TMV166 Linjär algebra för M

Veckoprogram 3: Matrisalgebra

Kapitel 2.1–2.5. Matrisalgebra.

Den här veckan ska vi studera matrisalgebra, kapitel 2.1–2.5. Vi introducerade matriser i Vecka 2 som ett effektivt sätt att arbeta med linjära ekvationssystem. Nu ska vi studera hur man räknar med matriser: bilda linjär kombination av matriser, multiplicera matriser och invertera matriser. Jag presenterar detta i föreläsning **F7** och **F8** och de rekommenderade övningarna och datorövningarna är på detta material.

Nästa vecka kommer att handla om vektorrum, kapitel 4. Vektorrum behandlas redan i kapitel 2.8–2.9. Redan i föreläsning **F9** kommer jag att presentera 2.8 och 4.1–4.2 parallellt. Övningar på detta material kommer i Vecka 4.

Dugga 1 kommer att vara öppen från söndag 2 februari till söndag 10 februari.

Rekommenderade uppgifter

(PP är förkortning av Practice problems. Här menas att du bör inleda med att göra alla dessa. Du hittar dem direkt före övningarna till respektive avsnitt, lösningar finns i slutet av avsnittet.)

(PP är förkortning av Practice problems. Uppgifter i fet stil demonstreras av övningsledare.)

Avsnitt	
2.1	PP, 2, 5, 7, 10, 11, 15, 16, 17, 33, 22–25
2.2	PP, 1, 5, 7, 9, 10, 13, 26, ***31***, 32, 21–25, 33, 34
2.3	PP, 1, 3, 11, 12, ***13***, 14, 33, 16, 17, 27, 35, 36
2.4	PP, 1, 2, 7, 10, 11, 12, ***15***, 17, 21
2.5	PP, 1, 4, 9, 18, 23, ***26***, 31, 19

Datorövningar

Vi utför operationer på matriser och jämför olika sätt att behandla linjära ekvationssystem.

Mål

- Addera, multiplicera och invertera matriser

2019–02–01

Matrisalgebra

Precis som för vektorer är addition av matriser och multiplikation av matris med skalär definierad elementvis. I MATLAB utförs dessa operationer med `+`, `-` och `*` som vanligt. Till skillnad från vektorer kan vi också multiplicera matriser om deras dimensioner passar ihop. Om $A = (a_{ij}) \in \mathbb{R}^{n\times m}$ och $B = (b_{ij}) \in \mathbb{R}^{m\times r}$ så är $AB \in \mathbb{R}^{n\times r}$ och dess element ges av

$$(AB)_{i,j} = \sum_{k=1}^{m} a_{ik}b_{kj}.$$

Detta är precis "rad-kolonn-regeln" och utförs med `*`.

MATLAB har även elementvis multiplikation $(AB)_{i,j} = a_{i,j}b_{i,j}$. Denna utförs med `.*`, dvs `A.*B`. Detta är användbart vid programmering, men vi studerar inte denna produkt i denna kurs.

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) Kontrollera att dimensionerna är korrekta med kommandot `size`.

b) Försök beräkna $A + B$, $-A$, $2A$, $3A - 5B$ och $A + C$. Förklara eventuella felmeddelanden.

c) Försök beräkna AC, AG och AB. Förklara eventuella felmeddelanden.

d) Jämför DG och GD.

e) Beräkna C^2 och C^{10}.

□

Fler matrisoperationer

Transponatet A^T av en matris A skrivs i MATLAB som `A'`. Exempel:

```
>> A = [1, 2, 3; 4, 5, 6]
A =
     1     2     3
     4     5     6

>> A'
ans =
     1     4
     2     5
     3     6
```

Detta är särskilt användbart för att se till att alla variabler som är tänkta att representera vektorer tolkas som antingen kolonn- eller rad-vektorer. Börjar man blanda uppstår fort problem. Man kan också snabbt definiera en kolonn-vektor genom t.ex. `x = [1 2 3 4]'` och därmed slippa alla semikolon.

Inversen A^{-1} till en matris A beräknas med hjälp av kommandot `inv` (att föredra) eller helt enkelt `A^-1`. Exempel:

```
>> A = [1, 2; 3, 4];
>> inv(A)
ans =
   -2.0000    1.0000
    1.5000   -0.5000
```

Om matrisen skulle sakna invers får man en varning, men ändå ett svar (som inte betyder något):

```
>> A = [1, 0; 0, 0];
>> inv(A)
Warning: Matrix is singular to working precision.

ans =
   Inf   Inf
   Inf   Inf
```

Uppgift 2. Låt A och G vara matriserna från Uppgift 1.

a) Beräkna A^T och 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 via definitionen att det som beräknades verkligen är inversen till A.

e) Verifiera räknelagen $(A^T)^{-1} = (A^{-1})^T$ för denna specifika matris A.

□

Ekvationssystem igen

Man kan använda matrisinversen för att lösa vissa linjära ekvationssystem. Om $A \in \mathbb{R}^{n\times n}$ (kvadratisk matris) och $Ax = b$ har precis en lösning så ges denna av $x = A^{-1}b$. Finns det flera lösningar eller inga lösningar alls så är inte A^{-1} definierad. Detta kan vara effektivt om man skall lösa många system med samma matris A men med olika högerled b. Annars är det dock alltid bättre att radreducera (Gauss-eliminera) istället. Detta kräver färre beräkningar och introducerar mindre fel (från avrundningar, störningar i högerledet, etc.). Även i fallet med många högerled bör man ofta t.ex. LU-faktorisera A istället för att bilda inversen.

Att lösa $Ax = b$ i MATLAB via radreducering kan göras i flera steg via `rref` som vi såg i Vecka 2, men även direkt via `x=A\b`. Om A inte är kvadratisk så får man istället en så kallad minsta-kvadrat-lösning, vilket vi skall studera senare i kursen. Om systemet inte är lösbart är en sådan "lösning" i en viss mening det bästa man kan åstadkomma. Värt att nämna är även syntaxen `A/B`, matrisdivision från höger, vilket motsvarar AB^{-1}, dvs multiplikation med inversen från andra hållet. Liksom för \ gäller detta bara då B^{-1} existerar.

Uppgift 3. Upprepa Uppgift 2 från Vecka 2, men testa istället \ och `inv`. Jämför resultaten med det du fick från `rref`. □