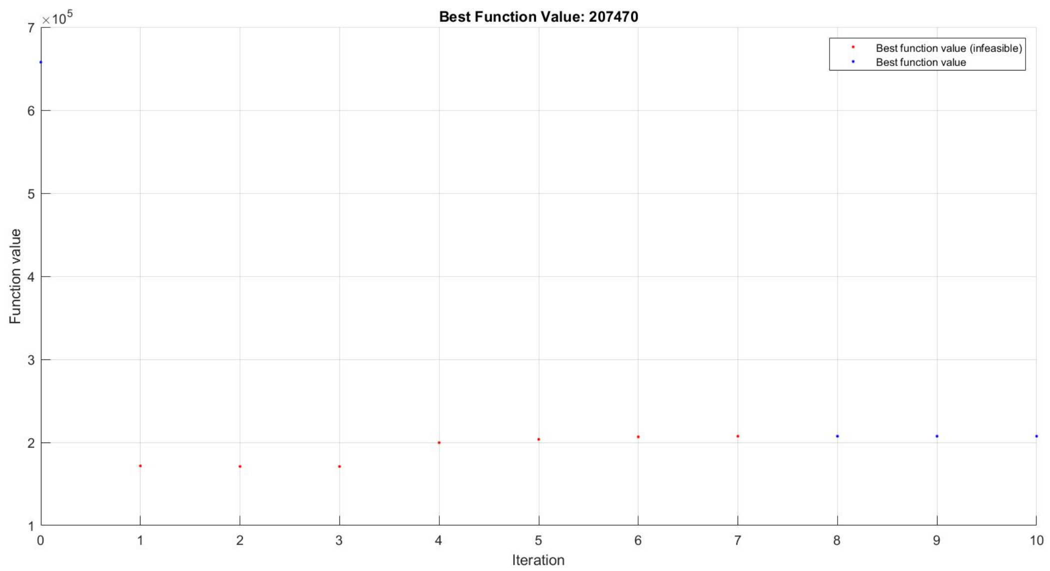

圖 3: 目標函數值與疊帶次數關係圖

References

[1] SOLab. *SOLab_manual_v1.3.pdf*

[2] SOLab. *Finite Element Truss.pdf*