Numerical Analysis
homework 04: Linear Iterative Methods

Due on Tuesday, March 28, 2017

102061149 Fu-En Wang

1 Introduction

To solve such linear system:

$$Ax = b \tag{1}$$

We had used **LU Decomposition** to get x in previous homework. In this project, we will solve it with the following iterative methods:

1. **Jacobi Method**
2. **Gauss-Seidel Method**
3. **Symmetric Gauss-Seidel Method**

To evaluate the performance of three method, we will use Question.4 in previous homework(20 resistors at each side).

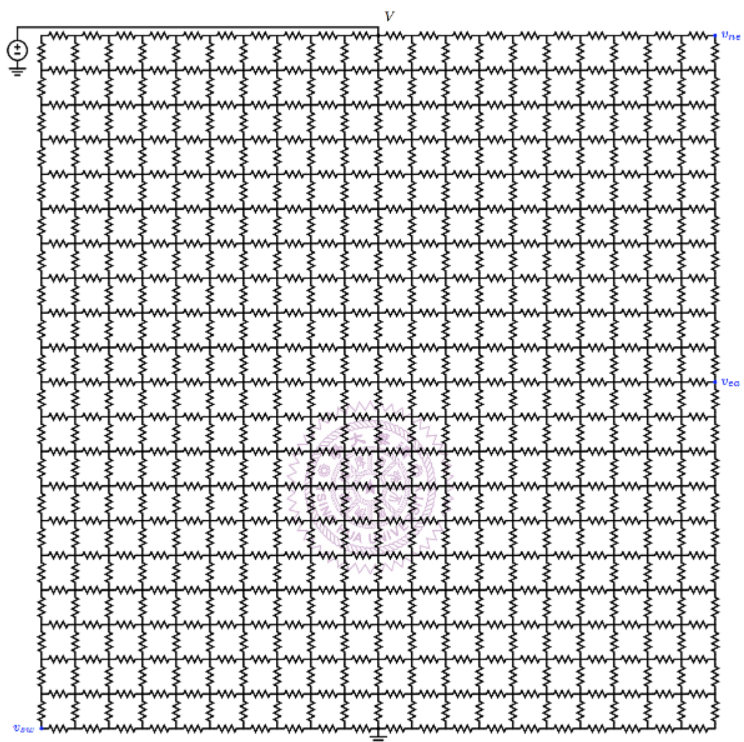

Figure 1: Simple resistor network

To calculate the error, we will use the following error formula:

1. $\|x\|_1 = \sum_{i=1}^{n} |x_i|$
2. $\|x\|_2 = (\sum_{i=1}^{n} x_i^2)^{\frac{1}{2}}$
3. $\|x\|_\infty = max_{i=1}^{n} |x_i|$

2 Implementation

2.1 Jacobi Method

Algorithm 1 Jacobi Method

```
for it ∈ {1, ..., maxIter} do
    lastX = X
    for i ∈ {1, ..., N} do
        sum = 0
        for j ∈ {1, ..., N} do
            if i ≠ j then
                sum += A[i][j] * lastX[j]
            end if
        end for
        x[i] = (1 / A[i][i]) * (b[i] -sum)
    end for
    if Error of (lastX - x) ≤ tol then
        break
    end if
end for
```

2.2 Gauss-Seidel Method

Algorithm 2 Gauss-Seidel Method

```
for it ∈ {1, ..., maxIter} do
    lastX = X
    for i ∈ {1, ..., N} do
        sum1 = 0
        sum2 = 0
        for j ∈ {1, ..., i-1} do
            sum1 += A[i][j] * x[j]
        end for
        for j ∈ {i+1, ..., N} do
            sum2 += A[i][j] * lastX[j]
        end for
        x[i] = (1 / A[i][i]) * (b[i] - sum1 - sum2)
    end for
    if Error of (lastX - x) ≤ tol then
        break
    end if
end for
```

2.3 Symmetric Gauss-Seidel Method

Algorithm 3 Symmetric Gauss-Seidel Method

```
for it ∈ {1, ..., maxIter} do
    lastX = X
    for i ∈ {1, ..., N} do
        sum1 = 0
        sum2 = 0
        for j ∈ {1, ..., i-1} do
            sum1 += A[i][j] * x[j]
        end for
        for j ∈ {i+1, ..., N} do
            sum2 += A[i][j] * lastX[j]
        end for
        x[i] = (1 / A[i][i]) * (b[i] - sum1 - sum2)
    end for
    for i ∈ {N, ..., 1} do
        sum1 = 0
        sum2 = 0
        for j ∈ {1, ..., i-1} do
            sum1 += A[i][j] * x[j]
        end for
        for j ∈ {i+1, ..., N} do
            sum2 += A[i][j] * x[j]
        end for
        x[i] = (1 / A[i][i]) * (b[i] - sum1 - sum2)
    end for
    if Error of (lastX - x) ≤ tol then
        break
    end if
end for
```

2.4 Complexity

In the three algorithm, we only use double for-loop for each time of iteration. As a result, the complexity is $O(n^2)$. When the number of iteration is small, iterative method should be faster than LU Decomposition.

3 Discussion

In this project, we will discuss the following topic:

1. How many iteration and tolerance to get accurate V_{ne}, V_{eq} and V_{sw}(error $< 10^{-7}$).
2. Which algorithm has the fastest convegence speed.
3. Is Symmetric Gauss-Seidel better than Gauss-Seidel?

3.1 Jacobi Method

3.1.1 iteration time