113-2 學年度
有限元素法期末個人書面報告

指導老師：劉彥辰 老師

系級：土木碩一

學號：11372009

姓名：邱昱倫

日期：2025/06/18

目錄

物理問題介紹

本期末報告主要在比較一端固接，且半徑為 1 單位的圓柱構件受扭矩的位移解析解及利用切分二為三角形元素的數值解，同時將切分元素的過程及切分元素的位置呈現在本篇報告。

本物理問題的控制方程式(Governing Equation, G.E.)及邊界條件(Boundary Condition, B.C.)為：

G.E.:

$$-\nabla u^2 = 2G\theta \qquad (\text{in } \Omega)$$

B.C.:

$$u = 0 \qquad (\text{in } \Gamma)$$

其理論解可被表示為：

$$u(x,y) = G\theta\left(\frac{a^2b^2}{a^2+b^2}\right)\left(1-\frac{x^2}{a^2}-\frac{y^2}{b^2}\right)$$

其中 u 為應力，和空間座標 x 及 y 有關，而 a 及 b 為橢圓的長軸、短軸參數，但因為本題目為圓形柱體，所以 a＝b＝1，而 G 為材料的剪力模數，θ 則代表旋轉角，在本篇報告分析的純受扭問題中，$G\theta$=5 為常數。

第一部分 課堂講解方式切分元素流程(四個元素)

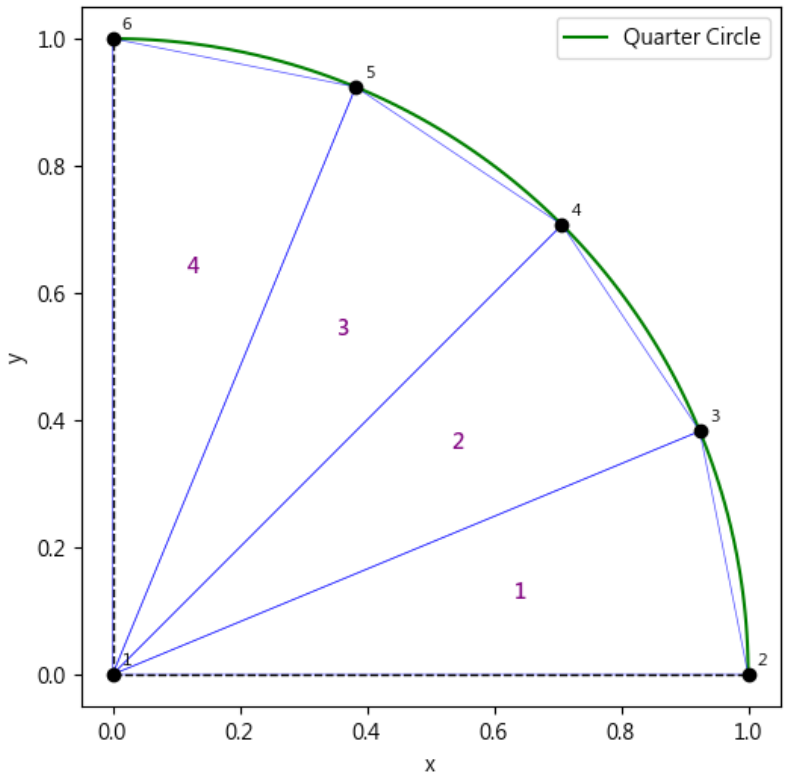

圖 1 課堂方式切分四個元素示意圖

如圖 1 所示，依照課堂所建議的節點數目有 6 個，各節點座標資訊如表 1 所示，所形成的三角形元素則有 4 個，依照課堂講解的方式計算每個元素的面積，詳見表 2。

表 1 課堂方式切分節點編號及座標

節點編號	節點座標
1	(0,0)
2	(1,0)
3	(0.9239, 0.3827)
4	(0.7071, 0.7071)
5	(0.3827, 0.9239)
6	(0,1)

如課程內容所示，面積計算方法：

$$A_e = 0.5 * \left| (x_j - x_i) * (y_k - y_i) - (x_k - x_i) * (y_j - y_i) \right|$$

表 2 課堂方式切分元素面積

元素編號	構成節點	元素面積
1	1,2,3	0.1913
2	1,3,4	0.1913
3	1,4,5	0.1913
4	1,5,6	0.1913

隨後，因為本物理問題切分線性三角形元素，且控制方程式中無交叉微分項，進需計算 β 及 γ 值搭配元素面積即可依照相對位置組成，先拆成四個元素內部所形成 3*3 的局部矩陣，再透過節點編號合併，組成整體系統的 6*6 矩陣k：

$$\beta_i = y_j - y_k$$

$$\gamma_i = x_k - x_j$$

以 3 號元素為例，構成 3 號元素的節點有 1、4、5 節點，依照逆時針規則，把 1 節點對應 Kronecker Delta 關係中的 i，4 節點對應 j，5 節點對應 k，計算個別 β 及 γ 值：

$$\beta_1 = y_2 - y_3 = 0.7071 - 0.9239 = -0.2168$$
$$\beta_2 = y_3 - y_1 = 0.9239 - 0 = 0.9239$$
$$\beta_3 = y_1 - y_2 = 0 - 0.7071 = -0.7071$$

$$\gamma_1 = x_3 - x_2 = 0.3827 - 0.7071 = -0.3244$$
$$\gamma_2 = x_1 - x_3 = 0 - 0.3827 = -0.3827$$
$$\gamma_3 = x_2 - x_1 = 0.7071 - 0 = 0.7071$$

$$k_{ii} = k_{jj} = k_{kk} = \frac{\beta_i\beta_i}{4A_e} + \frac{\gamma_i\gamma_i}{4A_e}$$
$$k_{ij} = k_{ji} = \frac{\beta_i\beta_j}{4A_e} + \frac{\gamma_i\gamma_j}{4A_e}$$
$$k_{ik} = k_{kj} = \frac{\beta_i\beta_k}{4A_e} + \frac{\gamma_i\gamma_k}{4A_e}$$
$$k_{jk} = k_{kj} = \frac{\beta_j\beta_k}{4A_e} + \frac{\gamma_j\gamma_k}{4A_e}$$

將每個元素所對應的節點依照逆時針規則帶入 i、j、k 位置之後，可以得到以下結果：

1 號元素(節點 1、節點 2、節點 3)

$$k^1 = \begin{bmatrix} 0.1989 & -0.0995 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 \\ -0.0995 & -1.2071 & 1.3066 \end{bmatrix}$$

2 號元素(節點 1、節點 3、節點 4)

$$k^2 = \begin{bmatrix} 0.1989 & -0.0995 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 \\ -0.0995 & -1.2071 & 1.3066 \end{bmatrix}$$

3 號元素(節點 1、節點 4、節點 5)

$$k^3 = \begin{bmatrix} 0.1989 & -0.0995 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 \\ -0.0995 & -1.2071 & 1.3066 \end{bmatrix}$$

4 號元素(節點 1、節點 5、節點 6)

$$k^4 = \begin{bmatrix} 0.1989 & -0.0995 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 \\ -0.0995 & -1.2071 & 1.3066 \end{bmatrix}$$

可發現，四個局部 k 矩陣都相同，這是因為每個的三角形元素形狀及面積皆全等，β 值及 γ 值也因為邊長關係相同所以一致。

下一步，透過節點編號合併出總體勁度矩陣 k：

$$\boldsymbol{k} = \begin{bmatrix} 0.7956 & -0.0995 & -0.1990 & -0.1990 & -0.1990 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 & 0.0000 & 0.0000 & 0.0000 \\ -0.1990 & -1.2071 & 2.6132 & -1.2071 & 0.0000 & 0.0000 \\ -0.1990 & 0.0000 & -1.2071 & 2.6132 & -1.2071 & 0.0000 \\ -0.1990 & 0.0000 & 0.0000 & -1.2071 & 2.6132 & -1.2071 \\ -0.0995 & 0.0000 & 0.0000 & 0.0000 & -1.2071 & 1.3066 \end{bmatrix}$$

接下來，定義外加條件 f 向量，為描述計算域(Ω)內部的外力大小。f的計算過程，也是先由單一元素所形成的 3*1 向量，因為這一物理問題的非齊次項 $G\theta$ 為定值，所以$G\theta = f_0 = 10$也是定值，每一項都可以被表示成：

$$f = A_e * f_0/3$$

因此，在同一個元素內的f向量都相同，因為面積都對應到A_e，若另一個元素面積不同，則計算的f才有所不同。

$$f^1 = f^2 = f^3 = f^4 = [0.6378 \quad 0.6378 \quad 0.6378]^T$$

透過節點編號的組合所合併出來的 6*1 向量：

$$\boldsymbol{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \end{bmatrix} = \begin{bmatrix} f_1^1 + f_1^2 + f_1^3 + f_1^4 \\ f_2^1 \\ f_3^1 + f_2^2 \\ f_3^2 + f_2^3 \\ f_3^3 + f_2^4 \\ f_3^4 \end{bmatrix} = \begin{bmatrix} 2.5513 \\ 0.6378 \\ 1.2756 \\ 1.2756 \\ 1.2756 \\ 0.6378 \end{bmatrix}$$

定義完計算域內部的節點外加條件，對於在邊界上(Γ)的節點，要用Q，一個 6*1 的向量，來描述外加條件對該點所造成的效應，所以純扭的物理問題中，對於在計算域內部的節點，Q向量所對應的元素數值為 0，其他在邊界上的點數值則需透過求解完物理量之後再帶回系統求解Q：

$$\boldsymbol{Q} = \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{bmatrix} = \begin{bmatrix} 0 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \\ Q_6 \end{bmatrix}$$

依照課堂講解過程，有了k矩陣、f向量、Q向量，可以定義u向量，以求解物理問題的核心物理量。

在此問題中，u所對應的物理量是應力大小，因為本篇報告在這種分割方式下產生了 6 個節點，所以u向量也是 6*1 的尺寸，將邊界條件帶入修改u向量，我們可以知道在邊界上的點，應力值都是 0，所以，u_2~u_6都為 0。

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} = \begin{bmatrix} u_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

如此一來，將原本的線性方程組：$\boldsymbol{k} * \boldsymbol{u} = \boldsymbol{f} + \boldsymbol{Q}$，觀察$u$向量的內部，可以發現此問題若要求解$u_1$可以化簡成一條方程式：

$$u_1 = \frac{f_1}{k_{11}} = \frac{2.5513}{0.7956} = 3.2067$$

得到u_1之後，我們可以再帶回其他方程式求解Q_2~Q_5，即：

$$k_{21} * u_1 = f_2 + Q_2$$

$$k_{31} * u_1 = f_3 + Q_3$$

$$k_{41} * u_1 = f_4 + Q_4$$

$$k_{51} * u_1 = f_5 + Q_5$$

$$k_{61} * u_1 = f_6 + Q_6$$

反推各個元素後，帶回原始 Q 向量，則物理問題即求解完畢：

$$\boldsymbol{u} = \begin{bmatrix} 3.2067 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \boldsymbol{f} = \begin{bmatrix} 2.5513 \\ 0.6378 \\ 1.2756 \\ 1.2756 \\ 1.2756 \\ 0.6378 \end{bmatrix} \boldsymbol{Q} = \begin{bmatrix} 0 \\ -0.9569 \\ -1.9138 \\ -1.9137 \\ -1.9138 \\ -0.9569 \end{bmatrix}$$

$$\boldsymbol{k} = \begin{bmatrix} 0.7956 & -0.0995 & -0.1990 & -0.1990 & -0.1990 & -0.0995 \\ -0.0995 & 1.3066 & -1.2071 & 0.0000 & 0.0000 & 0.0000 \\ -0.1990 & -1.2071 & 2.6132 & -1.2071 & 0.0000 & 0.0000 \\ -0.1990 & 0.0000 & -1.2071 & 2.6132 & -1.2071 & 0.0000 \\ -0.1990 & 0.0000 & 0.0000 & -1.2071 & 2.6132 & -1.2071 \\ -0.0995 & 0.0000 & 0.0000 & 0.0000 & -1.2071 & 1.3066 \end{bmatrix}$$

課堂方法與解析解的誤差比較

本篇期末報告意旨在練習切分二維有限元素中的三角形網格元素，為印證元素的數量及切分方式的不同會造成精準度的差別，需要透過和解析解的比較觀察不同分割情形所造成的誤差量。

$$u_{exact} = G\theta \left(\frac{1^2 1^2}{1^2 + 1^2} \right) \left(1 - \frac{x^2}{1^2} - \frac{y^2}{1^2} \right) \Big|_{x,y=0} \rightarrow 2.5$$

u_{exact}為解析解公式帶入計算域的節點座標所計算出來的應力大小，可以觀察到u_1的誤差量和理論公式絕對誤差約 28.269%。

$$rel_{error} = \left| \frac{u_i - u_{exact}}{u_{exact}} \right| * 100\% \rightarrow 28.269\%$$

第二部分 自行切分元素流程(四個元素)

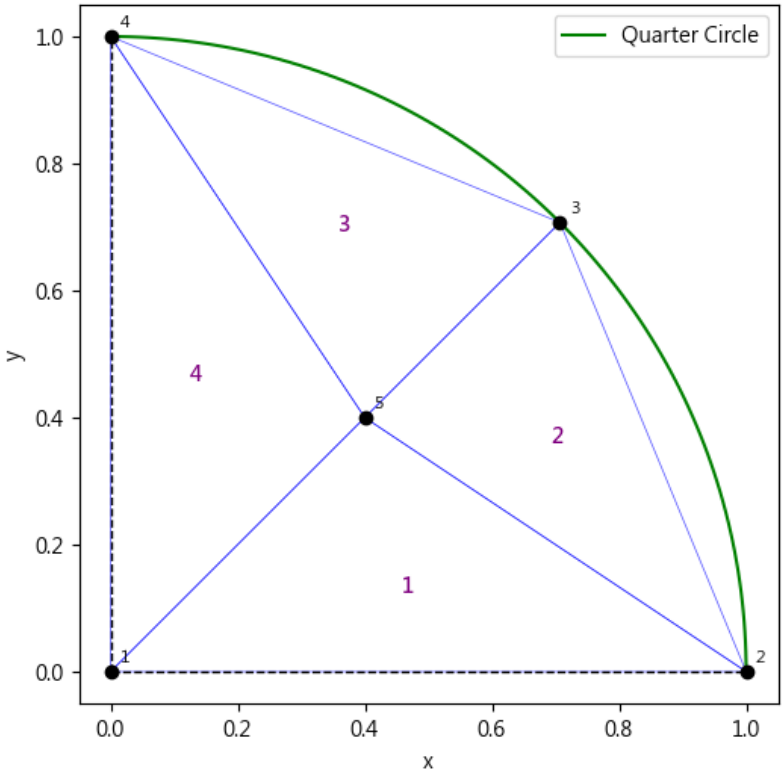

圖 2 自行分割四個元素示意圖

如圖 2 所示，自行切分的節點數目有 5 個，各節點座標資訊如表 3 所示，所形成的二維三角形元素則有 4 個，重複利用第一部份的公式計算面積，詳細資料詳見表 4。

表 3 自行切分節點編號及座標

節點編號	節點座標
1	(0,0)
2	(1,0)
3	(0.7071, 0.7071)
4	(0,1)
5	(0.4, 0.4)

表 4 自行切分元素面積

元素編號	構成節點	元素面積
1	1，2，5	0.2
2	2，3，5	0.1535
3	5，3，4	0.1535
4	1，5，4	0.2

計算完元素面積同時也有明確的節點資訊，即可開始計算 β 及 γ 數值，以拼湊出整體勁度矩陣 k，詳細計算過程同第一部分，將表 4 的元素構成節點依照 Kronecker Delta 關係計算 β 及 γ，其結果如表 5 所示。

表 5 自行切分元素的 β 及 γ

元素編號	$\beta_i = y_j - y_k$	$\gamma_i = x_k - x_j$
1	-0.4	-0.6
	0.4	-0.4
	0	1
2	0.3071	-0.3071
	0.4	0.6
	-0.7071	-0.2929
3	-0.2929	-0.7071
	0.6	0.4
	-0.3071	0.3071
4	-0.6	-0.4
	1	0
	-0.4	0.4

得到β及γ值之後即可計算個別元素的局部k矩陣，如同第一部分所陳述的公式：

1號元素(節點1、節點2、節點5)

$$k^1 = \begin{bmatrix} 0.65 & 0.1 & -0.75 \\ 0.1 & 0.4 & -0.5 \\ -0.75 & -0.5 & 1.25 \end{bmatrix}$$

2號元素(節點2、節點3、節點5)

$$k^1 = \begin{bmatrix} 0.3071 & -0.1 & -0.2071 \\ -0.1 & 0.8466 & -0.7466 \\ -0.2071 & -0.7466 & 0.9537 \end{bmatrix}$$

3號元素(節點5、節點3、節點4)

$$k^1 = \begin{bmatrix} 0.9537 & -0.7466 & -0.2071 \\ -0.7466 & 0.8466 & 0.1 \\ -0.2071 & 0.1 & 0.3071 \end{bmatrix}$$

4號元素(節點1、節點5、節點4)

$$k^1 = \begin{bmatrix} 0.65 & -0.75 & 0.1 \\ -0.75 & 1.25 & -0.5 \\ 0.1 & -0.5 & 0.4 \end{bmatrix}$$

接下來，依照節點編號組成整體勁度矩陣，將自行切分方式所建立的節點關係將各個3*3的k矩陣元素進行疊加，得到總體k矩陣：

$$\boldsymbol{k} = \begin{bmatrix} 1.3 & 0.1 & 0 & 0.1 & -1.5 \\ 0.1 & 0.7071 & -0.1 & 0 & -0.7071 \\ 0 & -0.1 & 1.6932 & 0.1 & 1.4932 \\ 0.1 & 0 & 0.1 & 0.7071 & -0.7071 \\ -1.5 & -0.7071 & -1.4932 & -0.7071 & 4.4074 \end{bmatrix}$$

依照第一部分流程，接下來將定義描述總體外加條件對計算域造成的影響f向量，如同前述內容，取得每個元素的面積A_e，再帶入非齊次項的數值f_0，可以得到以下局部f：

1 號元素(節點 1、節點 2、節點 5，$A_e = 0.2$，$f_0 = 5$)：

$$f^1 = \begin{bmatrix} 0.3333 \\ 0.3333 \\ 0.3333 \end{bmatrix}$$

2 號元素(節點 2、節點 3、節點 5，$A_e = 0.1535$，$f_0 = 5$)：

$$f^2 = \begin{bmatrix} 0.2558 \\ 0.2558 \\ 0.2558 \end{bmatrix}$$

3 號元素(節點 5、節點 3、節點 4，$A_e = 0.1535$，$f_0 = 5$)：

$$f^3 = \begin{bmatrix} 0.2558 \\ 0.2558 \\ 0.2558 \end{bmatrix}$$

4 號元素(節點 1、節點 5、節點 4，$A_e = 0.2$，$f_0 = 5$)：

$$f^4 = \begin{bmatrix} 0.3333 \\ 0.3333 \\ 0.3333 \end{bmatrix}$$

如此一來，即可合併f向量：

$$\boldsymbol{f} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} f_1^1 + f_1^4 \\ f_2^1 + f_1^2 \\ f_2^2 + f_2^3 \\ f_3^3 + f_3^4 \\ f_3^1 + f_3^2 + f_1^3 + f_2^4 \end{bmatrix} = \begin{bmatrix} 1.3333 \\ 1.1785 \\ 1.0237 \\ 1.1785 \\ 2.3570 \end{bmatrix}$$

再來定義Q向量，和f向量一般，是一個尺寸為 5*1 的向量，因為在邊界上的點有：節點 2、節點 3、節點 4，所以此三項不為 0，其餘都為 0：

$$\boldsymbol{Q} = \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \\ Q_5 \end{bmatrix} = \begin{bmatrix} 0 \\ Q_2 \\ Q_3 \\ Q_4 \\ 0 \end{bmatrix}$$

最後，將待求的物理量，u向量表示出來，再將邊界條件代入，即節點 2、節點 3、節點 4 的應力為 0，即可化簡整體計算難度，帶入線性聯立方程組$\boldsymbol{k} * \boldsymbol{u} = \boldsymbol{f} + \boldsymbol{Q}$，可以先求解$u_1$和$u_5$：

$$\boldsymbol{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} u_1 \\ 0 \\ 0 \\ 0 \\ u_5 \end{bmatrix}$$

經過化簡的k、u、f、Q結果如下：

$$\begin{bmatrix} k_{11} & k_{51} \\ k_{15} & k_{55} \end{bmatrix} \begin{bmatrix} u_1 \\ u_5 \end{bmatrix} = \begin{bmatrix} f_1 \\ f_5 \end{bmatrix}$$

$$\begin{bmatrix} 1.3 & -1.5 \\ -1.5 & 4.4074 \end{bmatrix} \begin{bmatrix} u_1 \\ u_5 \end{bmatrix} = \begin{bmatrix} 1.3333 \\ 2.357 \end{bmatrix}$$

經過計算，得u_1和u_5結果為：

$$\begin{bmatrix} u_1 \\ u_5 \end{bmatrix} = \begin{bmatrix} 2.7049 \\ 1.4554 \end{bmatrix}$$

再將u_1和u_5帶回線性方程組，即可反推得到Q向量，最終計算結果如下：

$$\boldsymbol{u}=\begin{bmatrix}2.7049\\0\\0\\0\\1.4554\end{bmatrix}\quad \boldsymbol{f}=\begin{bmatrix}1.3333\\1.1785\\1.0237\\1.1785\\2.3570\end{bmatrix}\quad \boldsymbol{Q}=\begin{bmatrix}0\\-1.9371\\-3.1968\\-1.9371\\0\end{bmatrix}$$

$$\boldsymbol{k}=\begin{bmatrix}1.3 & 0.1 & 0 & 0.1 & -1.5\\0.1 & 0.7071 & -0.1 & 0 & -0.7071\\0 & -0.1 & 1.6932 & 0.1 & 1.4932\\0.1 & 0 & 0.1 & 0.7071 & -0.7071\\-1.5 & -0.7071 & -1.4932 & -0.7071 & 4.4074\end{bmatrix}$$

自行切分元素與解析解的誤差比較

將計算域內部自行定義的點(節點 1、節點 5)座標值代入解析解公式計算應力值，得到u_{exaxt}，再進行誤差比較：

$$u_{exaxt}|_{(0,0)}=2.5$$

$$u_{exaxt}|_{(0.4,0.4)}=1.7$$

表 6 為絕對誤差大小，可以發現，和課堂方式相比，誤差值下降許多，但仍有 10%以上，本篇報告在下一部份會進行更詳細的節點切分，以提高準確率。

表 6 自行切分元素的誤差比較

節點編號	數值解u_i	解析解u_{exaxt}	絕對誤差rel_{error}(%)
1	2.7049	2.5	8.196
5	1.4554	1.7	14.391

第三及第四部分 自行切分元素(分割二十三個元素)

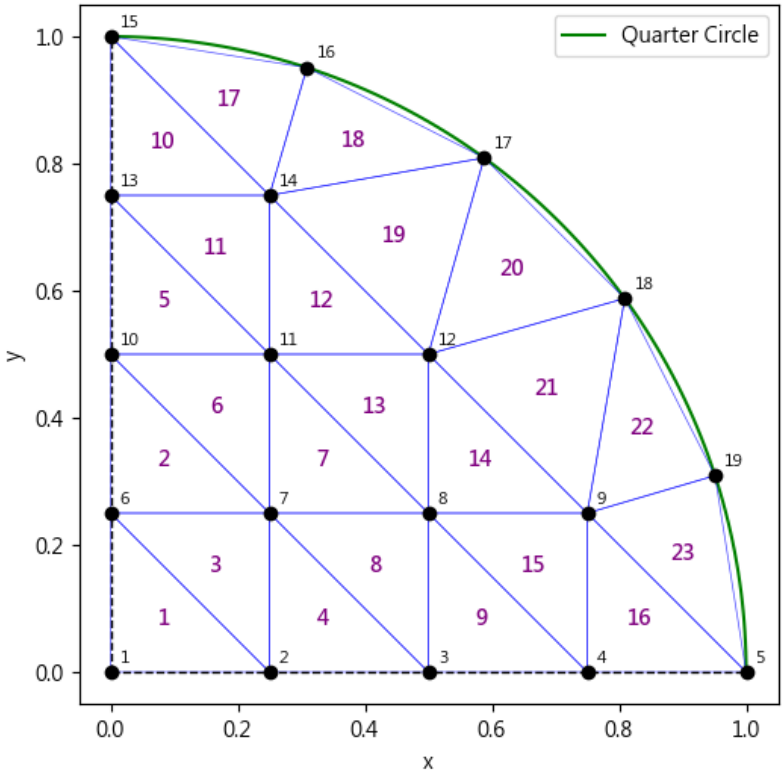

圖 3 自行分割二十三個元素示意圖

如圖 3 所示，本報告在最後一部分，利用 Python 進行輔助計算，因為節點數目龐大、元素複雜、矩陣和向量的合併都不再適合手動進行計算。

自行指定的節點數目有 19 個，各節點座標資訊如表 7 所示，所形成的三角形元素則有 23 個，第三部分依舊利用前述公式計算面積，詳細資料詳見表 8。

表 7 自行定義的 19 個節點座標

節點編號	節點座標
1	(0.0000, 0.0000)
2	(0.2500, 0.0000)
3	(0.5000, 0.0000)
4	(0.7500, 0.0000)
5	(1.0000, 0.0000)
6	(0.0000, 0.2500)
7	(0.2500, 0.2500)
8	(0.5000, 0.2500)
9	(0.7500, 0.2500)
10	(0.0000, 0.5000)
11	(0.2500, 0.5000)
12	(0.5000, 0.5000)
13	(0.0000, 0.7500)
14	(0.2500, 0.7500)
15	(0.0000, 1.0000)
16	(0.3090, 0.9510)
17	(0.5878, 0.8090)
18	(0.8090, 0.5878)
19	(0.9510, 0.3090)

表 8 自行定義的 23 個元素面積

元素編號	構成節點	元素面積
1	(1, 2, 6)	0.0312
2	(6, 7, 10)	0.0312
3	(6, 2, 7)	0.0312
4	(2, 3, 7)	0.0312
5	(10, 11, 13)	0.0312
6	(10, 7, 11)	0.0312
7	(7, 8, 11)	0.0312
8	(7, 3, 8)	0.0312
9	(8, 3, 4)	0.0312
10	(15, 13, 14)	0.0312
11	(13, 11, 14)	0.0312
12	(11, 12, 14)	0.0312
13	(11, 8, 12)	0.0312
14	(8, 9, 12)	0.0312
15	(8, 4, 9)	0.0312
16	(9, 4, 5)	0.0312
17	(15, 14, 16)	0.0325
18	(16, 14, 17)	0.0322
19	(14, 12, 17)	0.0496
20	(17, 12, 18)	0.0439
21	(18, 12, 9)	0.0496
22	(18, 9, 19)	0.0322
23	(9, 5, 19)	0.0325

依照前述流程，將每一個元素的構成節點利用程式的運算將β和γ值計算出來，所得結果如下：

表 9 23 個元素所對應的β和γ值

元素編號	β_1	β_2	β_3	γ_1	γ_2	γ_3
1	-0.25	0.25	0	-0.25	0	0.25
2	-0.25	0.25	0	-0.25	0	0.25
3	-0.25	0	0.25	0	-0.25	0.25
4	-0.25	0.25	0	-0.25	0	0.25
5	-0.25	0.25	0	-0.25	0	0.25
6	-0.25	0	0.25	0	-0.25	0.25
7	-0.25	0.25	0	-0.25	0	0.25
8	-0.25	0	0.25	0	-0.25	0.25
9	0	-0.25	0.25	0.25	-0.25	0
10	0	-0.25	0.25	0.25	-0.25	0
11	-0.25	0	0.25	0	-0.25	0.25
12	-0.25	0.25	0	-0.25	0	0.25
13	-0.25	0	0.25	0	-0.25	0.25
14	-0.25	0.25	0	-0.25	0	0.25
15	-0.25	0	0.25	0	-0.25	0.25
16	0	-0.25	0.25	0.25	-0.25	0
17	-0.201	-0.049	0.25	0.059	-0.309	0.25
18	-0.059	-0.142	0.201	0.3378	-0.2788	-0.059
19	-0.309	0.059	0.25	0.0878	-0.3378	0.25
20	-0.0878	-0.2212	0.309	0.309	-0.2212	-0.0878
21	0.25	-0.3378	0.0878	0.25	0.059	-0.309
22	-0.059	-0.2788	0.3378	0.201	-0.142	-0.059
23	-0.309	0.059	0.25	-0.049	-0.201	0.25

接下來得以利用表 9 的資訊將所有局部勁度矩陣計算出來，並進行合併，得到整體勁度矩陣 k(19*19)，(因為版面尺寸受限，請見附錄)。再來，以程式碼搭配節點下標進行合併，即可把f向量(19*1)定義出來，如表 10 所示：

表 10 f 向量

節點編號	f_i
1	0.104167
2	0.3125
3	0.3125
4	0.3125
5	0.2125
6	0.3125
7	0.625
8	0.625
9	0.693528
10	0.3125
11	0.625
12	0.789454
13	0.3125
14	0.693528
15	0.2125
16	0.215695
17	0.418982
18	0.418982
19	0.215695

依照前述步驟，假設Q向量，並將未在計算邊界上的點指定為 0，即除了節點 5、15、16、17、18、19 共 6 個節點不為 0，其餘的編號Q值都為 0。

表 11 Q 向量

節點編號	Q_i
1	0
2	0
3	0
4	0
5	Q_5
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	Q_{15}
16	Q_{16}
17	Q_{17}
18	Q_{18}
19	Q_{19}

再者，產生一個(19*1)的u向量，代入邊界條件後，即可發現，在邊界上的節點 5、15、16、17、18、19 的應力值都為 0，如表 12 所示。

表 12 u 向量

節點編號	u_i
1	u_1
2	u_2
3	u_3
4	u_4
5	0
6	u_6
7	u_7
8	u_8
9	u_9
10	u_{10}
11	u_{11}
12	u_{12}
13	u_{13}
14	u_{14}
15	0
16	0
17	0
18	0
19	0

利用程式碼進行化簡及計算之後，可以成功得到u向量的未知數值，隨後再將其帶回線性聯立方程組求出Q向量，得到的結果如表 13 所示：

表 13 自行切分 23 元素所得各向量

Node	u_i	f_i	Q_i
1	2.40874	0.104167	0
2	2.304574	0.3125	0
3	1.859325	0.3125	0
4	1.091416	0.3125	0
5	0	0.2125	1
6	2.304574	0.3125	0
7	2.162615	0.625	0
8	1.708156	0.625	0
9	0.94067	0.693528	0
10	1.859325	0.3125	0
11	1.708156	0.625	0
12	1.245012	0.789454	0
13	1.091416	0.3125	0
14	0.94067	0.693528	0
15	0	0.2125	1
16	0	0.215695	1
17	0	0.418982	1
18	0	0.418982	1
19	0	0.215695	1

自行切分 23 個元素與解析解的誤差比較

依照第一部分觀念，將u向量和解析解進行誤差比較，如表 14 所示，可以發現誤差量大幅下降，跟第一部分 28.269%相比，以 23 個元素的切分方式，同一點的誤差只有 3.65%，是最大的誤差，其餘都小於 2%，可見，分割 23 個元素能有效提高計算精準度。

表 14 自行切分 23 個元素的誤差比較

節點編號	u	u_{exact}	$rel_{exact}(\%)$
1	2.40874	2.5	3.6503
2	2.304574	2.34375	1.6715
3	1.859325	1.875	0.8360
4	1.091416	1.09375	0.2134
5	0	0	-
6	2.304574	2.34375	1.6715
7	2.162615	2.1875	1.1376
8	1.708156	1.71875	0.6164
9	0.94067	0.9375	0.3381
10	1.859325	1.875	0.8360
11	1.708156	1.71875	0.6164
12	1.245012	1.25	0.3990
13	1.091416	1.09375	0.2134
14	0.94067	0.9375	0.3381
15	0	0	-
16	0	0.000295	-
17	0	2.54E-05	-
18	0	2.54E-05	-
19	0	0.000295	-

心得及討論

本篇報告練習了課堂所教學的切分二維三角形元素的流程，以手動方式可以推導的過程將所有節點、元素及各向量計算出來，熟悉流程之後，自行切分元素，最後再依照流程寫入 Python 程式碼，省略複雜的計算時間及手動合併的過程。

在此報告撰寫及練習過程，我學到有限元素不僅僅是一門將空間及物理量離散的學問，更要考慮到如何建立節點及方式會深深影響計算結果及效率，以這一個報告的最後一部分而言，我利用等腰直角三角形將大部分的區域做分割，如此一來，三角形的面積全等，可以有效降低k矩陣的計算複雜度，同時f向量也有很多項會全等，如此一來，簡化計算量；而等腰直角三角形以外的區域就直觀地連接邊界上的點，進行分割，其邊界上的點位則是以角度變化量每 18 度一個點進行指定，經過程式碼計算，發現雖然最中心點的誤差降低到 3.65%，但我認為還是不夠準確，日後可以再練習，處理誤差較大的部分，未來職場上可能會用到局部密集網格方式處理一些容易有誤差的部分，總之，我認為這個報告讓我受益良多。

附錄

1.Python 程式碼(自行切分 23 個元素)

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from numpy.linalg import solve
plt.rcParams['font.sans-serif'] = ['Microsoft JhengHei']  # 設定中文字型
plt.rcParams['axes.unicode_minus'] = False  # 正確顯示負號

# === 1. 定義節點與元素 ===
nodes = [
    (0.00, 0.00), (0.25, 0.00), (0.50, 0.00), (0.75, 0.00), (1.00, 0.00),
    (0.00, 0.25), (0.25, 0.25), (0.50, 0.25), (0.75, 0.25),
    (0.00, 0.50), (0.25, 0.50), (0.50, 0.50),
    (0.00, 0.75), (0.25, 0.75), (0.00, 1.00),
    (0.309, 0.951), (0.5878, 0.809), (0.809, 0.5878), (0.951, 0.309)
]
nodes = np.array(nodes)

elements = [
    [1, 2, 6], [6, 7, 10], [6, 2, 7], [2, 3, 7],
```

```
        [10, 11, 13], [10, 7, 11], [7, 8, 11], [7, 3, 8],
        [8, 3, 4], [15, 13, 14], [13, 11, 14], [11, 12, 14],
        [11, 8, 12], [8, 9, 12], [8, 4, 9], [9, 4, 5],
        [15, 14, 16], [16, 14, 17], [14, 12, 17], [17, 12, 18],
        [18, 12, 9], [18, 9, 19], [9, 5, 19]
    ]

    # === 2. 面積與剛度矩陣函數 ===
    def triangle_area(p1, p2, p3):
        return 0.5 * abs((p2[0]-p1[0])*(p3[1]-p1[1]) - (p3[0]-
p1[0])*(p2[1]-p1[1]))

    def compute_betas(p1, p2, p3):
        return [p2[1]-p3[1], p3[1]-p1[1], p1[1]-p2[1]]

    def compute_gammas(p1, p2, p3):
        return [p3[0]-p2[0], p1[0]-p3[0], p2[0]-p1[0]]

    def compute_k_matrix(area, beta, gamma):
        k = np.zeros((3, 3))
        for i in range(3):
            for j in range(3):
                k[i][j] = (beta[i]*beta[j] + gamma[i]*gamma[j]) / (4 *
area)
        return k
```

```
# === 3. 初始化 K, f, q ===
num_nodes = len(nodes)
K_global = np.zeros((num_nodes, num_nodes))
f_vector = np.zeros((num_nodes, 1))
q_vector = np.zeros((num_nodes, 1))
f0 = 10
areas = []

for e in elements:
    n1, n2, n3 = e
    p1, p2, p3 = nodes[n1-1], nodes[n2-1], nodes[n3-1]
    area = triangle_area(p1, p2, p3)
    areas.append(area)
    beta = compute_betas(p1, p2, p3)
    gamma = compute_gammas(p1, p2, p3)
    k = compute_k_matrix(area, beta, gamma)

    # 組裝 K_global
    for i_local, i_global in enumerate([n1, n2, n3]):
        for j_local, j_global in enumerate([n1, n2, n3]):
            K_global[i_global-1, j_global-1] += k[i_local, j_local]

    # 加入 f 向量（平均分配到 3 個節點）
    for node in [n1, n2, n3]:
```

```
            f_vector[node-1] += f0 * area / 3

    # === 4. 定義邊界條件並求解 u ===
    boundary_nodes = [5, 15, 16, 17, 18, 19]
    interior_nodes = [i for i in range(1, num_nodes+1) if i not in
boundary_nodes]
    interior_idx = [i-1 for i in interior_nodes]

    # 邊界點強制項（可依需求調整值，這裡設為 1）
    for node in boundary_nodes:
        q_vector[node-1] = 1.0

    # 求解 u（僅對內部點）
    K_reduced = K_global[np.ix_(interior_idx, interior_idx)]
    rhs = f_vector[interior_idx] + q_vector[interior_idx]
    u_reduced = solve(K_reduced, rhs)

    # 還原完整 u 向量
    u_vector = np.zeros((num_nodes, 1))
    for i, idx in enumerate(interior_idx):
        u_vector[idx] = u_reduced[i]

    # === 5. 加入解析解與誤差 ===
    u_exact = []
```

```
for x, y in nodes:
    u_val = 5 * 0.5 * (1 - x**2 - y**2)
    u_exact.append(u_val)
u_exact = np.array(u_exact).flatten()


rel_error = []
for i in range(len(u_exact)):
    denom = u_exact[i]
    num = u_vector[i, 0]
    if abs(denom) > 1e-12:
        rel_error.append(abs((num - denom) / denom * 100))
    else:
        rel_error.append(np.nan)
rel_error = np.array(rel_error).flatten()


# === 6. 輸出總結果表格 ===
df_result = pd.DataFrame({
    "x": nodes[:, 0],
    "y": nodes[:, 1],
    "u": u_vector.flatten(),
    "f": f_vector.flatten(),
    "q": q_vector.flatten(),
    "u_exact": u_exact,
    "rel_error(%)": rel_error
})
```

```
df_result.index += 1
df_result.index.name = "Node"

# 節點資訊以 DataFrame 輸出
node_df = pd.DataFrame({
    '節點編號': np.arange(1, len(nodes)+1),
    '節點座標': [f"({x:.4f}, {y:.4f})" for x, y in nodes]
})

# 元素資訊以 DataFrame 格式輸出
# 構成節點以 1-based 顯示
el_df = pd.DataFrame({
    '元素編號': np.arange(1, len(elements)+1),
    '構成節點': [f"({e[0]}, {e[1]}, {e[2]})" for e in elements],
    '元素面積': [f"{a:.4f}" for a in areas]
})

# 輸出每個元素的 beta, gamma 參數
betas_list = []
gammas_list = []
for e in elements:
    n1, n2, n3 = e
    p1, p2, p3 = nodes[n1-1], nodes[n2-1], nodes[n3-1]
```

```
        beta = [p2[1]-p3[1], p3[1]-p1[1], p1[1]-p2[1]]
        gamma = [p3[0]-p2[0], p1[0]-p3[0], p2[0]-p1[0]]
        betas_list.append(beta)
        gammas_list.append(gamma)

beta_gamma_df = pd.DataFrame({
        '元素編號': np.arange(1, len(elements)+1),
        'beta1': [b[0] for b in betas_list],
        'beta2': [b[1] for b in betas_list],
        'beta3': [b[2] for b in betas_list],
        'gamma1': [g[0] for g in gammas_list],
        'gamma2': [g[1] for g in gammas_list],
        'gamma3': [g[2] for g in gammas_list]
})
# === 7. 輸出到終端機 ===
print("\n 節點解與向量彙總（u, f, q, u_exact, rel_error）:")
print(df_result.round(6).to_string())

print("\n 節點資訊：")
print(node_df.to_string(index=False))

print("\n 元素資訊：")
print(el_df.to_string(index=False))
```

```
# 輸出元素 beta, gamma 參數

print("\n 元素 beta, gamma 參數：")

print(beta_gamma_df.round(6).to_string(index=False))


# K_global 矩陣以 DataFrame 輸出

print("\n 整體勁度矩陣 K_global：")

df_K = pd.DataFrame(K_global)

print(df_K.round(4).to_string(index=False, header=False))


# 輸出 f 向量

print("\nf 向量：")

df_f = pd.DataFrame(f_vector, columns=["f"])

df_f.index += 1

df_f.index.name = "Node"

print(df_f.round(6).to_string())


# 匯出所有表格到同一份 CSV

with open('23_element_data.csv', 'w', encoding='utf-8-sig') as f:

    f.write('節點資訊：\n')

    node_df.to_csv(f, index=False)

    f.write('\n 元素資訊：\n')

    el_df.to_csv(f, index=False)
```

```
            f.write('\n 元素 beta, gamma 參數：\n')
            beta_gamma_df.to_csv(f, index=False)
            f.write('\nf 向量：\n')
            df_f.to_csv(f)
            f.write('\n 整體勁度矩陣 K_global：\n')
            df_K.to_csv(f, index=False, header=False)
            f.write('\n 節點解與向量彙總（u, f, q, u_exact, rel_error）：
\n')
            df_result.to_csv(f)

        # 劃出元素切割示意圖
        plt.figure(figsize=(6, 6))
        plt.scatter(nodes[:, 0], nodes[:, 1], c='black', zorder=3)
        for i, (x, y) in enumerate(nodes):
            plt.text(x + 0.015, y + 0.015, str(i+1), fontsize=8,
color='black')

        # 畫出每個元素的三角形邊，並在幾何中心標註元素編號
        for idx, e in enumerate(elements):
            n1, n2, n3 = [idx - 1 for idx in e]  # 轉成 0-based index
            tri_x = [nodes[n1][0], nodes[n2][0], nodes[n3][0],
nodes[n1][0]]
            tri_y = [nodes[n1][1], nodes[n2][1], nodes[n3][1],
```

```
nodes[n1][1]]
            plt.plot(tri_x, tri_y, 'b-', linewidth=0.5, alpha=0.7)
            # 計算三個節點的幾何中心
            cx = (nodes[n1][0] + nodes[n2][0] + nodes[n3][0]) / 3
            cy = (nodes[n1][1] + nodes[n2][1] + nodes[n3][1]) / 3
            plt.text(cx, cy, str(idx+1), fontsize=10, color='purple',
ha='center', va='center', fontweight='bold')

        theta = np.linspace(0, np.pi / 2, 100)
        x_arc = np.cos(theta)
        y_arc = np.sin(theta)
        plt.plot(x_arc, y_arc, 'g-', label="Quarter Circle", zorder=1)

        plt.plot([0, 1], [0, 0], 'k--', linewidth=1)
        plt.plot([0, 0], [0, 1], 'k--', linewidth=1)
        plt.gca().set_aspect('equal')
        plt.grid(False)
        plt.xlabel("x")
        plt.ylabel("y")
        plt.legend()
        plt.show()
```

2.總體勁度矩陣 k

1	-0.5	0	0	0	-0.5	0	0	0	0	0	0	0	0	0	0	0	0	0
-0.5	2	-0.5	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0
0	-0.5	2	-0.5	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0
0	0	-0.5	2	-0.5	0	0	0	-1	0	0	0	0	0	0	0	0	0	0
0	0	0	-0.5	0.837554	0	0	0	-0.06448	0	0	0	0	0	0	0	0	0	-0.27308
-0.5	0	0	0	0	2	-1	0	0	-0.5	0	0	0	0	0	0	0	0	0
0	-1	0	0	0	-1	4	-1	0	0	-1	0	0	0	0	0	0	0	0
0	0	-1	0	0	0	-1	4	-1	0	0	-1	0	0	0	0	0	0	0
0	0	0	-1	-0.06448	0	0	-1	4.032893	0	0	-0.24138	0	0	0	0	0	-0.37259	-1.35444
0	0	0	0	0	-0.5	0	0	0	2	-1	0	-0.5	0	0	0	0	0	0
0	0	0	0	0	0	-1	0	0	-1	4	-1	0	-1	0	0	0	0	0
0	0	0	0	0	0	0	-1	-0.24138	0	-1	3.742841	0	-0.24138	0	0	-0.63004	-0.63004	0
0	0	0	0	0	0	0	0	0	-0.5	0	0	2	-1	-0.5	0	0	0	0
0	0	0	0	0	0	0	0	0	0	-1	-0.24138	-1	4.032893	-0.06448	-1.35444	-0.37259	0	0
0	0	0	0	0	0	0	0	0	0	0	0	-0.5	-0.06448	0.837554	-0.27308	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	-1.35444	-0.27308	1.874265	-0.24675	0	0
0	0	0	0	0	0	0	0	0	0	0	-0.63004	0	-0.37259	0	-0.24675	1.558478	-0.3091	0
0	0	0	0	0	0	0	0	-0.37259	0	0	-0.63004	0	0	0	0	-0.3091	1.558478	-0.24675
0	0	0	0	-0.27308	0	0	0	-1.35444	0	0	0	0	0	0	0	0	-0.24675	1.874265