数值分析实验报告

第三章

姓名	王琛
学号	2016011360
班级	计 65
实验日期	2019 年 4 月 10 日
报告日期	2019 年 4 月 10 日

第 3 章第 6 题

问题描述

编程序生成 Hilbert 矩阵 $\boldsymbol{H}_n$，以及 n 维向量 $\boldsymbol{b}=\boldsymbol{H}_n\boldsymbol{x}$，其中，$\boldsymbol{x}$ 为所有分量都是 1 的向量。用 Cholesky 分解算法求解方程 $\boldsymbol{b}=\boldsymbol{H}_n\boldsymbol{x}$，得到近似解 $\hat{\boldsymbol{x}}$，计算残差 $\boldsymbol{r}=\boldsymbol{b}-\boldsymbol{H}_n\hat{\boldsymbol{x}}$ 和误差 $\boldsymbol{\Delta x}=\hat{\boldsymbol{x}}-\boldsymbol{x}$ 的 ∞ 范数。

(1) 设 n=10，计算 $||\boldsymbol{r}||_\infty$、$||\boldsymbol{\Delta x}||_\infty$;
(2) 在右端项上施加 10^{-7} 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 $||\boldsymbol{r}||_\infty$、$||\boldsymbol{\Delta x}||_\infty$ 的变化情况，通过这个实验说明了什么问题?

解题思路

首先生成 Hilbert 矩阵,由于其是对称正定矩阵,因此可以进行 Cholesky 分解，即 $H=LL^T$，其中 L 是下三角矩阵。

然后，$Hx=b$ 可写成 $LL^Tx=b$ 的形式，利用后代和前代的方法可以求解出方程的近似解 $\hat{x}$。并且计算残差和误差，施加扰动后再次计算残差和误差，可以探究其中的规律。

实验结果

当 n 为 8，10，12 时，结果如下表:

	扰动前		扰动后	
n	$\|\|\boldsymbol{r}\|\|_\infty$	$\|\|\boldsymbol{\Delta x}\|\|_\infty$	$\|\|\boldsymbol{r}\|\|_\infty$	$\|\|\boldsymbol{\Delta x}\|\|_\infty$
8	2.2204e-16	4.1154e-07	2.2204e-16	0.0216
10	2.2204e-16	4.4459e-04	4.4409e-16	0.7007
12	4.4409e-16	0.3358	4.4409e-16	23.6202

实验结论

可以看出，随着 n 的增加，$||\boldsymbol{r}||_\infty$ 变化并不明显，并且加上扰动基本不会改变 $||\boldsymbol{r}||_\infty$ 的结果，这是因为计算残差时，是用变化后的 b 作为被减数，抵消了扰动。随着 n 的增加，误差的范数也在很快增大，求解方程的结果也就越来越不准确。另外，我们发现，很小的扰动对于误差的范数也造成了巨大的影响，正说明了 Hilbert 矩阵的病态性，n 越大，条件数越来越大，病态性越来越明显。

实验心得

这次实验我对于 Hilbert 矩阵的病态性有了深入的了解，我深刻体会到了小的扰动所带来的巨大变化。并且从 n=8 到 n=12，问题的误差竟然发生了指数级别的改变。因此，在实际问题的求解时，我们要十分注意系数矩阵的性质，最好计算出条件数，看是否符合要求。

主要代码

生成 hilbert 矩阵

```
function [h,b] = hilbert(n)
h=zeros(n);
x=ones(n,1);
for i=1:n
    for j=1:n
        h(i, j)=1/(i+j-1);
    end
end
b=h*x;
```

根据 Cholesky 分解求解方程组

```
function[x] = lu_solve(l, u, b)
%lu_solve
```

```
% solve x using l, u decompsition. NOTE that the diagonal of l
    is not
% required to be 1.
[m,n]=size(l);
if m~=n
    fprintf("error! l is not square matrix!");
end
y=zeros(n,1);
for i=1:n
    y(i)=b(i);
    for j=1:i-1
        y(i)=y(i)-l(i,j)*y(j);
    end
    y(i)=y(i)/l(i,i);
end
x=zeros(n,1);
for i=n:-1:1
    if u(i,i)==0
        fprintf("error, cannot find pivot!");
    end
    x(i)=y(i);
    for j=n:-1:i+1
        x(i)=x(i)-u(i,j)*x(j);
    end
    x(i)=x(i)/u(i,i);
end
```