



Labtentamen VT07-01

Beakta:

- Innan Du titta på mönsterlösningen lös uppgifterna själva och låt en kamrat granska dina lösningar.
- Efter kamratgranskning kan ni använda mönsterlösningar.
- Det finns flera sätt att lösa de givna problem. Mönsterlösningen är bara *ett* sätt.

Uppgifter: 5 Poäng: 20p Godkänt: 10p

Uppgift 1 (4p):

```
>> v=[1,2,3]';w=[0,1,6]';z=[-1,0,5]';
>> X=[v,w,z];
>> A=X*Lambda*inv(X)
```

A =

```
   -1.2500    1.5000   -0.2500
  -20.0000   17.0000   -4.0000
 -108.7500   88.5000  -21.7500
```

```
>> [egenvektor,egenvarde]=eig(A)
```

egenvektor =

```
   -0.0000   -0.1961   -0.2673
   -0.1644   -0.0000   -0.5345
   -0.9864    0.9806   -0.8018
```

egenvarde =

```
  -7.0000         0         0
         0    0.0000         0
         0         0    1.0000
```

v , w och z skiljer sig från MATLABs egenvektorer bara i längden.
Efter normalisering är de identiska.

Uppgift 2 (4p):

2

```
A= [58    38    50
     38    42    23
     50    23    79];
b=[1 2 3]';c=[0 2 0]';
[L,U]=lu(A)
```

L =

```
1.0000    0    0
0.6552    1.0000    0
0.8621   -0.5706    1.0000
```

U =

```
58.0000   38.0000   50.0000
    0   17.1034   -9.7586
    0    0   30.3286
```

```
y=L\b;x=U\y
```

x =

```
-0.1527
 0.1333
 0.0958
```

```
disp('Test')
```

Test

A*x-b

ans =

```
1.0e-014 *
```

```
-0.0888
-0.0888
-0.2665
```

```
y=L\c;x=U\y;
```

```
y=L\c;x=U\y;
```

```
disp('Test')
```

Test

A*x-c

```
ans =
    1.0e-015 *
    0.4441
    0.8882
    0.4441
```

Uppgift 3 (4p):

Först skriver vi systemet i matris/vektorform

```
A =
    1    0    3
   -7   -4   -5
    2    5    0
    0    1    3
    1    6    0
```

```
>> b=[1 6 0 0 0]'
```

```
b =
    1
    6
    0
    0
    0
```

För att systemet hade en lösning så måste rangen av matrisen $[A, b]$ vara densamma som rangen av A , dvs b måste vara linjärt beroende av A s kolonner. Låt oss testa detta:

```
>> rank(A)
```

```
ans =
    3
```

```
>> rank([A, b])
```

```
ans =
    4
```

Altså har systemet ingen lösning.

Vi löser systemet mha minsta kvadrat metoden:

```
>> x=(A'*A)\(A'*b)
```

4

```
x =  
  
-1.0627  
0.2375  
0.1842
```

Residualvektorn och dess längd:

```
>> r=A*x-b
```

```
r =  
  
-1.5101  
-0.4320  
-0.9381  
0.7900  
0.3620
```

```
>> norm(r)
```

```
ans =
```

```
2.0254
```

Mätningar som gör att systemet ha en entydig lösning erhålls t.ex. från minsta kvadratlösningen:

```
>> b1=A*x
```

```
b1 =
```

```
-0.5101  
5.5680  
-0.9381  
0.7900  
0.3620
```

Test:

```
>> rank([A,b1])
```

```
ans =
```

```
3
```

Fint.

Uppgift 4 (4p):

```
>> A=[1 2  
3 4  
-4 5]
```

A =

```

      1      2
      3      4
     -4      5

```

Vi bildar P :

```
>> P=A*inv(A'*A)*(A')
```

P =

```

    0.1526    0.3554    0.0547
    0.3554    0.8510   -0.0229
    0.0547   -0.0229    0.9965

```

... eller på ett alternativt sätt (lägg märke till klamrar!):

```
>> P=A*((A'*A)\A')
```

P =

```

    0.1526    0.3554    0.0547
    0.3554    0.8510   -0.0229
    0.0547   -0.0229    0.9965

```

En ortogonalprojektion beskrivs av en symmetrisk matris som har egenskapen $PP = P$. Symmetrin ser man direkt. Det andra testa vi:

```
>> P*P-P
```

ans =

```

    1.0e-015 *
    0.0278    0.0555         0
    0.1110    0.2220   -0.0035
    0.0278    0.0520    0.1110

```

Rangen kan avläsas från antalet icke noll elementer i R s diagonal:

```
>> [Q,R]=qr(P)
```

Q =

```

   -0.3906    0.0000   -0.9206
   -0.9099   -0.1521    0.3860
   -0.1400    0.9884    0.0594

```

R =

```

   -0.3906   -0.9099   -0.1400
         0   -0.1521    0.9884

```

```

0      0      0.0000
Ty rangen är 2.
>> [egenvektor, egenvarde]=eig(P)

```

```
egenvektor =
```

```

-0.9206    0.3906   -0.0211
 0.3860    0.9099   -0.2009
 0.0594    0.1400    0.9794

```

```
egenvarde =
```

```

0      0      0
0      1      0
0      0      1

```

```
>> v=egenvektor(:,2),w=egenvektor(:,3)
```

```
v =
```

```

0.3906
0.9099
0.1400

```

```
w =
```

```

-0.0211
-0.2009
0.9794

```

Planets ekvation i parameterform är $z = tv + sw$.

Uppgift 5 (4p):

Programmet

```

function z=BmultVektor(v)
% v 20 x 1 vector (input)
% z 20 x 1 vector (output), where z=B*v

n=length(v)
if n~= 20
    error('Input vector has wrong dimension')
end
B=diag([1:1:20])+diag([1:1:19],1)+diag([1:1:19],-1);
for i= 1:20
    sum=0;
    for j=max(i-1,1):min(i+1,n); % det g{\aa}r ocks{\aa} att g\{o}ra med if-ko
        sum=sum+B(i,j)*v(j);
    end
end

```

```
        end;  
        z(i)=sum;  
    end;  
    z=z';  
    Test  
>> z=BmultVektor(v);  
>> B=diag([1:1:20])+diag([1:1:19],1)+diag([1:1:19],-1);  
>> zz=B*v;  
>> z-zz
```

```
ans =
```

```
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0
```

```
Hurra.
```