CHALMERS | GÖTEBORGS UNIVERSITET
Matematiska institutionen
Carl-Henrik Fant, Lennart Falk

8th November 2002

LAB 1: MATRISER OCH EKVATIONSSYSTEM

REDOVISNING:
Starta en dagbok genom att ge kommandot **diary lab1**. Skriv in alla beräkningar som efterfrågas i uppgifterna i dagboken. Glöm inte **diary off** om det skrivna inte ska sparas. Med hjälp av denna diary, kan man lätt redovisa vad man gjort under hela lab-passet.

1 Matrishantering

Detta avsnitt förutsätter att läsaren är bekant med den linjära algebrans mest grundläggande begrepp (t.ex. matrismultiplikation, matrisinvers och system av linjära ekvationer).
Hjälp till detta kapitel hittar du i `helpwin` under `matlab/ops`, `matlab/elmat` och `matlab/matfun`.
I *Pärt-Enander, Sjöberg: Användarhandledning för MATLAB*, se kapitel 5-6.

1.1 Grundläggande matrisoperationer

Man matar in matriser radvis med mellanslag eller kommatecken mellan radelementen och med semikolon eller tryck på return/enter-tangenten mellan raderna. Så till exempel ger inmatningen

```
>>M=[1 2 3; 4 5 6; 7 8 0]
```

resultatet

```
M =
   1  2  3
   4  5  6
   7  8  0
```

En radvektor (radmatris) med konstant differens d mellan elementen kan inmatas enligt mönstret: `R=[a:d:b]`.
Exempel:

```
>>R=[3:2:15]
```

ger resultatet

```
 R=
     3  5  7  9  11  13  15
```

Om ingen differens anges, sätts den till 1:

```
>>S=[3:9]
```

ger resultatet

```
 S=
     3  4  5  6  7  8  9
```

Addition, subtraktion och multiplikation av matriser betecknas med +, − respektive `*`.
Exempel: Med x=[-1;0;2] och M som ovan,

```
>>b= M*x
b =
     5
     8
    -7
```

Uppgift 1: Definiera egna matriser A och B av typ 2×3,
C av typ 3×3, D av typ 1×3 samt F och G av typ 3×1.

a. Testa kommandot *size* på några av matriserna. ($size(A)$)

b. Beräkna $A + B$, $-A$, $2A$ och $3A - 5B$.

c. Försök beräkna $A + C$.

d. Beräkna AC.

e. Beräkna AG.

f. Försök beräkna AB.

g. Jämför DG och GD.

h. Beräkna C^2 och C^{10}.

□

1.2 Transponering och inversmatris

Symbolen för *transponering* är `'` . Transponering av en radvektor ger till exempel kolonnvektor:

```
>> x=[-1 0 2]'
```

ger alltså resultatet

```
x =
   -1
    0
    2
```

Med `M` enligt ovan får man dess invers $N = M^{-1}$:

```
>> N=inv(M)
N =
   -1.7778    0.8889   -0.1111
    1.5556   -0.7778    0.2222
   -0.1111    0.2222   -0.1111
```

Man kan också skriva `A^-1`.
Multiplikation med inversmatris från höger respektive vänster kan skrivas kortare med bråkstreck (slash och backslash): M/A respektive $A\backslash N$ i stället för $M * inv(A)$ respektive $inv(A) * N$.

Uppgift 2: Låt A och G vara matriserna ur uppgift 1.

a. Beräkna A^t (transponatet av A). Jämför med A.

b. Inför en tredje rad i A genom att sätta den lika med G^t. Låt A vara den nya matrisen.

c. Beräkna A^{-1}. Om den skulle sakna invers, ändra något av elementen i A!

d. Testa om vi verkligen har fått inversen till `A`.

e. Jämför $(A^t)^{-1}$ och $(A^{-1})^t$.

f. Testa M/A och $A\backslash N$, där M och N är lämpligt valda matriser. Jämför med $M * inv(A)$ respektive $inv(A) * N$.

□

1.3 Rader, kolonner och enskilda matriselement. Nya matriser av gamla.

Om `M` är en given matris betecknar `M(r,:)` den r:te raden i `M`, `M(:,k)` är den k:te kolonnen och `M(r,k)` ä r elementet på plats (r,k). `M` kan, som nämnts tidigare, uppfattas som en lista med elementen uppräknade kolonnvis. Om `size(M) = [m,n]` så innehåller listan `m*n` element. `M(p)` är element på plats `p` i denna lista. Alltså är `M(r,k) = M(p)` där `p = (k-1)m+r`. Om `u` och `v` är två vektorer med heltalskomponenter så betecknar `M(u,v)` den undermatris av `M` som består av de rader vars index ges av vektorn `u` och vars kolonner är kolonnerna i `M` med index givna av vektorn `v`. Låt oss exemplifiera:

```
>> M=[11:15;21:25;31:35]
M =
   11  12  13  14  15
   21  22  23  24  25
   31  32  33  34  35
>> M(1,:)
ans =
   11  12  13  14  15
>> M(:,1)
ans =
   11
   21
   31
>> M(3,4)
ans =
   34
>> N=M([2 3],[1 3 5])
N =
   21  23  25
   31  33  35
```

Man kan ändra enstaka element eller hela rader och kolonner genom att direkt tilldela dem nya värden. Exempel:

```
>> M(3,4)=134
M =
   11  12  13   14  15
   21  22  23   24  25
   31  32  33  134  35
>> M(:,5)=[1:3]'
M =
```

```
   11   12   13    14   1
   21   22   23    24   2
   31   32   33   134   3
```

Man kan ändra storlek på en matris med direkta tilldelningskommandon. Matrisen utökas eventuellt med tillägg av extra nollor. Exempel:

```
>> N(4,:)=[1 1 1]
N =
   21   23   25
   31   33   35
    0    0    0
    1    1    1
```

Man kan även bygga matriser med andra matriser som ”block”. Exempel:

```
>> P=[[M;2*[1:5]] N]
P =
   11   12   13    14    1   21   23   25
   21   22   23    24    2   31   33   35
   31   32   33   134    3    0    0    0
    2    4    6     8   10    1    1    1
```

Man kan också göra om formen på en $m \times n$-matris med `reshape(M,r,k)`, under förutsättning att $r * k = m * n$. `M(:)` gör om `M` till en kolonnmatris. Ordningen på elementen i listan `M` ändras inte av dessa kommandon.

1.4 Speciella matriser

Det finns ett antal kommandon som är avsedda för att bygga upp nya matriser. Här följer ett urval:

`zeros(n)`	$n \times n$-matris med bara nollor
`zeros(n,m)`	$n \times m$-matris med bara nollor
`eye(n)`	enhetsmatris av typ $n \times n$
`eye(n,m)`	$n \times m$-matris med ettor i diagonalen och nollor för övrigt
`ones(n)`	$n \times n$-matris med bara ettor
`ones(n,m)`	$n \times m$-matris med bara ettor
`rand(n)`	$n \times n$-matris vars element är tal mellan 0 och 1, slumpmässigt utvalda.
`rand(n,m)`	$n \times m$-matris vars element är tal mellan 0 och 1, slumpmässigt utvalda.

Ytterligare några användbara kommandon för att bygga upp nya matriser är

`tril(A)`	ger den undre triangulärmatrisen i den givna matrisen `A`
`triu(A)`	ger den övre triangulärmatrisen i `A`
`diag(V)`	ger, om `V` är en rad- eller kolonnmatris, en matris med `V` på diagonalen och 0 på övriga platser.
`diag(A)`	ger, om `A` är en matris, en kolonnvektor med diagonaleelementen i `A` som element.

Det finns även ett antal speciella matriser som den nyfikne läsaren kan inspektera, dessa hittar du i `helpwin` under `matlab/elmat Specilized matrices`

Uppgift 3: Stora matriser byggda av mindre block.

Bygg en valfri `6x8`-matris med hjälp av de tidigare matriserna `A-G`, specialmatriser ur detta avsnitt och de flesta av metoderna som ges i föregående avsnitt.

□

2 Ekvationssystem

Betrakta ett linjärt ekvationssystem `A*X = b`, där `A` är en $m \times n$-matris, `X` en $n \times 1$-matris (n obekanta) och `b` en $m \times 1$-matris (högerled, m ekvationer). Systemet kan alltid lösas med hjälp av Gauss-elimination i ekvationssystemet tills koefficientmatrisen är trappstegsformad. Ofta väljer man att presentera räkningarna utan att ta med de obekanta. Man bildar då systemets *totalmatris* eller *utökade koefficientmatris* 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. På denna matris gör man sedan radoperationer tills koefficientmatrisen är trappstegsformad, därefter kan man enkelt lösa ut de obekanta ev. på parameterform, eller se att systemet saknar lösning. I Matlab erhålls trappstegsformen genom att man bildar systemets totalmatris `A1 = [A b]` och sedan beräknar `T = rref(A1)`, där `rref` är förkortning av *row reduced echelon form* = *radreducerad trappstegsform*. Alla pivotelement i trappstegsmatrisen `T` är då ettor, och alla element ovanför ett pivotelement är nollor. Ur denna matris `T` kan man bestämma `X`, om lösning finns. Det finns en pedagogisk variant av detta kommando: `rrefmovie`. Man kan då genom tangenttryckningar steg för steg studera eliminationsprocessen. MATLAB utför s k pivotering: före varje eliminationssteg ser man till genom erfordeliga radbyten, att det pivotelement som står på tur har maximalt belopp. Därigenom hålls beräkningsfelen nere.

Om matrisen `A` är kvadratisk och inverterbar, så har systemet entydig lösning `X=inv(A)*b`, vilket som vi tidigare sett också kan skrivas `X=A\b`. Om matrisen `A` inte är inverterbar (men fortfarande kvadratisk), så finns ingen eller oändligt många lösningar. Kommandot `X=A\b` ger dock ett svar, tillsammans med en varning. Då bör man övergå till `rref`.
Om `A` inte är kvadratisk så är `X=A\b` en lösning till `A*X = b` med minstakvadratmetoden (som behandlas senare i kursen i linjär algebra). Det innebär att även om systemet saknar lösning så får man alltså ett svar (“det närmaste man kan komma en lösning”).

I *Pärt-Enander, Sjöberg: Användarhandledning för MATLAB*, kapitel 7, finns ytterligare information om behandling av linjära ekvationssystem med MATLAB.

Några exempel.
Med `A =`

$$\begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 0 \end{matrix}$$

och `b =`

$$\begin{matrix} 5 \\ 8 \\ -7 \end{matrix}$$

ger både `X = inv(A)*b` och `X = A\b`
`X =`

$$\begin{matrix} -1 \\ 0 \\ 2 \end{matrix}$$

Vidare ger `rref([A b])` matrisen

$$\begin{matrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \end{matrix}$$

ur vilken vi direkt kan utläsa samma lösning som ovan.
Låter vi `A =`

$$\begin{matrix} 1 & 2 & 3 \\ -2 & 4 & 1 \\ 3 & -2 & 2 \end{matrix}$$

och `b =`

$$\begin{matrix} 4 \\ 5 \\ 2 \end{matrix}$$

så ger `T = rref([A b])`

```
T =
```

$$\begin{matrix} 1.0000 & 0 & 1.2500 & 0 \\ 0 & 1.0000 & 0.8750 & 0 \\ 0 & 0 & 0 & 1.0000 \end{matrix}$$

Den sista raden ger oss ekvationen `0 = 1` vilket visar att ekvationssystemet `AX = b` saknar lösning.
Däremot ger `X = A\b`

```
X =
```

$$1.0e+15*$$

$$\begin{matrix} 4.2221 \\ 2.9555 \\ -3.3777 \end{matrix}$$

Här ges emellertid en varning som säger oss att detta kanske är fel.
Om vi ändrar högerledet till `b =`

$$\begin{matrix} 4 \\ 5 \\ -1 \end{matrix}$$

så ger r `T = rref([A b])`

```
T =
```

$$\begin{matrix} 1.0000 & 0 & 1.2500 & 0.7500 \\ 0 & 1.0000 & 0.8750 & 1.6250 \\ 0 & 0 & 0 & 0 \end{matrix}$$

Ur denna matris kan vi se att systemet har oändligt många lösningar. Observera att lösningarna mycket lätt kan formuleras i parameterform, med hjälp av trappstegsmatrisen ovan!
Nu ger `X = A\b`

```
X =
```

$$\begin{matrix} -0.5000 \\ 0.7500 \\ 1.0000 \end{matrix}$$

Även nu ges en varning, som måste tas på allvar: den angivna lösningen är ju bara en bland oändligt många.

Uppgift 4: I denna uppgift skall du undersöka några olika ekvationssystem och jämföra lösningarna som erhålls med de olika metoderna som beskrivs ovan. Definiera först följande matriser (E står här för enhetsmatrisen):

$$A = \begin{bmatrix} 3 & -3 & 1 & 0 & 4 \\ 6 & 0 & 2 & 5 & 5 \\ -7 & 7 & 0 & 4 & 1 \\ 4 & 2 & 1 & 6 & 1 \\ -3 & 1 & 1 & 4 & 10 \end{bmatrix}, \text{ B=A-E, } C = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{bmatrix}, \quad D = B * \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \end{bmatrix}.$$

Talen p_1 - p_5 och q_1 - q_5 är de 5 sista siffrorna i era personnummer.
I varje deluppgift, testa alla metoderna: `rref`, `rrefmovie`, användning av `inv(A)` och användning av bråkstreck (backslash). Ange också lösningen/lösningarna där sådana finns. Behandla följande ekvationssysytem:

a. `AX=C`.

b. `BX=C`.

c. `BX=D`.

□