A

Jacobi method

$$x_i^{(k+1)} = -\sum_{j\neq i}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k+1)} + \frac{b_i}{a_{ii}}$$

```
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> g++ .\7_1_1.cpp -o 1
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> ./1
(a) Jacobi Method Solution:
x1 = 1.01905
x2 = 1.85714
x3 = 3.55238
x4 = 2.21905
x5 = 3.85714
x6 = 3.35238
iterations: 32
```

B

Gauss-Seidel method

Gauss-Seidel 方法和 Jacobi 方法類似，不同之處在於 Gauss-Seidel 每次計算新值時，會立刻使用已經更新過的新變數值，而不是像 Jacobi 一樣全部用舊值。

為了求這個方程組的解$\vec{x}$，我們使用疊代法。k用來計量疊代步數。給定該方程組解的一個近似值$\vec{x}^k \in \mathbb{R}^n$。在求k+1步近似值時，我們利用第m個方程式求解第m個未知量。在求解過程中，所有已解出的k+1步元素都被直接使用。這一點與雅可比法不同。對於每個元素可以使用如下公式

$$x_m^{k+1} = \frac{1}{a_{mm}} \left(b_m - \sum_{j=1}^{m-1} a_{mj} \cdot x_j^{k+1} - \sum_{j=m+1}^{n} a_{mj} \cdot x_j^{k} \right), \quad 1 \leq m \leq n.$$

```
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> g++ .\7_1_2.cpp -o 2
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> ./2
(b) Gauss-Seidel Method Solution:
x1 = 1.01905
x2 = 1.85714
x3 = 3.55238
x4 = 2.21905
x5 = 3.85714
x6 = 3.35238
iterations: 18
```

C

SOR method

SOR 方法是在 Gauss-Seidel 方法的基礎上，加入一個鬆弛參數 ω 來加速收斂

$$x_i^{(k+1)} = (1-\omega)x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n.$$

```
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> g++ .\7_1_3.cpp -o 3
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> ./3
(c) SOR Method Solution:
x1 = 1.01905
x2 = 1.85714
x3 = 3.55238
x4 = 2.21905
x5 = 3.85714
x6 = 3.35238
iterations: 14
```

D

Conjugate gradient method

$$\bar{v}^{(k)} = b - A\bar{x}^{(k)}, \quad t_k = (\bar{v}^{(k)}, \bar{v}^{(k)}) / (\bar{v}^{(k)}, A\bar{v}^{(k)}),$$

$$\bar{x}^{(k+1)} = \bar{x}^{(k)} + t_k \bar{v}^{(k)}.$$

```
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> g++ .\7_1_4.cpp -o 4
PS C:\Users\yunyu\Documents\大學\三下\數值方法\Numerical_class\F74114095_numerical_hw7> ./4
(d) Conjugate Gradient Method Solution :
x1 = 1.01905
x2 = 1.85714
x3 = 3.55238
x4 = 2.21905
x5 = 3.85714
x6 = 3.35238
iterations: 32
```