Algorithmische Methoden in der Numerik - Uebung2

Felix Dreßler (k12105003)
Elisabeth Köberle (k12110408)

16. Juni 2022

1 Aufgabe a - QRFact

```
function [A, D, pi , k ] = QRFact (A)

[m,n] = size(A);

pi = 1:n; %p=pi
si = zeros(n,1);
D = zeros(min(m,n),1);
nq = n;

for j = n:-1:1
    si(j) = dot(A(:,j),A(:,j));
    if si(j) == 0
        temp1 = pi(j); %alternative (maybe less efficient) [pi(j),pi(nq)] = deal(pi(nb), pi(j));
        pi(j) = pi(nq);
        pi(nq) = temp1;
        nq = nq-1;
    end
end
siq = si;

for i = 1:nq%different loop iterator than in script, here i is k

    [val,piv] = max(si(pi(i:nq))./siq(pi(i:nq)));

    piv = piv+i-1;

    if val <= -1 %piv < k wenn val <= -1
        k =i-1;
        return;
    end

    temp1 = pi(i);
    pi(i) = pi(piv);
    pi(piv) = temp1;

    si(pi(i)) = dot(A(i:m,pi(i)),A(i:m,pi(i)));

    if si(pi(i)) < m * eps^2 * siq(pi(i))
        k = i-1;
        return;
    end

    if sign(A(i,(pi(i)))) == 0 %to compensate for the different sign function
        D(i,1) = -sqrt(si(pi(i)));
    else
        D(i,1) = -sign(A(i,(pi(i)))) * sqrt(si(pi(i)));
    end

    A(i,pi(i)) = A(i, pi(i)) - D(i,1);

    for j = i+1:nq
        gamma = dot(A(i:m,pi(j)),A(i:m,pi(i))) / (-D(i,1)*A(i,pi(i))); %dot() is scalarproduct
        A(i:m, pi(j)) = A(i:m, pi(j)) - gamma * A(i:m, pi(i));
        si(pi(j)) = si(pi(j)) - A(i, pi(j))^2;
        if si(pi(j)) < m * eps * siq(pi(j))
```

```
            si(pi(j)) = dot(A(i+1:m,pi(j)),A(i+1:m,pi(j)));
        end
    end
end
k = nq;
%return;
end
```

2 Aufgabe b - QRSolve

Unter Verwendung von den in *Aufgabe c* berechneten Q und R wurde in *Aufgabe b* der Vektor x berechnet.

```
function [ x ] = QRSolve (B,D, p , k , b )

[~,n] = size(B);

if k < n
    x = zeros(n:1);
else

    Q = CompQ(B,p,k);
    Qt = transpose(Q);
    c= Qt * b;

    R=B (:,p) ;
    x= zeros (n,1) ;

    pi(p) = 1:length(p);

    x(k) = c(k)/D(k);
    for i=k -1: -1:1
        x(i) =(c(i)-dot(R(i,i+1:k) , x(i+1:k))) / D(i);
    end
    x = x(pi);

end
end
```

3 Aufgabe c

3.1 CompR

```
function [R] = CompR(B,D, p , k )

R= triu ( B (: , p ));
R= full ( spdiags (D ,0 , R) );

end
```

3.2 CompQ

```
function [Q] = CompQ(B, p , k )

[m ,~]= size (B);
Q= eye (m );

for j =1: k

    v= zeros (m ,1) ;
    if(j >1)
        v (1: j -1) =0;
    end
    v(j:m)=B (j:m ,p(j) );

    P= eye (m) -(2/ dot (v ,v)) *( v* transpose (v ));
    Q=Q *P;
end
end
```

4 Tests

Aufgrund der besseren leserlichkeit wurde auf genauere Darstellung der Zahlen großteils verzichtet. Die Tests wurden dafür alle als Matlab Workspace gespeichert und beigelegt.

4.1 QRFact Tests

Im folgenden wird mit Matrizen der Größe 2x2, 4x2, 10x5 und 1000x100 getestet.

4.1.1 2x2 Matrix

```
>> A1=randMatrix(2,2,2)

A1 =

0.5377   -2.2588
1.8339    0.8622
>> [B1,D1,p1,k1] = QRFact(A1)

B1 =

2.4487   -0.1918
1.8339    4.8203


D1 =

-1.9111
-2.4102


p1 =

1     2


k1 =

2
```

4.1.2 4x2 Matrix

```
>> A2 = randMatrix(4,2,2)

A2 =

0.3188    3.5784
-1.3077    2.7694
-0.4336   -1.3499
0.3426    3.0349

>> [B2,D2,p2,k2] = QRFact(A2)

B2 =

1.7738    0.5881
-1.3077   10.5562
-0.4336   -0.6189
0.3426    2.4573


D2 =

-1.4550
-5.5823


p2 =

1     2


k2 =

2
```

4.1.3 10x5 Matrix

```
A3 = randMatrix(10,5,5)

A3 =

0.7254    0.7172   -1.0689    0.3192   -1.2141
-0.0631    1.6302   -0.8095    0.3129   -1.1135
0.7147    0.4889   -2.9443   -0.8649   -0.0068
-0.2050    1.0347    1.4384   -0.0301    1.5326
-0.1241    0.7269    0.3252   -0.1649   -0.7697
1.4897   -0.3034   -0.7549    0.6277    0.3714
1.4090    0.2939    1.3703    1.0933   -0.2256
1.4172   -0.7873   -1.7115    1.1093    1.1174
0.6715    0.8884   -0.1022   -0.8637   -1.0891
-1.2075   -1.1471   -0.2414    0.0774    0.0326

>> [B3,D3,p3,k3] = QRFact(A3)

B3 =

3.7619   -0.4257    1.5019   -0.9862   -0.0047
-0.0631   -1.2958    0.6194    0.5266   -4.0112
0.7147    0.4595   -6.4680    0.5611    0.2229
-0.2050    4.5881    0.7601    0.5911    1.4667
-0.1241    0.1682    0.5374   -1.8159   -0.8096
1.4897   -0.1315   -0.0490    0.0594    0.8503
1.4090    0.0244    2.2497    0.0118    0.2274
1.4172   -0.0580   -1.3203    0.8758    1.5730
0.6715    0.0407    0.6771   -1.2240   -0.8732
-1.2075   -1.0379   -0.9361    0.5721   -0.3556


D3 =

-3.0365
```

```
        2.8775
        3.9304
        -2.4170
        1.6228


        p3 =

        1     5     3     2     4


        k3 =

        5
```

4.1.4 1000x100 Matrix

Dieser Test wird aus übersichtlichkeitsgründen nicht im PDF angeführt. Beiliegend ist jedoch die Matlab Workspace-Datei *TestsQRFact.mat* in der alle Tests mit Inputs und Outpus abgespeichert sind.

4.2 QRSolve Tests

4.2.1 Test 1

```
          >> b1 = randMatrix(3,1,1)

          b1 =

          1.4188
          -1.9819
          -0.2029

          >> A1 = randMatrix(3,3,3)

          A1 =

          -1.2212   -1.7193   -1.2536
          -0.0602    0.1326   -1.8723
          -1.6034   -0.2888   -0.8403

          >> [B1,D1,p1,k1] = QRFact(A1)

          B1 =

          -3.2377    1.2670    1.4834
          -0.0602    0.1428   -3.7142
          -1.6034    2.3929    0.5151


          D1 =

          2.0164
          1.8928
          -1.1964


          p1 =

          1     3     2


          k1 =

          3

          >> x1 = QRSolve(B1, D1, p1, k1, b1)

          x1 =

          -0.1169
          -1.4422
          0.9601

          >> A1 * x1

          ans =

          1.4188
          -1.9819
```

```
-0.2029

x1alt = linsolve(A1,b1)

x1alt =

-0.1169
-1.4422
0.9601

>> format long
>> F_abs = norm(x1 - x1alt)

F_abs =

4.284169974453670e-16

>> F_rel = F_abs / norm(x1)

F_rel =

2.467087919033295e-16
```

4.2.2 Test 2

Test wurde mit einer 100x100 Matrix durchgeführt, die jedoch nur Rang 99 hat. Wie erwartet ist das Ergebnis der 0 Vektor.

```
A2 = randMatrix(100,100,99)

...

[B2,D2,p2,k2] = QRFact(A2)

...

b2 = randMatrix(100,1,1)

...

x2 = QRSolve(B2,D2,p2,k2,b2)

x2 =

[]
```

4.2.3 Test 3

```
>> A3 = randMatrix(5,5,5)

A3 =

1.6703    0.9527   -0.5493    1.1881   -0.3618
-1.5417    0.1314   -0.3175   -1.2128   -0.4264
-0.2720   -1.7419   -0.5827   -0.3812    0.3871
0.3416    0.6678   -0.1642    1.3227    0.9028
```

```
        0.4844    0.8982   -0.9351    0.3853    0.7178

        >> b3 = randMatrix(5,1,1)

        b3 =

        1.7361
        -1.0088
        -0.1800
        -2.0265
        0.4089

        >> [B3,D3,p3,k3] = QRFact(A3)

        B3 =

        4.0350   -1.0681    0.3292   -1.9437   -0.2553
        -1.5417   -0.0763   -0.4769    0.6954   -1.7814
        -0.2720   -0.1612   -1.8231    0.3048    0.3800
        0.3416    3.1299   -0.1800   -0.4438    0.9118
        0.4844    1.6687   -0.9020   -1.2232    0.7306


        D3 =

        -2.3647
        1.3144
        1.1435
        -2.0098
        0.6116


        p3 =

        1     5     3     2     4


        k3 =

        5

        >> x3 = QRSolve(B3,D3,p3,k3,b3)

        x3 =

        2.3944
        0.1683
        -0.0477
        -2.1173
        -0.1821

        >> x3alt = linsolve(A3,b3)

        x3alt =

        2.3944
        0.1683
        -0.0477
        -2.1173
        -0.1821
```

```

          >> format long
          >> F_abs = norm(x3 - x3alt)

          F_abs =

          7.157831469485814e-16

          >> F_rel = F_abs / norm(x3)

          F_rel =

          2.232497222110345e-16
```

4.2.4 Test 4

```
          >> A4 = randMatrix(100,100,100)

          ...

          >> b4 = randMatrix(100,1,1)

          ...

          >> [B4,D4,p4,k4] = QRFact(A4)

          ...

          >> x4 = QRSolve(B4,D4,p4,k4,b4)

          ...

          >> x4alt = linsolve(A4,b4)

          ...

          >> F_abs = norm(x4 - x4alt)

          F_abs =

          1.951423563331391e-12

          >> F_rel = F_abs / norm(x4)

          F_rel =

          4.991915427983470e-14
```

4.3 CompR, CompQ Tests

Für diese Tests wurden die Matrizen aus den Tests für QRFact verwendet.

4.3.1 2x2 Matrix

```
>> R1 = CompR(B1,D1,p1,k1)

R1 =

-1.9111   -0.1918
0   -2.4102

>> Q1 = CompQ(B1,p1,k1)

Q1 =

-0.2813    0.9596
-0.9596   -0.2813

>> F_rel1 = norm(Q1*R1-A1(:,p1))/norm(A1)

F_rel1 =

9.1373e-17

>> [Q1alt,R1alt,e1alt] = qr(A1,'vector')

Q1alt =

-0.9343    0.3566
0.3566    0.9343


R1alt =

2.4178    0.1516
0    1.9051


e1alt =

2     1

>> F_rel1alt = norm(Q1alt*R1alt-A1(:,e1alt))/norm(A1)

F_rel1alt =

4.5686e-17
```

4.3.2 4x2 Matrix

```
>> R2 = CompR(B2,D2,p2,k2)

R2 =

-1.4550    0.5881
0   -5.5823
0         0
0         0

>> Q2 = CompQ(B2,p2,k2)

Q2 =

-0.2191   -0.6641    0.3896   -0.5993
0.8987   -0.4014   -0.1764    0.0016
0.2980    0.2732    0.8983    0.1723
-0.2355   -0.5685    0.1011    0.7818

>> F_rel2 = norm(Q2*R2-A2(:,p2))/norm(A2)

F_rel2 =

2.2812e-16

>> [Q2alt,R2alt,e2alt] = qr(A2,'vector')

Q2alt =

-0.6375    0.2875    0.3544   -0.6208
-0.4934   -0.8517   -0.1760    0.0118
0.2405   -0.3250    0.9067    0.1202
-0.5407    0.2937    0.1460    0.7746


R2alt =

-5.6132    0.1525
0    1.4470
0         0
0         0


e2alt =

2     1

>> F_rel2alt = norm(Q2alt*R2alt-A2(:,e2alt))/norm(A2)

F_rel2alt =

1.1206e-16
```

4.3.3 10x5 Matrix

```
        >> R3 = CompR(B3,D3,p3,k3)

        R3 =

        -3.0365   -0.0047    1.5019   -0.4257   -0.9862
        0    2.8775    0.6194   -1.2958    0.5266
        0         0    3.9304    0.4595    0.5611
        0         0         0   -2.4170    0.5911
```

```
         0         0         0         0    1.6228
         0         0         0         0         0
         0         0         0         0         0
         0         0         0         0         0
         0         0         0         0         0
         0         0         0         0         0

>> Q3 = CompQ(B3,p3,k3)

Q3 =

   -0.2389   -0.4223   -0.1141   -0.0500    0.2462   -0.3985   -0.4760   -0.4206
      -0.1194    0.3345
    0.0208   -0.3869   -0.1529   -0.4998    0.5659    0.3017    0.1566    0.0997
       0.3001   -0.2022
   -0.2354   -0.0028   -0.6587   -0.2846   -0.3437   -0.0747    0.4742   -0.1845
      -0.1931    0.0990
    0.0675    0.5327    0.2562   -0.6769    0.0076   -0.0422   -0.1075   -0.0992
       0.0718    0.3993
    0.0409   -0.2674    0.1093   -0.1438    0.0246    0.1095    0.0050    0.5769
     -0.6716    0.3126
   -0.4906    0.1283   -0.0248    0.1385    0.0052    0.7773   -0.2156   -0.2018
      -0.0914    0.1528
   -0.4640   -0.0792    0.5384    0.1049    0.1930   -0.1720    0.6139   -0.1323
      -0.0023    0.1275
   -0.4667    0.3876   -0.3182    0.1397    0.3333   -0.2805   -0.1240    0.5259
       0.1685    0.0337
   -0.2211   -0.3788    0.1182   -0.1030   -0.5470    0.0164   -0.1078    0.3184
       0.5527    0.2551
    0.3977    0.0120   -0.2153    0.3572    0.2298    0.1346    0.2492   -0.0186
       0.2448    0.6888

>> F_rel3 = norm(Q3*R3-A3(:,p3))/norm(A3)

F_rel3 =

   4.7262e-16

>> [Q3alt,R3alt,e3alt] = qr(A3,'vector')

Q3alt =

   -0.2513   -0.3896    0.1830   -0.0500    0.2462   -0.4375   -0.2117   -0.5951
      -0.1097    0.2934
   -0.1903   -0.3632   -0.0734   -0.4998    0.5659    0.3197    0.0105    0.1914
       0.2934   -0.1773
   -0.6923    0.0991   -0.0154   -0.2846   -0.3437    0.0292    0.5331   -0.0704
      -0.1457    0.0190
    0.3382    0.4887    0.0276   -0.6769    0.0076   -0.0606   -0.0196   -0.1645
       0.0729    0.3890
    0.0765   -0.2816    0.0012   -0.1438    0.0246    0.0270   -0.0534    0.4917
     -0.7104    0.3790
   -0.1775    0.1565    0.4492    0.1385    0.0052    0.7380   -0.2776   -0.2266
      -0.1132    0.1929
    0.3222   -0.1265    0.6258    0.1049    0.1930   -0.0459    0.6585    0.0184
       0.0526    0.0371
   -0.4024    0.4515    0.3217    0.1397    0.3333   -0.3563   -0.1957    0.4559
       0.1312    0.0992
   -0.0240   -0.3790    0.2493   -0.1030   -0.5470   -0.0433   -0.1913    0.2757
       0.5207    0.3134
```

```
-0.0568    0.0198   -0.4483    0.3572    0.2298    0.1598    0.2865   -0.0016
       0.2585    0.6637


R3alt =

4.2529    0.4174   -1.0723    0.0856    0.2470
0    2.8470    0.1622   -1.3215    0.4976
0         0    2.8362    0.5637    1.1207
0         0         0   -2.4170    0.5911
0         0         0         0    1.6228
0         0         0         0         0
0         0         0         0         0
0         0         0         0         0
0         0         0         0         0
0         0         0         0         0


e3alt =

3     5     1     2     4

>> F_rel3alt = norm(Q3alt*R3alt-A3(:,e3alt))/norm(A3)

F_rel3alt =

3.8923e-16
```

4.3.4 1000x100 Matrix

```
>> R4 = CompR(B4,D4,p4,k4)

...

>> Q4 = CompQ(B4,p4,k4)

...


>> F_rel4 = norm(Q4*R4-A4(:,p4))/norm(A4)

F_rel4 =

2.0532e-15

>> [Q4alt,R4alt,e4alt] = qr(A4,'vector')

...

>> F_rel4alt = norm(Q4alt*R4alt-A4(:,e4alt))/norm(A4)

F_rel4alt =

5.5636e-16
```