Report of Assignment 2----Jacobi Method

Meng Wei

1. Results

When I enter n=10, I get my output solution of x as follows:

0.915438

0.844013

0.790375

0.757464

0.746363

0.756320

0.784929

0.828449

0.882216

0.941108

1. Some experience got from doing this homework

From this homework, I get a lot of knowledge in algorithms, the language of C and C++ and debuging.

First I studied Jacobi method to solve Ax=b from Google by myself, then I run the source code of Jacobi method without errors.

Second, I can understand “for and while” loop more deeply after I finish this homework. And I can use some commands which I didn’t know before, such as malloc, free and scanf.

Third, I met some difficulties when I did this homework. Then I debug my code again and again. It’s worth mentioning that there were no errors, but the solution was always same with x0, when I run the code for the first time. So I checked my code from beginning to the end, I realized that there was something wrong with the “for loop” in the function of Jacobi. Then I found that iteration was not executed at all, because I assigned (b[i] - s) / A[i\*n + i] to y[i] which is invalid in the “for loop”. Then I corrected the code here to be x[i] = (b[i] - s) / A[i\*n + i]. Then it worked.

Above all, I get a lot of experience from doing this homework.

3. Source codes

// Jacobi method.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <stdio.h>

#include <malloc.h>

#include <math.h>

#define MAX 20

double \*Jacobi(double \*A, double \*b)

{

int n;

scanf\_s("%d", &n);

double \*x, s;

int i, j, k = 0;

x = (double\*)malloc(n\*sizeof(double));

for (i = 0; i < n; i++)

x[i] = 0;

while (k< MAX)

{

for (i = 0; i < n; i++)

{s = 0;

for (j = 0; j < n; j++)

if (j != i)

s = s + A[i\*n + j] \* x[j];

x[i] = (b[i] - s) / A[i\*n + i];

}

k++;

}

return x;

}

int main()

{

int n;

scanf\_s("%d", &n);

int i, j;

double \*A = (double\*)malloc((n\*n)\*sizeof(double));

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

if (i == j - 1)

{

A[i\*n + j] = -1.0;

}

else if (i == j)

{

A[i\*n + j] = 2.0;

}

else if (i == j + 1)

{

A[i\*n + j] = -1.0;

}

else { A[i\*n + j] = 0.0; }

}

}

/\*for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

printf("%lf", A[i\*n + j]);

}\*/

double \*b = (double\*)malloc(n\*sizeof(double));

for (i = 0; i < n; i++)

{

if (i == 0 || i == n - 1)

b[i] = 1;

else b[i] = 0;

}

/\*for (i = 0; i < n; i++)

printf("%lf\n", b[i]);\*/

double\*x = (double\*)malloc(n\*sizeof(double));

x = Jacobi(A, b);

for (i = 0; i < n; i++)

printf("%lf\n", x[i]);

free(A);

free(b);

}