Datorlaboration ::

Använding av Markovkedjor på flöden

Laborationen går ut på att lösa Laborationsuppgifterna 1 till 6 nedan. Laborationen redovisas individuellt genom att skicka laborationens Mathematicafil via mail till mfg@hig.se
Förklara gärna i ord vad du gör.
Mathematicafilen skall kunna köras utan felmeddelanden. (testa sjäv: Evaluation¿Evaluate Notebook)

Problembeskrivning

Ett vanligt problem är att undersöka tidsutvecklingar av olika *tillstånd* . Man beskriver därvid ett tillstånd hos ett system genom en vektor. T ex kan vektorns komponenter vara antalet bilar som finns inne på olika stationer som ett hyrbilsföretag har (så om det finns 100 stationer får vi en vektor med 100 komponenter), eller så kan det vara hur en marknad är uppdelad mellan olika produkter eller varumärken. Hur många äter kalalspuffar, corn flakes, rispuffar, musli, eller inget av dessa. ”Flingtillståndet” ges i detta fall av en vektor med fem komponenter. Tillståndet (eng. State) för uthyrningsfirman eller tillståndet på flingmarknaden beskrivs helt (för de ändamål man är intresserade av) av sin tillståndsvektor.
Tillståndet varierar för det mesta med tiden, och man är förstås mycket intresserad hur detta sker. Vi betraktar här någon viss tidsenhet, t ex en vecka, ett år eller en mikrosekund (man får bestämma sig), så tillståndet vid tiden k (t ex k veckor) ges av en tillståndsvektor med n komponenter

$$\mathbf{x}^{(k)} = \begin{pmatrix} x_1^{(k)} \\ x_2^{(k)} \\ \cdots \\ x_n^{(k)} \end{pmatrix}$$

I fallet med 100 stationer i en hyrfirma har vi alltså $n = 100$ och $x_i^{(k)}$ anger antalet bilar på stationen i vid tidpunkten k, medan i fallet med flingor har vi $n = 5$ och $x_3^{(10)}$ kan ange andelen personer som föredrar rispuffar år 10.
Man frågar sig alltså hur de olika komponenterna $x_i^{(k)}$ beror på tiden k. Vad händer t ex när k blir stor? Svaret på dessa frågor styr hur många platser det bör finnas på de olika hyrstationerna vid olika tider, alternativt hur mycket kalaspuffar en grossist bör köpa in (så att utbudet möter efterfrågan).
Naturligtvis kommer detta tidsberoende i allmänhet vara mycket komplicerat. Men en första approximation är att *tillståndet vid tiden* $k+1$ *bara beror på vad tillståndet var vid tiden* k, dvs att är möjligt att bestämma tillståndet vid tiden $k+1$ bara genom att veta vad det är vid tiden k . Denna egenskap hos det dynamiska systemet kallas *Markovegenskapen.* Vi har alltså

Man säger: systemet har inget ”minne”

$$\mathbf{x}^{(k+1)} = \mathbf{A}(\mathbf{x}^{(k)});$$

för att bestämma $\mathbf{x}^{(k+1)}$ räcker det att veta $\mathbf{x}^{(k)}$. Här kan dock $\mathbf{A}(\mathbf{x}^{(k)})$ vara en komplicerad funktion av alla variabler i $\mathbf{x}^{(k)}$. Nästa approximation är att anta att denna funktion är linjär, vilket ofta är räcker långt (speciellt för oss som befinner oss i en kurs i linjär algebra.), dvs

$$\mathbf{x}^{(k+1)} = A\mathbf{x}^{(k)};$$

där vi har matrisprodukt i högerledet och A är en $n \times n$-matris. Detta innebär t. ex. att om man vet tillståndet vid tiden $k = 0$ (när man startar), så får man tillståndet efter en tidsenhet $\mathbf{x}^{(1)} = A\mathbf{x}^{(0)}$ och efter två tidsenheter $\mathbf{x}^{(2)} = A\mathbf{x}^{(1)} = A(A\mathbf{x}^{(0)}) = A^2\mathbf{x}^{(0)}$, där $A^2 = AA$ är vanliga matrisprodukten. Man får mer allmänt

$$\mathbf{x}^{(k)} = A^k\mathbf{x}^{(0)}$$

där man behöver veta den k:e potensen av matrisen A. Matrisen A brukar kallas systemets *övergångsmatris* (transition matrix)

Vi skall nu betrakta en situation när inget försvinner eller skapas under varje tidssteg, det sker bara en omfördelning. Detta kan man betrakta som *flöden*: allt som strömmar in kommer att strömma ut, allt som strömmar ut har strömmat in. I våra exempel ovan gäller det att totala antalet bilar är oförändrat och totala antalet personer som äter olika flingor är konstant. Det är i sådana situationer praktiskt att låta tillståndsvektorns komponenter ange *andelar* istället för absoluta antal, så att komponenterna adderas ihop till 100 % (dvs till 1)[1]. Vi antar alltså att tillståndet $\mathbf{x}$ har egenskapen

$$\sum_{i=1}^{n} x_i = x_1 + \cdots + x_n = 1$$

och vi vill att $\mathbf{x}^{(1)} = A\mathbf{x}$ också ska ha denna egenskap. T ex om man väljer $\mathbf{x} = \mathbf{e}_j = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ (en etta i position j) så får man att summan av alla komponenter i j:e kolonnen i A skall vara 1. Detta måste då gälla att *alla* kolonner i A har denna egenskap, dvs de adderar ihop till talet 1. Sådana matriser kallas (vänster) *stokastiska matriser*[2]. T ex är följande matriser stokastiska matriser

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & \frac{3}{4} \end{pmatrix}, \begin{pmatrix} 0.2 & 0.4 & 0.3 \\ 0.3 & 0 & 0.3 \\ 0.5 & 0.6 & 0.4 \end{pmatrix}, \begin{pmatrix} 0.606403 & 0.460389 & 0.365792 \\ 0.0445923 & 0.472038 & 0.486586 \\ 0.349005 & 0.0675734 & 0.147622 \end{pmatrix}.$$

Observera att man inte kräver att summan av elementen i en rad blir 1.

För att läsa mer om stokastiska matriser och sk Markovkedjor, se Lays kapitel 4.9 (speciellt Th 18). Markovkedjor är ett viktigt analysverktyg i tillämpningarna.

Tillståndet i en biluthyrningsfirma

Hyrbilsföretaget RentaCar har 10 hyrstationer i en stad med en hyrbilsflotta om sammanlagt 1000 bilar. Varje station har plats för 100 bilar vardera och kunderna kan hyra på vilken station som helst och lämna till baka bilen på vilken annan station som helst.

Vi numrerar stationerna 1-10 och låter vektorn $\mathbf{x} = (x_1, \ldots, x_n)$ beteckna tillståndet för hyrbilarna, dvs x_i är *andelen* bilar som finns på station i.

RentaCar har genomfört en noggrann statistisk undersökning för att se hur bilarna som hyrs på en station fördelar sig på stationerna vid återlämningen. RentaCar utgår från att det går att modellera systemet som en linjärt system med markovegenskapen, dvs fördelningen på stationerna i morgon beror bara på dagens fördelning, och detta beroende är linjärt. Överföringsmatrisen för tillståndet från en dag till nästa har mätts upp till

Observera att summan av varje rad är 100%, eftersom varje bil som hyrs ut antas återlämnas på någon station. Om vi låter matrisen A vara *transponatet* av matrisen i tabellen ovan så blir därmed A en stokastisk matris. Om man vet tillståndet en viss dag $\mathbf{x}^{(0)}$ så fås därmed tillståndet dagen efter genom

$$\mathbf{x}^{(1)} = A\mathbf{x}^{(0)}. \tag{E}$$

T ex blir andelen bilar på station 7 efter ett dygn givet av (se på kolonn nr 7 i tabellen)

$$x_7^{(1)} = 0.03 \cdot x_1^{(0)} + 0.03 \cdot x_2^{(0)} + 0.12 \cdot x_3^{(0)} + 0.02 \cdot x_4^{(0)} + 0.06 \cdot x_5^{(0)} + \\ + 0.01 \cdot x_6^{(0)} + 0.67 \cdot x_7^{(0)} + 0.1 \cdot x_8^{(0)} + 0.01 \cdot x_9^{(0)} + 0 \cdot x_{10}^{(0)}$$

[1]Om man sedan vet (i fallet med hyrbilar) det totala antalet bilar, t ex 1000 bilar, så kommer 100$\mathbf{x}$ ange fördelningar av bilar på de olika stationerna

[2]Man kan tolka komponenterna i matrisen som sannolikheter för att ett tillstånd ska byta till ett annat, därav beteckningen stokastisk matris.

		Stationer för återlämning									
		$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S7$	$S8$	$S9$	$S10$
Uthyrningsstationer	$S1$	63	2	3	1	5	9	3	5	2	7
	$S2$	11	53	7	2	8	9	3	2	4	1
	$S3$	12	1	47	3	11	8	12	3	2	1
	$S4$	7	2	4	59	5	17	2	1	2	1
	$S5$	3	6	7	2	61	8	6	2	3	2
	$S6$	1	3	2	2	4	79	1	3	3	2
	$S7$	13	1	2	1	5	8	67	0	0	3
	$S8$	2	3	1	0	4	2	10	78	0	0
	$S9$	0	2	1	1	0	0	1	0	82	13
	$S10$	0	1	0	1	0	0	0	0	7	91

Tabell 1: En rad i denna tabell visar hur många **procent** av de bilar uthyrda från stationen som återlämnas på de övriga stationerna. T.ex. den sista raden anger hur många procent av de bilar som hyrdes ut från station $S10$ som återlämnas på de övriga stationerna.

Laborationsuppgifter

1. Ställ upp (för hand) alla 10 ekvationer för alla stationer motsvarande ekvation (E) som ett ekvationssystem. Skriv sedan detta system på matrisform där vi kallar den 10×10 matris som uppstår för A.
2. Beräkna vad som händer efter *en dag*, *en vecka*, *en månad* (30 dagar) och *ett år* för RentaCars stationer som dag noll har 100 bilar var.
3. Beräkna vad som händer efter *en dag, en vecka, en månad och ett år* om RentaCar dag noll har alla sina bilar på station 1.
4. Försök se ett mönster. Hur ser fördelningen ut i båda två ovanstående fall. Hur ser fördelningarna ut efter ett år?
5. RentalCar har ett problem. Man tror att genom att *fördela bilarna* på ett annat sätt vid en viss tidpunkt på de olika stationerna så skulle man kunna få en jämnare beläggning av bilar på de olika stationerna och därmed kunna optimera kostnaderna för bilförvaring och minska bilförflyttningskostnaderna. Är detta möjligt att göra, givet att vår modell är korrekt?
6. Ange hur många platser de tio stationerna behöver ha när det har gått lång tid (många dagar) för att det ska finna plats för bilarna om de totalt har 1000 bilar.

Ett förenklat exempel

Om vi tittar på en enklare situation där RentaCar bara har tre stationer med följande överföringsmatris

	$S1$	$S2$	$S3$
$S1$	74	21	5
$S2$	12	68	20
$S3$	11	12	77

Vi ställer upp ekvationerna för denna situation

$$\begin{aligned}
x_1^{(1)} &= 0.74 \cdot x_1^{(0)} + 0.12 \cdot x_2^{(0)} + 0.11 \cdot x_3^{(0)} \\
x_2^{(1)} &= 0.21 \cdot x_1^{(0)} + 0.68 \cdot x_2^{(0)} + 0.2 \cdot x_3^{(0)} \\
x_3^{(1)} &= 0.05 \cdot x_1^{(0)} + 0.2 \cdot x_2^{(0)} + 0.77 \cdot x_3^{(0)}
\end{aligned}$$

Ställer vi upp detta som en matrisekvation så får vi

$$\bar{x}^{(1)} = \underbrace{\begin{pmatrix} 0.74 & 0.12 & 0.11 \\ 0.21 & 0.68 & 0.12 \\ 0.05 & 0.2 & 0.77 \end{pmatrix}}_{=A} \mathbf{x}^{(0)} \quad \text{där} \quad \mathbf{x}^{(1)} = \begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \end{pmatrix} \quad \text{och} \quad \mathbf{x}^{(0)} = \begin{pmatrix} x_1^{(0)} \\ x_2^{(0)} \\ x_3^{(0)} \end{pmatrix}$$

Notera att matrisen A har kolonner vars element summeras ihop till 1. Detta ska vi ha eftersom procentsatserna i varje rad i tabellen ovansummeras till 100%. A blir således transponatet till matrisen i tabellen (och alla komponenter är dividerade med 100).

Låt oss nu se vad som händer om vid dag noll varje station har 100 bilar var. Efter n dagar har man fördelning av bilar mellan stationerna som ges av vektorn $\mathbf{x}^{(n)}$ given av

$$\mathbf{x}^{(n)} = A^n \mathbf{x}^{(0)}$$

där $\mathbf{x}^{(0)}$ är fördelningen av bilar som vi har dag noll. Varje multiplikation med matrisen A svarar mot en uthyrningsdag i vårt resonemang. Om

$$\mathbf{x}^{(0)} = \begin{pmatrix} 100 \\ 100 \\ 100 \end{pmatrix}$$

I Mathematica matar vi in matrisen

```
A=Transpose[{{.74,.21,.05},{.12,.68,.20},{.11,.12,.77}}]
```

och tillståndsvektorn

```
x0={100,100,100}
```

och utför matrisprodukten

```
A.x0
```

vilket svara mot det första resultatet nedan, Vi får (mha Mathematica)

$$\mathbf{x}^{(1)} = \begin{pmatrix} 97. \\ 101. \\ 102. \end{pmatrix}, \quad \mathbf{x}^{(7)} = \begin{pmatrix} 92.238 \\ 100.839 \\ 106.923 \end{pmatrix}, \quad \mathbf{x}^{(30)} = \begin{pmatrix} 91.9086 \\ 100.618 \\ 107.474 \end{pmatrix}, \quad \mathbf{x}^{(365)} = \begin{pmatrix} 91.9086 \\ 100.618 \\ 107.474 \end{pmatrix}$$

Notera att vi får samma svar för en månad och för ett år. Det verkar alltså som om bilfördelningen är stabil efter en månad.
Vi undersöker vad som händer om vi använder en annan startfördelning. Låt

$$\mathbf{x}^{(0)} = \begin{pmatrix} 300 \\ 0 \\ 0 \end{pmatrix}$$

Då får vi

$$\mathbf{x}^{(1)} = \begin{pmatrix} 222. \\ 63. \\ 15. \end{pmatrix}, \quad \mathbf{x}^{(7)} = \begin{pmatrix} 222. \\ 63. \\ 15. \end{pmatrix}, \quad \mathbf{x}^{(30)} = \begin{pmatrix} 91.9086 \\ 100.618 \\ 107.474 \end{pmatrix}, \quad \mathbf{x}^{(365)} = \begin{pmatrix} 91.9086 \\ 100.618 \\ 107.474 \end{pmatrix}$$

Vi får alltså samma slutvektor även från denna startpunkt. Vi borde nog gissa att detta är den fördelning som RentaCar söker efter. Om vi sätter

$$\mathbf{x}^{(0)} = \begin{pmatrix} 92 \\ 101 \\ 107 \end{pmatrix}$$

så får vi

$$\mathbf{x}^{(1)} = \begin{pmatrix} 91.97 \\ 100.84 \\ 107.19 \end{pmatrix} \approx \begin{pmatrix} 92 \\ 101 \\ 107 \end{pmatrix}$$

vilket betyder att fördelningen av bilar håller sig konstant från dag till dag.

$\blacksquare$

Egenvärdet 1

I exemplet så såg vi att vi genom att upprepa multiplikationen med A så kunde vi till slut få fram ett stabilt tillstånd, dvs varje station får in lika återlämnade bilar som de hyr ut. För matrisekvationen innebär det att vi söker en tillståndsvektor $\mathbf{x}$ som uppfyller[3]

$$A\mathbf{x} = \mathbf{x}$$

Detta är en matrisekvation som vi borde kunna lösa. Vi subtraherar fördelningsvektorn från båda led och bryter ut fördelningsvektorn, vilket leder till

$$A\mathbf{x} - \mathbf{x} = 0 \quad \Leftrightarrow \quad (A - I)\mathbf{x} = 0,$$

där I är identitetsmatrisen. Fördelningsvektorn $\mathbf{x}$ är alltså en vektor i nollrummet till matrisen $A - I$. Nu löser vi denna ekvation med Mathematica. Först matar vi in vektorn A

```
A=Transpose[{{.74,.21,.05},{.12,.68,.20},{.11,.12,.77}}]
```

och sedan konstruerar vi matrisen $B = A - I$ genom kommandot

```
B=A-IdentityMatrix[3]
```

Mathematica svarar då med matrisen

$$\begin{pmatrix} -0.26 & 0.12 & 0.11 \\ 0.21 & -0.32 & 0.12 \\ 0.05 & 0.2 & -0.23 \end{pmatrix}$$

Vi vill alltså beräkna nollrummet (kärnan) till denna matris, vilket är enkelt med Mathematicas kommando

```
NullSpace[B]
```

vilket ger resultatet

```
{{0.529561, 0.579742, 0.619245}}
```

Detta är en lista av vektorer; klammerparenteser anger sk listor i Mathematica, så här har vi en lista av listor, där vi uppfattar inre listan som just en vektor. Vill man se denna vektor i mer läsbar form så kan man istället skriva

```
NullSpace[B]//MatrixForm
```

som ger

$$\begin{pmatrix} 0.529561 & 0.579742 & 0.619245 \end{pmatrix}$$

Vill man plocka ut ett element i en lista så använder man konstruktionen [[i]] (i:e elementet i listan), så här

```
u= NullSpace[B][[1]]
```

Vektorn u spänner alltså upp nollrummet till B och Mathematica har sett till att vektorn har längden 1, vilket vi kan verifiera genom att beräkna längden, eller normen som det också kallas genom

```
Norm[u]
```

och här svarar Mathematica med 1. Däremot om man summerar komponenterna (man använder kommandot Total på en lista)

```
Total[u]
```

[3]Denna ekvation är en egenvärdesekvation för egenvärdet 1. Egenvärdesproblem studeras i kursen

som ger

$$1.72855$$

vilket svarar mot en situation med totala antalet bilar 1.72. Det är bättre att ha en tillståndsvektor som anger den procentuella fördelningen av bilar, dvs kompenterna skall addera ihop till 1. Detta fixas dock enkelt genom att multiplicera vektorn $u = (u_1, u_2, u_3)$ med en lämplig skalär k, så att $\mathbf{x} = ku$ får den önskade egenskapen. Dvs

```
Total[k u] =1
```

vilket ger $k = 1/(u_1 + u_2 + u_3)$. Vi sätter

```
x= u/Total[u]
```

varpå Mathematica ger

```
{0.306362, 0.335392, 0.358246}
```

och Total[x]=1. Eftersom vi startade med 100 bilar på varje station, dvs totalt 300 bilar, får vi fördelningen av bilar genom $300x$, dvs

```
{91.9086,100.618,107.474}
```

Du ser att detta är mycket nära resultatet vi kom fram till efter lång tid ovan (efter 365 dagar)! Ovanstående exempel visar i princip hur man kan lösa laborationens uppgifter. Lös motsvarande system, men nu med 10 stationer och beräkna tillståndsvektorerna vid de olika tispunkterna, när man antar att RentaCar har 1000 bilar.

Redovisning

Hela Laborationen redovisas individuellt i ett Mathematicaarbetsblad. Använd gärna mallen som heter "dittnamn.nb" som finns här , och **byt namn på filen så att ditt namn verkligen står där**.
Observera att vissa sk celler i Mathematica är till för att ge input till Mathematica som då evalueras, medan andra är textceller där du kan skriva på vanlig svenska. Det finns goda möjligheter att skapa en bra rapport i Mathematica. Texten skall vara väl presenterad.
Svara på samtliga frågor.
Skicka labben till mig via mail: mailadress mfg@hig.se

DEADLINE

Före tentan som går 20 januari 2015, dvs lämna in senast, 20160119, klockan 23:59
Observera att detta är ett obligatoriskt moment i kursen.

Användbara Mathematicakommandon

Följande mathematicakommandon kan vara till nytta. Använd naturligtvis Mathematicas inbyggda hjälpsystem för att ta reda på exakt vad kommandon gör och hur man ska använda dem.

`MatrixPower, IdentityMatrix, RowReduce, NullSpace, Norm, Transpose, Total, Inverse,`