

# **Numerical Analysis: homework 03**

Due on Tuesday, March 21, 2017

**102061149 Fu-En Wang**

# 1 Introduction

For modern circuit design, the scale of circuit is always such large that human cannot handle it manually. To overcome this, we frequently use a lot of EDA tool such as **hspice** to solve these problems. However, the license of these EDA tool is usually expensive. As a result, EDA provider such as **Synopsys** benefits a lot from it every year, which means these circuit analysis technology is quite valuable.

In this project, we will implement several simple **Resistor Networks** and use **LU Decomposition** to analyze it.

## 1.1 Resistor Networks

Each resistor networks only consists of several resistors as shown in Figure 1.

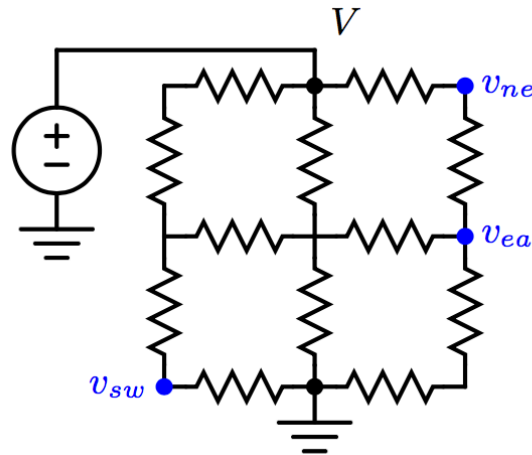


Figure 1: Simple resistor network

In this assignment, there are six networks to implement:

1. 2 resistors each side and resistance is  $1K\Omega$
2. 4 resistors each side and resistance is  $500\Omega$
3. 10 resistors each side and resistance is  $200\Omega$
4. 20 resistors each side and resistance is  $100\Omega$
5. 40 resistors each side and resistance is  $50\Omega$
6. 50 resistors each side and resistance is  $40\Omega$

For each of them, we have to show the equivalent resistance and the voltage of  $V_{ne}$ ,  $V_{ea}$  and  $V_{sw}$ .

## 2 Implementation

### 2.1 Algorithm

To model the problem into  $A$  and  $b$ , I divide the network into two segments by horizontal and vertical resistor as shown in Figure 1. And then update matrix  $A$  and  $b$  with **Algorithm 1.4.1** in class material.

---

**Algorithm 1.4.1** System Equation for a Resistor Network
 

---

For a network with  $N$  nodes in each side, create  $N^2 \times N^2$  zero matrix  $A$  and  $N^2$  zero vector  $b$ .

nodeIndex = 0

$G = resistance^{-1}$

**for** each  $i \in \{1 \dots N\}$  **do**

**for** each  $j \in \{1 \dots N-1\}$  **do**

$A[nodeIndex][nodeIndex] += G$

$A[nodeIndex][nodeIndex+1] -= G$

$A[nodeIndex+1][nodeIndex+1] += G$

$A[nodeIndex+1][nodeIndex] -= G$

    nodeIndex += 1

**end for**

nodeIndex += 1

**end for**

nodeIndex = 0

**for** each  $i \in \{1 \dots N-1\}$  **do**

**for** each  $j \in \{1 \dots N\}$  **do**

$A[nodeIndex][nodeIndex] += G$

$A[nodeIndex][nodeIndex+N] -= G$

$A[nodeIndex+N][nodeIndex+N] += G$

$A[nodeIndex+N][nodeIndex] -= G$

    nodeIndex += 1

**end for**

**end for**

**for**  $v, i \in$  voltage and index at fixed voltage point **do**

$A[i] = 0$

$A[i][i] = 1$

$b[i] = v$

**end for**

$LU = luDecompose(A)$

$Y = forwardSub(LU, b)$

$X = backwardSub(LU, Y)$

$X$  is the voltage of each node.

---

## 3 Discussion

### 3.1 Complexity

In all the following part, I will refer  $N$  as the square of the number of nodes at each side of networks. For the network in Figure 1,  $N$  is  $3 \times 3 = 9$ .

When solving the problem, the most time-consuming part is the **LU Decomposition**, which is  $O(n^3)$ . As a result, the system is a  $O(n^3)$  problem.

### 3.2 Performance Evaluation

In this project, I use five numbers to indicate the efficiency of our method:

1. **Runtime**(total execution time of program)
2. **PROBLEM**(execution time of modeling problem into martrix A and vector b)
3. **LU**(time taken by LU Decomposition)
4. **FWD**(time taken by forward substitution)
5. **BCK**(time taker by backward substitution)

The detail result of each resistor networks is show in Table 1

N	9	25	121	441	1681	2601
R(each)	1000	500	200	100	50	40
R(equivalent)	1000	681.818182	376.009408	229.423008	136.042809	114.395519
$V_{ne}$	0.75	0.7	0.648693	0.622178	0.603088	0.598084
$V_{ea}$	0.5	0.5	0.5	0.5	0.5	0.5
$V_{sw}$	0.25	0.3	0.351307	0.377822	0.396912	0.401916
Runtime(s)	0.003	0.003	0.007	0.2	9.948	36.1
PROBLEM(s)	1.50E-05	2.10E-05	0.000199	0.001173	0.020578	0.052503
LU(s)	2.00E-06	3.30E-05	0.003537	0.194482	9.89144	35.9966
FWD(s)	1.00E-06	5.00E-06	4.40E-05	0.000489	0.008348	0.017034
BCK(s)	1.00E-06	3.00E-06	4.60E-05	0.000532	0.010656	0.019895

Table 1: Experiment result

### 3.3 Runtime and LU

Because LU Decomposition should be the most time-consuming part of algorithm, so after N is large enough, LU should be very close to Runtime. Figure 2 shows the result of Runtime/LU and  $N^3$  with log-scale:

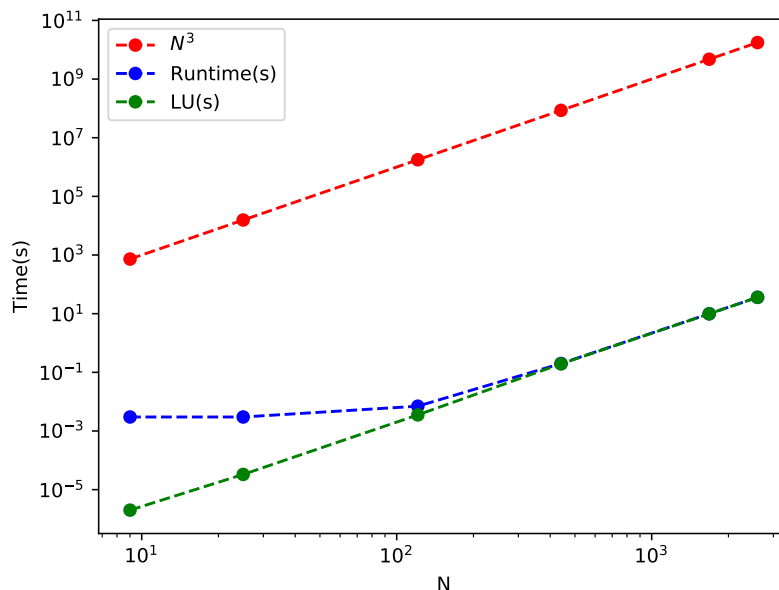


Figure 2: Runtime/LU vs N

Because when  $N$  is small, execution time will have huge error, so we should ignore former part of the line of Runtime. However, when  $N$  is large, we can see that Runtime and LU almost overlap and their slope is same as  $N^3$ , which means the system is a  $O(n^3)$  problem and satisfies my analysis in Section 3.1.

### 3.4 Forward/Backward Substitution

Because the complexity of Forward/Backward Substitution is  $O(n^2)$  as we discuss in last homework assignment, the slope of them should be same as  $N^2$  in log-scale plot. Figure 3 shows the result with log-scale.

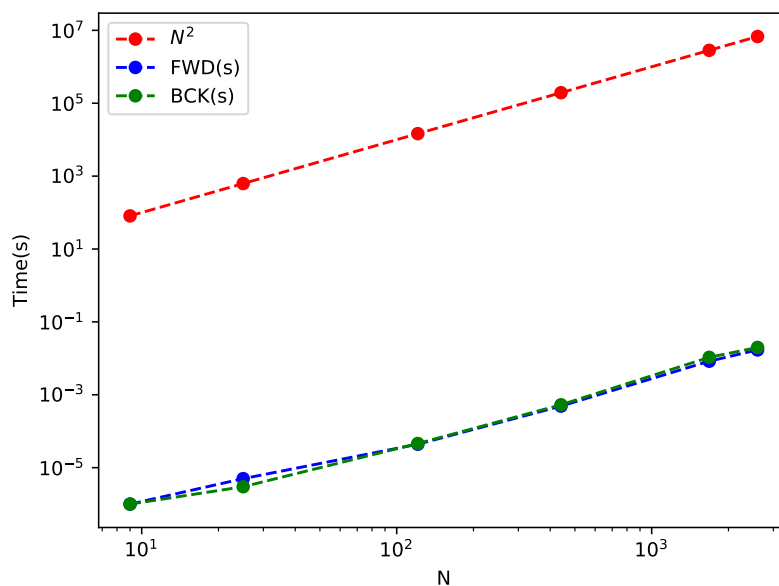


Figure 3: FWD/BCK vs N

From Figure 3, we can clearly see that the slope of FWD and BCK is same as  $N^2$ , which means they are  $O(n^2)$  problem.

## 4 Code Usage

To compile the code, just type `$ g++ hw03.cpp MAT.cpp VEC.cpp` in terminal.

To run the program, use `$ ./a.out 10`, 10 means there are 10 resistors at each side of the resistor network.