# 上机题第四题实验报告

王伟任 计33 2013011333

1. **题目要求及分析**

**第四章上机题1**：考虑10阶Hilbert矩阵作为系数阵的方程组，其中***b***=。取初始解，编写程序用Jacobi与SOR迭代法求解该方程组，将作为终止迭代的判据。

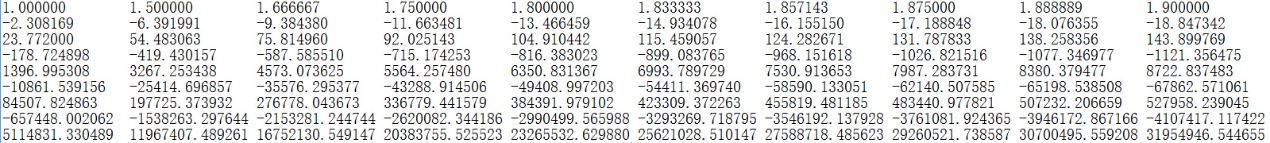
（1）分别用Jacobi与SOR（迭代法求解，观察收敛情况；

（2）改变的值，试验SOR迭代法的效果，考察解的准确度。

迭代法求解是否收敛的判断依据就是迭代矩阵的谱半径是否小于一，当谱半径小于1时迭代法收敛。但是由于谱半径求解算法实现起来较为复杂，因此不另作计算。

1. **实验结果及分析**
2. 利用Jacobi迭代法求解：

利用Jacobi迭代法求解时，迭代法不收敛，得到的结果很快达到数据类型上限。迭代过程记录储存在test1.txt中，记录了每一次迭代的近似解。部分截图如下：



由此可知，此时的迭代矩阵的谱半径大于等于一，不满足收敛条件。

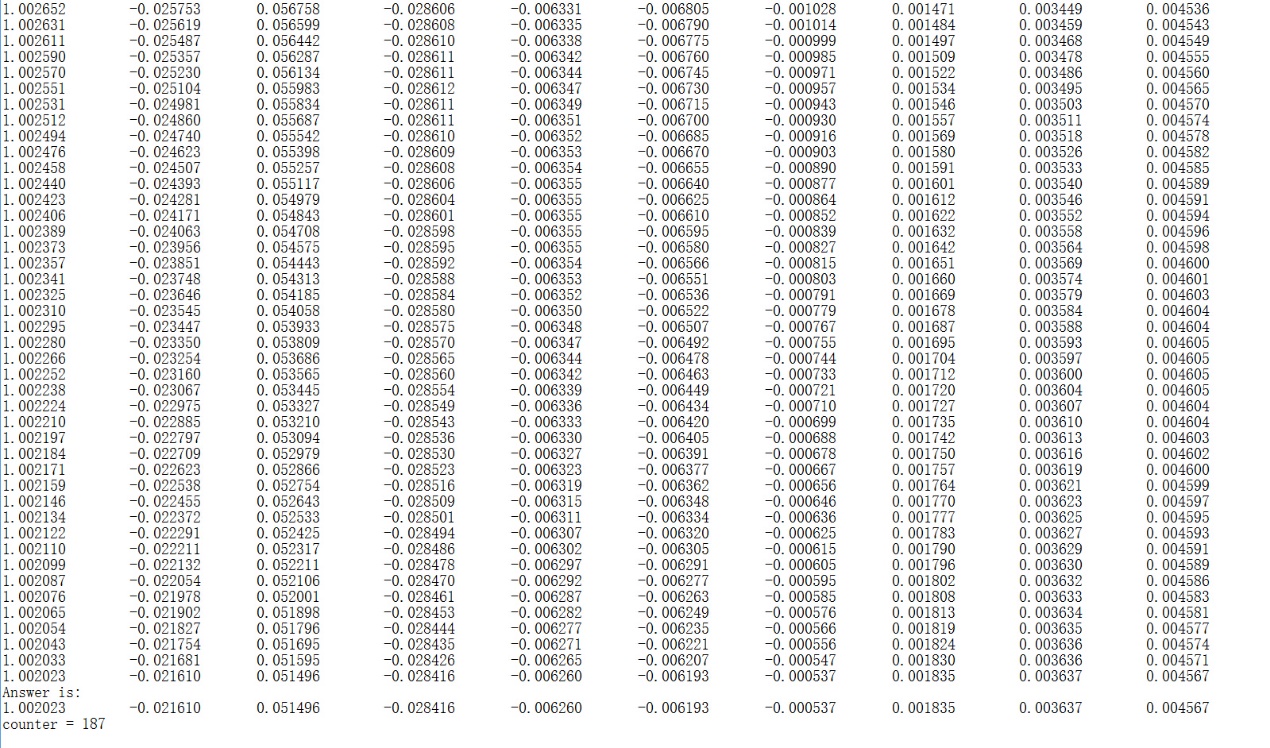
1. 利用SOR（）迭代法求解：

利用SOR（）迭代法求解时，迭代法收敛。

最终得到的近似解为

一共迭代187步。

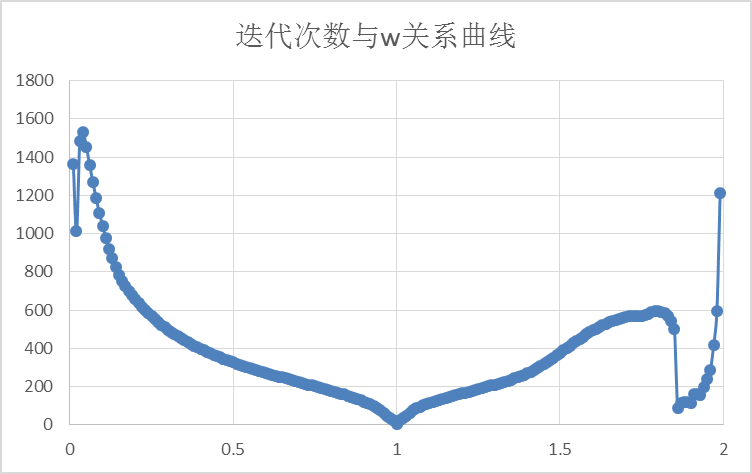
迭代过程记录储存在test2.txt中，记录了每一次迭代的近似解。部分截图如下：

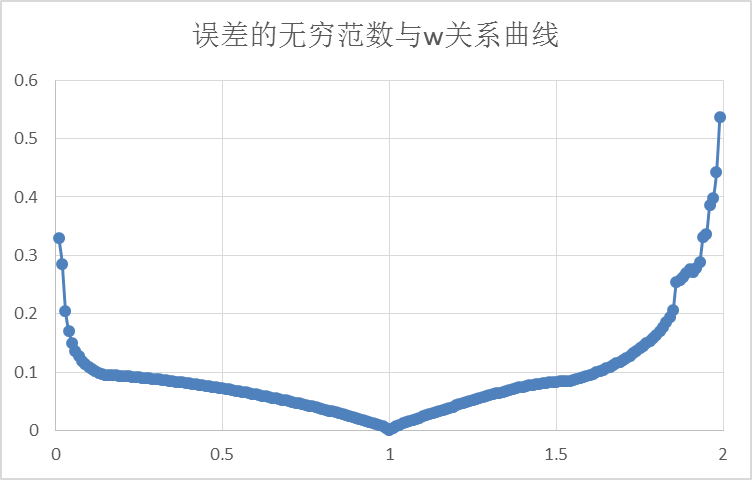


由以上可知，此时的迭代矩阵的谱半径小于一，满足收敛条件。

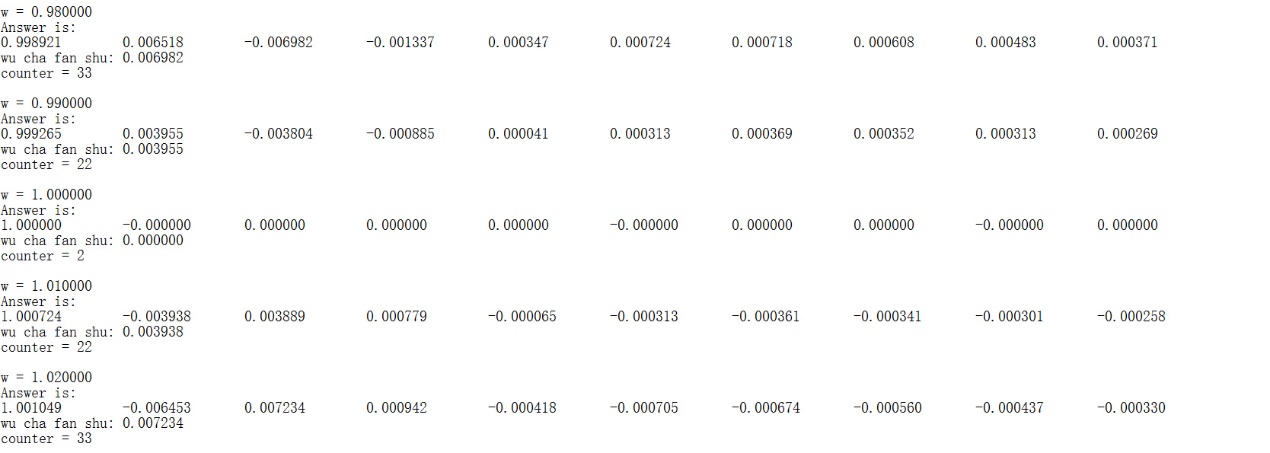
1. 改变的值，试验SOR迭代法的效果

令从0.01变化到1.99，每次增加0.01，观察在每一个下迭代法的收敛情况。计算结果显示，在0.01-1.99范围内，无论取何值，SOR迭代法均收敛，但是收敛速度不同，得到近似解的误差大小也不同。当=1.00时，迭代速度最快，只需要迭代两步即可得到解，而且误差也最小（为零）。





全部计算结果储存在test0.txt中。部分截图如下：



在这些情况下迭代矩阵的谱半径均小于1，满足收敛条件。

1. **实验代码**

采用C++语言实现。

仅附Jacobi迭代法和试验不同ω条件下SOR迭代法效果的代码：

Jacobi迭代法代码：

**#include <cstdio>**

**#include <cmath>**

**int n = 10;**

**double\*\* createHilbert(){**

**double\*\* temp = new double\*[n];**

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

**temp[i] = new double[n];**

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

**temp[i][j] = 1.0 / (i + j + 1);**

**}**

**return temp;**

**}**

**double\* multiply(double\*\* A, double\* bl){**

**double\* ans = new double[n];**

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

**ans[i] = 0;**

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

**ans[i] += A[i][j] \* bl[j];**

**}**

**return ans;**

**}**

**double norm(double\* A, double\* B){**

**double mmax = 0.0;**

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

**if (fabs(A[i] - B[i]) > mmax) mmax = fabs(A[i] - B[i]);**

**return mmax;**

**}**

**int main(){**

**double\*\* Hilbert = createHilbert();**

**double\* b = new double[n];**

**for (int i = 0; i < n; i++) b[i] = 1.0 / (i + 1);**

**double\* x = new double[n];**

**double\* y = new double[n];**

**for (int i = 0; i < n; i++) x[i] = 0;**

**FILE\* fp = fopen("test1.txt", "w");**

**do{ //Jacobi**

**for (int i = 0; i < n; i++) y[i] = x[i];**

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

**double s = 0.0;**

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

**if (j != i) s += Hilbert[i][j] \* y[j];**

**x[i] = (b[i] - s) / Hilbert[i][i];**

**}**

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

**fprintf(fp, "%f\t", x[i]);**

**fprintf(fp, "\n");**

**} while(norm(x, y) >= 0.0001);**

**fprintf(fp, "Answer is:\n");**

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

**fprintf(fp, "%f\t", x[i]);**

**fprintf(fp, "\n");**

**fclose(fp);**

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

**delete[] Hilbert[i];**

**delete[] Hilbert;**

**delete[] b;**

**delete[] x;**

**delete[] y;**

**return 0;**

**}**

SOR迭代法代码：

**#include <cstdio>**

**#include <cmath>**

**int n = 10;**

**double w = 0.01;**

**double\*\* createHilbert(){**

**double\*\* temp = new double\*[n];**

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

**temp[i] = new double[n];**

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

**temp[i][j] = 1.0 / (i + j + 1);**

**}**

**return temp;**

**}**

**double norm(double\* A, double\* B){**

**double mmax = 0.0;**

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

**if (fabs(A[i] - B[i]) > mmax) mmax = fabs(A[i] - B[i]);**

**return mmax;**

**}**

**int main(){**

**double\*\* Hilbert = createHilbert();**

**double\* b = new double[n];**

**for (int i = 0; i < n; i++) b[i] = 1.0 / (i + 1);**

**double\* x = new double[n];**

**double\* y = new double[n];**

**double\* x0 = new double[n];**

**x0[0] = 1.0;**

**for (int i = 1; i < n; i++) x0[i] = 0;**

**FILE\* fp = fopen("test0.txt", "w");**

**for (;w < 2; w += 0.01){**

**for (int i = 0; i < n; i++) x[i] = 0;**

**int counter = 0;**

**do{ //SOR**

**for (int i = 0; i < n; i++) y[i] = x[i];**

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

**double s = 0.0;**

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

**if (j != i) s += Hilbert[i][j] \* x[j];**

**x[i] = (1 - w) \* x[i] + w \* (b[i] - s) / Hilbert[i][i];**

**}**

**counter++;**

**} while(norm(x, y) >= 0.0001);**

**fprintf(fp, "w = %f\n", w);**

**fprintf(fp, "Answer is:\n");**

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

**fprintf(fp, "%f\t", x[i]);**

**fprintf(fp, "\n");**

**double nor = norm(x, x0);**

**fprintf(fp, "wu cha fan shu: %f\n", nor);**

**fprintf(fp, "counter = %d\n\n", counter);**

**}**

**fclose(fp);**

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

**delete[] Hilbert[i];**

**delete[] Hilbert;**

**delete[] b;**

**delete[] x;**

**delete[] y;**

**return 0;**

**}**