Oblig 2 - MAT1120

Fredrik Meyer

23. september 2009

Oppgave 1

Linkmatrise:

$$A = \begin{pmatrix} 0 & \frac{1}{3} & 0 & \frac{1}{2} \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 1 & \frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{3} & \frac{1}{2} & 0 \end{pmatrix}$$

En basis til nullrommet til matrisen $A - I$ kan finnes ved å bruke MATLAB. Jeg kjører kommandoen `rref(A-I)` og får følgende:

```
>> rref(A-I)
ans =

    1.0000         0         0   -0.8750
         0    1.0000         0   -1.1250
         0         0    1.0000   -1.2500
         0         0         0         0
```

Dette betyr at $x_1 = \frac{7}{8}x_4$, $x_2 = \frac{9}{8}x_4$, $x_3 = \frac{5}{4}x_4$, og $x_4 \in R$. For at de skal summere til én, løser vi ligningen $\frac{7}{8}x_4 + \frac{9}{8}x_4 + \frac{5}{4}x_4 + x_4 = 1$ for x_4. Dette gir oss at $x_4 = \frac{4}{17}$. Med andre ord har vi score-vektoren $\mathbf{x} = (\frac{7}{34}, \frac{9}{34}, \frac{20}{34}, \frac{16}{34})$. Dermed ser vi at rangeringen for linkmatrisen er:

- Nr 1 - link 3
- Nr 2 - link 4
- Nr 3 - link 2
- Nr 4 - link 1

En basis for nullrommet til $A - I$ er gitt ved den ene vektoren $\mathbf{v} = \mathbf{x} = (\frac{7}{34}, \frac{9}{34}, \frac{20}{34}, \frac{16}{34})$.

Oppgave 2

Linkmatrisen er gitt ved

$$A = \begin{pmatrix} 0 & 1 & \frac{1}{2} & 0 & 0 \\ 1 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Igjen lønner det seg å bruke MATLAB. Jeg kjører kommandoen `rref(A-I`, og får følgende resultat:

```
>> rref(A-I)
ans =

     1    -1     0     0     0
     0     0     1     0     0
     0     0     0     1    -1
     0     0     0     0     0
     0     0     0     0     0
```

Dette betyr på samme måte som i forrige oppgave at nullrommet er alle lineærkombinasjoner av typen

$$\mathbf{x} = x_2 \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + x_5 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

hvor $x_2, x_5 \in R$. De to vektorene utgjør en basis for nullrommet. Legger vi til kravet at elementene skal summere til 1, får vi likningen $2x_2 + 2x_5 = 1$, som gir at $x_2 = \frac{1}{2} - x_5$. Dette kan vi sette inn i lineærkombinasjonen over, og vi får følgende:

$$\mathbf{x} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ 0 \\ 0 \\ 0 \end{pmatrix} + x_5 \begin{pmatrix} -1 \\ -1 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

For at vi skal unngå negative verdier ser vi at $x_5 \in [0, \frac{1}{2}]$. Rangeringen er ikke unik. Dette ser vi hvis vi f.eks setter $x_5 = 0$, for så å sette $x_5 = \frac{1}{2}$. Dette gir to motsatte rangeringer.

Oppgave 3

a)

Ja, linkmatrisene i oppg. 1 og 2 er stokastiske. Det ser vi lett ved å summere elementene i kolonnene. En ikke-stokastisk matrise oppstår f.eks når vi tar med en side som ikke linker til noen andre sider.

b)

Skal forklare at M gitt ved $M = (1-m)A + mS$ er stokastisk og regulær når A er stokastisk. Siden elementene i kolonnene i A summerer til 1, vil elementene i kolonnene i $(1-m)A$ summere til $1-m$. S er også stokastisk, siden $n(\frac{1}{n} + ... + \frac{1}{n}) = 1$, og derfor vil elementene i kolonnene til mS summere til m. Legger vi matrisene sammen, blir "kolonnesummen" derfor $(1-m) + m = 1$. At M også er regulær, er en selvfølge: den består kun av positive ikke null-elementer, og enhver kombinasjon av disse vil også være positiv (ingen minus-operasjoner oppstår under matrisemultiplikasjon).

c)

Skriver opp matrisen (m=0.15) ved hjelp fra MATLAB:

```
>> S = [1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1; 1 1 1 1 1];
>> S = S/5;
>> M = 0.85*A+0.15*S
M =

    0.0300    0.8800    0.4550    0.0300    0.0300
    0.8800    0.0300    0.4550    0.0300    0.0300
    0.0300    0.0300    0.0300    0.0300    0.0300
    0.0300    0.0300    0.0300    0.0300    0.8800
    0.0300    0.0300    0.0300    0.8800    0.0300
```

Finner null-rommet, igjen med hjelp fra MATLAB:

```
>> null(M-eye(5,5))

ans =
   -0.5777
   -0.5777
   -0.0608
   -0.4054
   -0.4054
```

Denne vektoren summerer dog ikke til 1, så vi må skalere den. Vi summerer elementene, og får -2.027. Deler vi hvert element på dette tallet, vil vi få en unik score-vektor: $\mathbf{x} \approx (0.285, 0.285, 0.03, 0.2, 0.2)$.

Oppgave 4

a)

Skal studere en MATLAB-kode.

```
function A=randlinkmatrix(n)        (1)
    A = round(rand(n,n));           (2)
    for k=1:(n-1)
        A(k,k)=0;                   (3)
        if (A(:,k) == 0)            (4)
            A(n,k)=1;
        end
        s = sum(A(:,k));            (5)
        A(:,k) = (1/s)*A(:,k);
    end
```

Det første som skjer (1), er at en funksjon med navnet `randlinkmatrix` blir definert. Den returnerer en matrise A. Så (2) lages nxn-matrisen A slik at den består av tilfeldige tall mellom 0 og n. En for-løkke settes så i gang, og går gjennom matrisen (bortsett fra siste kolonne): først gjøres alle diagonalelementene om til 0. Så sjekker den om en rad består kun av nuller. Hvis den gjør det, legges en null til på nederst på kolonnen. Til slutt skaleres kolonnene slik at summen i hver kolonne er én.

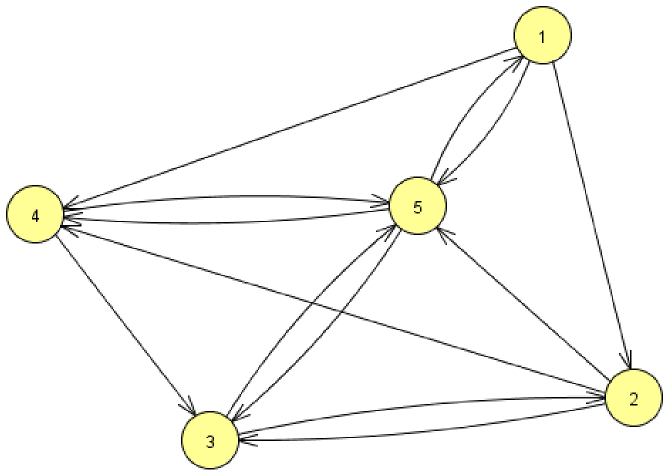
```
A(n,n) = 0;
if (A(:,n) == 0)                (6)
    A(1,n) = 1;
end
s = sum(A(:,n));
A(:,n) = (1/s)*A(:,n);
```

Her skjer omtrent det samme som over. Eneste forskjellen er at hvis siste kolonne (6) kun består av nuller, så blir førstelementet gjort om til 1.

Vi skjønner nå at programmet returnerer en linkmatrise.

b)

Jeg kjører metoden fra a), og utra matrisen kan jeg tegne følgende web:



Oppgave 5

a)

Skal lage funksjonen `ranking(A)` som returnerer score-vektoren for en stokastisk matrise. Programmet gir også feilmelding om A ikke er stokastisk. (denne testingen gjør programmet litt langt og stygt)

```
function x=ranking(A)
    n = size(A);
    if n(1)~=n(2)
        fprintf('Matrisen er ikke kvadratisk!\n');
    end
    for k=1:n(1)-1
        for i=1:n(2)-1
            if A(k,i) < 0
                fprintf('Matrisen er ikke stokastisk!\n');
                return
            end
```

```
        end
    end
    for k=1:n(1)-1
        if sum(A(1))-1 > 0.00001
            fprintf('Matrisen er ikke stokatisk!\n');
            return
        end
    end
    S = (1/n(1))*ones(size(A));
    M = 0.85*A+0.15*S;
    x = null(M-eye(size(A)));
    summ = sum(x);
    x = x/summ;
```

Det er altså de siste linjene som er interessante. Det er her vi lager M, finner nullrommet, og returnerer **x**.

b)

La $x_k = Mx_{k-1}, k \geq 1$. Vi skal programmere funksjonen som returnerer x_k som er slik at $max_j|x_k(j) - x_{k-1}(j)| < \delta$, hvor $x(i)$ viser til element i i vektoren **x**. Denne gangen har vi ikke programmert inn tester for å sjekke om matrisen er stokastisk eller kvadratisk. (om den ikke er stokastisk, kan programmet gå i all evighet).

```
function x=rankingapprox(A,m,delta)
    [t,n] = size(A);
    x = ones(n,1)/n;
    S = ones(n,n)/n;
    M = (1-m)*A+m*S;
    xk = M*x;
    while 1 % for alltid inntil avbrudd
        fo = abs(xk(1)-x(1));
            for k=2:n
                d = abs(xk(k)-x(k));
                if d > fo
                    fo = d;
                end
            end
        if fo < delta
            return
        end
        x = xk;
        xk = M*x;
    end
```

Setter $\delta = 0.005, m = 0.15$. Vi bruker linkmatrisen fra oppgave 1, og ser på hva vi får om vi tester funksjonene våre:

```
>> ranking(A)
```

```
ans =

    0.2116
    0.2614
    0.2915
    0.2354

>> rankingapprox(A, 0.15, 0.005)

ans =

    0.2094
    0.2624
    0.2916
    0.2366
```

Resultatene er, som forventet, ganske like. Den første funksjonen er nøyaktig, mens den andre er en tilnærming.

SLUTT.