中華民國第四十四屆中小學科學展覽會

作者說明書

高中組數學科

040418

國立新竹女子高級中學

指導老師姓名

徐以誠

作者姓名

劉育珍

龔怡嘉

具錯誤修正能力的高斯-喬登消去法

一、摘要

利用電腦程式做大量且密集運算的時候，會因為電腦硬體暫時性的失效而產生計算上的錯誤，為了減少重新計算所要花費的時間，並提高整體計算的穩定性與可靠度，具有錯誤偵測與錯誤修正能力的容錯計算方式是很重要的。本篇報告提出一種具有容錯能力之高斯-喬登消去法，藉由以直觀方式加入的檢查碼及修正的矩陣基本列運算，使得每個階段的運算過程中具有錯誤偵測與錯誤修正的功能，與以往的方法比較起來，多了容錯的能力，但是時間複雜度依然是相同的$O(n^3)$。

二、動機與目的

在二年級上學期學到以高斯消去法求線性聯立方程組的解，在經過大量的計算後，要驗證所求得的解是否正確，必須將解代回原線性聯立方程組來做檢驗，若是發現所得的解有錯誤，則需完全從頭再來過，這對人腦而言，真是一件沉重的負擔。若以電腦程式來解決此類問題，電腦硬體也會因暫時性的失效而產生計算上的錯誤，最後驗證時發現所得的解有誤，也得重頭再算一次。我們是否能夠找到一種方法，能夠在計算的過程中就發現錯誤，並將錯誤找出且更正它，然後從產生錯誤的地方繼續計算下去，那就太好了。

三、研究過程

3-1 線性方程式系統與矩陣基本列運算

由參考資料[4]得知，很多工程和科學上的問題，或是以數學方法解決社會科學問題時，或是商業及經濟問題的定量分析時，都會遭遇下列具有n個未知數$x_1,x_2,\cdots,x_n$的線性方程式系統

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

其中a_{ij}與b_i均為已知係數，i,j均為正整數，且$1\le i\le n$，$1\le j\le n$。此線性方程式系統可用矩陣的方式來表示

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

通常我們先將此線性方程式系統的增廣矩陣$M^{(0)}$如下列的方式寫出

$$M^{(0)}=\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right]$$

為了省掉反向取代求解的步驟，我們使用高斯-喬登消去法來轉換矩陣。首先，選擇 a_{11} 為基準元素，經過第 1 階段的矩陣基本列運算後，將矩陣 $M^{(0)}$ 轉換為 $M^{(1)}$，$M^{(1)}$ 的第 1 行元素中，除了基準元素 $a_{11}^{(1)}$ 外，其餘元素均為 0，形式如下

$$M^{(1)}=\left[\begin{array}{cccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1n}^{(1)} & b_1^{(1)} \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} & b_n^{(1)} \end{array}\right]$$

第 2 階段時，選擇 $a_{22}^{(1)}$ 為基準元素，經過矩陣基本列運算後，將 $M^{(1)}$ 轉換為 $M^{(2)}$，$M^{(2)}$ 的第 2 行元素中，除了基準元素 $a_{22}^{(2)}$ 外，其餘元素均為 0。以同樣的方式繼續下去，經過 n 個階段後，矩陣 $M^{(0)}$ 轉換為 $M^{(n)}$

$$M^{(n)}=\left[\begin{array}{cccc|c} a_{11}^{(n)} & 0 & \cdots & 0 & b_1^{(n)} \\ 0 & a_{22}^{(n)} & \cdots & 0 & b_2^{(n)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & a_{nn}^{(n)} & b_n^{(n)} \end{array}\right]$$

則此線性方程式系統的解如下，其中 i 為正整數，且 $1 \le i \le n$。

$$x_i=\frac{b_i^{(n)}}{a_{ii}^{(n)}}$$

在對增廣矩陣作轉換的時候，可供使矩陣的基本列運算有下列三種類型：

(1) $R_{ij}M$ 可將矩陣 M 第 i 列和第 j 列各元素互相對調。矩陣 M 經過如下列的運算後，第 1 列與第 2 列各元素會互相對調。

$$R_{12}=\begin{bmatrix} 0 & 1 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

(2) $R_i^k M$ 可將矩陣 M 第 i 列各元素均乘上一非零常數 k。矩陣 M 經過如下列的運算後，第 1 列各元素均會乘上常數 $1/a_{11}$。

$$R_1^{\frac{1}{a_{11}}}=\begin{bmatrix} \frac{1}{a_{11}} & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

(3) $R_{ij}^k M$ 可將矩陣 M 第 i 列各元素均乘上一非零常數 k 後，再加入第 j 列各相互對應的元素。矩陣 M 經過如下列的運算後，第 1 列各元素均乘上常數 $-a_{21}/a_{11}$ 後，再加入第 2 列各相互對應的元素中。

$$R_{12}^{\frac{-a_{21}}{a_{11}}}=\begin{bmatrix}1 & 0 & \cdots & 0\\ \frac{-a_{21}}{a_{11}} & 1 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 1\end{bmatrix}$$

3-2 容錯計算的概念與相關文獻簡述

利用高斯-喬登消去法來解線性方程式系統，僅可在最後求得解時，再將所得的解代回原線性方程式系統，來驗證所得的解是否正確。若是驗證時發現所得的解有錯誤，則需完全從頭來過，無法在計算的過程中就發現錯誤，將錯誤找出並更正它，然後能夠從產生錯誤的地方繼續計算下去。

由參考資料[5]、[6]、[7]、[8]知道，若是以電腦程式來解決此類問題，也會因電腦硬體暫時性的失效而產生計算上的錯誤。為了減少重新計算所要花費的時間，並提高整體計算的穩定性與可靠度，具有錯誤偵測與錯誤修正能力的計算方式是很重要的。

容錯計算的概念主要是在輸入的資料中再增加些額外的資訊，而原來的計算程序也需配合著輸入資料的改變做適當的修正，藉著這些額外的資訊得以在運算的過程中做錯誤偵測、錯誤定位、錯誤修正。參考資料[6]、[7]，分別提出了「列總和檢查碼」與「列加權總和檢查碼」的方法，在矩陣每一列的最右邊加入些檢查碼，可用來偵測在作反矩陣運算時是否有錯誤的發生。他們的方法可以偵測出運算時發生的錯誤是在哪一列，但無法正確的指出是哪一個元素有錯誤並且將之更正，整個運算必須暫停下來，然後重頭來過。

3-3 以直觀的方式加入檢查碼

「總和檢查碼」的觀念給了我們一個重要的啟示，既然在矩陣每一列的最右邊加入些檢查碼可以偵測出運算時發生的錯誤是在哪一列，那麼在矩陣每一行的最下邊也加入些檢查碼，不就可以正確的知道矩陣中是哪一列哪一行的元素發生錯誤了嗎？若是每列或每行的檢查碼在運算的過程中均維持是矩陣中每列或每行元素的總和，那麼將發生錯誤的元素修正將是可行的方法，線性方程式系統具有「列與行總和檢查碼」的增廣矩陣定義如下。

定義 1： 具有 n 個未知數 $x_1,x_2,\cdots,x_n$ 的線性方程式系統

$$\begin{array}{ccccccccc}a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1\\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2\\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots\\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n\end{array}$$

此線性方程式系統具有「列與行總和檢查碼」的增廣矩陣為 $M_{CS}^{(0)}$

$$M_{CS}^{(0)}=\begin{bmatrix}a_{11} & a_{12} & \cdots & a_{1n} & b_1 & \sum_{j=1}^{n}a_{1j}+b_1\\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 & \sum_{j=1}^{n}a_{2j}+b_2\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n & \sum_{j=1}^{n}a_{nj}+b_n\\ \sum_{i=1}^{n}a_{i1} & \sum_{i=1}^{n}a_{i2} & \cdots & \sum_{i=1}^{n}a_{in} & \sum_{k=1}^{n}b_k & \sum_{i=1}^{n}\sum_{j=1}^{n}a_{ij}+\sum_{k=1}^{n}b_k\end{bmatrix}$$

$M_{CS}^{(0)}$ 是一$(n+1)\times(n+2)$的矩陣，為了方便後面的說明，我們將 $M_{CS}^{(0)}$ 中第 i 列第 j 行的

元素改為a_{ij}來表示，其中i,j均為正整數，且$1\le i\le n+1$，$1\le j\le n+2$。

$$M_{CS}^{(0)}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & a_{1(n+1)} & a_{1(n+2)} \\ a_{21} & a_{22} & \cdots & a_{2n} & a_{2(n+1)} & a_{2(n+2)} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & a_{n(n+1)} & a_{n(n+2)} \\ a_{(n+1)1} & a_{(n+1)2} & \cdots & a_{(n+1)n} & a_{(n+1)(n+1)} & a_{(n+1)(n+2)} \end{bmatrix}$$

3-4 基本列運算的修正

我們試著在$M_{CS}^{(0)}$上使用傳統高斯-喬登消去法的矩陣基本列運算，其第 1 階段的過程如下

$$R_{1(n+1)}^{-\frac{a_{(n+1)1}}{a_{11}}}\ R_{1n}^{-\frac{a_{n1}}{a_{11}}}\cdots R_{12}^{-\frac{a_{21}}{a_{11}}}\ M_{CS}^{(0)}$$

由於矩陣的基本列運算具有合併性，上述過程可改為

$$\begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -\frac{a_{21}}{a_{11}} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\frac{a_{n1}}{a_{11}} & 0 & \cdots & 1 & 0 \\ -\frac{a_{(n+1)1}}{a_{11}} & 0 & \cdots & 0 & 1 \end{bmatrix} M_{CS}^{(0)}$$

運算後的結果$M_{CS}^{(1)}$如下

$$M_{CS}^{(1)}=\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n+1)} & a_{1(n+2)} \\ 0 & a_{22}-a_{12}\frac{a_{21}}{a_{11}} & \cdots & a_{2(n+1)}-a_{1(n+1)}\frac{a_{21}}{a_{11}} & a_{2(n+2)}-a_{1(n+2)}\frac{a_{21}}{a_{11}} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n2}-a_{12}\frac{a_{n1}}{a_{11}} & \cdots & a_{n(n+1)}-a_{1(n+1)}\frac{a_{n1}}{a_{11}} & a_{n(n+2)}-a_{1(n+2)}\frac{a_{n1}}{a_{11}} \\ 0 & a_{(n+1)2}-a_{12}\frac{a_{(n+1)1}}{a_{11}} & \cdots & a_{(n+1)(n+1)}-a_{1(n+1)}\frac{a_{(n+1)1}}{a_{11}} & a_{(n+1)(n+2)}-a_{1(n+2)}\frac{a_{(n+1)1}}{a_{11}} \end{bmatrix}$$

$$=\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & a_{1(n+1)}^{(1)} & a_{1(n+2)}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} & \cdots & a_{2(n+1)}^{(1)} & a_{2(n+2)}^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1}^{(1)} & a_{n2}^{(1)} & \cdots & a_{n(n+1)}^{(1)} & a_{n(n+2)}^{(1)} \\ a_{(n+1)1}^{(1)} & a_{(n+1)2}^{(1)} & \cdots & a_{(n+1)(n+1)}^{(1)} & a_{(n+1)(n+2)}^{(1)} \end{bmatrix}$$

檢驗$M_{CS}^{(1)}$中的$a_{i(n+2)}^{(1)}$，發現仍維持是第i列，除$a_{i(n+2)}^{(1)}$外其餘元素的總和，也就是說

$$a_{i1}^{(1)}+\cdots+a_{i(n+1)}^{(1)}=\sum_{k=1}^{n+1}a_{ik}^{(1)}=a_{i(n+2)}^{(1)}\qquad(1\le i\le n)$$

但是行的總和就無法維持了，$a_{(n+1)j}^{(1)}$不是第j行除$a_{(n+1)j}^{(1)}$外其餘元素的總和，也就是說

$$a_{1j}^{(1)}+\cdots+a_{nj}^{(1)}=\sum_{k=1}^{n}a_{kj}^{(1)}\ne a_{(n+1)j}^{(1)}\qquad(1\le j\le n+1)$$

雖然行的總和無法維持，可是我們察覺到一個現象，那就是

$$a_{2j}^{(1)}+a_{3j}^{(1)}+\cdots+a_{nj}^{(1)}=\sum_{k=2}^{n}a_{kj}^{(1)}=a_{(n+1)j}^{(1)}\qquad(1\le j\le n+1)$$

也就是說 $a^{(1)}_{(n+1)j}$（$1 \le j \le n+1$）是第 j 行除了 $a^{(1)}_{(n+1)j}$ 與 $a^{(1)}_{1j}$ 外，其餘元素的總和，這是一個很好的現象，我們只要再加上一個矩陣基本列運算 $R^1_{1(n+1)}$，如此可以將 $a^{(1)}_{1j}$ 加入 $a^{(1)}_{(n+1)j}$ 中，可能就可以達到我們的要求，修正後的第 1 階段運算過程如下

$$R^1_{1(n+1)} R_{1(n+1)}^{-\frac{a_{(n+1)1}}{a_{11}}} R_{1n}^{-\frac{a_{n1}}{a_{11}}} \cdots R_{12}^{-\frac{a_{21}}{a_{11}}} M^{(0)}_{CS} = R_{1(n+1)}^{1-\frac{a_{(n+1)1}}{a_{11}}} R_{1n}^{-\frac{a_{n1}}{a_{11}}} \cdots R_{12}^{-\frac{a_{21}}{a_{11}}} M^{(0)}_{CS}$$

$$= \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -\frac{a_{21}}{a_{11}} & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ -\frac{a_{n1}}{a_{11}} & 0 & \cdots & 1 & 0 \\ 1-\frac{a_{(n+1)1}}{a_{11}} & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n+2)} \\ a_{21} & a_{22} & \cdots & a_{2(n+2)} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{n(n+2)} \\ a_{(n+1)1} & a_{(n+1)2} & \cdots & a_{(n+1)(n+2)} \end{bmatrix}$$

運算後的結果 $M^{(1)}_{CS}$ 如下

$$M^{(1)}_{CS} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1(n+2)} \\ 0 & a_{22} - a_{12}\frac{a_{21}}{a_{11}} & \cdots & a_{2(n+2)} - a_{1(n+2)}\frac{a_{21}}{a_{11}} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2} - a_{12}\frac{a_{n1}}{a_{11}} & \cdots & a_{n(n+2)} - a_{1(n+2)}\frac{a_{n1}}{a_{11}} \\ a_{11} & a_{(n+1)2} - a_{12}(\frac{a_{(n+1)1}}{a_{11}} - 1) & \cdots & a_{(n+1)(n+2)} - a_{1(n+2)}(\frac{a_{(n+1)1}}{a_{11}} - 1) \end{bmatrix}$$

$$= \begin{bmatrix} a^{(1)}_{11} & a^{(1)}_{12} & \cdots & a^{(1)}_{1(n+2)} \\ a^{(1)}_{21} & a^{(1)}_{22} & \cdots & a^{(1)}_{2(n+2)} \\ \vdots & \vdots & \ddots & \vdots \\ a^{(1)}_{n1} & a^{(1)}_{n2} & \cdots & a^{(1)}_{n(n+2)} \\ a^{(1)}_{(n+1)1} & a^{(1)}_{(n+1)2} & \cdots & a^{(1)}_{(n+1)(n+2)} \end{bmatrix}$$

我們先檢驗 $a^{(1)}_{i(n+2)}$ 是否等於 $\sum_{k=1}^{n+1} a^{(1)}_{ik}$ （$1 \le i \le n$）

當 $i = 1$ 時，由於第 1 列的各元素的值都沒有改變，所以 $a^{(1)}_{1(n+2)} = \sum_{k=1}^{n+1} a^{(1)}_{1k}$

當 $2 \le i \le n$ 時

$$\begin{aligned} a^{(1)}_{i(n+2)} &= a_{i(n+2)} - a_{1(n+2)}\frac{a_{i1}}{a_{11}} \\ &= [a_{i1} + a_{i2} + a_{i3} + \cdots + a_{i(n+1)}] - \frac{a_{i1}}{a_{11}}[a_{11} + a_{12} + a_{13} + \cdots + a_{1(n+1)}] \\ &= [a_{i1} + a_{i2} + a_{i3} + \cdots + a_{i(n+1)}] - a_{i1} - \frac{a_{i1}}{a_{11}}[a_{12} + a_{13} + \cdots + a_{1(n+1)}] \\ &= [a_{i2} + a_{i3} + \cdots + a_{i(n+1)}] - \frac{a_{i1}}{a_{11}}[a_{12} + a_{13} + \cdots + a_{1(n+1)}] \\ &= 0 + [a_{i2} - \frac{a_{i1}}{a_{11}}a_{12}] + \cdots + [a_{i(n+1)} - \frac{a_{i1}}{a_{11}}a_{1(n+1)}] \\ &= \sum_{k=1}^{n+1} a^{(1)}_{ik} \end{aligned}$$

再來檢驗 $a^{(1)}_{(n+1)j}$ 是否等於 $\sum_{k=1}^{n} a^{(1)}_{kj}$ （$1 \le j \le n+1$）

$$
\begin{aligned}
a^{(1)}_{(n+1)j} &= a_{(n+1)j} - a_{1j}\left(\frac{a_{(n+1)1}}{a_{11}} - 1\right) \\
&= [a_{1j} + a_{2j} + a_{3j} + \cdots + a_{nj}] - \frac{a_{1j}}{a_{11}}[a_{11} + a_{21} + a_{31} + \cdots + a_{n1} - a_{11}] \\
&= a_{1j} + [a_{2j} + a_{3j} + \cdots + a_{nj}] - \frac{a_{1j}}{a_{11}}[a_{21} + a_{31} + \cdots + a_{n1}] \\
&= a_{1j} + [a_{2j} - \frac{a_{1j}}{a_{11}} a_{21}] + \cdots + [a_{nj} - \frac{a_{1j}}{a_{11}} a_{n1}] \\
&= \sum_{k=1}^{n} a^{(1)}_{kj}
\end{aligned}
$$

最後檢驗 $a^{(1)}_{(n+1)(n+2)}$ 是否等於 $\sum_{k=1}^{n+1} a^{(1)}_{(n+1)k}$ 及 $\sum_{k=1}^{n} a^{(1)}_{k(n+2)}$

$$
\begin{aligned}
a^{(1)}_{(n+1)(n+2)} &= a_{(n+1)(n+2)} - a_{1(n+2)}\left(\frac{a_{(n+1)1}}{a_{11}} - 1\right) \\
&= [a_{(n+1)1} + a_{(n+1)2} + \cdots + a_{(n+1)(n+1)}] - \frac{a_{(n+1)1}}{a_{11}}[a_{11} + a_{12} + \cdots + a_{1(n+1)}] + a_{1(n+2)} \\
&= [a_{(n+1)2} + \cdots + a_{(n+1)(n+1)}] - \frac{a_{(n+1)1}}{a_{11}}[a_{12} + \cdots + a_{1(n+1)}] + [a_{11} + a_{12} + \cdots + a_{1(n+1)}] \\
&= a_{11} + [a_{(n+1)2} + \cdots + a_{(n+1)(n+1)}] - \frac{a_{(n+1)1}}{a_{11}}[a_{12} + \cdots + a_{1(n+1)}] + [a_{12} + \cdots + a_{1(n+1)}] \\
&= a^{(1)}_{(n+1)1} + [a_{(n+1)2} - a_{12}\frac{a_{(n+1)1}}{a_{11}} + a_{12}] + \cdots + [a_{(n+1)(n+1)} - a_{1(n+1)}\frac{a_{(n+1)1}}{a_{11}} + a_{1(n+1)}] \\
&= a^{(1)}_{(n+1)1} + a^{(1)}_{(n+1)2} + \cdots + a^{(1)}_{(n+1)(n+1)} \\
&= \sum_{k=1}^{n+1} a^{(1)}_{(n+1)k}
\end{aligned}
$$

$$
\begin{aligned}
a^{(1)}_{(n+1)(n+2)} &= a_{(n+1)(n+2)} - a_{1(n+2)}\left(\frac{a_{(n+1)1}}{a_{11}} - 1\right) \\
&= a_{(n+1)(n+2)} - a_{(n+1)1}\frac{a_{1(n+2)}}{a_{11}} + a_{1(n+2)} \\
&= a_{1(n+2)} + [a_{1(n+2)} + a_{2(n+2)} + \cdots a_{n(n+2)}] - \frac{a_{1(n+2)}}{a_{11}}[a_{11} + a_{21} + \cdots + a_{n1}] \\
&= a_{1(n+2)} + [a_{2(n+2)} + \cdots + a_{n(n+2)}] - \frac{a_{1(n+2)}}{a_{11}}[a_{21} + \cdots + a_{n1}] \\
&= a_{1(n+2)} + [a_{2(n+2)} - a_{1(n+2)}\frac{a_{21}}{a_{11}}] + \cdots + [a_{n(n+2)} - a_{1(n+2)}\frac{a_{n1}}{a_{11}}] \\
&= a^{(1)}_{1(n+2)} + a^{(1)}_{2(n+2)} + \cdots + a^{(1)}_{n(n+2)} \\
&= \sum_{k=1}^{n} a^{(1)}_{k(n+2)}
\end{aligned}
$$

透過上述的檢驗，我們可以知道，$M^{(0)}_{CS}$ 經過第 1 階段修正後的矩陣基本列運算而轉換為 $M^{(1)}_{CS}$ 後，仍保有定義 1 中列與行總和檢查碼增廣矩陣的性質。

3-5 各階段矩陣轉換所使用的基本列運算

定義 2： $(n+1)\times(n+2)$ 的矩陣 $M^{(s-1)}_{CS}$，在第 s 階段轉換時，所使用的修正後矩陣基本列運算如下，其中 $s = 1,2,\cdots,n$。

$$
s=1 \qquad R^{1}_{1(n+1)} R_{1(n+1)}^{-\frac{a_{(n+1)1}}{a_{11}}} R_{1n}^{-\frac{a_{n1}}{a_{11}}} \cdots\cdots R_{12}^{-\frac{a_{21}}{a_{11}}}
$$

$$
s=2 \qquad R^{1}_{2(n+1)} R_{2(n+1)}^{-\frac{a^{(1)}_{(n+1)2}}{a^{(1)}_{22}}} R_{2n}^{-\frac{a^{(1)}_{n2}}{a^{(1)}_{22}}} \cdots\cdots R_{23}^{-\frac{a^{(1)}_{32}}{a^{(1)}_{22}}} R_{21}^{-\frac{a^{(1)}_{12}}{a^{(1)}_{22}}}
$$

$$\vdots$$

$$s=t \qquad R^{1}_{t(n+1)}R^{-\frac{a^{(t-1)}_{(n+1)t}}{a^{(t-1)}_{tt}}}_{t(n+1)}R^{-\frac{a^{(t-1)}_{nt}}{a^{(t-1)}_{tt}}}_{tn}\cdots\cdots R^{-\frac{a^{(t-1)}_{(t+1)t}}{a^{(t-1)}_{tt}}}_{t(t+1)}R^{-\frac{a^{(t-1)}_{(t-1)t}}{a^{(t-1)}_{tt}}}_{t(t-1)}\cdots\cdots R^{-\frac{a^{(t-1)}_{1t}}{a^{(t-1)}_{tt}}}_{t1}$$

$$\vdots$$

$$s=n-1 \qquad R^{1}_{(n-1)(n+1)}R^{-\frac{a^{(n-2)}_{(n+1)(n-1)}}{a^{(n-2)}_{(n-1)(n-1)}}}_{(n-1)(n+1)}R^{-\frac{a^{(n-2)}_{n(n-1)}}{a^{(n-2)}_{(n-1)(n-1)}}}_{(n-1)n}R^{-\frac{a^{(n-2)}_{(n-2)(n-1)}}{a^{(n-2)}_{(n-1)(n-1)}}}_{(n-1)(n-2)}\cdots\cdots R^{-\frac{a^{(n-2)}_{1(n-1)}}{a^{(n-2)}_{(n-1)(n-1)}}}_{(n-1)1}$$

$$s=n \qquad R^{1}_{n(n+1)}R^{-\frac{a^{(n-1)}_{(n+1)n}}{a^{(n-1)}_{nn}}}_{n(n+1)}R^{-\frac{a^{(n-1)}_{(n-1)n}}{a^{(n-1)}_{nn}}}_{n(n-1)}\cdots\cdots R^{-\frac{a^{(n-1)}_{1n}}{a^{(n-1)}_{nn}}}_{n1}$$

3-6 推展到最後階段

第 1 階段是成功了，但是接下來的階段呢？是否都能保有如定義 1 所述列與行總和檢查碼增廣矩陣的性質？從我們學過的證明方法中,「歸納法」可能是比較適合的方法,從第 1 階段、第 $s-1$階段，看看是否可以推展到第 s 階段。

定理 1： 具有 n 個未知數 $x_1,x_2,\cdots,x_n$ 的線性方程式系統，其 $M^{(s-1)}_{CS}$ 在第 s 階段，使用定義 2 中的矩陣基本列運算轉換後，仍保有如定義 1 所述列與行總和檢查碼增廣矩陣的性質，其中 $s=1,2,\cdots,n$。

證明 ：我們使用歸納法來證明，$s=1$的時候，前面已經檢驗過，可以符合定義 1 所述列與行總和檢查碼增廣矩陣的性質。假設第 $s-1$階段時，$M^{(s-2)}_{CS}$ 經過定義 2 中的矩陣基本列運算轉換為 $M^{(s-1)}_{CS}$ 後，保有下列如定義 1 中所述的性質

$$a^{(s-1)}_{i(n+2)}=\sum_{k=1}^{n+1}a^{(s-1)}_{ik} \qquad (1\le i\le n)$$

$$a^{(s-1)}_{(n+1)j}=\sum_{k=1}^{n}a^{(s-1)}_{kj} \qquad (1\le j\le n+1)$$

$$a^{(s-1)}_{(n+1)(n+2)}=\sum_{k=1}^{n+1}a^{(s-1)}_{(n+1)k}=\sum_{k=1}^{n}a^{(s-1)}_{k(n+2)}$$

現在我們來證明第 s 階段，$M^{(s-1)}_{CS}$ 轉換為 $M^{(s)}_{CS}$ 後，仍保有如定義 1 中所述的性質

第 s 階段的運算過程如下

$$M^{(s)}_{CS}=R^{1}_{s(n+1)}R^{-\frac{a^{(s-1)}_{(n+1)s}}{a^{(s-1)}_{ss}}}_{s(n+1)}\cdots\cdots R^{-\frac{a^{(s-1)}_{(s+1)s}}{a^{(s-1)}_{ss}}}_{s(s+1)}R^{-\frac{a^{(s-1)}_{(s-1)s}}{a^{(s-1)}_{ss}}}_{s(s-1)}\cdots\cdots R^{-\frac{a^{(s-1)}_{1s}}{a^{(s-1)}_{ss}}}_{s1}M^{(s-1)}_{CS}$$

$$=\begin{bmatrix}
1 & 0 & \cdots & 0 & -\frac{a^{(s-1)}_{1s}}{a^{(s-1)}_{ss}} & 0 & \cdots & \cdots & 0\\
0 & \ddots & \ddots & \vdots & \vdots & \vdots & & & \vdots\\
\vdots & \ddots & \ddots & 0 & \vdots & \vdots & & & \vdots\\
\vdots & & \ddots & 1 & \vdots & \vdots & & & \vdots\\
\vdots & & & 0 & 1 & 0 & & & \vdots\\
\vdots & & & \vdots & \vdots & 1 & \ddots & & \vdots\\
\vdots & & & \vdots & \vdots & 0 & \ddots & \ddots & \vdots\\
\vdots & & & \vdots & \vdots & \vdots & \ddots & \ddots & 0\\
0 & \cdots & \cdots & 0 & 1-\frac{a^{(s-1)}_{(n+1)s}}{a^{(s-1)}_{ss}} & 0 & \cdots & 0 & 1
\end{bmatrix}M^{(s-1)}_{CS}$$

$$=\begin{bmatrix} a_{11}^{(s-1)}-a_{s1}^{(s-1)}\frac{a_{1s}^{(s-1)}}{a_{ss}^{(s-1)}} & \cdots & 0 & \cdots & a_{1(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}\frac{a_{1s}^{(s-1)}}{a_{ss}^{(s-1)}} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{s1}^{(s-1)} & \cdots & a_{ss}^{(s-1)} & \cdots & a_{s(n+2)}^{(s-1)} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1}^{(s-1)}-a_{s1}^{(s-1)}\frac{a_{ns}^{(s-1)}}{a_{ss}^{(s-1)}} & \cdots & 0 & \cdots & a_{n(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}\frac{a_{ns}^{(s-1)}}{a_{ss}^{(s-1)}} \\ a_{(n+1)1}^{(s-1)}-a_{s1}^{(s-1)}(\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}-1) & \cdots & a_{ss}^{(s-1)} & \cdots & a_{(n+1)(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}(\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}-1) \end{bmatrix}$$

$$=\begin{bmatrix} a_{11}^{(s)} & \cdots & a_{1s}^{(s)} & \cdots & a_{1(n+2)}^{(s)} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{s1}^{(s)} & \cdots & a_{ss}^{(s)} & \cdots & a_{s(n+2)}^{(s)} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1}^{(s)} & \cdots & a_{ns}^{(s)} & \cdots & a_{n(n+2)}^{(s)} \\ a_{(n+1)1}^{(s)} & \cdots & a_{(n+1)s}^{(s)} & \cdots & a_{(n+1)(n+2)}^{(s)} \end{bmatrix}$$

檢驗 $a_{i(n+2)}^{(s)}=\sum_{k=1}^{n+1}a_{ik}^{(s)} \quad (1\le i\le n)$

當 $i=s$ 時, $a_{s(n+2)}^{(s)}=a_{s(n+2)}^{(s-1)}=\sum_{k=1}^{n+1}a_{sk}^{(s-1)}=\sum_{k=1}^{n+1}a_{sk}^{(s)}$

當 $i\ne s$ 時,

$$a_{i(n+2)}^{(s)} = a_{i(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}$$

$$= [a_{i1}^{(s-1)}+a_{i2}^{(s-1)}+\cdots+a_{i(n+1)}^{(s-1)}]-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{s1}^{(s-1)}+\cdots+a_{ss}^{(s-1)}+\cdots+a_{s(n+1)}^{(s-1)}]$$

$$= [a_{i1}^{(s-1)}+\cdots+a_{i(n+1)}^{(s-1)}]-a_{is}^{(s-1)}-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{s1}^{(s-1)}+\cdots+a_{s(s-1)}^{(s-1)}+a_{s(s+1)}^{(s-1)}+\cdots+a_{s(n+1)}^{(s-1)}]$$

$$= [a_{i1}^{(s-1)}-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s1}^{(s-1)}]+\cdots+[a_{i(s-1)}^{(s-1)}-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(s-1)}^{(s-1)}]+0+[a_{i(s+1)}^{(s-1)}-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(s+1)}^{(s-1)}]+\cdots+$$

$$[a_{i(n+1)}^{(s-1)}-\frac{a_{is}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+1)}^{(s-1)}]$$

$$= a_{i1}^{(s)}+\cdots+a_{i(s-1)}^{(s)}+a_{is}^{(s)}+a_{i(s+1)}^{(s)}\cdots+a_{i(n+1)}^{(s)}$$

$$= \sum_{k=1}^{n+1}a_{ik}^{(s)}$$

檢驗 $a_{(n+1)j}^{(s)}=\sum_{k=1}^{n}a_{kj}^{(s)} \quad (1\le j\le n+1)$

$$a_{(n+1)j}^{(s)} = a_{(n+1)j}^{(s-1)}-a_{sj}^{(s-1)}(\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}-1)$$

$$= [a_{1j}^{(s-1)}+\cdots+a_{sj}^{(s-1)}+\cdots+a_{nj}^{(s-1)}]-\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{1s}^{(s-1)}+\cdots+a_{ss}^{(s-1)}+\cdots+a_{ns}^{(s-1)}-a_{ss}^{(s-1)}]$$

$$= a_{sj}^{(s-1)}+[a_{1j}^{(s-1)}+\cdots a_{(s-1)j}^{(s-1)}+a_{(s+1)j}^{(s-1)}+\cdots+a_{nj}^{(s-1)}]-$$

$$\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{1s}^{(s-1)}+\cdots+a_{(s-1)s}^{(s-1)}+a_{(s+1)s}^{(s-1)}+\cdots+a_{ns}^{(s-1)}]$$

$$= [a_{1j}^{(s-1)}-\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}a_{1s}^{(s-1)}]+\cdots+[a_{(s-1)j}^{(s-1)}-\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}a_{(s-1)s}^{(s-1)}]+a_{sj}^{(s-1)}+[a_{(s+1)j}^{(s-1)}-\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}a_{(s+1)s}^{(s-1)}]+\cdots+$$

$$[a_{nj}^{(s-1)}-\frac{a_{sj}^{(s-1)}}{a_{ss}^{(s-1)}}a_{ns}^{(s-1)}]$$

$$= a_{1j}^{(s)}+\cdots+a_{(s-1)j}^{(s)}+a_{sj}^{(s)}+a_{(s+1)j}^{(s)}\cdots+a_{nj}^{(s)}$$

$$= \sum_{k=1}^{n}a_{kj}^{(s)}$$

檢驗 $a_{(n+1)(n+2)}^{(s)}=\sum_{k=1}^{n+1}a_{(n+1)k}^{(s)}=\sum_{k=1}^{n}a_{k(n+2)}^{(s-1)}$

$$a_{(n+1)(n+2)}^{(s)} = a_{(n+1)(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}(\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}-1)$$

$$= [a_{(n+1)1}^{(s-1)}+\cdots+a_{(n+1)(n+1)}^{(s-1)}]-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{s1}^{(s-1)}+\cdots+a_{s(n+1)}^{(s-1)}]+a_{s(n+2)}^{(s-1)}$$

$$= [a_{(n+1)1}^{(s-1)}+\cdots+a_{(n+1)(s-1)}^{(s-1)}+a_{(n+1)(s+1)}^{(s-1)}+\cdots+a_{(n+1)(n+1)}^{(s-1)}]-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{s1}^{(s-1)}+\cdots+a_{s(s-1)}^{(s-1)}+$$

$$a_{s(s+1)}^{(s-1)}+\cdots+a_{s(n+1)}^{(s-1)}]+a_{ss}^{(s-1)}+[a_{s1}^{(s-1)}+\cdots+a_{s(s-1)}^{(s-1)}+a_{s(s+1)}^{(s-1)}+\cdots+a_{s(n+1)}^{(s-1)}]$$

$$= [a_{(n+1)1}^{(s-1)}-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s1}^{(s-1)}+a_{s1}^{(s-1)}]+\cdots+[a_{(n+1)(s-1)}^{(s-1)}-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(s-1)}^{(s-1)}+a_{s(s-1)}^{(s-1)}]+a_{ss}^{(s-1)}+$$

$$[a_{(n+1)(s+1)}^{(s-1)}-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(s+1)}^{(s-1)}+a_{s(s+1)}^{(s-1)}]+\cdots+[a_{(n+1)(n+1)}^{(s-1)}-\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+1)}^{(s-1)}+a_{s(n+1)}^{(s-1)}]$$

$$= a_{(n+1)1}^{(s)}+\cdots+a_{(n+1)(s-1)}^{(s)}+a_{(n+1)s}^{(s)}+a_{(n+1)(s+1)}^{(s)}+\cdots+a_{(n+1)(n+1)}^{(s)}$$

$$= \sum_{k=1}^{n+1}a_{(n+1)k}^{(s)}$$

$$a_{(n+1)(n+2)}^{(s)} = a_{(n+1)(n+2)}^{(s-1)}-a_{s(n+2)}^{(s-1)}(\frac{a_{(n+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}-1)$$

$$= [a_{1(n+2)}^{(s-1)}+\cdots+a_{n(n+2)}^{(s-1)}]-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}[a_{1s}^{(s-1)}+\cdots+a_{ns}^{(s-1)}]+a_{s(n+2)}^{(s-1)}$$

$$= [a_{1(n+2)}^{(s-1)}-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}a_{1s}^{(s-1)}]+\cdots+[a_{(s-1)(n+2)}^{(s-1)}-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}a_{(s-1)s}^{(s-1)}]+[a_{s(n+2)}^{(s-1)}-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}a_{ss}^{(s-1)}+$$

$$+a_{s(n+2)}^{(s-1)}]+[a_{(s+1)(n+2)}^{(s-1)}-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}a_{(s+1)s}^{(s-1)}]+\cdots+[a_{n(n+2)}^{(s-1)}-\frac{a_{s(n+2)}^{(s-1)}}{a_{ss}^{(s-1)}}a_{ns}^{(s-1)}]$$

$$= [a_{1(n+2)}^{(s-1)}-\frac{a_{1s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+2)}^{(s-1)}]+\cdots+[a_{(s-1)(n+2)}^{(s-1)}-\frac{a_{(s-1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+2)}^{(s-1)}]+[a_{s(n+2)}^{(s-1)}]+$$

$$[a_{(s+1)(n+2)}^{(s-1)}-\frac{a_{(s+1)s}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+2)}^{(s-1)}]+\cdots+[a_{n(n+2)}^{(s-1)}-\frac{a_{ns}^{(s-1)}}{a_{ss}^{(s-1)}}a_{s(n+2)}^{(s-1)}]$$

$$= a^{(s)}_{1(n+2)} + \cdots + a^{(s)}_{(s-1)(n+2)} + a^{(s)}_{s(n+2)} + a^{(s)}_{(s+1)(n+2)} + \cdots + a^{(s)}_{n(n+2)}$$

$$= \sum_{k=1}^{n} a^{(s-1)}_{k(n+2)}$$

3-7 其他情況的處理

定理 1 確保了我們修正後的矩陣基本列運算可於每階段的轉換後，列與行總和檢查碼增廣矩陣仍保有如定義 1 所述的性質，這也意味著在運算過程中具有錯誤偵測與錯誤修正能力的線性方程式系統解法是可以做得到的，此外，我們尚須思考下列三種情況時的處理方法，使得整個運作模式更為完備。

(1)選擇的基準元素等於 0 的時候

當選擇的基準元素等於 0 的時候，會有除數為 0 的的情形。另外，若是使用太小的基準元素，將會造成嚴重的電腦系統捨位誤差（roundoff error)。根據參考資料[4]，我們可於每階段的矩陣基本列運算轉換前，檢視基準元素所在行由基準元素開始往下的各元素，將含絕對值最大元素的那一列與基準元素所在列交換，來克服上述的問題。

(2)線性方程式系統沒有唯一解的時候

當我們選擇絕對值最大的基準元素時候，若是可供選取的元素均為 0，那就是代表本線性方程式系統沒有唯一解，有可能是無解，也有可能有無限多解。參考資料[2]中，有說明利用矩陣的秩（rank）來判別線性方程式系統是無解或是無限多解，但是我們並不去探討這個問題，碰到上述情況，就以「本線性方程式系統沒有唯一解」做結束。

(3)計算時錯誤偵測與錯誤修正的方法

我們可於每個階段的矩陣基本列運算完成後，檢驗「列與行總和檢查碼」增廣矩陣中每列或每行的檢查碼是否等於該列或該行其他各元素的總和。若是有一個非檢查碼的元素錯誤，則該元素所在列與所在行的檢查碼在檢驗時均會發現錯誤，因此可將該元素的位置偵測出來，並將之更正。若是錯誤發生在檢查碼呢？檢驗時只有該檢查碼發現錯誤，只要將該檢查碼更正即可。

四、研究結果與實例

4-1 演算法運作流程

我們的方法主要分為下列四個部分：

(1)線性方程式系統「列與行總和檢查碼」增廣矩陣的產生

分別計算每一列與每一行各元素的總和，並將之填入該列與該行檢查碼的位置，藉以產生線性方程式系統的「列與行總和檢查碼」增廣矩陣。

(2)選擇最大基準元素並進行行列交換

為了避免基準元素等於 0 及產生嚴重的捨位誤差，我們由可供選擇的元素中，選擇絕對值最大的元素當做新的基準元素，並將該列與原來的基準元素所在列交換。若是可供選擇的元素均為 0，就顯示「本線性方程式系統沒有唯一解」，並結束程式的運作。

(3)使用基本列運算轉換矩陣

使用我們修正後的矩陣基本列運算來轉換線性方程式系統的「列與行總和檢查碼」增廣矩陣，轉換後的矩陣，各檢查碼仍保有其性質。

(4)進行錯誤偵測與錯誤修正

檢驗「列與行總和檢查碼」增廣矩陣中每列或每行的檢查碼是否等於該列或該行其他各元素的總和，若有錯誤的發生，則將之偵測出來並更正。本項檢驗需於每階段矩陣轉換完成後施行，因為一個錯誤的元素，會在下一個階段轉換時，造成多個元素的錯誤。

整個具有錯誤偵測與錯誤修正能力之線性方程式系統解法的運作流程，可用下面的 Modified Fault Tolerant Gauss Jordan（MFTGJ）演算法來表示，輸入的資料包含 M 與 n，M 代表線性方程式系統的增廣矩陣，n 表示線性方程式系統的未知數及方程式數目。

Procedure MFTGJ (M,n)

/* M 為線性方程式系統的增廣矩陣
n 為線性方程式系統的未知數及方程式數目 */

begin

(1) 產生線性方程式系統的「列與行總和檢查碼」增廣矩陣 $M_{CS}^{(0)}$ ；

for s = 1 to n

begin

(2) 在 $M_{CS}^{(s-1)}$ 中選擇最大基準元素並進行行列交換，若是可供選擇的元素均為 0，則本線性方程式系統沒有唯一解並停止程式；

(3) 使用基本列運算轉換矩陣 $M_{CS}^{(s-1)}$ 為 $M_{CS}^{(s)}$ ；

(4) 對 $M_{CS}^{(s)}$ 進行錯誤偵測與錯誤修正，若只有一個錯誤則更正此錯誤，若是超過一個以上的錯誤，則需重新計算；

end

(5) 由 $M_{CS}^{(n)}$ 中取得線性方程式系統的解；

end

4-2 一個簡單的範例

求下列線性方程式系統的解？

$$\begin{aligned} 3x_1 + x_2 + 3x_3 &= 14 \\ 2x_1 + x_2 + 3x_3 &= 12 \\ x_1 + x_2 + x_3 &= 6 \end{aligned}$$

產生本線性方程式系統的「列與行總和檢查碼」增廣矩陣 $M_{CS}^{(0)}$

$$M_{CS}^{(0)} = \begin{bmatrix} 3 & 1 & 3 & 14 & 21 \\ 2 & 1 & 3 & 12 & 18 \\ 1 & 1 & 1 & 6 & 9 \\ 6 & 3 & 7 & 32 & 48 \end{bmatrix}$$

選擇第 1 階段最大基準元素 3，不用進行列交換

$$M_{CS}^{(0)} = \begin{bmatrix} 3 & 1 & 3 & 14 & 21 \\ 2 & 1 & 3 & 12 & 18 \\ 1 & 1 & 1 & 6 & 9 \\ 6 & 3 & 7 & 32 & 48 \end{bmatrix}$$

準備第 1 階段轉換的矩陣基本列運算

$$R_{14}^{1}R_{14}^{-\frac{6}{3}}R_{13}^{-\frac{1}{3}}R_{12}^{-\frac{2}{3}}=\begin{bmatrix}1&0&0&0\\-\frac{2}{3}&1&0&0\\-\frac{1}{3}&0&1&0\\1-\frac{6}{3}&0&0&1\end{bmatrix}$$

進行第 1 階段的矩陣轉換

$$M_{CS}^{(1)}=\begin{bmatrix}1&0&0&0\\-\frac{2}{3}&1&0&0\\-\frac{1}{3}&0&1&0\\1-\frac{6}{3}&0&0&1\end{bmatrix}\begin{bmatrix}3&1&3&14&21\\2&1&3&12&18\\1&1&1&6&9\\6&3&7&32&48\end{bmatrix}=\begin{bmatrix}3&1&3&14&21\\0&\frac{1}{3}&1&\frac{8}{3}&4\\0&\frac{2}{3}&0&\frac{4}{3}&2\\3&2&4&18&27\end{bmatrix}$$

檢驗 $M_{CS}^{(1)}$ 中的檢查碼是否等於該列或該行其他各元素的總和

經檢驗無誤

選擇第 2 階段最大基準元素 $\frac{2}{3}$，並進行列交換 $R_{23}M_{CS}^{(1)}$，經過列交換的 $M_{CS}^{(1)}$ 如下

$$M_{CS}^{(1)}=\begin{bmatrix}3&1&3&14&21\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&\frac{1}{3}&1&\frac{8}{3}&4\\3&2&4&18&27\end{bmatrix}$$

準備第 2 階段轉換的矩陣基本列運算

$$R_{24}^{1}R_{24}^{-\frac{2}{\frac{2}{3}}}R_{23}^{-\frac{\frac{1}{3}}{\frac{2}{3}}}R_{21}^{-\frac{1}{\frac{2}{3}}}=\begin{bmatrix}1&-\frac{3}{2}&0&0\\0&1&0&0\\0&-\frac{1}{2}&1&0\\0&1-3&0&1\end{bmatrix}$$

進行第 2 階段的矩陣轉換

$$M_{CS}^{(2)}=\begin{bmatrix}1&-\frac{3}{2}&0&0\\0&1&0&0\\0&-\frac{1}{2}&1&0\\0&1-3&0&1\end{bmatrix}\begin{bmatrix}3&1&3&14&21\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&\frac{1}{3}&1&\frac{8}{3}&4\\3&2&4&18&27\end{bmatrix}=\begin{bmatrix}3&0&3&12&18\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&2&3\\3&\frac{2}{3}&4&\frac{46}{3}&23\end{bmatrix}$$

檢驗 $M_{CS}^{(2)}$ 中的檢查碼是否等於該列或該行其他各元素的總和

經檢驗無誤

選擇第 3 階段最大基準元素 1，不用進行列交換

$$M_{CS}^{(2)}=\begin{bmatrix}3&0&3&12&18\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&2&3\\3&\frac{2}{3}&4&\frac{46}{3}&23\end{bmatrix}$$

準備第 3 階段轉換的矩陣基本列運算

$$R_{34}^{1}R_{34}^{-\frac{4}{1}}R_{32}^{-\frac{0}{1}}R_{31}^{-\frac{3}{1}}=\begin{bmatrix}1&0&-3&0\\0&1&0&0\\0&0&1&0\\0&0&1-4&1\end{bmatrix}$$

進行第 3 階段的矩陣轉換

$$M_{CS}^{(3)}=\begin{bmatrix}1&0&-3&0\\0&1&0&0\\0&0&1&0\\0&0&_{1-4}&1\end{bmatrix}\begin{bmatrix}3&0&3&12&18\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&2&3\\3&\frac{2}{3}&4&\frac{46}{3}&23\end{bmatrix}=\begin{bmatrix}3&0&0&6&9\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&2&3\\3&\frac{2}{3}&1&\frac{28}{3}&14\end{bmatrix}$$

檢驗 $M_{CS}^{(3)}$ 中的檢查碼是否等於該列或該行其他各元素的總和

經檢驗無誤

此線性方程式系統的解為

$$x_1=\frac{6}{3}=2,\quad x_2=\frac{\frac{4}{3}}{\frac{2}{3}}=2,\quad x_3=\frac{2}{1}=2$$

4-3 錯誤偵測與錯誤修正範例

承上例，假設經過第 2 階段的矩陣轉換後， $M_{CS}^{(2)}$ 中 $a_{34}^{(2)}$ 產生了錯誤（正確為 $a_{34}^{(2)}=2$）

$$M_{CS}^{(2)}=\begin{bmatrix}3&0&3&12&18\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&3&3\\3&\frac{2}{3}&4&\frac{46}{3}&23\end{bmatrix}$$

檢驗 $M_{CS}^{(2)}$ 中的檢查碼是否等於該列或該行其他各元素的總和時，發現

$$RowSum(3)=a_{35}^{(2)}-[a_{31}^{(2)}+a_{32}^{(2)}+_{33}^{(2)}+a_{34}^{(2)}]=3-[0+0+1+3]=-1\neq 0$$

$$ColSum(4)=a_{44}^{(2)}-[a_{14}^{(2)}+a_{24}^{(2)}+_{34}^{(2)}]=\frac{46}{3}-[12+\frac{4}{3}+3]=-1\neq 0$$

所以我們知道 $a_{34}^{(2)}$ 產生了錯誤，現將其更正

$$a_{34}^{(2)}=a_{34}^{(2)}+RowSum(3)=3+(-1)=2$$

更正後如下

$$M_{CS}^{(2)}=\begin{bmatrix}3&0&3&12&18\\0&\frac{2}{3}&0&\frac{4}{3}&2\\0&0&1&2&3\\3&\frac{2}{3}&4&\frac{46}{3}&23\end{bmatrix}$$

4-4 程式實作

我們使用 Visual Basic 6.0 將「具錯誤修正能力的高斯-喬登消去法」實作，其主要運作的子程序程式碼如下：

```
Public Sub MFTGJ (M() As Double, ByVal n As Integer)
    Const Zero As Double = 0.0000000001
    Dim Mcs(), RowSum(), ColSum() As Double
    Dim McsRow, McsColum As Integer
    Dim RowFaultPosition, ColFaultPosition, RowFaultNumber, ColFaultNumber As Integer
    Dim temp As Double
    Dim s, i, j, MaxPivot As Integer
    McsRow = n+1
    McsColum = n+2
    Redim Mcs(1 To McsRow, 1 To McsColum)
    Redim RowSum(McsRow)
    Redim ColSum(McsColum)
    For i = 1 To n + 1
        RowSum(i) = 0
    Next
    For j = 1 To n + 2
        ColSum(j) = 0
    Next
```

```
‘產生列與行總和檢查碼增廣矩陣
For i = 1 To n + 1
    For j = 1 To n + 2
        If i = n + 1 Then
            Mcs(i, j) = ColSum(j)
        Else
            If j = n + 2 Then
                Mcs(i, j) = RowSum(i)
                ColSum(j) = ColSum(j) + RowSum(i)
            Else
                Mcs(i, j) = M(i, j)
                RowSum(i) = RowSum(i) + M(i, j)
                ColSum(j) = ColSum(j) + M(i, j)
            End If
        End If
    Next
Next
‘n 個階段的矩陣轉換、錯誤偵測與更正
For s = 1 To n
    ‘選擇最大基準元素並進行行列交換
    MaxPovit = s
    For i = s + 1 To n
        If Abs(Mcs(i, s)) > Abs(Mcs(MaxPovit, s)) Then MaxPovit = i
    Next
    If Abs(Mcs(MaxPovit, s)) < Zero Then
        Print “本線性方程式系統沒有唯一解！”
        Exit Sub
    Else
        If MaxPovit <> s Then ‘列交換
            For i = s To n + 2
                temp = Mcs(s, i)
                Mcs(s, i) = Mcs(MaxPovit, i)
                Mcs(MaxPovit, i) = temp
            Next
        End If
    End If
    ‘使用基本列運算轉換矩陣
    For i = 1 To n + 1
        If i < s Then
            RowSum(i) = Mcs(i, i)
        Else
            RowSum(i) = 0
        End If
    Next
    For j = 1 To n + 2
        If j < s Then
            ColSum(j) = Mcs(j, j)
        Else
            ColSum(j) = 0
        End If
    Next
    For i = 1 To n + 1
        For j = n + 2 To s Step –1
            Select Case i
                Case s
                    ‘不做任何事
                Case n + 1
                    Mcs(i, j) = Mcs(i, j) – Mcs(s, j) * (Mcs(i, s) / Mcs(s, s) – 1)
                Case Else
                    Mcs(i, j) = Mcs(i, j) – Mcs(s, j) * Mcs(i, s) / Mcs(s, s)
            End Select
            If (i <> n+1) And (j <> n+2) Then ‘累加轉換後的新矩陣各列與各行之和
                RowSum(i) = RowSum(i) + Mcs(i, j)
                ColSum(j) = ColSum(j) + Mcs(i, j)
            End If
        Next
```

```
            Next
            ‘錯誤偵測與錯誤更正
            RowFaultNumber = 0 ‘紀錄列總和檢查碼發現錯誤的數目
            ColFaultNumber = 0 ‘紀錄行總和檢查碼發現錯誤的數目
            For i = 1 To n
                  RowSum(i) = Mcs(i, n+2) – RowSum(i)
                  If Abs(RowSum(i)) > Zero Then
                        RowFaultNumber = RowFaultNumber + 1 ‘紀錄列總和檢查碼發現錯誤的數目
                        RowFaultPosition = i ‘紀錄發生錯誤的列位置
                  End If
            Next
            For j = 1 To n + 1
                  ColSum(j) = Mcs(n+1,j) – ColSum(j)
                  If Abs(ColSum(j)) > Zero Then
                        ColFaultNumber = ColFaultNumber + 1 ‘紀錄行總和檢查碼發現錯誤的數目
                        ColFaultPosition = j ‘紀錄發生錯誤的行位置
                  End If
            Next
            If (RowFaultNumber >= 2) Or (ColFaultNumber >= 2) Then
                  Print “無法更正超過 1 個以上的錯誤，必須重新計算！”
                  Exit Sub
            End If
            If (RowFaultNumber = 1) And (ColFaultNumber = 0) Then ‘列檢查碼錯誤
                  Mcs(RowFaultPosition, n + 2) = Mcs(RowFaultPosition, n + 2) – RowSum(RowFaultPosition)
            End If
            If (RowFaultNumber = 0) And (ColFaultNumber = 1) Then ‘行檢查碼錯誤
                  Mcs(n + 1, ColFaultPosition) = Mcs(n + 1, ColFaultPosition) – ColSum(ColFaultPosition)
            End If
            If (RowFaultNumber = 1) And (ColFaultNumber = 1) Then ‘非檢查碼元素錯誤
                  Mcs(RowFaultPosition, ColFaultPosition)= Mcs(RowFaultPosition, ColFaultPosition) + RowSum(RowFaultPosition)
            End If
      Next
      ‘取得線性方程式系統的解
      For i = 1 To n
            Print “x(“ & i & “) = “ & Mcs(i, n + 1) / Mcs(i, i)
      Next
End Sub
```

五、分析與討論

5-1 時間複雜度分析

我們可將演算法分為下列五個部分來分析，整個程式的時間複雜度為$O(n^3)$，就時間複雜度來說，我們的方法與傳統的高斯-喬登消去法是相同的。

(1)在產生列與行總和檢查碼增廣矩陣的部分

需$n\times(n+1)$個運算步驟，其時間複雜度為$O(n^2)$。

(2)選擇最大基準元素並進行列交換部分

每階段最多需要$(n-1)$個選擇最大基準元素與$(n+2)$個列元素交換步驟，共有n個階段，故時間複雜度為$O(n^2)$。

(3)使用基本列運算轉換矩陣部分

每階段需$n\times(n+2)$個運算步驟，共有n個階段，其時間複雜度為$O(n^3)$。

(4)進行錯誤偵測與錯誤修正部分

每階段至少需要$n\times(n+1)$個錯誤偵測的運算步驟，共有n個階段，其時間複雜度為$O(n^3)$。

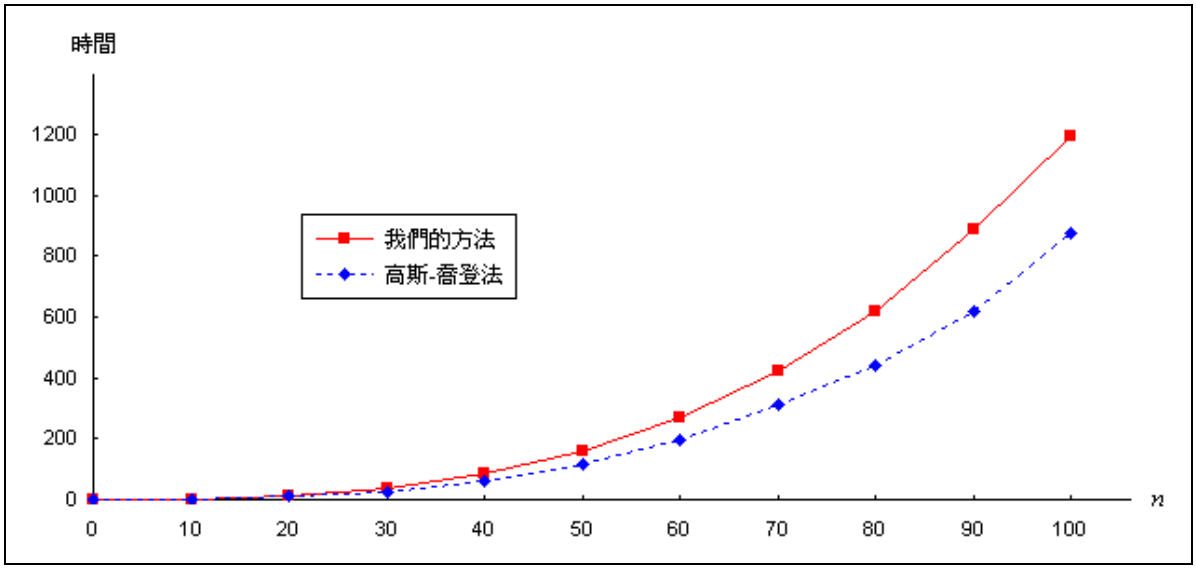
(5)取得線性方程式系統的解部分

需n個運算步驟，其時間複雜度為$O(n)$。

時間複雜度分析完成後，我們利用亂數來產生具有 n 個未知數的線性方程式系統

（$n=10,20,\cdots,100$），每個 n 值各有 100 組，分別交由傳統的高斯－喬登法與我們的方法去處理，計算出每一組線性方程式系統平均所需花費的計算時間，數據資料與折線圖整理如下（表中時間單位為1×10^{-3}秒）。

方法 \ n	10	20	30	40	50	60	70	80	90	100
高斯-喬登法	1.5	9.3	27.3	61.3	116.1	197.4	312.1	439.5	620.1	874.2
具容錯之高斯-喬登法	2.13	12.5	37.4	85.3	161.4	270.6	420.9	618.8	887.9	1196.6
額外花費時間的比例	42.0%	34.4%	37.0%	39.2%	39.0%	37.1%	34.9%	40.8%	43.2%	36.9%



5-2 錯誤偵測與錯誤修正能力的探討

我們的方法可於每個階段的運算中偵測並且更正一個錯誤，對於超過一個以上的錯誤僅能偵測出有錯誤的發生但無法更正，下面的例子中均有兩個元素發生錯誤，其中$\times_{ij}$表示第 i 列第 j 行錯誤的元素，我們的方法可以偵測出有錯誤的發生，但無法分辨出是$\times_{11}$與$\times_{22}$發生錯誤，還是$\times_{12}$與$\times_{21}$發生錯誤，故無法將錯誤修正。

$$\begin{bmatrix} \times_{11} & & \times_{1(n+2)} \\ & \times_{22} & \times_{2(n+2)} \\ & & \\ \times_{(n+1)1} & \times_{(n+1)2} & \end{bmatrix} \quad \begin{bmatrix} & \times_{12} & \times_{1(n+2)} \\ \times_{21} & & \times_{2(n+2)} \\ & & \\ \times_{(n+1)1} & \times_{(n+1)2} & \end{bmatrix}$$

偵測出有錯誤的發生且無法更正錯誤的時候，重新計算是無法避免的，我們可用兩個$(n+1)\times(n+2)$矩陣來存放具列與行總和檢查碼的增廣矩陣，一個存放上一階段轉換後正確的結果，另一個用來存放本階段轉換後的結果，兩個矩陣交替使用，如此，只要從上一階段重新計算即可。

是否只要有錯誤均能偵測得出來？很不幸，答案是「否定」的，Why？請看下面的例子，若是$\times_{11}+\times_{12}=a_{11}+a_{12}$、$\times_{21}+\times_{22}=a_{21}+a_{22}$、$\times_{11}+\times_{21}=a_{11}+a_{21}$、$\times_{12}+\times_{22}=a_{12}+a_{22}$，其中$\times_{ij}$表示第 i 列第 j 行錯誤的元素，a_{ij}表示第 i 列第 j 行正確時的元素，若有上述的情形，檢查碼$a_{1(n+2)}$、$a_{2(n+2)}$、$a_{(n+1)1}$、$a_{(n+2)2}$在檢驗的過程中，是無法偵測出錯誤來的。

$$\begin{bmatrix} \times_{11} & \times_{12} & a_{1(n+2)} \\ \times_{21} & \times_{22} & a_{2(n+2)} \\ & & \\ & & \\ a_{(n+1)1} & a_{(n+1)2} & \end{bmatrix}$$

為了偵測及修正更多的錯誤，勢必於原始的增廣矩陣中加入更多的額外資訊，相對地，也會增加運算時所需額外花費的時間，如何在容錯能力與計算時間上做取捨？額外的資訊要如何加入？計算程序要如何修改？有待日後的探討與研究。

5-3 捨位誤差造成的影響

當我們的程式開始運作測試後，常會偵測到許多的錯誤的發生，利用 Visual Basic 6.0 的偵錯模式追蹤變數，發現發生錯誤的值與正確的值都僅差一點點，探究其原因，就是電腦系統在浮點數運算時捨位誤差造成的。捨位誤差是因為實數 x 在電腦系統中以浮點數表示成 $m \times b^e$，其中 m 為分數部分（或稱 mantissa）、e 為指數部分、b 為基底，因為存放分數部分 m 的位元（bit）數目有限，無法真實的表示所有的實數，僅能以近似值來代替。

這樣會影響到我們的程式去判別到底是計算時硬體失效造成的錯誤還是捨位誤差造成的錯誤，參考資料[6]有提到可於參考資料[9]中找到如何去設定捨位誤差的容許值（tolerant value）來解決這個問題，但由於參考資料[9]年代久遠，我們無法取得，程式中就以 $Zero = 1 \times 10^{-10}$ 來當做捨位誤差的容許值（因為 Vsiual Basic 6.0 中的雙精度浮點數 *Double* 的分數部分約可精準至 1×10^{-14}，經過運算放大誤差後，取 1×10^{-10} 當容許值是可以接受的）。

5-4 可否應用到其他的矩陣運算

參考資料[8]提到許多的矩陣運算演算法，如高斯消去法（Gauss elimination）、高斯-喬登消去法（Gauss-Jordan elimination）、LA 分解（LA decomposition）、反矩陣運算（Matrix Inversion）、Cholesky 演算法、Faddeeva 演算法等，具有相同的運算架構與特性，所以我們的容錯計算模式經過適當的修正後，應可運用到其他的矩陣運算演算法上。

六、結論

我們在線性方程式系統原有的增廣矩陣中加入列與行的總和檢查碼，並將原來的矩陣基本列運算做適當的修改，使得列與行總和檢查碼於每階段的轉換後仍保有其總和檢查碼的性質，藉以達到在運算的過程中錯誤偵測與錯誤修正的目的。整個演算法的時間複雜度與以往的方法同為 $O(n^3)$，但是我們的方法可於每個階段的運算中更正一個錯誤及偵測多個錯誤。由於許多著名的矩陣運算演算法具有相同的運算架構與模式，我們的檢查碼編碼方式與修改矩陣基本列運算的概念，應可推廣應用到其他的矩陣運算演算法上。

七、參考資料

1. 高中數學第三冊，龍騰文化事業有限公司。
2. 林義雄，初等線性代數，第二冊—矩陣，九章出版社。
3. 蕭世文譯，“演算法導論，”文魁資訊股份有限公司，第 28 章。
4. 林聰哲譯，“數值分析，”儒林圖書有限公司，第 6 章。
5. Johnson, B. W., “Design and Analysis of Fault-Tolerant Digital Systems,” Chap. 3,

Addison-Wesley Publishing Company, Inc., U.S.A.
6. Huang, K. H., and Abraham, J. A. “Algorithm-based fault tolerance for matrix operations,” *IEEE Trans. Comput.* vol. C-33, 6 (1984), pp. 518-528.
7. Jou, J. Y., and Abraham, J. A. “Fault tolerant matrix arithmetic and signal processing on highly concurrent computing structures,” *Proc. IEEE*, vol.74 no.5, (1986), pp. 732-741.
8. Vijay, M., and Mittal, R. “Algorithm-based fault tolerance:a review,” Microprocessors and Micro-systems, vol.21, (1997), pp. 151-161.
9. Wilkinson, J. H. “Rounding Errors in Algebraic Processes,” Englewood Cliffs, NJ, Prentice-Hall, 1963.

評 語

040418 高中組數學科

具錯誤修正能力的高斯-喬登消去法

研究題材超過中學數學程度，同學們對於研究問題並未進入狀況。