HF1006 Laboration MATLAB
Inlämningsuppgift Linjäralgebra, HT22

Här ska vi titta på scriptfiler där vi samla alla kommandon i en fil samt lära oss fler kommandon som användas vid beräkningar med vektorer. Vi ska även titta på hur man löser ekvationer och hur MATLAB kan användas vid matrisberäkningar.

Bra att arbeta igenom exemplen för att öva innan du ger dig på uppgifterna.

Alla arbetsuppgifter ska skrivas i editorn och sparas som m-filer. När du är klar med alla uppgifter ska du ladda upp m-filerna i canvas senast på redovisningsdagen. En m-fil per uppgift. Lägg till kommentarer i dina m-filer! Hur man skriver kommentarer se längre ner.

Scriptfiler

Normalt arbetar MATLAB i en kommandodriven mod, där man rad för rad matar in kommandon via tangentbordet. MATLAB kan också exekvera en sekvens av kommandon som samlas i en fil. En serie kommandon samlade i en fil utgör ett program eller ett script. Genom att skriva namnet på filen vid kommandoprompten kommer programmet att exekveras. Exekveringen innebär att kommandona i filen utförs i ordning uppifrån och ner. Om något blir fel kan man gå in i filen, göra ändringar och köra filen på nytt. I MATLAB har filen som innehåller kommandon alltid extensionen **.m** och kallas därför m-filer. M-filer skapas bäst i MatLabs editor vilken startas genom att välja *New Script* under *home*-menyn. Filnamnet för inte börja med siffror eller innehålla punkter.

Exempel på tillåtna namn: uppgift1 uppgift_1 OvningMATLAB

Exempel på otillåtna namn: u 1a 1uppgift U-1 U.1 övning

Scriptfiler används huvudsakligen då det är opraktiskt att skriva direkt i Matlab-fönstret, till exempel om man skall skriva in en lång rad av kommandon, stora matriser och liknande, eller om man vill ha kvar de kommandon som skrivs in för senare bruk.

Man kan skriva egna kommentarer var som helst i en m-fil, bara man låter kommentaren inledas med tecknet %. MATLAB ignorerar då resten av raden. Det är klokt att ta för vana att sätta in kommentarer och hjälp-information i sina m-filer. Man glömmer lättare än vad man tror. Dessutom är det ofta hopplöst svårt att begripa vad okommenterade program egentligen gör, till och med om man själv skrivit dem.

Exempel:

```
% M-fil som beräknar summan av serien
% 1 + 1/2 + 1/4 + 1/8 + 1/16 + ...
% Vi summerar de 1000 första elementen i denna serie:
clear %raderar variabler
clc   % rensar kommandofönstret
s = 0;
for n = 1:1000
    s = s + 1/(2*n);
end
s % Visa resultatet av beräkningarna
```

Spara filen som summa1.m, gå sedan över till MATLABs kommandofönster. Skriv vid kommandoprompten summa1 så kommer alla kommandon i denna fil att utföras och resultera i:

```
s =

    3.7427
```

Om man råkar definiera en variabel eller m-fil med samma namn som någon av MATLABs fördefinierade funktioner (definierat_namn) så har det namnet man själv definierat högre prioritet. Vill man då återfå den inbyggda funktionens namn raderar man bara den nya med hjälp av clear definierat_namn.

Vektorer

En vektor är en samling ordnande reella eller komplexa tal $v = (v_1, v_2, \dots, v_n)$.

I MATLAB fås vektorer genom att skriva elementen inom hakparentes v = [v1 v2 … vn]

Eller genom att tilldela ett och ett v(1)=v1, v(2)=v2, … , v(n)=vn

Låt $\vec{u} = (x_1, y_1, z_1)$ och $\vec{v} = (x_2, y_2, z_2)$ vara två vektorer i rummet. Med den skalära produkten $\vec{u} \bullet \vec{v}$ av vektorerna menas det reella talet $|\vec{u}||\vec{v}|\cos\alpha$.

Normen av vektorn $\vec{u}$ som motsvarar absolutbelopp ges av $|\vec{u}| = \sqrt{x_1^2 + y_1^2 + z_1^2}$.

Vektorerna $\vec{u}$ och $\vec{v}$ är ortogonala eller vinkelräta om $\vec{u} \bullet \vec{v} = 0$.

En enhetsvektor är en vektor $\vec{u}$ med $|\vec{u}| = 1$. Enhetsvektor $\vec{u}$ med samma riktning som vektorn $\vec{v}$ ges av $\vec{u} = \frac{1}{|\vec{v}|}\vec{v}$ som i MATLAB skrivs:

```
>> u=v/norm(v)
```

Vektorprodukten $\vec{u} \times \vec{v}$ är definierad som vektorn med längden $|\vec{u}||\vec{v}|\sin\alpha$ som är ortogonal mot både $\vec{u}$ och $\vec{v}$.

Nedan visas en lista över några kommandon som användbara i denna övning:

norm(u)	ger längden på vektorn $\vec{u}$
dot(u,v)	beräknar skalärprodukten $\vec{u} \bullet \vec{v}$
cross(u,v)	beräknar vektorprodukten $\vec{u} \times \vec{v}$
solve(p,r)	löser ekvationen *p=0*, där *p* är ett symboliskt uttryck med avseende på variabeln *r*. Då r utelämnas löses ekvationen med avseende på defaultvariabeln.
solve(p_1, p_2, …, p_n,r_1, r_2, …, r_n)	löser ekvationssystemet $p_1=0$, $p_2=0$, … $p_n=0$ med avseende på variablarna r_1, r_2, …, r_n.
subs(p)	ersätter symboliska variabler i uttrycket p med motsvarande variabelvärde.
subs(p,old,new)	ersätter den symboliska variabel old i uttrycket p med new, där new är en symbolisk eller numerisk variabel eller ett uttryck. Fungerar även för ersättning av flera symboliska variabler, vilka då skrivs inom klammerparanteser.
acos(x)	arccos x, svarar i radianer
acosd(x)	arccos x, svarar i grader

Nedan visas exempel på användning av kommandot subs:

```
% Definierar ett symboliskt uttryck som lagras i plan
syms x y z
plan=3*x+4*y-8*z +2;

x=1;y=2;z=-8;
subs(plan) % erätter variablerna x,y,z i uttrycket plan med variabelvärdena 1, 2 och -8
```

Vilket ger:

```
ans =
77
```

Ett annat sätt att skriva exemplet ovan:

```
syms x y z
plan=3*x+4*y-8*z +2;
subs(plan,{x y z},{1 2 -8}) % erätter variablerna x,y,z (old) i uttrycket plan med variabelvärdena 1, 2 och -8 (new)
```

Nedan visas fyra exempel på hur MATLAB har använts för att lösa problem med vektorer, plan och linjer.

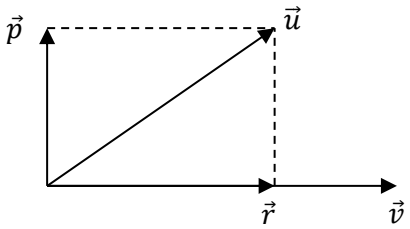
Exempel 1: Ett plan bestäms av en normalvektor $\vec{n}$ och en punkt A. Punkten P=(x,y,z) ligger i planet precis då vektorn $\overrightarrow{AP}$ är vinkelrät mot normalvektorn, dvs så $\vec{n} \bullet \overrightarrow{AP} = 0$. Detta definierar planets ekvation. Låt $\vec{n} = (1,2,3)$ och en punkt A=(4,5,6). Vänsteledet i planets ekvation beräknas då genom:

```
syms x y z % Deklarerar att x,y och z är symboliska variabler
n=[1,2,3];
A=[4,5,6];
P=[x,y,z];
AP=P-A;
planet=dot(n,AP)
```

Exempel 2: Låt $\vec{r}$ vara projektionen av vektorn $\vec{u}$ på vektorn $\vec{v}$, se figuren till höger. Enligt projektionsformeln är

$\vec{r} = t\overrightarrow{v_e}$ där $t = \vec{u} \bullet \overrightarrow{v_e}$ och $\overrightarrow{v_e}$ enhetsvektorn, $\overrightarrow{v_e} = \frac{\vec{v}}{|\vec{v}|}$,

dvs $\vec{r} = \frac{\vec{u} \bullet \vec{v}}{|\vec{v}|^2}\vec{v}$ ($|\vec{v}|^2 = \vec{v} \bullet \vec{v}$).



Den ortogonala projektionen $\vec{p}$ fås genom:

$$\vec{p} = \vec{u} - \vec{r}$$

I MATLAB:

```
u=[1,-2,3];
v=[-2,5,1];
r=(dot(u,v)/dot(v,v))*v
p=u-r
```

ger utskriften:

```
r =

    0.6000   -1.5000   -0.3000


p =

    0.4000   -0.5000    3.3000
```

Om man lägger till u=sym(u) och v=sym(v) (eller börjar med syms u v) fås exakta svar.

```
u=[1,-2,3]; u=sym(u);
v=[-2,5,1]; v=sym(v);
r=(dot(u,v)/dot(v,v))*v
p=u-r
```

Vilket ger:

```
r =

[3/5, -3/2, -3/10]


p =

[2/5, -1/2, 33/10]
```

Exempel 3: Beräkna skärningspunkten mellan planet 9x-10y+31z-112=0 och linjen genom punkterna P=(1,2,-1) och A=(3,3,3).

I MATLAB:

```
format compact % tätare utkrift
syms x y z % Deklarerar att x,y och z är symboliska variabler
Q=[x,y,z]; % Godtycklig punkt
plan=9*x-10*y+31*z-112; % ej nödvändigt att skriva HL=0.
P=[1,2,-1];
A=[3,3,3];
% skriver linjen på parameterform
syms t
linje=P+t*(A-P)  % ger x=2t+1, y=t+2, z=4t-1
% insättning av x, y och z i planets ekvation för att bestämma
% parametern t(här t0)
linjeplan=subs(plan,Q,linje)  % kan skrivas subs(plan,{x y z},linje) istället för Q.
t0=solve(linjeplan)  % löser ekvationen 132t-154=0.
punkt=subs(linje,t,t0) % insättning av t i linjens ekvation för att bestämma skärningspunkten
subs(plan,Q,punkt) % kontroll av att punkten ligger i planet
```

Utskriften blir då:

```
linje =
[2*t + 1, t + 2, 4*t - 1]
linjeplan =
132*t - 154
t0 =
7/6
punkt =
[10/3, 19/6, 11/3]
ans =
0
```

Lös följande uppgifterna först förhand och sedan i MATLAB.

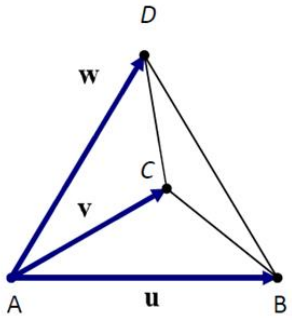
Uppgift 1: Låt A=(1, 2, 2), B=(2, 4, 3), C=(4, 8, 4) och D=(3, 5, 5) vara fyra punkter i R^3.

a) Bestäm vektorerna $\vec{u} = \overrightarrow{AB}$, $\vec{v} = \overrightarrow{AC}$ och $\vec{w} = \overrightarrow{AD}$.

b) Beräkna arean av triangeln ABC.

c) Bestäm ekvationen för planet Π som innehåller punkterna A, B och C.

d) Beräkna volymen av pyramiden ABCD.



Uppgift 2: Beräkna den spetsiga vinkeln mellan planet 3x – 5y + 4z = 5 och linjen

$$\frac{x+1}{4} = y-2 = \frac{z-3}{-1}$$

Uppgift 3: Låt $\vec{a} = (1,2,1)$ och $\vec{b} = (1,2,-2)$ vara två vektorer i R^3. Bestäm två vektorer $\vec{p}$ och $\vec{q}$ sådana att $\vec{a} = \vec{p} + \vec{q}$ och att $\vec{p}$ är parallell med $\vec{b}$, medan $\vec{q}$ är vinkelrät mot $\vec{b}$.

Uppgift 4: Givet i R^3 är:

punkterna P=(0, -1, 1) och Q=(3, 0, -3),

den räta linjen $L = \begin{cases} x = 2t \\ y = t \\ z = 2 + 2t \end{cases}$, $t \in \mathbb{R}$,

sfären S: $x^2 + y^2 + z^2 - 2x - 2y + 2z - 6 = 0$ och

planet Π: $2x - y + 4 = 0$.

a) Planet Ω innehåller punkten *P* och den räta linjen *L*. Visa att $x + 2y - 2z + 4 = 0$ är en ekvation för Ω.

b) Visa att planen Π och Ω är vinkelräta mot varandra.

*c) Bestäm koordinaterna för centrum *C* och radien *R* till sfären *S*. Frivilligt och för den intresserade!

***Uppgift 5:** Betrakta planen Π_1: x – 2y + 2z + 10 = 0 och Π_2: $\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 6 \end{pmatrix} + s\begin{pmatrix} 2 \\ 0 \\ -1 \end{pmatrix} + t\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$.

Beräkna avståndet mellan Π_1 och Π_2. Frivilligt och för den intresserade!

Linjära och icke-linjära ekvationssystem

Linjära ekvationssystem är vanligt förekommande i olika tillämpningar.

Följande kommando kommer att användas:

solve(p_1, p_2, …, p_n,r_1, r_2, …, r_n) löser ekvationssystemet $p_1=0$, $p_2=0$, … $p_n=0$ med avseende på variablarna r_1, r_2, …, r_n.

Exempel 4: Följande linjära ekvationssystem har samma antal ekvationer som antalet obekanta:

$$\begin{cases} x+y+2z=3 \\ 2x+y+z=4 \\ 2x+y+3z=4 \end{cases}$$ har lösningen

I MATLAB skrivs:

```
clc
clear
format compact % tätare utskrift
syms x y z
p1=x+y+2*z==3;  % Observera ==3, går bra att i stället skriva x+y+2*z-3
p2= 2*x+y+z==4;
p3=2*x+y+3*z==4;
[X,Y,Z]=solve(p1,p2,p3,x,y,z); % löser systemet, går bra att utelämna x,y,z i parentesen
S=[X,Y,Z] % lösningen
```

Exempel 5: Undersök om planen $x+y+z=3$ och $x+2y+2z=4$ skär varandra.

Två möjligheter när det handlar om plan: 1. Planen är parallella därmed saknar skärnings eller 2. Planen är ej parallella och skär varandra i en linje.

De två planen ger ett ekvationssystem som är underbestämd. Om det skrivs:

```
syms x y z
ekv1=x+y+z==3;  % här kan man istället skriva ekv1=x+y+z-3
ekv2= x+2*y+2*z==4; % här kan man istället skriva ekv2=x+2*y+2*z-4
[x y z]=solve(ekv1,ekv2,x,y,z)
```

så ger MATLAB bara en lösning av oändligt många:

```
x =
2
y =
1
z =
0
```

Om det istället skrivs:

```
syms x y z
ekv1=x+y+z==3;
ekv2= x+2*y+2*z==4;
S=solve(ekv1,ekv2,x,y,z,'ReturnConditions',true) % returnerar alla lösningar och övrig info i en struktur
[S.x S.y S.z] %skriver ut lösningen
```

Vilket ger följande svar:

```
S =
  struct with fields:

            x: [1×1 sym]
            y: [1×1 sym]
            z: [1×1 sym]
   parameters: [1×1 sym]
   conditions: [1×1 sym]
ans =
[2, 1 - z1, z1]
```

z1 tolkas som en fri parameter och betecknas tex med bokstaven *t*. Linjens ekvation är då:

$$\begin{cases} x = 2 \\ y = 1 - t, \ t \in \mathbb{R} \\ z = t \end{cases}$$

Om planen hade varit parallella och saknar skärningslinje hade svaret i stället blivit:

```
S =
  struct with fields:

            x: [0×1 sym]
            y: [0×1 sym]
            z: [0×1 sym]
   parameters: [1×0 sym]
   conditions: [0×1 sym]
ans =
Empty sym: 0-by-3
```

Uppgifter 6: Lös ekvationssystemen

a) $\begin{cases} x + y + 2z = 9 \\ x + 2y + z = 6 \\ 2x + 3y + 3z = 7 \end{cases}$ b) $\begin{cases} x - y + 2z + u + 3w = 1 \\ 2x + z + 2u + 4w = -2 \\ x + 2y - 2z + 3w = -3 \end{cases}$

Uppgifter 7: Bestäm skärningspunkten mellan de räta linjerna

$$L: \begin{cases} x = 5 + s \\ y = 1 - 2s \\ z = 2 - s \end{cases} \quad \text{och} \quad M: \begin{cases} x = t \\ y = 2 + t \\ z = -2 + 2t \end{cases}.$$

Uppgift 8: För vilka värden på konstanten *a* har ekvationssystemet exakt en lösning? När finns oändligt många lösningar och när saknas lösning?

$$\begin{cases} x+y+z=0 \\ x+2y+az=1 \\ x+ay+2z=-1 \end{cases}$$

Matriser

I MATLAB skrivs matriser genom att skriva elementen inom hakparentes. Alla element kan ges på samma rad där semikolon skiljer raderna åt och elementen skiljs åt med mellanrum eller med ett komma. Transponatet av A, dvs A^T skrivs i MATLAB med specialtecknet ’ (apostrof), se exemplet nedan.

```
>> A=[1 4 -7 6]

A =

     1     4    -7     6

>> B=A'

B =

     1
     4
    -7
     6
```

Addition och subtraktion av matriser görs med matriser av samma dimension och med kommandona + och -. Multiplikation av två matriser, tex A*B kräver att antalet kolonner i A är lika med antalet rader i B. En skalär kan multipliceras med en godtycklig matris, varvid varje element multipliceras med skalären.

Exempel 6:

```
format compact
A=[2,4,-2;-4,5,1;9,-3,-5];
B=[1 1 3;-3 -3 -3;0 1 5];
Addition=A+B
Multiplikation=A*B
MultMedSkalar=pi*A
```

Ger svaret:

```
Addition =
     3     5     1
    -7     2    -2
     9    -2     0
Multiplikation =
   -10   -12   -16
   -19   -18   -22
    18    13    11
MultMedSkalar =
    6.2832   12.5664   -6.2832
  -12.5664   15.7080    3.1416
   28.2743   -9.4248  -15.7080
```

Numeriska matriser kan omvandlas till symbolisk form. Matrisen MultMedSkalar fås på symbolisk form genom

```
>> sym(MultMedSkalar)
ans =
[ 2*pi,  4*pi, -2*pi]
[-4*pi,  5*pi,    pi]
[ 9*pi, -3*pi, -5*pi]
```

Matrisen A av formen $n \times n$ är inverterbar om $A^{-1}A = I$ och $AA^{-1} = I$ där I är enhetsmatrisen. I MATLAB beräknas den inversa matrisen till A genom kommandot inv(A). Kvadratiska matriser som saknar inverser kallas singulära.

Exempel 7:

```
A=[1 2;-1 -1];
A=sym(A);
C=[1;2;3];
InversenA=inv(A)
inv(C)
```

```
InversenA =
[-1, -2]
[ 1,  1]


Error using inv
Matrix must be square.
Error in matriser (line 5)
inv(C)
```

Determinanten till en matris A betecknas det(*A*) eller $|A|$. I MATLAB beräknas determinanten till den kvadratiska matrisen A genom kommandot det(A).

I MATLAB används divisionssymbolerna \ på matriser enligt följande: Om A är en kvadratisk och inverterbar matris så ger X=A\B lösningen till ekvationen $AX = B$, dvs $X = inv(A) * B$.

I MATLAB:

```
% Lösa ekvationen AX=B
A=[1 2 -3;-1 -1 4;1 1 1];
A=sym(A);
B=[1;2;3];
losning1=A\B
Losning2=inv(A)*B
```

```
losning1 =
0
2
1
Losning2 =
0
2
1
```

Exempel 8: För att lösa ekvationssystemet

$$\begin{cases} -x+2y=1 \\ -2x+4y=0 \end{cases}$$

definierar vi koefficientmatrisen och högerledet: $A=\begin{bmatrix} -1 & 2 \\ -2 & 4 \end{bmatrix}$ och $B=\begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

Koefficientmatrisen A är singulär, dvs inte inverterbar eftersom andra raden är en multipel av den första raden. I detta fall saknas lösningar till ekvationssystemet. Nedan visas hur MATLAB hanterar detta.

AX=B

```
A=[-1 2;-2 4];
B=[1;0];
X=A\B
X=inv(A)*B
```

```
Warning: Matrix is singular to working precision.
X =
   Inf
   Inf
Warning: Matrix is singular to working precision.
X =
   NaN
   NaN
```

Inf står för *infinity* och NaN för *Not a Number*.

För inverterbara och icke-inverterbara matriser kan lösningen till AX=B erhållas med hjälp av Gausseliminering i ekvationssytemet till koefficientmatrisen är triangulär, dvs trappstegsformad. Bildar systemets totalmatris som består av koefficientmatrisen följd av högerledet, ofta separerade med ett lodrätt streck för att förtydliga att det handlar om en totalmatris till ett ekvationssystem. I MATLAB erhålls trianguleringen genom att man bildar systemets totalmatris A1=[A B] och sedan beräknar T=rref(A1), kommandot rref är förkortning för *row reduced echelon form*.

AX=B

```
A=[-1 2;-2 4];
B=[1;0];
A1=[A B]
T=rref(A1) % går även bra att hoppa över definition av A1 och direkt T=rref([A B])
```

```
A1 =
    -1     2     1
    -2     4     0
T =
     1    -2     0
     0     0     1
```

Den sista raden i T ger oss ekvationen 0=1 vilket visar att ekvationssystemet $AX=B$ saknar lösning. Även ekvationen $XA=B$ kan lösas med hjälp av rref men då måste ekvationen först transponeras. $(XA)^T=B^T$, som är samma som $A^TX^T=B^T$.

Uppgift 9: Definiera matriserna $A=\begin{bmatrix}1 & -2 & 6\\ 7 & -3 & 5\end{bmatrix}$, $B=\begin{bmatrix}4 & -3 & -2\\ 4 & 4 & 0\end{bmatrix}$, $C=\begin{bmatrix}1 & 2 & -3\\ -1 & -1 & 4\\ 1 & 1 & 1\end{bmatrix}$,

$D=\begin{bmatrix}4 & 0 & -2\end{bmatrix}$ och $E=\begin{bmatrix}2\\ -1\\ -3\end{bmatrix}$.

a) Beräkna A(2,1) +B(2,3) och tolka vad du har beräknat.
b) Beräkna $-2A$
c) Beräkna A+3B
d) Beräkna AE
e) Jämför DE och ED
g) Beräkna och jämför C^2 och C.^2.
h) Beräkna det(C).
i) Beräkna A^2 och tolka svaret.

Uppgift 10: Låt A, C, D och E vara matriserna från uppgift 9.

a) Beräkna A^T. Jämför med A.
b) Inför en tredje rad i A genom att sätta den lika med E^T. Låt A vara den nya matrisen.
c) jämför $(A^T)^{-1}$ och $(A^{-1})^T$.
d) Beräkna $(AC)^T$ och $C^T A^T$. Förklara.

Uppgift 11: I denna uppgift skall du undersöka några olika ekvationssystem och jämför lösningarna som erhålls med de olika metoderna som beskrivs ovan. Matriserna $C=\begin{bmatrix}1 & 2 & -3\\ -1 & -1 & 4\\ 1 & 1 & 1\end{bmatrix}$,

$D=\begin{bmatrix}4 & 0 & -2\end{bmatrix}$ och $E=\begin{bmatrix}2\\ -1\\ -3\end{bmatrix}$, matris A är den du fick i uppgift 10.

a) Lös ekvationssystemen $AX=E$ och $YA=D$ med hjälp av rref.
b) Lös ekvationssystemen $AX=E$ och $YA=D$ med hjälp av MATLABs operationer \.
c) Lös ekvationssystemet $CX=E$ med rref och \.