暨南大学本科实验报告专用纸

课程名称 数值计算实验 成绩评定

实验项目名称 Inconsistent problem 指导教师 Liangda Fang

实验项目编号 03 实验项目类型 综合性 实验地点 N116

学生姓名 甄洛生、吴凯一 学号 2018054625、2018050821

学院 信息科学技术学院 系 计算机系 专业 计算机科学技术

实验时间 2020 年 11 月 20 日下午 10:30~ 12:10

I. Problems

Given two inconsistent systems as follows:

$$\text{(a)} \begin{bmatrix} 3 & -1 & 2 \\ 4 & 1 & 0 \\ -3 & 2 & 1 \\ 1 & 1 & 5 \\ -2 & 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 10 \\ -5 \\ 15 \\ 0 \end{bmatrix} \text{ and (b)} \begin{bmatrix} 4 & 2 & 3 & 0 \\ -2 & 3 & -1 & 1 \\ 1 & 3 & -4 & 2 \\ 1 & 0 & 1 & -1 \\ 3 & 1 & 3 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \\ 2 \\ 0 \\ 5 \end{bmatrix}.$$

1. Write a program that implements classical Gram-Schmidt to find the full QR factorization, and report the matrices Q and R.
2. Repeat the first question, but implement Householder reflections and report each Householder reflector Hi of every step, the matrices Q and R.
3. Report the least squares solution and 2-norm error.

II. Algorithm summary

● Least squares

What is least squares? Let's look at the following function:

$$\phi(x) = \sum (Observe - Predict)^2$$

Our goal is to **minimize** the function $\phi(x)$, $Observe$ is some observed value like the value on the temperature sensor, $Predict$ is the value we want to forecast. That is why we called "**least squares**" And this is what we often say in our lives--**Data Fitting**.

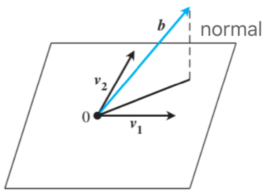
In the field of mathematical modeling, data fitting is usually an effective way to **expand** the dataset. Now consider using a linear function to fit these three points:

$$Fitting\ function \rightarrow f(x) = c_0 + c_1 x$$
$$Dataset \rightarrow (1,2), (-1,1), (1,3)$$

Represented by matrix:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} c_0 \\ c_1 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$$

Obviously, this is an **unsolved** system of equations! In other words, this is an **inconsistent system of equations**. But we still want to find the **closest solution**. If we look at our system of equations again, we can see that the dimension of the space stretched by our matrix A is **smaller** than that of vector b. Then we can get the nearest solution by **projecting** the vector b into the geometric space where A is located. The error of the solution obtained in this way happens to be the normal length of the vector b projected into space A. The following is a schematic diagram:



Expressed by mathematical formula：

$$\min(\phi(x)) = \min\left(||b - Ax||_2\right)$$

Noted that vector $normal$ is perpendicular to space A, and:

$$normal = b - A\overline{x}$$
$$normal \perp Ax, x \in R^n$$
$$\therefore (Ax)^T * normal = 0$$
$$\therefore x^T A^T * normal = 0$$
$$\because x \neq \vec{0} \therefore A^T(b - A\overline{x}) = 0$$
$$\therefore A^T A\overline{x} = A^T b$$

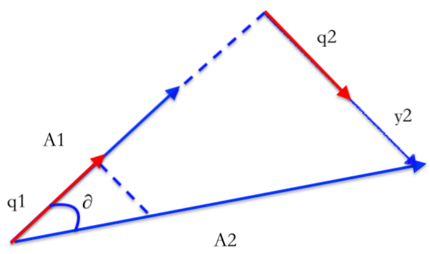
$\overline{x}$ is the closest solution what we want to find!

$$\therefore \overline{x} = (A^T A)^{-1} * b \qquad (1-1)$$

● Classical Gram-Schmidt orthogonalization

We can use the normal equation to solve the least squares problem, but once the **condition number** of the $A^T A$ matrix constructed by the normal equation is **too large**, this method will be invalid! Can we avoid calculating for $(A^T A)^{-1}$? Now we introduce the concept of **QR decomposition**, which decomposes the matrix A into two matrices Q and R, where Q is a **standard orthogonal matrix** and R is an **upper triangular matrix**. So how do we break it down? Noted that Q is a standard orthogonal matrix, so we first use matrix A to generate a set of standard orthogonal bases, and the method of generation is to use the **Gramm-Schmidt orthogonalization** described below:

$$A\ is\ a\ m \times n\ matrix, and\ rank(A) = n$$
$$\therefore A = [A_1, A_2, \dots, A_n]$$
$$Standard\ orthogonal\ bases < q_1, q_2, \dots, q_n >$$
$$q_i\ is\ a\ m - dimension\ vector$$
$$For\ q_1, we\ just\ use\ A_1\ replace\ it. But\ maybe\ ||A_1||_2 \neq 1$$
$$\therefore q_1 = \frac{A_1}{||A_1||_2}$$



Now we want to use A_2 to generate q_2, noted that:

$$\therefore q_1^T A_2 = |q_1||A_2| \cos \partial$$
$$\therefore y_2 = A_2 - q_1 * (q_1^T A_2)$$
$$\therefore q_2 = \frac{y_2}{\|y_2\|_2}$$

Now, maybe you've already guessed it, for q_n:

$$y_n = A_n - q_1 * (q_1^T A_n) - q_2 * (q_2^T A_n) - \cdots - q_{n-1} * (q_{n-1}^T A_n) \qquad (2-1)$$
$$q_n = \frac{y_n}{\|y_n\|_2}$$

It is also very simple to prove its orthogonality, as follows:

$$For\ i < j < n$$
$$Step1.\ (j = 2)$$
$$q_1^T * y_2 = q_1^T A_2 - q_1^T * q_1 * (q_1^T A_2) = 0 \quad Proof!$$
$$Step2.\ (j < k)$$
$$Assume\ q_i^T y_j = 0$$
$$Step3.\ (j = k)$$
$$\begin{aligned} q_i^T * y_j &= q_i^T * \left(A_j - \sum_{c=1}^{j-1} q_c * \left(q_c^T A_j\right) \right) \\ &= q_i^T * A_j - q_i^T * q_i * \left(q_i^T A_n\right) \\ &= 0 \quad Proof! \end{aligned}$$

Now we have generated a set of **standard orthogonal bases** q_i, but how should we construct the upper triangular matrix R? If we look at the Gramm-Schmidt orthogonal procedure again, we can deform $(2-1)$ a little:

$$A_n = q_1(q_1^T A_n) + \cdots + q_{n-1}(q_{n-1}^T A_n) + y_n$$
$$A_n = q_1(q_1^T A_n) + \cdots + q_{n-1}(q_{n-1}^T A_n) + q_n * \|y_n\|_2 \qquad (2-2)$$

That is to say:

$$\begin{aligned} A &= Q * R \\ &= [q_1, q_2, \dots, q_n] * \begin{bmatrix} \|y_1\|_2 & q_1^T A_2 & \dots & q_1^T A_n \\ 0 & & \dots & \dots \\ 0 & & 0 & \|y_n\|_2 \end{bmatrix} \end{aligned} \qquad (2-3)$$

We make it, isn't it? Basically, more detailed:

$$r_{ij}\ means\ R(i,j)$$
$$r_{ii} = \|y_i\|_2$$
$$r_{ij} = q_i^T A_j$$

Think it's over? If $n < m$, then only n orthogonal bases can be obtained, which can only describe n-dimensional space, **not m-dimensional space**, so it is called **reduced QR decomposition**. We can add $m - n$ vectors, which are independent of $A_i, i = 1,2, \dots, n$

linearly, and then do the above steps, we can get m orthogonal bases, which is now called **full QR decomposition**.

Now we can use Gramm-Schmidt orthogonalization to decompose matrix A to get matrices Q and R. Don't forget that our task is to **find the closest solution** to the system of inconsistent equations:

$$\min \phi(x) = \min(b - Ax)$$
$$\because A = QR$$
$$\therefore QRx = b$$
$$\because Q^T Q = I$$
$$\therefore Q^T = Q^{-1}$$
$$\therefore Rx = Q^T b$$
$$Noted\ that\ R\ is\ m \times n, Q\ is\ m \times m.$$
$$\therefore e = Rx - Q^T b, size(e) is\ m \times 1$$
$$\therefore \min(b - Ax) = \min(e)$$
$$\because e = \begin{bmatrix} e_1 \\ \dots \\ e_n \\ e_{n+1} \\ \dots \\ e_m \end{bmatrix}, we\ only\ can\ minimal\ (e_1, \dots, e_n)\ to\ (0, \dots, 0)\ when\ \hat{R}x = \hat{d}.$$
$$where\ \ R = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}\ and\ d = Q^T b\ and\ d = \begin{bmatrix} \hat{d} \\ d_{n+1} \\ \dots \\ d_m \end{bmatrix}$$

Therefore, the closest solution $\bar{x}$:

$$\bar{x} = \hat{R}^{-1}\hat{d}$$
$$error = \sqrt{d_{n+1}^2 + \dots + d_m^2} \qquad (2-4)$$

● Householder reflections

We already know that using QR decomposition can avoid calculating large conditional number matrices such as $A^T A$, but using standard Gramm-Schmidt orthogonalization to QR decomposition is **not stable** enough. The reason is that the **rounding error** of the computer may cause the generated orthogonal base to be not orthogonal! So, we want to introduce a new concept, the House Holder reflection matrix:

$$\exists x, \omega\ and\ \|x\|_2 = \|\omega\|_2$$
$$let\ u = \omega - x$$
$$let\ v = \frac{u}{\|u\|_2}$$
$$H = I - 2vv^T$$

The H is the so-called House Holder reflection matrix! It has these properties:

$$Hx = \omega$$
$$H\omega = x$$

Now, we will use the House Holder reflection to decompose A into QR decomposition:

$$\because R\ is\ a\ upper\ triangular\ matrix$$
$$\therefore let\ A_{12}, \dots, A_{1n}\ be\ 0$$
$$\therefore x = A_1, \omega = (\|x\|_2, 0, \dots, 0)$$
$$\therefore \|x\|_2 = \|\omega\|_2$$
$$\therefore u = \omega - x, v = \frac{u}{\|u\|_2}$$
$$\therefore \widehat{H_1} = I - 2vv^T$$
$$\therefore H_1 = \widehat{H_1}\ (we\ should\ keep\ H_i\ size\ is\ m \times m)$$
$$H_1 A = \begin{bmatrix} x & x & x \\ 0 & & \\ \dots & x & x \\ 0 & & \\ 0 & x & x \end{bmatrix}$$
$$\therefore let\ A_{23}, \dots, A_{2n}\ be\ 0$$
$$Now\ just\ x = (A_{22}, A_{23}, \dots A_{2n}), \omega = (\|x\|_2, 0, \dots, 0)$$
$$And\ other\ procedure\ same\ as\ above.$$
$$\therefore H_2 = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & & & \\ \dots & & \widehat{H_2} & \\ 0 & & & \end{bmatrix} (keep\ H_i\ size\ is\ m \times m)$$

And H_n can be obtained by above procedures. Eventually we will get:

$$H_n H_{n-1} \dots H_2 H_1 A = R$$
$$\because H_i = H_i^{-1}$$
$$\therefore A = H_1 \dots H_n R$$
$$\therefore Q = H_1 \dots H_n$$

That is the application of Householder in QR decomposition.

III. Result analysis

1) Result of question 1:

For inconsistent system **(a)**:

变量 - Q_a

Q_a

5x5 double

	1	2	3	4	5	6
1	0.4804	-0.2697	0.4057	0.6503	0.7073	
2	0.6405	0.5494	-0.2236	0.1326	-0.2081	
3	-0.4804	0.6592	-0.0310	0.5083	0.5636	
4	0.1601	0.4295	0.6914	-0.4989	-0.2117	
5	-0.3203	-0.0799	0.5535	0.2286	-0.3065	

Figure 1 Matrix Q

变量 - R_a

R_a

5x3 double

	1	2	3
1	6.2450	-0.6405	0.3203
2	0	2.5670	2.0277
3	0	0	5.8980
4	0	0	0
5	0	0	0
6			

Figure 2 Matrix R

For inconsistent system **(b)**:

	1	2	3	4	5
1	0.7184	0.2115	0.2259	0.6086	0.1332
2	-0.3592	0.7685	0.5228	-0.0516	0.0666
3	0.1796	0.5993	-0.7688	-0.1320	-0.0133
4	0.1796	-0.0564	0.0525	-0.4071	0.8922
5	0.5388	0.0494	0.2862	-0.6662	-0.4261

Figure 3 Matrix Q

	1	2	3	4
1	5.5678	1.4368	3.5921	-1.2572
2	0	4.5755	-2.4393	1.9247
3	0	0	4.1408	-1.6395
4	0	0	0	1.4237
5	0	0	0	0

Figure 4 Matrix R

2) Result of question 2:

For inconsistent system **(a)**:

	1	2	3	4	5
1	0.4804	-0.2697	0.4057	0.7098	-0.1676
2	0.6405	0.5494	-0.2236	0.0182	0.4874
3	-0.4804	0.6592	-0.0310	0.5587	-0.1469
4	0.1601	0.4295	0.6914	-0.4258	-0.3614
5	-0.3203	-0.0799	0.5535	0.0502	0.7630

Figure 5 Matrix Q

	1	2	3
1	6.2450	-0.6405	0.3203
2	-4.1259e-16	2.5670	2.0277
3	5.4594e-16	8.5582e-17	5.8980
4	7.0566e-16	-1.8790e-16	-3.2713e-18
5	-9.0984e-17	-5.6474e-17	2.0182e-16

Figure 6 Matrix R

	1	2	3	4	5
1	0.4804	0.6405	-0.4804	0.1601	-0.3203
2	0.6405	0.2105	0.5922	-0.1974	0.3948
3	-0.4804	0.5922	0.5559	0.1480	-0.2961
4	0.1601	-0.1974	0.1480	0.9507	0.0987
5	-0.3203	0.3948	-0.2961	0.0987	0.8026

Figure 7 Matrix H1

	1	2	3	4	5
1	1	0	0	0	0
2	0	0.2169	0.9086	0.3464	0.0863
3	0	0.9086	-0.0542	-0.4019	-0.1001
4	0	0.3464	-0.4019	0.8468	-0.0382
5	0	0.0863	-0.1001	-0.0382	0.9905

Figure 8 Matrix H2

	1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	-0.0853	0.9387	0.3339
4	0	0	0.9387	0.1880	-0.2888
5	0	0	0.3339	-0.2888	0.8973

Figure 9 Matrix H3

For inconsistent system **(b)**:

变量 – Q_b

5x5 double

	1	2	3	4	5
1	0.7184	0.2115	0.2259	0.6086	-0.1332
2	-0.3592	0.7685	0.5228	-0.0516	-0.0666
3	0.1796	0.5993	-0.7688	-0.1320	0.0133
4	0.1796	-0.0564	0.0525	-0.4071	-0.8922
5	0.5388	0.0494	0.2862	-0.6662	0.4261

Figure 10 Matrix Q

变量 – R_b

5x4 double

	1	2	3	4
1	5.5678	1.4368	3.5921	-1.2572
2	-3.8433e-16	4.5755	-2.4393	1.9247
3	-1.1060e-16	-2.8016e-16	4.1408	-1.6395
4	-6.4884e-16	-3.2757e-16	-6.1102e-17	1.4237
5	1.2987e-16	4.1615e-17	-1.2132e-16	-4.0857e-17

Figure 11 Matrix R

变量 – H_b{1, 1}

H_b{1, 1}

	1	2	3	4	5
1	0.7184	-0.3592	0.1796	0.1796	0.5388
2	-0.3592	0.5418	0.2291	0.2291	0.6874
3	0.1796	0.2291	0.8854	-0.1146	-0.3437
4	0.1796	0.2291	-0.1146	0.8854	-0.3437
5	0.5388	0.6874	-0.3437	-0.3437	-0.0311

Figure 12 Matrix H1

变量 – H_b{1, 2}

H_b{1, 2}

	1	2	3	4	5
1	1	0	0	0	0
2	0	0.4986	0.7342	0.0785	0.4541
3	0	0.7342	-0.0751	-0.1150	-0.6649
4	0	0.0785	-0.1150	0.9877	-0.0711
5	0	0.4541	-0.6649	-0.0711	0.5887

Figure 13 Matrix H2

变量 – H_b{1, 3}

H_b{1, 3}

	1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	-0.2811	0.2333	0.9309
4	0	0	0.2333	0.9575	-0.1695
5	0	0	0.9309	-0.1695	0.3235

Figure 14 Matrix H3

变量 – H_b{1, 4}

H_b{1, 4}

	1	2	3	4	5
1	1	0	0	0	0
2	0	1	0	0	0
3	0	0	1	0	0
4	0	0	0	-0.3227	-0.9465
5	0	0	0	-0.9465	0.3227

Figure 15 Matrix H4

3) Result of question 3:

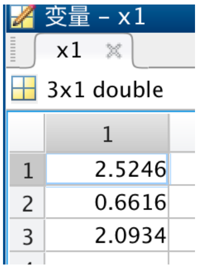


Figure 16 (a) closest solution x_1

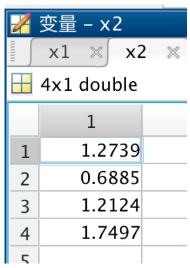


Figure 17 (b) closest solution x_2

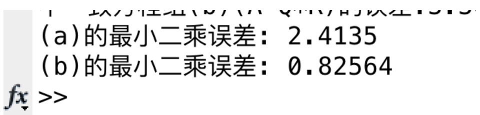


Figure 18 error

IV. Experimental summary

Through this experiment, we understand the principle of **data fitting** and the application of **QR decomposition**. In this experiment, two different methods are used to calculate the QR decomposition of matrix A, one is the **classical Gram-Schmidt method**, and the other is the **House Holder reflection method**. Generally speaking, the QR decomposition based on House Holder reflection matrix is **more stable** and takes up **less space-time overhead**.

The experimental results show that the errors of the closest solutions obtained by the two methods are relatively small.

V. Appendix: Source code

ClassicalGS.m: 基于施密特正交化 QR 分解

```
function [Q, R] = ClassicalGS(A)
    %% 基于经典格拉姆施密特正交的 QR 分解

    % 我们已经假设 A 是 n 列线性无关的 m 维向量
    % 由于 A 可能是  n ≤ m  的，其中 n 是 A 的列数，m 是行数
    % 我们需要扩展一下 A，才能生成 m 个正交基底
    % 毕竟 n 个向量至多张成 n 维空间


    % 获取 A 的行数列数
    m = size(A, 1);
    n = size(A, 2);
    % 初始化返回值
    Q = zeros(m, m);
    R = zeros(m, n);
    % 检查 A 是否为 n 个线性无关的向量组
    if rank(A) < n
        disp('A 不是由  n  个线性无关的向量组成的矩阵');
        Q = 0;
```

```
        R = 0;
        return ;
    end
    % 若 n < m 我们要填充向量使得 A 的 rank 为 m
    if n < m
        % 保证 rank 为 m 因为只是很大概率 并不是说一定
        while rank(A) ~= m
            for k = 1 : m - n
                % 由于 A 里已经是 n 个线性无关向量的组合
                % 我们可以依次取前 m - n 个向量，并进行随机变换
                % 这样我们可以以很大概率不会与之前的向量有线性相关关系
                A(:, n+k) = rand(m, 1) .* A(:, k);
            end
        end
    end
    % 现在进行经典的嘎拉姆施密特正交分解
    for k = 1 : m
        y = A(:, k);
        for count = 1 : k-1
            if count > n
                break
            end
            R(count, k) = Q(:, count)' * A(:, k);
            y = y - R(count, k) * Q(:, count);
        end
        R(k, k) = norm(y);
        Q(:, k) = y / R(k, k);
    end
    % 只保留 R 的 m*n 的部分
    R = R(1:m, 1:n);
end
```

HouseHolder.m: 基于豪斯霍尔德反射的 QR 分解

```
function [Q, R, H_list] = HouseHolder(A)
    %% 基于豪斯霍尔德反射的 QR 分解
    % H_list 是每一步反射子矩阵

    m = size(A, 1);
    n = size(A, 2);
    R = A;
    H_list = cell(1, n);
    for k = 1 : n
        x = R(k:end, k);
        w = zeros(size(x));
```

```
        w(1) = norm(x);

        v = w - x;
        P = (1 / (v' * v))*(v .* v');
        H_hat = eye(size(P)) - 2 * P;
        H = eye(m);
        H(k:end,k:end) = H_hat;
        R = H * R;  % 等 k = n 时，R 才是真正意义上的 R
        % 其他时候都是 H_i * H_(i-1) * ... * H_2 * H_1 * A

        H_list{k} = H;
    end
    % 计算 Q
    Q = H_list{1};
    for k = 2 : n
        Q = Q * H_list{k};
    end
end
```

init.m：初始化数据

```
A1 = [
    3   -1  2;
    4   1   0;
    -3  2   1;
    1   1   5;
    -2  0   3
];
b1 = [10 10 -5 15 0]';

A2 = [
    4   2   3   0;
    -2  3   -1  1;
    1   3   -4  2;
    1   0   1   -1;
    3   1   3   -2
];
b2 = [10 0 2 0 5]';

% 矩阵维度
m1 = size(A1,1);
n1 = size(A1,2);

m2 = size(A2,1);
n2 = size(A2,2);
```

q1.m: 问题 1 求解脚本

```
clear;clc
% 导入数据
run init.m

% QR 分解
[Q_a, R_a] = ClassicalGS(A1);
[Q_b, R_b] = ClassicalGS(A2);

% QR 分解误差
disp(['不一致方程组(a)(A-Q*R)的误差:', num2str(norm(A1-Q_a*R_a))])
disp(['不一致方程组(b)(A-Q*R)的误差:', num2str(norm(A2-Q_b*R_b))])

% 作者甄洛生 抄袭死 m
```

q2.m: 问题 2 求解脚本

```
clear;clc
run init.m

% QR 分解 基于豪斯活儿霍尔德反射
[Q_a, R_a, H_a] = HouseHolder(A1);
[Q_b, R_b, H_b] = HouseHolder(A2);

% QR 分解误差
disp(['不一致方程组(a)(A-Q*R)的误差:', num2str(norm(A1-Q_a*R_a))])
disp(['不一致方程组(b)(A-Q*R)的误差:', num2str(norm(A2-Q_b*R_b))])
```

q3.m: 问题 3 求解脚本

```
clear;clc

run q2.m

% 解（a）
d1 = Q_a' * b1;
x1 = inv(R_a(1:n1,1:end)) * d1(1:n1);
e1 = norm(d1(n1+1:end));

% 解（b）
d2 = Q_b' * b2;
x2 = inv(R_b(1:n2,1:end)) * d2(1:n2);
e2 = norm(d2(n2+1:end));

```

```
disp(['(a)的最小二乘误差：', num2str(e1)])
disp(['(b)的最小二乘误差：', num2str(e2)])
```