# 上机题第三题实验报告

王伟任 计33 2013011333

1. **题目要求及分析**

**第三章上机题6**：编程序生成Hilbert矩阵，以及n维向量，其中**x**为所有分量都是1的向量。用Cholesky分解算法求解方程，得到近似解，计算残差 和误差 的-范数。

（1）设n=10，计算、；

（2）在右端项上施加的扰动然后解方程组，观察残差和误差的变化情况；

（3）改变n的值为8和12，求解相应的方程，观察、的变化情况，通过这个实验说明了什么问题？

利用Cholesky分解算法求解方程，只需要执行一次前代过程和一次回代过程（经过Cholesky分解后，原方程可化为，可以先求解方程，再求解方程），复杂度较小。值得注意的是在右端项添加扰动之后解方程组得到的结果与原结果的差异取决于Hilbert矩阵的条件数大小，因此为了更加清楚的了解这个情况，另行编写程序计算Hilbert矩阵的条件数。

1. **实验结果及分析**
2. 计算和

经计算，得到的近似解结果为：

误差范数为0.0004485575，残差范数为0.00000000000000022204。

结果储存在test0.txt中。

可以看到此时的误差和残差都比较小，在可接受范围内。

1. 添加扰动后得到的结果

由于题目中并未说明扰动加在向量的哪个分量上，因此将扰动依次加在十个分量上，比较得到的结果。得到的结果如下：

·加到**第一分量**上，得到的近似解结果为：

误差范数为0.9601524784，残差范数为0.00000000000000044409。

·加到**第二分量**上，得到的近似解结果为：

误差范数为83.2280159311，残差范数为0.00000000000000222045。

·加到**第三分量**上，得到的近似解结果为：

误差范数为1775.5243125789，残差范数为0.00000000000003153033。

·加到**第四分量**上，得到的近似解结果为：

误差范数为16157.2657752129，残差范数为0.00000000000027400304。

·加到**第五分量**上，得到的近似解结果为：

误差范数为77114.1798725634，残差范数为0.00000000000280797607。

·加到**第六分量**上，得到的近似解结果为：

误差范数为212063.8995017390，残差范数为0.00000000000369249076。

·加到**第七分量**上，得到的近似解结果为：

误差范数为348002.1559917043，残差范数为0.00000000000458877381。

·加到**第八分量**上，得到的近似解结果为：

误差范数为336334.3279038350，残差范数为0.00000000000634092778。

·加到**第九分量**上，得到的近似解结果为：

误差范数为176575.4661780928，残差范数为0.00000000000346078721。

·加到**第十分量**上，得到的近似解结果为：

误差范数为38830.2426978130，残差范数为0.00000000000078559381。

原结果储存在test1.txt中。

可以看出，扰动加在不同的位置对结果的影响有一定差异，但是误差都会变得很大，就连误差最小的情况（加在第一分量上）也有接近1的误差，当加在其他分量上误差最大能达到348002。因此，可以得出结论，Hilbert矩阵是一个极为病态的矩阵，添加扰动对于结果的影响极大。

以上的结果是从直观上来看Hilbert矩阵的病态，从理论上来看，Hilbert矩阵的条件数是35352489096888.9531250000，足见其病态性。

文件test.txt中储存了Hilbert矩阵的逆矩阵。

1. 改变n的值，观察、的变化情况

当n=8时，得到的近似解结果是：

误差范数为0.0000006968，残差范数为0.00000000000000044409。

当n=12时，得到的近似解结果是：

误差范数为0.5184688315，残差范数为0.00000000000000044409。

原结果分别储存在test2.txt和test3.txt中。

综合n=8，n=10，n=12三种情况下的解可以得出结论，当n越大时，方程组的解误差越大。

1. **实验代码**

采用C++语言实现。

由于各问的代码基本类似，因此只附第二问和计算条件数的代码。

第二问代码：

**#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\*\* Cholesky(double\*\* Hilbert){**

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

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

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

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

**for (int j = i + 1; j < n; j++)**

**L[i][j] = 0;**

**L[0][0] = sqrt(Hilbert[0][0]);**

**for (int i = 1; i < n; i++)**

**L[i][0] = Hilbert[i][0] / L[0][0];**

**for (int j = 1; j < n; j++){**

**double ss = 0.0;**

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

**ss += L[j][i] \* L[j][i];**

**L[j][j] = sqrt(Hilbert[j][j] - ss);**

**for (int i = j + 1; i < n; i++){**

**double s = 0.0;**

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

**s += L[i][ii] \* L[j][ii];**

**L[i][j] = (Hilbert[i][j] - s) / L[j][j];**

**}**

**}**

**return L;**

**}**

**double\* solve(double\*\* L, double\* b){**

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

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

**y[i] = b[i] / L[i][i];**

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

**y[i] = y[i] - y[j] \* L[i][j] / L[i][i];**

**}**

**return y;**

**}**

**double\* solve2(double\*\* L, double\* y){**

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

**for (int i = n-1; i >= 0; i--){**

**x[i] = y[i] / L[i][i];**

**for (int j = i + 1; j < n; j++)**

**x[i] = x[i] - x[j] \* L[j][i] / L[i][i];**

**}**

**return x;**

**}**

**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\*\* L = Cholesky(Hilbert);**

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

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

**double \*x, \*y, \*r, \*b;**

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

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

**b = multiply(Hilbert, x0);**

**b[k] = b[k] + 0.0000001;**

**y = solve(L, b);**

**x = solve2(L, y);**

**r = multiply(Hilbert, x);**

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

**fprintf(fp, "Jin si jie x:\n");**

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

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

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

**fprintf(fp, "Wu cha fan shu: %.10f\n", norm(x, x0));**

**fprintf(fp, "Can cha fan shu: %.20f\n\n", norm(r, b));**

**}**

**fclose(fp);**

**delete[] b;**

**delete[] x0;**

**delete[] y;**

**delete[] x;**

**delete[] r;**

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

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

**delete[] Hilbert;**

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

**delete[] L[i];**

**delete[] L;**

**return 0;**

**}**

计算条件数代码：

**#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 \* 2];**

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

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

**for (int j = n; j < 2 \* n; j++)**

**temp[i][j] = 0;**

**temp[i][n + i] = 1.0;**

**}**

**return temp;**

**}**

**double norm(double\*\* h){**

**double mmax = 0;**

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

**double s = 0;**

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

**s += fabs(h[i][j]);**

**if (s > mmax) mmax = s;**

**}**

**return mmax;**

**}**

**int main(){**

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

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

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

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

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

**Hilbert[i][j] = HI[i][j];**

**}**

**for (int i = 1; i < n; i++)**

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

**double bk = HI[i][j] / HI[j][j];**

**for (int k = j; k < 2 \* n; k++)**

**HI[i][k] = HI[i][k] - HI[j][k] \* bk;**

**}**

**for (int i = n-2; i >= 0; i--)**

**for (int j = n-1; j > i; j--){**

**double bk = HI[i][j] / HI[j][j];**

**for (int k = j; k < 2 \* n; k++)**

**HI[i][k] = HI[i][k] - HI[j][k] \* bk;**

**}**

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

**double bk = HI[i][i];**

**for (int j = 0; j < 2 \* n; j++)**

**HI[i][j] = HI[i][j] / bk;**

**}**

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

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

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

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

**invHilbert[i][j] = HI[i][j + n];**

**}**

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

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

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

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

**fprintf(fp, "%26f\t", Hilbert[i][j]);**

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

**}**

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

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

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

**fprintf(fp, "%26f\t", invHilbert[i][j]);**

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

**}**

**fclose(fp);**

**printf("%.10f\n", norm(Hilbert) \* norm(invHilbert));**

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

**delete[] HI[i];**

**delete[] HI;**

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

**delete[] invHilbert[i];**

**delete[] invHilbert;**

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

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

**delete[] Hilbert;**

**return 0;**

**}**