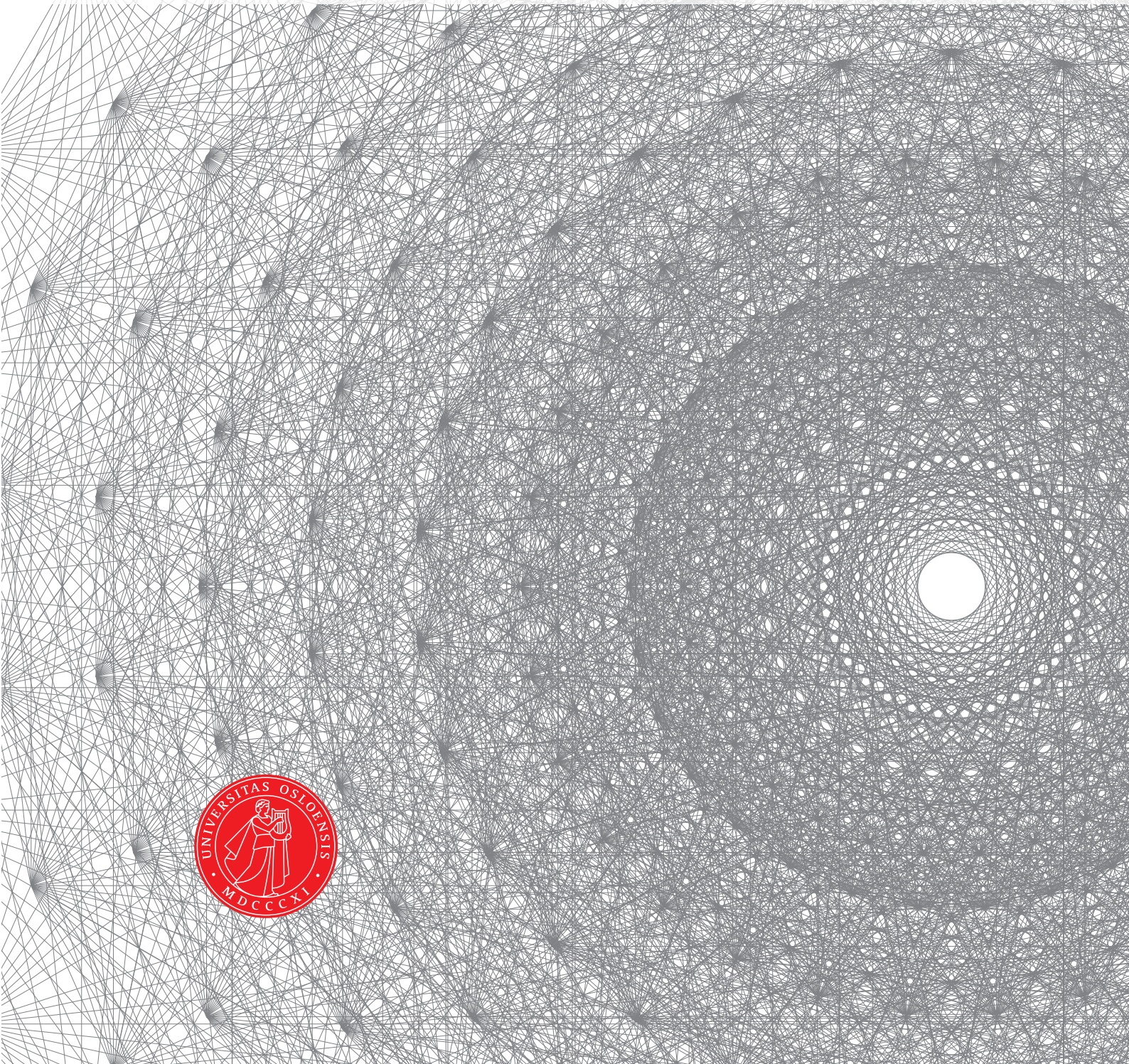
UiO : Matematisk institutt
Det matematisk-naturvitenskapelige fakultet

Dobbeltstokastiske matriser, Skalering og Rangering

Olai Sveine Johannessen

Masteroppgave for graden Lektorprogrammet med masterspesialisering i matematikk,
MAT5930L, våren 2016



Annerkjennelse

Før jeg starter på min masteroppgave ønsker jeg bare å takke min veileder Geir Dahl for at han sa seg villig til å veilede meg, og alle timene han har brukt for at jeg skal få til det jeg ønsket, uten han hadde dette ikke vært mulig. Jeg må også takke matematisk institutt på UiO som har gitt meg muligheten til å skrive denne oppgaven hos dem, selv om jeg har gått på Lektorprogrammet. Til slutt ønsker jeg å dedikere denne oppgaven til min nylig avdøde far, Kjell-Eirik Johannessen som opp igjennom årene alltid har vært der for meg, pushet meg og hjulpet meg.

Blindern, 30/05/2016 Olai

Innhold

Annerkjennelse **3**

Introduksjon **6**

1 Teori **9**
1.1 Radsumvektor og kolonnesumvektor 9
1.2 Dobbeltstokastiske matriser og permutasjonsmatriser 10
1.3 Total support og matrisemønster 14
1.4 Skalering av matriser 18
1.5 Sinkhorn-Knopp algoritmen 20

2 Rangering **29**
2.1 Hva er egentlig rangering? 29
2.2 Get-ligaen og scorematriser 30
2.3 Offens-Defense Modellen(ODM) 35
2.4 ODM anvendt på scorematriser fra Get-ligaen 43
2.5 Tolkning av rangeringene 49
2.6 Kan en kombinasjonsranging forutse utfallet av en kamp? . . . 51
2.7 Tolkning av elementene i en dobbeltstokastisk matrise 53

A Programmer og Scorematriser **57**
A.1 Skalering av en matrise A til en dobbeltstokastisk matrise . . . 58
A.2 ODM 59
A.3 Scorematrisene tilhørende kapittel 2.4 60

Bibliografi **64**

Introduksjon

Målet med denne masteroppgaven er å ta en nærmere titt på matematikken bak en bestemt rangeringsmetode. Jeg vil forklare hvorfor rangeringen ser ut som den gjør og kan derfra se om det er en fornuftig rangering av dataene. Jeg baserer mesteparten av min teori fra [2] og [4] hvor det er snakk om skalering av matriser og rangeringsmodellen kalt Offens-Defens modellen.

Jeg vil i første del av oppgaven skrive om teorien rundt matriseskaleringer, jeg skal ta for meg begreper, teoremer og algoritmer, hvor jeg til slutt ender opp med at en matrise med noen enkle betingelser kan skaleres til en dobbelstokastisk matrise, ved hjelp av en algoritme. I del to av oppgaven ser jeg først på hva en rangering egentlig er og hvordan teorien fra første del kan anvendes i en rangeringsmodell. Jeg vil deretter studere en rangeringsmodell og anvende denne til å rangere lagene i Get-ligaen (Eliteserien i ishockey), vil vi få ut en fornuftig rangering? Hvorfor? Helt til slutt kommer jeg med en alternativ rangering hvor jeg kombinerer de ulike rangeringene og lager en kombinasjonsrangering.

Jeg kommer gjennom hele oppgaven til å være så presis som mulig slik at det skal være enkelt å forstå mine tanker og forklaringer, jeg kommer også til å bruke en del eksempler som støtte til teorien. Men før vi kommer igang med diskusjon av rangeringer må vi få grunnleggende teori på plass. Hva er egentlig *matriseskalering*? Hva innebærer *total support*? Hva er en *dobbelstokastisk* matrise og hvordan henger alt dette sammen og relateres til rangering?

Kapittel 1

Teori

Målet med dette kapittelet er å få frem matriseteorien som ligger til grunn for det vi skal se på i kapittel 2. Jeg ønsker å være presis og grundig i mine forklaringer og defenisjoner av teorien, derfor bruker jeg en del eksempler som støtte. Mesteparten av teorien er hentet fra R.A. Brualdi sin bok *Combinatorial Matrix Classes* [2]. Kapittelet ender med en forklaring av Sinkhorn-Knopp algoritmen som kan komme godt med når vi skal se på ulike scorematriser fra Get-ligaen(Eliteserien i ishockey) i kapittel 2.

1.1 Radsumvektor og kolonnesumvektor

Vi har en $n \times n$ matrise $A = [a_{ij}]$, hvor a_{ij} gir oss verdien av det elementet på (i, j)-plassen i matrisen A. Senere kommer vi til å se på to forskjellige typer matriser, *positive* matriser og *ikke-negative* matriser. I en positiv matrise er $a_{ij} > 0$ for alle i og j, og i en ikke-negativ matrise er $a_{ij} \geq 0$ for alle i og j. Summerer vi alle elementene langs en rad får vi det som kalles en *radsum*, radsummen til den i'te raden i A er gitt som:

$$r_i = \sum_{j=1}^{n} a_{ij}$$

På samme måte kan vi få en *kolonnesum* når vi summerer elementene nedover i samme kolonne. Kolonnesummen til den j'te kolonnen i A blir dermed:

$$s_j = \sum_{i=1}^{n} a_{ij}$$

Radsumvektoren(R) og kolonnesumvektoren(S) er to vektorer som består av alle radsummene og kolonnesummene til matrisen, de er definert slik:

$$R = (r_1, r_2, ..., r_n) = (\sum_{j=1}^{n} a_{1j}, \sum_{j=1}^{n} a_{2j}, ..., \sum_{j=1}^{n} a_{ij})$$

$$S = (s_1, s_2, ..., s_n) = (\sum_{i=1}^{n} a_{i1}, \sum_{i=1}^{n} a_{i2}, ..., \sum_{i=1}^{n} a_{ij})$$

Her er da r_1 radsummen til den øverste/første raden i A, mens s_1 er kolonnesummen til den kolonnen lengst til venstre i A. Både r_1 og s_1 inneholder ett felles element, nemlig a_{11}. Dette kan vi også se av eksempelet under.

Eksempel 1.1.1
Vi har 4×4 matrisen:

$$H = \begin{bmatrix} 2 & 4 & 1 & 1 \\ 5 & 1 & 2 & 6 \\ 3 & 3 & 4 & 2 \\ 1 & 4 & 2 & 5 \end{bmatrix}$$

Summerer vi alle elementene som står i samme rad får vi 4 tall som utgjør radsumvektoren R_H og summerer vi elementene i samme kolonne får vi 4 tall som utgjør kolonnesumvektoren S_H.

Altså er $r_1 = 2+4+1+1 = 8$ og $s_1 = 2+5+3+1 = 11$ de to første tallene i hver av vektorene. Vektorene blir ut i fra matrisen H seende slik ut:

$$R_H = (8, 14, 12, 12)$$

$$S_H = (11, 12, 9, 14)$$

1.2 Dobbeltstokastiske matriser og permutasjonsmatriser

Før vi ser på hva en dobbeltstokastisk matrise er må vi kjapt definere hva en stokastisk matrise er. En matrise $B = [b_{ij}]$ kalles stokastisk dersom den er kvadratisk, $b_{ij} \in [0, 1]$ og enten radsumvektoren R_B eller kolonnesumvektoren

S_B kun inneholder énere. Om radsumvektoren R_B kun inneholder énere sier vi at matrisen er rad-stokastisk og på samme måte hvis kolonnesumvektoren S_B kun inneholder énere sier vi at matrisen er kolonne-stokastisk.

En $n \times n$ matrise $A = [a_{ij}]$ er en dobbelstokastisk matrise (ds-matrise) dersom $a_{ij} \in [0, 1]$ og både radsumvektoren $R_A = (r_1, r_2, ..., r_n)$ og kolonnesumvektoren $S_A = (s_1, s_2, ..., s_n)$ kun inneholder énere. Altså at

$$r_i = \sum_{j=1}^{n} a_{ij} = 1 \quad (i = 1, 2, ..., n)$$

og

$$s_j = \sum_{i=1}^{n} a_{ij} = 1 \quad (j = 1, 2, ..., n)$$

Ω_n betegner klassen av alle $n \times n$ dobbeltstokastiske matriser med orden n. For å være helt sikre på at vi har forstått hva en ds-matrise er ser vi på følgende eksempel:

Eksempel 1.2.1

$$A = \begin{bmatrix} \frac{3}{6} & \frac{3}{6} & 0 \\ \frac{1}{6} & \frac{1}{6} & \frac{4}{6} \\ \frac{2}{6} & \frac{2}{6} & \frac{2}{6} \end{bmatrix}$$

Vi ser her at A er en ds-matrise fordi alle $a_{ij} \in [0, 1]$ og radsumvektoren til A, $R_A = (r_1\ r_2\ r_3) = (1\ 1\ 1)$ og kolonnesumvektoren til A, $S_A = (s_1\ s_2\ s_3) = (1\ 1\ 1)$. Altså får vi at $A \in \Omega_n$

Fra [2] har vi et teorem som sier at

Teorem 1.2.1
La $P^{(1)} = [p_{ij}^{(1)}]$ og $P^{(2)} = [p_{ij}^{(2)}]$ være ds-matriser, da vil $A = P^{(1)}P^{(2)} = [a_{ij}]$ også være en ds-matrise.

Bevis. Fordi $P^{(1)}$ og $P^{(2)}$ er ikke-negative matriser med elementer i $[0, 1]$, er det opplagt fra definisjonen av matriseprodukt at elementene i A har de samme betingelsene som elementene i $P^{(1)}$ og $P^{(2)}$. Når det gjelder radsummene og kolonnesummene ser vi at

$$\sum_{i=1}^{n} a_{ij} = \sum_{i=1}^{n} p_{ij}^{(1)} p_{ij}^{(2)} = \sum_{i=1}^{n} p_{ij}^{(1)} \cdot \sum_{i=1}^{n} p_{ij}^{(2)} = 1 \cdot 1 = 1$$

$$\sum_{j=1}^{n} a_{ij} = \sum_{j=1}^{n} p_{ij}^{(1)} p_{ij}^{(2)} = \sum_{j=1}^{n} p_{ij}^{(1)} \cdot \sum_{j=1}^{n} p_{ij}^{(2)} = 1 \cdot 1 = 1$$

Eller dersom vi sier at $e = (1, 1, ..., 1)$ kan vi skrive dette enklere ved:

$$eA = eP^{(1)}P^{(2)} = eP^{(2)} = e$$

$$Ae^T = P^{(1)}P^{(2)}e^T = P^{(1)}e^T = e^T$$

Altså må også A være en ds-matrise. □

Det er vikitg å stille spørsmål rundt matematiske temaer slik som, hvorfor er dobbeltstokastiske matriser så interessante? og hva er det de egentlig kan gi oss av informasjon og ikke minst hvordan oppstår disse matrisene?

En stokastisk matrise kalles også en sannsynlighetsmatrise hvor elementene i samme kolonne representerer sannsynligheter for at de ulike utfallene (hver rad i kolonnen) skal intreffe (hver rad i kolonnen representerer et bestemt utfall). En dobbeltstokastisk matrise vil naturlig nok også være stokastisk og derfor også en sannsynlighetsmatrise, men noe mer avansert. For en dobbeltstokastisk matrise vil både matrisen selv og den transponerte av matrisen være stokastikse matriser, og dette er noe av grunnen til at den er litt spesiell når det kommer til sannsynlighetsberegninger. Alle verdiene i matrisen vil også her kunne representere sannsynligheter for at ulike utfall skal intreffe, men hvert element blir ikke bare påvirket av hvilken kolonne det er plassert i, men også i hvilken rad elementet står, vi skal se mer på dette litt senere.

Hvordan man lager en ds-matrise skal vi komme tilbake til i kapittel 1.4 om *skalering*, jeg kan røpe så mye at vi skal se nærmere på Sinkhorn-Knopp algoritmen som er en fornuftig og enkel måte å lage ds-matriser på.

Nå skal vi se nærmere på noen spesielle tilfeller av dobbeltstokastiske matriser som kalles *permutasjonsmatriser*. En såkalt $(0, 1)$-matrise er en matrise som kun inneholder nuller og énere, altså hvor $a_{ij} = 0$ eller 1 for alle $i, j \leq n$. En permutasjonsmatrise er en kvadratisk $(0, 1)$-matrise som kun inneholder én éner i hver rad og én éner i hver kolonne, dette vil da som sagt være en spesiell ds-matrise.

Eksempel 1.2.2
En permutasjon er en forflyttning som tar deg fra f.eks. 4 til 2 til 3 til 1. Denne permutajsonen skriver vi slik: $\sigma = (4, 2, 3, 1)$ som da vil ha følgende tilhørende permutasjonsmatrise:

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Som du ser er dette en $(0, 1)$-matrise med kun én éner i hver rad og hver kolonne.

Det finnes et endelig antall permutasjonsmatriser av orden n, nemlig $n!$. Dette kan forklares kort og enkelt ved kombinatorikk, altså vil det være n énere som skal inn i n forskjellige kolonner og vi får et ordnet utvalg uten tilbakelegging som vil si at

$$\text{Mulige utfall} = \frac{n!}{(n-r)!}.$$

I vår situasjon vil $r = n$, fordi alle kolonnene må inneholde én éner og dermed ser vi at det finnes $n!$ permutasjonsmatriser. Eksempelvis ser vi at de eneste permutasjonsmatrisene for $n = 2$ er:

$$P_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{Og} \quad P_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Under følger et lite kjent teorem av G. Birkhoff om sammenhengen mellom ds-matriser og permutasjonsmatriser, jeg utelater beviset, men du kan se både beviset og teoremet både i [2] og [1]. Jeg velger i stedet å se på et par eksempler og anvendelser av teoremet.

Teorem 1.2.2
Hvis A er en dobbeltstokastisk matrise av orden n, så finnes det en $t \geq 1$, permutasjonsmatriser $P_1, P_2, ...P_t$ og positive reelle tall $c_1, c_2, ..., c_t$ med $c_1 + c_2 + ... + c_t = 1$ slik at

$$A = c_1P_1 + c_2P_2 + ... + c_tP_t$$

Dette betyr altså at enhver ds-matrise kan skrives som en konveks kombinasjon av permutasjonsmatriser, noe som det står mer om i [3]. A er en

konveks kombinasjon dersom

$$A = \sum_{t=1}^{n} c_t P_t \quad \text{hvor} \quad \sum_{t=1}^{n} c_t = 1$$

og P_t er ulike permutasjonsmatriser.

Eksempel 1.2.3
Matrisene A og B er to ulike ds-matriser, fra teorem 1.2.2 skal begge disse kunne skrives som konvekskombinasjoner av permutasjonsmatriser av samme orden som A og B.

$$A = \begin{bmatrix} 0.25 & 0.75 \\ 0.75 & 0.25 \end{bmatrix} =$$

$$\frac{1}{4}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \frac{3}{4}\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.35 & 0.15 & 0.50 \\ 0.65 & 0.15 & 0.20 \\ 0.00 & 0.70 & 0.30 \end{bmatrix} =$$

$$\frac{3}{20}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \frac{3}{20}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \frac{1}{5}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Vi ser altså at både A og B kunne skrives som konvekskombinasjoner av permutasjonsmatriser av samme orden. Dette beviser ikke teoremet, men er eksempler på hva teoremet sier.

1.3 Total support og matrisemønster

Mønsteret til en ikke-negativ matrise A er definert som $patt(A)$. $patt(A)$ er en $(0,1)$-matrise hvor vi bytter ut alle ikke-null elementene i A med tallet 1 i $patt(A)$ og lar de resterende null elementene i A være null også i $patt(A)$. Vi ønsker å se på hvor matrisen A har null og ikke-null elementer og da er det enklere å jobbe med $patt(A)$ enn A fordi verdien av de positive tallene

er uviktig.

$$A = \begin{bmatrix} 1 & 4 & 9 & 0.5 \\ 12 & 0 & 0 & 3 \\ 0.8 & 7 & 1 & 0 \\ 0 & 5 & 0 & 2 \end{bmatrix} \quad \Rightarrow \quad patt(A) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Kjører vi matrisen A gjennom koden under får vi ut $patt(A)$:

```
A = [1 4 9 0.5 ; 12 0 0 3 ; 0.8 7 1 0 ; 0 5 0 2]
n = 4;
% Lager matrisen om til en (0,1)-matrise
for i = 1:n
        for j = 1:n
                if A(i,j)>=1
                        A(i,j) = 1;
                else
                        A(i,j)=0;
                end
        end
end
A
```

Men hvorfor er det så viktig å se på hvor en matrisen har null og ikke-null elementer? Når vi senere ser på Sinkhorn-Knopp algoritmen er det en viktig forutsetning at matrisen vi jobber med har såkalt *total support*. Om en matrise har total support eller ikke avhenger bare av hvor matrisen har null elementer, derfor vil det være slik at dersom $patt(A)$ har total support vil også A ha total support. Vi kommer derfor til å se om $patt(A)$ ha total support fordi denne er letter å jobbe med. Definisjonen for om en matrise har total support er:

Definisjon 1.3.1
En ikke-negativ matrise A av orden n sies å ha total support dersom hvert av matrisens ikke-null elementer hører til en positiv diagonal. En diagonal er de n posisjonene til énerene i en permutasjonsmatrise.

Her ser vi faktisk et bruksområde for permutasjonsmatriser, vi bruker disse til å kontrollere om en matrise har total support eller ikke. Nå skal vi se på hva det faktisk betyr at en matrise har total support. Vi skal altså gjennom et lite eksempel, hvor vi ønsker å finne ut om matrisen A har total support eller ikke. Spørsmålet da blir om det er mulig å dekke alle ikke-null elementer i A (vi jobber med $patt(A)$) med énere fra permutasjonsmatriser

uten at en éner i en av permutasjonsmatrisene havner på en plass hvor A og $patt(A)$ har et null-element:

Eksempel 1.3.2
Ønsker å se om 4×4 matrisen A ovenfor har total support. Vi jobber for enkelhetsskyld med mønstermatrisen til A, nemlig $patt(A)$.

$$patt(A) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Vi skal nå prøve å dekke alle énerene i $patt(A)$ med énere fra permutasjonsmatriser uten at noen av énerene fra permutasjonsmatrisene havner på en posisjon hvor $patt(A)$ har et null-element. Vi må bruke permutasjonsmatriser av orden 4, som det finnes $4! = 4 \cdot 3 \cdot 2 \cdot 1 = 24$ stykker av.
Nedenfor til venstre har vi de ulike permutasjonsmatrisene som passer og til høyre markeres de énerene i $patt(A)$ som dekkes av nettopp disse permutasjonsmatrisene. Om noen énere i $patt(A)$ dekkes flere ganger har det ingen betydning.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \textit{ og } \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \xrightarrow[over]{Dekker} \begin{bmatrix} \boldsymbol{1} & \boldsymbol{1} & 1 & 1 \\ \boldsymbol{1} & 0 & 0 & \boldsymbol{1} \\ 1 & 1 & \boldsymbol{1} & 0 \\ 0 & \boldsymbol{1} & 0 & \boldsymbol{1} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \textit{ og } \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \xrightarrow[over]{Dekker} \begin{bmatrix} 1 & 1 & \boldsymbol{1} & 1 \\ \boldsymbol{1} & 0 & 0 & \boldsymbol{1} \\ \boldsymbol{1} & \boldsymbol{1} & 1 & 0 \\ 0 & \boldsymbol{1} & 0 & \boldsymbol{1} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \xrightarrow[over]{Dekker} \begin{bmatrix} 1 & 1 & 1 & \boldsymbol{1} \\ \boldsymbol{1} & 0 & 0 & 1 \\ 1 & 1 & \boldsymbol{1} & 0 \\ 0 & \boldsymbol{1} & 0 & 1 \end{bmatrix}$$

De fem permutasjonsmatrisene vi har valgt ut til venstre dekker til sammen over alle énerene i $patt(A)$ uten å dekke over noen nuller i $patt(A)$. Da vet vi at $patt(A)$ har total support og dermed har også A total support. Jeg omtaler metoden i dette eksempelet som prøve-feile metoden, og det viser seg at den kan bli litt tungvint.

Nå har vi fått en forståelse av hva som må til for at en matrise skal ha total support. Denne prøve feile metoden viser seg å bli litt tungvint dersom vi øker dimensjonen til matrisen vi ser på, altså øker n. Senere i oppgaven skal vi jobbe med 10×10 matriser og da vil det finnes $10! = 3628800$ forskjellige permutasjonsmatriser som kan brukes for å finne ut om matrisen har total support eller ikke. Vi skal se på matriser av typen B:

$$B = \begin{bmatrix} 0 & 6 & 9 & 1 & 2 & 4 & 6 & 6 & 7 & 2 \\ 3 & 0 & 2 & 8 & 1 & 3 & 5 & 4 & 0 & 2 \\ 0 & 0 & 1 & 4 & 3 & 0 & 1 & 4 & 6 & 5 \\ 0 & 0 & 8 & 7 & 6 & 3 & 8 & 2 & 0 & 1 \\ 1 & 3 & 5 & 0 & 2 & 1 & 1 & 3 & 0 & 5 \\ 3 & 5 & 4 & 6 & 2 & 4 & 0 & 3 & 2 & 1 \\ 0 & 3 & 1 & 3 & 0 & 0 & 2 & 3 & 1 & 3 \\ 0 & 2 & 4 & 0 & 3 & 1 & 0 & 3 & 6 & 0 \\ 6 & 0 & 2 & 6 & 0 & 1 & 0 & 3 & 0 & 5 \\ 1 & 4 & 1 & 0 & 1 & 1 & 2 & 3 & 0 & 1 \end{bmatrix}$$

Når vi da jobber med slike matriser senere er vi avhengige av at de har total support. Det er flere måter å finne ut om en matrise har total support på, men en vanlig metode er å løse det som et lineært optimeringsproblem hvor vi går igjennom hvert element i matrisen. Vi bruker den tilhørende (0,1)-matrisen og ser på hver posisjon som inneholder en éner, og for hver av disse ser vi om det er mulig å finne en éner i hver av de andre kolonnene og hver av de andre radene. Slik går vi igjennom matrisen for alle posisjoner som inneholder en éner. Dersom det plutselig ikke finnes en éner i noen av de andre radene eller kolonnene vil ikke matrisen ha total support. Det man gjør er enkelt og greit at man velger ut en éner og ser om dette er en diagonal i en permutasjonsmatrise som "passer" med matrisen. Med "passer" mener jeg at ingen av permutasjonsmatrisens énere havner på en plass hvor matrisen har et null element.

Det finnes også teori på ulike betingelser og egenskaper en matrise må ha dersom vi skal se om den har total support uten å faktisk kontrollere det. Det er nemlig slik at den må være ikke-negativ og kvadratisk, men i tillegg til disse kravene er det slik at dersom matrisen inneholder for mange nuller kan den umulig ha total support. Vi kan fort forstå at dersom en matrise inneholder en nullrad eller nullkolonne kan den umulig ha total support. Jeg skal ikke gå noe mer inn på dette i denne oppgaven, dersom dette er av interesse anbefaler jeg å se i [2].

1.4 Skalering av matriser

Generelt vil en skalering av et tall bare være å endre størrelsen på tallet ved å multiplisere tallet med en skalar større enn null. Dersom vi skalerer tallet 8 med skalaren 5 får vi: $8 \cdot 5 = 40$. Men når det kommer til matriser er det litt mer komplisert, her er en liten definisjon som hjelper oss med forståelsen.

Definisjon 1.4.1
En matriseskalering er en operasjon hvor man multipliserer alle elementene a_{ij} i en rad eller en kolonne med den samme skalaren $k > 0$.

Det kan være hvilken som helst rad eller kolonne, og ofte skalerer vi flere rader eller flere kolonner samtidig, de forksjellige skalarene vi skalerer med har ingenting med hverandre å gjøre og kan godt være ulike. Fra definisjonen ser vi at vi må skalere rader og kolonner hver for seg. Skalerer vi alle elementene i en eller flere rader i matrisen kaller vi det en *radskalering* og hvis vi skalerer alle elementene i en eller flere av kolonnene er det en *kolonneskalering*. Generelt omtaler vi bare begge deler som skalering. Baktanken med skalering er at om vi velger disse skalarene smart så kan vi skalere matrisen til å ha en bestemt form. Dette skal vi se nærmere på i kapittel 1.5 som tar for seg Sinkhorn-Knopp algoritmen.

Her har vi et eksempel på en radskalering og en kolonneskalering:

Her ser vi 3×3 matrisen A som vi skal bruke i flere eksempler i dette kapittelet for å forklare alle aspekter ved skalering.

$$A = \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix}$$

Eksempel 1.4.2
En **radskalering** av A vil f.eks. være at vi multipliserer alle elementene i rad nr. 1 med skalaren p, slik:

$$\begin{bmatrix} p \cdot 4 & p \cdot 2 & p \cdot 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix}$$

Eksempel 1.4.3
En **kolonneskalering** av A vil f.eks. være at vi multipliserer alle elementene i kolonne nr. 2 og 3 henholdsvis med skalarene k og r:

$$\begin{bmatrix} 4 & k \cdot 2 & r \cdot 3 \\ 5 & k \cdot 4 & r \cdot 1 \\ 3 & k \cdot 8 & r \cdot 1 \end{bmatrix}$$

Når vi ser på eksempel 1.4.2 og 1.4.3 må vi huske at de radene og kolonnene som tilsynelatende ikke blir multiplisert med noen skalar faktisk blir multiplisert med skalaren 1. Altså blir rad 2 og 3 i eksempel 1.4.2 og kolonne 1 i eksempel 1.4.3 multiplisert med 1.

Nå vet vi at alle radene eller alle kolonnene multipliseres med en skalar hver gang vi skalerer en matrise. En matriseskalering vil altså være å multiplisere matrisen vår med en annen matrise. Samtidig vet vi at matrisen vi starter med og matrisen vi ender opp med etter skaleringen har samme dimensjoner. Den eneste muligheten for at dette skal stemme er om matrisen vi skalerer med har samme dimensjon som matrisen vi ønsker å skalere. Altså dersom vi ønsker å skalerer en $n \times n$ matrise A må vi multiplisere A med en annen $n \times n$ matrise. Spørsmålet nå er bare hvordan den matrisen vi multipliserer med skal se ut? Ut i fra algoritmen for matriseprodukter og det faktum at multiplikasjon med identitetsmatrisen I ikke endrer matrisen A:

$$A = A \cdot I = I \cdot A = A$$

Ser vi at dersom vi ønsker å skalere en matrise A må vi multiplisere med en diagonalmatrise av samme orden, men skal vi multiplisere fra høyre eller fra venstre? Vi ser på noen eksempler og hva som skjer:

Eksempel 1.4.4
Vi ser på hva som skjer når vi multipliserer A med en diagonalmatrise fra venstre:

$$\begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \cdot \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} = \begin{bmatrix} a \cdot 4 & a \cdot 2 & a \cdot 3 \\ b \cdot 5 & b \cdot 4 & b \cdot 1 \\ c \cdot 3 & c \cdot 8 & c \cdot 1 \end{bmatrix}$$

Vi endte altså opp med at matrisen A har blitt radskalert. Er det da slik at matrisen A blir kolonneskalert om vi multipliserer med en diagonalmatrise fra høyre?

$$\begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} \cdot \begin{bmatrix} d & 0 & 0 \\ 0 & e & 0 \\ 0 & 0 & f \end{bmatrix} = \begin{bmatrix} d \cdot 4 & e \cdot 2 & f \cdot 3 \\ d \cdot 5 & e \cdot 4 & f \cdot 1 \\ d \cdot 3 & e \cdot 8 & f \cdot 1 \end{bmatrix}$$

Nå ser vi at matrisen A har blitt kolonneskalert, og det var dette vi ville finne ut.

Det vi kan lære fra eksempl 1.4.4 og det viktigste å få med seg før vi går inn i neste kapittel er:

- Radskalering $\Rightarrow$ multipliser med en diagonalmatrise fra venstre
- Kolonneskalering $\Rightarrow$ multipliser med en diagonalmatrise fra høyre
- Elementet på plass (i, j) i diagonalmatrisen (når $i = j$) vil henholdsvis skalere A i rad i dersom vi multipliserer fra venstre eller kolonne j dersom vi multipliserer fra høyre.

1.5 Sinkhorn-Knopp algoritmen

Nå skal vi se litt på anvendelser av matrieskaleringer, hva er målet med en matriseskalering og hvordan når vi målet? I vårt tilfelle så ønsker vi å skalere matriser slik at vi ikke bare får en stokastisk matrise, men en dobbeltstokastisk matrise. Det er flere matematikere som har gjort kjent arbeid innenfor dette feltet, men ut i fra flere kilder, blant annet [2] og [4] kom Sinkhort og Knopp i tiden 1964-1967 med sin teori som først sier følgende:

Teorem 1.5.1 - *Sinkhorn-Knopp teoremet*
For enhver positiv kvadratisk matrise A finnes det diagonalmatriser $D_1, D_2 > 0$ slik at D_1AD_2 er dobbeltstokastisk.

Bevis.
Beviset for dette teoremet er mer eller mindre Sinkhorn-Knopp algoritmen. Den sier at vi må radskalerer og kolonneskalerer med spesielle diagonalmatriser annenhver gang, fortsetter man med dette vil vi ende opp med at matrisen konvergerer mot å bli en dobbeltstokastisk matrise, samtidig som radsummene og kolonnesummene alle konvergerer mot 1.
□

Litt senere hadde Sinkhorn, Knopp, Brualdi, Parter og Schneider uavhengig av hverandre kommet frem til at den samme algoritmen også kunne fungere når det var snakk om å skalere en kvadratisk ikke-negativ matrise til en dobbeltstokastisk matrise. Den eneste betingelsen var at matrisen måtte ha total support.

Teorem 1.5.2
Enhver ikke-negativ kvadratisk matrise A kan bli skalert ved bruk av diagonalmatriser $D_1, D_2 > 0$ til en dobbeltstokastisk matrise D_1AD_2 kun i de tilfellene hvor matrisen A har total support.

Bevis.
Ifølge [2] er det gitt titalls bevis for dette teoremet, et av dem kan du lese i [2], men det blir unødvendig komplisert å ta med i denne oppgaven.
□

Altså vil Sinkhorn-Knopp algoritmen skalere kvadratiske matriser A ved hjelp av diagonalmatriser D_1, D_2 slik at vi ender opp med at D_1AD_2 konvergerer mot å bli en dobbeltstokastiske matrise. Dette gjelder kun i de tilfellene hvor matrisen er positiv eller hvis matrisen er ikke-negaitv, men har total support. Dersom matrisen ikke skulle ha total support vil ikke algoritmen konvergere og vi vil ikke ende opp med en dobbeltstokastisk matrise. Vi skal i denne oppgaven ikke se på hvorfor dette bare gjelder når den ikke-negative matrisen har total support, vi skal heller fokusere på å forstå algoritmen.

For at en matrise skal bli dobbelstokastisk må radsumene og kolonnesumene være lik 1, men hvordan blir en radsum plutselig lik 1 etter en skalering? I eksempel 1.5.1 ser vi på hvordan dette henger sammen.

Eksempel 1.5.1
Vi har en tilfeldig tallrekke a som kan være en rad(eller kolonne) i en tenkt matrise. Radsummen R_a til a er ikke lik 1. Vi ønsker å skalere a til a^* slik at $R_{a^*} = 1$.

$$R_a = 2 + 5 + 6 + 4 + 3 = 20$$

Når vi skalerer så multipliserer vi altså en skalar k med alle elementene i rekka a. Hva må k være for at $R_{a^*} = 1$? Når vi vet radsummen R_a vil det være slik at $R_{a^*} = 1$ dersom

$$k = \frac{1}{R_a}$$

Vi tester ved å multiplisere alle leddene i a med $\frac{1}{R_a} = \frac{1}{20}$:

$$R_{a^*} = \frac{2}{20} + \frac{5}{20} + \frac{6}{20} + \frac{4}{20} + \frac{3}{20} = \frac{20}{20} = 1$$

Slik vil det være for alle rekker, ønsker man at summen av rekka skal bli lik 1 må du multiplisere alle leddene i rekka med den inverse av summen til rekka.

Nå har vi sett teknikken for at én radsum (eller kolonnesum) kan bli 1. Vi ønsker å kunne gjøre dette med alle radene (eller kolonnene) i en matrise. Dersom elementene langs diagonalen i diagonalmatrisen er de inverse av de tilhørende radsumene (eller kolonnesumene) vil dette føre til at alle radsumene (eller kolonnesummene) blir lik 1. Se eksempel 1.5.2 og 1.5.3 for en mer detaljert forklaring.

Eksempel 1.5.2
Vi skal her radskalere matrisen A til en matrise vi kaller A^* slik at $R_{A^*} = 1$

$$A = \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} \quad \Rightarrow \quad R_A = \begin{bmatrix} 9 & 10 & 12 \end{bmatrix}$$

Den inverse av R_A vil være:

$$R_A^{-1} = \begin{bmatrix} \frac{1}{9} & \frac{1}{10} & \frac{1}{12} \end{bmatrix}$$

Som vi deretter putter inn langs diagonalen i en diagonalmatrise:

$$D_1 = \begin{bmatrix} \frac{1}{9} & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & \frac{1}{12} \end{bmatrix}$$

Vi multipliserer fra venstre og får $D_1 \cdot A = A^*$ som er radskalert:

$$\begin{bmatrix} \frac{1}{9} & 0 & 0 \\ 0 & \frac{1}{10} & 0 \\ 0 & 0 & \frac{1}{12} \end{bmatrix} \cdot \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} = \begin{bmatrix} \frac{4}{9} & \frac{2}{9} & \frac{3}{9} \\ \frac{5}{10} & \frac{4}{10} & \frac{1}{10} \\ \frac{3}{12} & \frac{8}{12} & \frac{1}{12} \end{bmatrix} = A^*$$

Som en kontroll ser vi på radsumene til A^*:

$$R_{A^*} = \begin{bmatrix} \frac{4}{9} + \frac{2}{9} + \frac{3}{9} & \frac{5}{10} + \frac{4}{10} + \frac{1}{10} & \frac{3}{12} + \frac{8}{12} + \frac{1}{12} \end{bmatrix}$$

Vi ser her at $R_{A^*} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ som var målet vårt, men legg også merke til at dette betyr at A^* er en rad-stokastisk matrise. A^* er åpenbart ikke dobbelstokastisk fordi kolonnesumvektoren ikke består av énere.

Vi kan gjøre det på samme måten når det er snakk om kolonneskalering og kolonnesumvektoren den eneste forskjellen er at vi multipliserer med diagonalmatrisen fra høyre, se eksempel 1.5.3 under.

Eksempel 1.5.3
I dette eksempelet ønsker vi å kolonneskalere A til matrisen A^*, slik at kolonnesumene til A^* blir lik 1.

$$A = \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} \quad \Rightarrow \quad S_A = \begin{bmatrix} 12 & 14 & 5 \end{bmatrix}$$

Finner den inverse til kolonnesumvektoren S_A

$$S_A^{-1} = \begin{bmatrix} \frac{1}{12} & \frac{1}{14} & \frac{1}{5} \end{bmatrix}$$

Vi putter så elementene fra S_A^{-1} inn langs diagonalen i en diagonalmatrise og får D_2.

$$D_2 = \begin{bmatrix} \frac{1}{12} & 0 & 0 \\ 0 & \frac{1}{14} & 0 \\ 0 & 0 & \frac{1}{5} \end{bmatrix}$$

Regner nå ut $A \cdot D_2 = A^*$, hvor da A^* er kolonneskalert.

$$\begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{12} & 0 & 0 \\ 0 & \frac{1}{14} & 0 \\ 0 & 0 & \frac{1}{5} \end{bmatrix} = \begin{bmatrix} \frac{4}{12} & \frac{2}{14} & \frac{3}{5} \\ \frac{5}{12} & \frac{4}{14} & \frac{1}{5} \\ \frac{3}{12} & \frac{8}{14} & \frac{1}{5} \end{bmatrix} = A^*$$

Som en kontroll ser vi på kolonnesumene til A^*:

$$S_{A^*} = \begin{bmatrix} \frac{4}{12} + \frac{5}{12} + \frac{3}{12} & \frac{2}{14} + \frac{4}{14} + \frac{8}{14} & \frac{3}{5} + \frac{1}{5} + \frac{1}{5} \end{bmatrix}$$

Ser enkelt her da at $S_{A^*} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$ som betyr at A^* i dette eksempelet er en kolonne-stokastisk matrise, men fortsatt ikke dobbeltstokastisk fordi radsumvektoren ikke består av énere.

Fra eksemplene 1.5.2 og 1.5.3 ser vi at metoden fungerte og vi enkelt kan skalere en matrise til en stokastisk matrise. Når det gjelder å skalere den til å bli en dobbeltstokastisk matrise må det litt mer jobb til. Som nevnt tidligere beviste Sinkhorn at dette var mulig ved å radskalere og kolonneskalere matrisen annenhver gang helt til matrisen vi endte opp med var dobbeltstokastisk. Eksempel 1.5.2 og 1.5.3 vil bare være hvert sitt første steg i Sinkhorn-Knopp

algoritmen mot at matrisen blir en dobbeltstokastisk matrise. Algoritmen går som følger:

- Finn radsumvektoren R til matrisen
- Putt de inverse radsumene inn på diagonalen i en diagonalmatrise
- Multipliser diagonalmatrisen med matrisen fra venstre

  På dette tidspunktet vil R kun inneholde énere og matrisen vi har fått er rad-stokastisk

- Finn nå kolonnesumvektoren S til den rad-stokastiske matrisen
- Putt de inverse kolonnesumene inn på diagonalen i en diagonalmatrise
- Multipliser diagonalmatrisen med den rad-stokastiske matrisen fra høyre

  Nå vil S kun inneholde énere og matrisen er kolonne-stokastisk. R vil være forandret

- Begynn algoritmen fra starten igjen med den kolonne-stokastiske matrisen du nå har.

Om man begynner med en radskalering eller en kolonneskalering spiller ingen rolle. Husk også at matrisen enten må være positiv eller ikke-negativ med total support.

Når du kjører en egnet matrise gjennom denne algoritmen vil du underveis etter hver skalering annenhver gang få en rad-stokastisk og kolonne-stokastisk matrise. Dette betyr også at radsumvektoren og kolonnesumvektoren annenhver gang vil bestå av énere. Til slutt vil algoritmen føre til at matrisen konvergerer mot en dobbeltstokastisk matrise. Samtidig som vi kan se av eksempel 1.5.4 vil radsummene og kolonnesummene nærme seg 1 når vi ikke regner med annenhver gang hvor de er lik 1 som følge av en skalering.

I vedlegg A.1 har jeg skrevet en matlabkode som gjør akkurat dette, den radskalerer og kolonneskalerer annahver gang helt til både radsumvektoren og kolonnesumvektoren er tilnærmet lik 1. Nå kommer et lite eksempel hvor jeg gjennomfører et par steg i algoritmen.

Eksempel 1.5.4
Vi ønsker nå å skalere A til å bli en tilnærmet stokastisk matrise

$$A = \begin{bmatrix} 4 & 2 & 3 \\ 5 & 4 & 1 \\ 3 & 8 & 1 \end{bmatrix} \Rightarrow \qquad \begin{aligned} R_A &= \begin{bmatrix} 9 & 10 & 12 \end{bmatrix} \\ S_A &= \begin{bmatrix} 12 & 14 & 5 \end{bmatrix} \end{aligned}$$

Vi radskalerer A i henhold til algoritmen og får:

$$A^* = \begin{bmatrix} \frac{4}{9} & \frac{2}{9} & \frac{3}{9} \\ \frac{5}{10} & \frac{4}{10} & \frac{1}{10} \\ \frac{3}{12} & \frac{8}{12} & \frac{1}{12} \end{bmatrix} \Rightarrow \qquad \begin{aligned} R_{A^*} &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \\ S_{A^*} &= \begin{bmatrix} \frac{43}{36} & \frac{58}{45} & \frac{31}{60} \end{bmatrix} \end{aligned}$$

Vi kolonneskalerer A^* i henhold til algoritmen og får:

$$A^{**} = \begin{bmatrix} \frac{16}{43} & \frac{5}{29} & \frac{20}{31} \\ \frac{18}{43} & \frac{9}{29} & \frac{6}{31} \\ \frac{9}{43} & \frac{15}{29} & \frac{5}{31} \end{bmatrix} \Rightarrow \quad \begin{aligned} R_{A^{**}} &= \begin{bmatrix} 1.189 & 0.922 & 0.88 \end{bmatrix} \\ S_{A^{**}} &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \end{aligned}$$

Vi skal nå radskalere A^{**} i henhold til algortimen, men det blir bare mange desimaler og unøyaktigheter så vi stopper her og lar datamaskinen gjøre resten. Vi får følgende resultater.

$$B = \begin{bmatrix} 0.3017 & 0.1299 & 0.5683 \\ 0.4564 & 0.3144 & 0.2292 \\ 0.2419 & 0.5556 & 0.2025 \end{bmatrix} \Rightarrow \qquad \begin{aligned} R_B &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \\ S_B &= \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \end{aligned}$$

Jeg fikk denne løsningsmatrisen B ved å kjøre A gjennom programmet mitt fra Vedlegg A.1 hvor den ble radskalert og kolonneskalert 15 ganger før matrisen ble tilnærmet dobbeltstokastisk. Om vi ønsker å være mer nøyaktig, må vi bare endre betingelsene i programmet og programmet vil fortsette å radskalere og kolonneskalere til vi er fornøyd. Altså er konklusjonen at den konvergerer mot å bli dobbeltstokastisk.

Nå kommer spørsmålet om hvorfor dette er nødvendig og hva vi kan bruke dette til? I kapittel 2 som omhandler *Rangering* skal vi se nærmere på hva de dobbeltstokastiske matrisene forteller oss, men først ønsker jeg å nevne et kjekt bruksområde for skaleringsalgoritmen.

Vi ønsker å løse en helt vanlig matriseligning $Ax = B$ hvor A er kvadratisk invertibel og enten positiv eller ikke-negativ med total support. Problemet er at A er en matrise det er vanskelig å jobbe med grunnet veldig stor variasjon i elementene sine. For å gjøre A "pen" kan vi kjøre den gjennom Sinkhorn-Knopp algoritmen og bruke den dobbeltstokastiske matrisen D_1AD_2 i stedet for A, hvordan vi implementerer D_1AD_2 skal jeg vise nå.
Begynner med likningen vi har:

$$Ax = B$$

Deretter multipliserer vi med D_1 fra venstre på begge sider av likheten

$$D_1Ax = D_1B$$

Deretter må vi få inn D_2. Vi bruker at $I = D_2D_2^{-1}$ og multiplikasjon med identitetsmatrisen forandrer ingenting. Dermed får vi:

$$D_1AD_2D_2^{-1}x = D_1B$$

For at dette skal se ordentlig ut innfører vi en ny ukjent $y = D_2^{-1}x$ og ligningen blir:

$$D_1AD_2y = D_1B$$

Vi løser denne likningen for y og deretter må vi løse ligningen

$$y = D_2^{-1}x$$

for x, som betyr at $x = D_2y$
Dermed har vi løst likningen $Ax = B$ ved hjelp av en dobbeltstokastisk matrise. Alle trinnene i denne utregningen er mulige fordi A, D_1 og D_2 er invertible (D_1, D_2 grunnet positive elementer langs diagonalen)

Kapittel 2

Rangering

I dette kapittelet skal jeg se nærmere på hva rangering er og hvorfor vi ender opp med å rangere slik vi gjør. Hva forteller dataene oss og hvilken rangering av dataene vil være mest fornuftig? Hvorfor? Jeg kommer til å bruke teorien om rangering på selvlagde scorematriser fra Get-ligaen(Eliteserien i ishockey) hvor objektene er de ulike lagene i ligaen. Alle disse scorematrisene er ikke-negative og har total support. Til slutt vil jeg komme med min egen kombinasjonsrangering. Teorien om ragnering og rangeringsformer låner jeg for det meste fra A.Y.Govan, A.N.Langville og C.D.Meyer sin bok *Offense-defense approach to ranking team sports* [4]. Kan man bruke rangeringer til å forutse utfallet av kamper?

2.1 Hva er egentlig rangering?

De fleste er interessert i en eller annen sport, og har kanskje et favorittlag eller en favorittspiller. Tenk deg at du har et favorittlag(Lag B), lag B har vunnet over lag A i to av to kamper i år, men allikevel ligger lag A over lag B på den offesielle tabellen. Er det da noe galt med rangeringen? Lag B har jo vist flere ganger at det er bedre enn nettopp lag A, kan vi rangere lagene på en annen måte?

Rangeringer er viktig for oss om du vil eller ikke, vi rangerer objekter hele tiden og kan ofte handle ut ifra rangeringen. Objektene kan være alt fra klærne du har på deg, bilen du kjører, maten du spiser til lag innenfor sport. Har du noen gang spilt på oddsen? Hvor kommer egenlig tallene på oddsen fra og hvorfor er de så forskjellige? Jeg håper at denne oppgaven kanskje kan gi deg noen tanker om dette.

Du har et problem og spør Google, i hvor mange tilfeller scroller du ned og ser på googleresultatene på side 2 eller 3? Hvorfor ikke? Jo fordi Google's søkemotor gir deg en rangert liste hvor de beste nettsidene kommer øverst. Rangeringsalgoritmen Google bruker skal vi ikke se på i denne oppgaven, men den er ikke så ulik de som kan brukes innenfor sport og du kan lese om den i [5].

For å kunne rangere objekter fornuftig må vi først foreta en *vurdering* av de samme objektene. Vurderingen inneholder informasjon om hvert objekt og kan gi oss forklaring på hvorfor objektet er rangert slik det er. Vurderingen av objektene går på å sammenligne objektene og ut ifra denne sammenligningen kommer det en rangering. Vurderingen er altså grunnlaget for rangeringen av objektene, det er vanskelig å rangere objekter fornuftig uten en vurdering. Generelt kan man si at en rangering kun er en sortert liste av vurderingen de forskjellige objektene har fått.

Objektene vi ønsker å rangere må altså vureres først, hva vi vurderer objektene ut i fra kan variere stort som også fører til at rangeringene kan bli veldig forskjellige. Det er derfor viktig at vi er presise i hva rangeringen faktisk forteller oss når vi forsøker å tolke resultatene, altså rangeringen.

Vi skal senere vurdere og rangere lagene i Get-ligaen ut i fra forskjellige kriterier. Bakgrunnen for vurderingene vi kommer til å gjøre er ulike scorematriser som kan gi oss ulike rangeringer, hva en scorematrise er og hvordan kampene i Get-ligaen gjennomføres skal vi se nærmere på nå.

2.2 Get-ligaen og scorematriser

Vi må først ha litt grunnkunnskaper om hvordan Get-ligaen er bygget opp når det gjelder lag, poeng, kamper, runder, serie etc. Slik at vi senere forstår hvorfor de ulike scorematrisene ser ut som de gjør. Teorien og dataene i dette delkapittelet er en blanding av noe du kan lese på [7] og mitt eget hode. Offesielt rangeres lagene i ligaen etter poeng, hvor hvert lag kan få poeng hver kamp. Laget med flest poeng når serien er ferdig vinner. Her ser du den offesielle tabellen fra seriespillet i sesongen 15/16. Jeg ønsker å se om lagene vil være rangert i den samme rekkefølgen når vi senere skal vurdere scorematriser fra forskjellige aspekter rundt sesongen 15/16.

Get-ligatabellen 2015-2016:

1	Stavanger Oilers	99 poeng
2	Lørenskog IK	95 poeng
3	Frisk Asker	86 poeng
4	Sparta Sarpsborg	86 poeng
5	Vålerenga Ishockey	77 poeng
6	Storhamer Hockey	74 poeng
7	Stjernen Hockey	58 poeng
8	Manglerud Stars	43 poeng
9	Lillehammer IK	42 poeng
10	Kongsvinger Knights	15 poeng

(2.1)

Get-ligaen består av 45 kamper for hvert lag, hvor alle møter alle like mange ganger. Når alle lagene har spilt mot hverandre én gang kaller vi det en *runde*, det spilles altså 5 runder, 5 ganger mot hvert lag. Vi ser med en gang at antall hjemmekamper og bortekamper vil være ulikt ($3+2$ eller $2+3$), dette byttes året etter slik at det blir likt for alle etter to år.

Kampene varer i 3 perioder av 20 minutter effektiv spilletid, og laget som har scoret flest mål når 3. periode er slutt, vinner. Står det uavgjort etter $3 \cdot 20$ minutter vil det spilles én ekstraperiode på 5 minutter hvor det laget som scorer først vinner. Dersom ingen scorer i ekstraperioden vil det bli en straffeslagskonkurranse for å kåre en vinner. Det kan altså aldri bli uavgjort i hockey, vi vil alltid få en vinner. Det skal alltid deles ut 3 poeng i en kamp og hvordan disse tre poengene fordeles mellom de to lagene avhenger av hva som skjer i kampen.

Poengfordelingen mellom lagene i hver kamp er:

- Laget som vinner kampen etter spill i 3 perioder får **3 poeng**
- Laget som spiller uavgjort etter 3 perioder, men vinner i ekstraomgangen eller i straffeslagskonkurransen får **2 poeng**
- Laget som spiller uavgjort etter 3 perioder, men taper i ekstraomgangen eller i straffekonkurransen får **1 poeng**
- Laget som taper kampen etter spill i 3 perioder får naturlig nok **0 poeng**

Det er på bakgrunn av disse reglene vi får tabell 2.1, tabellen kårer hvem som vinner og taper Get-ligaen 15/16. Denne tabellen rangerer altså lagene ut ifra hvor mange poeng hvert lag har skaffet seg på 45 kamper. Vi skal senere rangere lagene ved hjelp av flere scorematriser og sammenligne det vi får med rangeringen i tabell 2.1. Jeg omtaler senere hvert lag som et tall, hvor tallet tilsvarer plasseringen i tabell 2.1, og vi bruker den samme rekkefølgen på lagene når vi ser på scorematriser.

Bare for å være helt presis så snakker vi om Stavanger som lag nr 1 $(i, j = 1)$, Lørenskog er lag nr 2 $(i, j = 2)$, Frisk er nr 3 osv ned til Kongsvinger som er lag nr 10 $(i, j = 10)$. Lagene blir altså representert ved et tall fremfor lagnavn slik at senere rangeringer av lagene kun vil være en tallfølge. Rangeringen fra tabell 2.1 som er den vi kommer til å sammenligne litt med er (1 2 3 4 5 6 7 8 9 10).

Jeg er ute etter å se om man kan rangere lagene på bakgrunn av andre kriterier enn poeng. Jeg har laget ulike scorematriser som viser forskjellige aspekter mellom lagene i løpet av sesongen. Disse scorematrisene kan gi oss mye informasjon om hvordan lagene har gjort det mot hverandre.

En scorematrise vil i vårt tilfelle være grunnlaget for den vurderingen vi gjør av de forskjellige lagene. Som vi snart skal se kan scorematriser basere seg på mange forskjellige aspekter ved kampene.

	Sta	Lør	Fri	Spa	Vål	Sto	Stj	Man	Lil	Kon
Stavanger	0	6	4	10	3	2	2	7	5	5
Lørenskog	2	0	2	2	4	3	4	1	2	13
Frisk	3	1	0	2	5	3	3	3	2	7
Sparta	2	4	1	0	1	5	5	10	1	6
Vålerenga	2	0	4	2	0	2	3	3	4	4
Storhamar	1	2	4	0	0	0	5	3	10	2
Stjernen	3	2	2	3	2	3	0	7	3	8
Manglerud	1	2	1	0	2	4	3	0	5	5
Lillehammer	2	3	3	0	1	1	1	2	0	3
Kongsvinger	0	1	3	3	1	0	3	1	2	0

Figur 2.1: En oversikt over antall mål radlaget har scort mot hvert av kolonnelagene i første runde av sesongen 15/16

Figur 2.1 er en komplett oversikt over alle resultatene fra kampene i første runde. Lagene er plassert både loddrett i første kolonne og vannrett i første rad, rekkefølgen er som du ser lik den i tabell 2.1. Lagene nedover første

kolonne kalles *"radlag"*, jeg kommer senere til å omtale radlagene som lag i. Lagene som står i første rad kalles *"kolonnelag"* og disse vil senere bli omtalt som lag j. Dette er fordi radlagene har sine resultater bortover i rader, mens kolonnelagene har sine resultater nedover i kolonner. Raden og kolonnen som inneholder navnene på lagene er bare med for å holde orden slik at vi forstår innholdet i figuren. Hva tallene forteller oss og hvorfor de står der de står kommer vi inn på straks.

Ved å sette alle tallene fra figur 2.1 inn i en 10×10-matrise vil vi få en *scorematrise*. Rekkefølgen på lagene vil være den samme i alle scorematrisene jeg lager. Diagonalen inneholder natrulig nok nuller fordi dette er kamper hvor lagene møter seg selv (som ikke er mulig).

Fra Figur 2.1 får vi scorematrisen A:

$$A = \begin{bmatrix} 0 & 6 & 4 & 10 & 3 & 2 & 2 & 7 & 5 & 5 \\ 2 & 0 & 2 & 2 & 4 & 3 & 4 & 1 & 2 & 13 \\ 3 & 1 & 0 & 2 & 5 & 3 & 3 & 3 & 2 & 7 \\ 2 & 4 & 1 & 0 & 1 & 5 & 5 & 10 & 1 & 6 \\ 2 & 0 & 4 & 2 & 0 & 2 & 3 & 3 & 4 & 4 \\ 1 & 2 & 4 & 0 & 0 & 0 & 5 & 3 & 10 & 2 \\ 3 & 2 & 2 & 3 & 2 & 3 & 0 & 7 & 3 & 8 \\ 1 & 2 & 1 & 0 & 2 & 4 & 3 & 0 & 5 & 5 \\ 2 & 3 & 3 & 0 & 1 & 1 & 1 & 2 & 0 & 3 \\ 0 & 1 & 3 & 3 & 1 & 0 & 3 & 1 & 2 & 0 \end{bmatrix}$$

Scorematrisen A inneholder som nevnt resultatene fra første runde av Get-ligaen 15/16. Matrisen består av elementene $[a_{ij}]$ hvor i representerer et "radlag" og j representerer et "kolonnelag". Dersom lag i spiller mot lag j vil a_{ij} være antall mål lag i scorte på lag j, og a_{ji} vil være antall mål lag j scorte på lag i. Du kan også tenker på en annen måte ved at a_{ij} er antall mål lag j slapp inn mot lag i, og motsatt for a_{ji}.

Eksempel 2.2.1
Vi ser på kampen mellom Lørenskog(lag 2) og Frisk(lag 3), den kampen endte $2 - 1$ til Lørenskog, det kunne vi også sett ut i fra matrisen A ved at $a_{23} = 2$ og $a_{32} = 1$

Vi vet nå at a_{ij} er antall mål lag i scorer mot lag j, da vil dette bety at

radsummen til lag i

$$\sum_{j=1}^{10} a_{ij} = \text{Antall mål lag } i \text{ har scort totalt}$$

Radsummene vil altså være antallet mål et lag har scort i den første runden. På tilsvarende måte vil kolonnesummen til lag j være antal mål lag j har sluppet inn i løpet av den første runden. Antall scorte mål og antall innslupne mål for hvert av lagene i den første runden kan bli sett på som rad- og kolonnesumvektorer:

$$R_A = \begin{bmatrix} 44 & 33 & 29 & 35 & 24 & 27 & 33 & 23 & 16 & 14 \end{bmatrix}$$
$$S_A = \begin{bmatrix} 16 & 21 & 24 & 22 & 19 & 23 & 29 & 37 & 34 & 53 \end{bmatrix}$$

Her vil da R_A være antall scorte mål og S_A være antall innslupne mål. Rekkefølgen på tallene forteller oss hvilket lag tallene hører til. De to første tallene 44 og 16 hører til $i = 1$ og $j = 1$ som vil si at dette er scorte og innslupne mål for Stavanger. Dersom vi nå ønsker å gjøre en kjapp rangering av lagene hvor vurderingsgrunnlaget er R_A og S_A kan vi f.eks. gjøre som i følgende eksempel:

Eksempel 2.2.2
Vi ønsker å se på forskjellen mellom r_i og s_j når $i = j$ for å finne målforskjellen lagene har etter ni spilte kamper.

$$R_A - S_A = \begin{bmatrix} 28 & 12 & 5 & 13 & 5 & 4 & 4 & -14 & -18 & -39 \end{bmatrix}$$

Det beste er å ha stor positiv målforskjell altså at $r_i - s_j$ er et stort positivt tall. Derfor får vi denne rangeringen, hvor det beste laget står først:

$$(1 \quad 4 \quad 2 \quad 3 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10)$$

Vi husker at Stavanger = nr 1, Lørenskog = nr 2 osv. fra tabell 2.1. Sammenligner vi denne rangeringen med tabell 2.1 ser vi faktisk at et par av lagene har byttet plass.

Ut i fra scorematrise A kan vi også rangere lagene på andre måter en den brukt i eksempel 2.2.2, som vi snart skal se.

På samme måte som jeg lagde matrisen A ut i fra tallene i Figur 2.1 kan man lage andre scorematriser hvor elementene a_{ij} betyr andre ting enn *"antall mål radlaget scorte mot kolonnelaget i første runde"*. Vi skal blant annet se på matriser hvor a_{ij} betyr:

- Antall mål radlaget i scorte mot kolonnelaget j i runde 2,3,4 og 5
- Antall mål radlaget i scorte mot kolonnelaget j totalt i hele sesongen
- Antall kamper radlaget i har vunnet mot kolonnelaget j
- Antall poeng radlaget i har tatt totalt i de fem kampene mot kolonnelaget j.

Hva a_{ij} betyr i scorematrisen er avhengig av hva vi ønsker å vurdere lagene på bakgrunn av. Rangeringene kan variere veldig på bakgrunn av hva a_{ij} betyr, derfor må vi alltid være presise når vi diskuterer rangeringene. Vi skal nå se nærmere på hvordan vi kan rangere lagene.

2.3 Offens-Defense Modellen(ODM)

En rangeringsmodell gir oss en vurdering av objekter som ender med en rangering av objektene. Mange matematikere har gjort arbeid innenfor dette feltet, derfor finnes det mange modeller og metoder for rangering. De fleste av disse metodene bygger på tolkning og vurdering av scorematriser. Scorematrisene behøver ikke være bygget opp på samme måte som den vi så på i kapittel 2.2.

Vi skal nå se nærmere på rangeringsmodellen fra [4] kjent som *Offens-Defens Modellen*(heretter ODM). Vi ender opp med at ODM baserer seg på algoritmen og teoremet til Sinkhorn og Knopp fra kapittel 1.5. Scorematrisene ODM anvendes på er bygget opp slik at hvis lag i spiller mot lag j vil a_{ij} være antall mål lag i scorte på lag j, eller om du tenker motsatt så vil a_{ij} være antall mål lag j slapp inn når de spilte mot lag i. Hvert element a_{ij} i scorematrisen kan enten ses på offensivt som mål lag i har scort eller

defensivt som mål lag j har sluppet inn. Det er denne tvetydigheten ODM bruker for å vurdere lagene.

ODM vurderer og rangerer lag ut i fra hvor godt laget totalt sett er. Ifølge ODM reflekteres hvor godt et lag er gjennom deres offensive og defensive kvaliteter. Modellen gir lagene en offensiv vurdering og en defensiv vurdering, før vurderingene samkjøres og vi får en total vurdering av lagene som igjen gir oss en rangering av lagene.

Et lag med en god offensiv vurdering har evnen til å score mange mål, og et lag med en god deffensiv vurdering vil slippe inn få mål. Det er viktig å få med seg og forstå at dette betyr at det beste offensive laget vil ha en høy offensiv skår(o_i vil ha en stor verdi), mens det beste defensive laget vil ha en lav defensiv skår(d_j har liten verdi).

Vi ser for oss en $n \times n$ scorematrise $\mathbf{A} = [a_{ij}]$, da vil den offensive vurderingen til lag i beregnes slik:

$$o_i = a_{i1}(1/d_1) + a_{i2}(1/d_2) + ... + a_{in}(1/d_n) \tag{2.2}$$

Hvor d_j er den defensive vurderingen til lag j. Den defensive vurderingen til lag j er definert som:

$$d_j = a_{1j}(1/o_1) + a_{2j}(1/o_2) + ... + a_{nj}(1/o_n) \tag{2.3}$$

Vi ser altså at den offensive vurderingen o_i og den defensive vurderingen d_j er gjensidig avhengige av hverandre. Vurderingene tar hensyn til hvilket lag man scorer og slipper inn mål mot. Dersom d_1 er det beste defensive laget ($d_1 < d_j$ for alle j) så vil laget som skårer mot lag 1 få en høyere offensiv vurdering o_i enn om de skårer mot et dårlig defensivt lag, det vil atlså "telle mer" å score mot et defensivt bra lag. Fra 2.2 og det vi vet om scorematrisen er o_i en sum av alle målene lag i har scort, på samme måte er d_j en sum av alle målene lag j har sluppet inn. o_i er en radsum og d_j er en kolonnesum, men de er ikke helt like de radsummene og kolonnesummene vi så på i kapittel 1.1.

Forskjellen mellom 2.2 og 2.3 og vanlige rad- og kolonnesummer er $(1/d_j)$ og $(1/o_i)$, disse verdiene som multipliseres med hvert ledd blir å regne som "vekter" for hvert ledd. Det er altså disse som gir oss en betegnelse på hvor bra prestasjon det er å score mål eller ikke slippe inn mål mot et bestemt lag

i eller j. Denne "vekten" forteller oss hvor bra det laget som scorte mål eller slapp in mål er vurdert, slik at o_i og d_j kan forstås som en vektet sum av antall mål scort og sluppet inn. Det er disse vektene som sørger for at den offensive vurderingen av et lag vil øke mer om man scorer mot det defensivt beste laget enn om man scorer mot det defensivt dårligste laget. På samme måte virker de om man slipper inn mål mot det offensivt beste laget kontra det offensivt dårligste laget. Det er altså disse vektene som gjør at rangeringen ikke bare baserer seg på en total målforskjell.

Fra ODM får vi ut o_i og d_j for alle $i, j \leq n$, vi putter alle vurderingene inn i to vektorer $\mathbf{o} = (o_1 \;\; o_2 \;\; ... \;\; o_n)$ og $\mathbf{d} = (d_1 \;\; d_2 \;\; ... \;\; d_n)$. Under følger formlene som må brukes for å finne de offensive og defensive vurderingene av lagene ut ifra scorematrisen $\mathbf{A} = [a_{ij}]$. For at formlene skal være fornuftige må jeg først definere at $1/\mathbf{d}^{(k)}$ og $1/\mathbf{o}^{(k)}$ er lik følgende kolonnevektorer:

$$1/\mathbf{d}^{(k)} = (1/d_1 \;\; 1/d_2 \;\; ... \;\; 1/d_n)^T$$
$$1/\mathbf{o}^{(k)} = (1/o_1 \;\; 1/o_2 \;\; ... \;\; 1/o_n)^T$$

Offens-Defense Modellen:

$$\mathbf{o}^{(k)} = \mathbf{A}\frac{1}{\mathbf{d}^{(k-1)}} \tag{2.4}$$

$$\mathbf{d}^{(k)} = \mathbf{A}^T\frac{1}{\mathbf{o}^{(k)}} \tag{2.5}$$

Fra [4] ser vi at for å løse dette systemet må vi sette inn en initialverdi, så for $k = 1$ vil $d_j = 1$ for alle j. I praksis betyr dette at vi starter med å vurderer alle lagene til å ha like gode defensive kvaliteter. Vi får altså $\mathbf{d}^{(0)} = (1 \;\; 1 \;\; ... \;\; 1)$ og $\mathbf{o}^{(1)} = \mathbf{A}\frac{1}{\mathbf{d}^{(0)}}$. Vi må deretter løse 2.5 når $k = 1$ og finne $\mathbf{d}^{(1)}$ for så å finne $\mathbf{o}^{(2)}$ og fortsette slik for flere k.

Når vi løser 2.4 og 2.5 for lave k vil dette bare være tilnærmede vurderinger av lagene, fordi vi begynte med en initialverdi. Dersom vi fortsetter å løse 2.4 og 2.5 annenhver gang for høyere k så vil $\mathbf{o}^{(k)}$ og $\mathbf{d}^{(k)}$ konvergere mot å bli vurderinger av de offensive og defensive kvalitetene til lagene. Hvorfor det er slik at de konvergerer skal vi straks se på, men jeg kan avsløre at det har noe med Sinkhorn-Knopp algoritmen å gjøre.

Jeg har laget et program som bruker ODM til å finne den offensive og defensive vurderingen av lag, ut i fra en scorematrise, dette programmet kan du se i vedlegg A.2, det er mistenkelig likt probrammet som omhandler Sinkhorn-Knopp algoritmen i A.1.

Vi ser på et eksempel av hvordan de offensive og defensive vurderingene vil konvergerer mot en tilnærmet korrekt vurdering. Lagene i eksempelet er lag nr 1, 2 og 3, altså Stavanger, Lørenskog og Frisk. Vi ønsker å rangere disse lagene ut i fra en scorematrise bestående av deres innbyrdes resultater i den første runden i Get-ligaen, altså de 9 elementene a_{ij} hvor $i, j \leq 3$, dette blir 3×3 matrisen øverst til venstre i matrisen A fra kapittel 2.2.

Eksempel 2.3.1
Resultatene fra første runde er:

$$\begin{aligned} \text{Stavanger} - \text{Lørenskog} &= 6 - 2 \\ \text{Stavanger} - \text{Frisk} &= 4 - 3 \\ \text{Lørenskog} - \text{Frisk} &= 2 - 1 \end{aligned}$$

Vi får scorematrisen Q, og skal videre finne tilnærmede verdier til de offensive og defensive vurderingene av lagene ved å bruke ligningene 2.4 og 2.5.

$$Q = \begin{bmatrix} 0 & 6 & 4 \\ 2 & 0 & 2 \\ 3 & 1 & 0 \end{bmatrix}$$

Vi har initialverdien:

$$\boldsymbol{d}^{(0)} = \begin{bmatrix} d_1^{(0)} & d_2^{(0)} & d_3^{(0)} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Deretter bruker vi ligningene fra 2.4 og 2.5 for flere verdier av k

$$\begin{aligned} \boldsymbol{o}^{(1)} &= \begin{bmatrix} o_1^{(1)} & o_2^{(1)} & o_3^{(1)} \end{bmatrix} = \begin{bmatrix} 10 & 4 & 4 \end{bmatrix} \\ \boldsymbol{d}^{(1)} &= \begin{bmatrix} d_1^{(1)} & d_2^{(1)} & d_3^{(1)} \end{bmatrix} = \begin{bmatrix} 1.25 & 0.85 & 0.9 \end{bmatrix} \\ \boldsymbol{o}^{(2)} &= \begin{bmatrix} o_1^{(2)} & o_2^{(2)} & o_3^{(2)} \end{bmatrix} = \begin{bmatrix} 11.5 & 3.82 & 3.57 \end{bmatrix} \\ \boldsymbol{d}^{(2)} &= \begin{bmatrix} d_1^{(2)} & d_2^{(2)} & d_3^{(2)} \end{bmatrix} = \begin{bmatrix} 1.36 & 0.801 & 0.871 \end{bmatrix} \end{aligned}$$

En foreløpig offensiv rangering av lagene fra best til dårligst får vi fra den offensive vurderingen $\mathbf{o}^{(2)}$:

$$\textit{Offensiv rangering} = (1 \quad 2 \quad 3)$$

En foreløpig defensiv rangering av lagene fra best til dårligst får vi fra den defenisve vurderingen $\mathbf{d}^{(2)}$:

$$\textit{Defensiv rangering} = (2 \quad 3 \quad 1)$$

Vi har altså brukt ODM og funnet en offensiv rangering og en defensiv rangering av lagene.

Nå har vi kanskje forstått hvordan vi kan finne en offensiv og en defensiv rangering ut ifra en scorematrise, men hva er det egentlig som skjer med scorematrisen A? og hvorfor konvergerer $\mathbf{o}^{(k)}$ og $\mathbf{d}^{(k)}$ når k øker?

For å få en helhetlig forståelse av det som skjer må vi igjen gå tilbake til likning 2.2 og se at når vi finner o_i for alle i vil alle elementene i samme kolonne j fra A multipliseres med den samme "vekten"(skalaren) $(1/d_j)$. Denne vekten representerer hvor godt defensivt et lag vurderes til å være. $(1/d_j)$ vil i de første iterasjonene ikke være helt korrekt fordi vi startet med en initialverdi. Vi brukte denne initialverdien til å finne o_i som vi igjen må bruke for å finne en ny d_j. Etterhvert som vi fortsetter å gjenta prosessen vil denne vekten bli mer og mer riktig. Det samme vil da gjelde for vekten $(1/o_i)$ fra likning 2.3.

Når vi regner ut o_i som vi fra tidligere vet er en radsum, ser vi nå at dette vil være en radsum av en kolonneskalert matrise A. På samme måte vil likning 2.3 være en kolonnesum av en radskalert matrise. Fra kapittel 1.4 vet vi at når en matrise skaleres så vil dette tilsvare at matrisen mulitpliseres med en diagonalmatrise fra høyre eller fra venstre. Scorematrisen vil i de to tilfellene være skalert av skalarene $(1/d_j)$ og $(1/o_i)$. Ser vi nå på 2.4 og 2.5 ser vi at disse ligningene minner veldig om det som skjer i Sinkhorn-Knopp algoritmen i kapittel 1.5, det er faktisk akkurat det samme.

Det er slik at o_i og d_j konvergerer mot å bli de inverse til elementene i henholdsvis D_1 og D_2 fra Sinkhorn-knopp algoritmen i kapittel 1.5. I dette kapittelet har vi også snakket om at algoritmen vil konvergere så lenge A er en kvadratisk ikke-negativ matrise med total support. Slik vil det også være med D_1 og D_2, de vil i algoritmen konvergere mot å bli diagonalmatriser som fører til at D_1AD_2 er dobbeltstokastisk. Derfor vil også $\mathbf{o}^{(k)}$ og $\mathbf{d}^{(k)}$ konvergere mot de inverse av elementene langs diagonalene til D_1 og D_2. Dersom scorematrisen A ikke har total support vil ikke de offensive og defensive vurderingene konvergere fordi D_1AD_2 ikke vil konvergere mot å bli dobbeltstokastisk. Vurderingene $\mathbf{o}^{(k)}$ og $\mathbf{d}^{(k)}$ vil da heller ikke gi noen som helst mening.

Eksempel 2.3.2
Vi skal se litt på scorematrisen Q fra eksempel 2.3.1. Vi skal først finne den offensive $\mathbf{o}^{(k)}$ og defensive $\mathbf{d}^{(k)}$ vurderingen av lagene ved hjelp av vedlegg A.2. Før vi skal kjøre matrisen Q gjennom Sinkhorn-Knopp algoritmen i vedlegg A.1 og skrive ut D_1QD_2:

$$Q = \begin{bmatrix} 0 & 6 & 4 \\ 2 & 0 & 2 \\ 3 & 1 & 0 \end{bmatrix}$$

Etter å ha kjørt Q gjennom vedlegg A.2 22 ganger er vi fornøyd fordi $\mathbf{o}^{(22)}$ og $\mathbf{o}^{(22)}$ endrer seg svært lite for $k > 22$:

$$\boldsymbol{o}^{(22)} = \begin{bmatrix} 12.3484 & 3.7397 & 3.3978 \end{bmatrix}$$
$$\boldsymbol{d}^{(22)} = \begin{bmatrix} 1.4177 & 0.7802 & 0.8587 \end{bmatrix}$$

Nå bruker vi Sinkhorn-Knopp algoritmen til å få ut den dobbeltstokastiske matrisen $B = D_1QD_2$, det måtte også her 22 iterasjoner, altså $22 \cdot 2$ skaleringer til før B var dobbeltstokastisk i henhold til våre betingelser:

$$\begin{bmatrix} 0 & 0.6228 & 0.3772 \\ 0.3772 & 0 & 0.6228 \\ 0.6228 & 0.3772 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} \frac{1}{12.3484} & 0 & 0 \\ 0 & \frac{1}{3.7397} & 0 \\ 0 & 0 & \frac{1}{3.3978} \end{bmatrix} \begin{bmatrix} 0 & 6 & 4 \\ 2 & 0 & 2 \\ 3 & 1 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{1.4177} & 0 & 0 \\ 0 & \frac{1}{0.7802} & 0 \\ 0 & 0 & \frac{1}{0.8587} \end{bmatrix}$$

Eksempelet viser at det er slik vi antydet, $\mathbf{o}$ og $\mathbf{d}$ er henholdsvis de inverse til elementene i diagonalmatrisene D_1 og D_2 som skalerer Q til en dobbelstokastisk matrise B.

ODM ender altså opp med å bli et matriseskaleringsproblem, dette er fordi ODM ønsker å ta hensyn til hvilket lag man scorer eller slipper inn mål mot. Vi skalerer altså scorematrisen for å ta hensyn til hvor bra eller hvor dårlig lagene er offensivt og defensivt, representert i ODM som $(1/o_i)$ eller $(1/d_j)$. o_i og d_j er som kjent radsummer og kolonnesummer og vi skalerer scorematrisen med de inverse til disse summene, slik vi også gjør i Sinkhorn-

Knopp algoritmen. Poenget med denne skaleringen er at om et lag scorer 10 mål mot et bra lag skal dette påvirke den offensive vurderingen mer enn om de scorer 10 mål mot et dårlig lag. Å score 10 mål mot en dårlig motstander kan faktisk ende opp med å ha svært liten påvirkning på den offensive vurderingen din. Det er jo naturlig at det er slik dersom også alle de andre lagene i ligaen også scorer 10 mål mot det dårlige laget.

ODM tar faktisk også hensyn til flere ting en hvem man scorer eller slipper inn mål mot. Modellen er bygget opp slik at den også tar hensyn til hvordan laget du spiller mot har spilt mot de andre lagene i ligaen, sammenlignet med hvordan du selv har spilt mot alle disse lagene. Dette er en konsekvens av matriseskaleringen, når vi får en dobbeltstokastisk matrise vil som jeg har nevnt tidligere alle elementene representere ulike sannsynligheter for utfallet av kamper. Vi har foreløpig ikke sett så mye på hva tallene i den dobbelstokastiske matrisen forteller oss om lagene, men dette skal vi komme tilbake til i et senere kapittel.

ODM tar hensyn til utfallet av de fleste kampene samtidig når den kommer frem til en offensiv og defensiv vurdering. Det er faktisk ikke bare kampene til lag i som påvirker vurderingen til lag i, alle de andre kampene i ligaen påvirker også vurderingen til lag i. Det er nettopp dette som gjør ODM til en bra og fornuftig rangeringsmodell.

Fra flere eksempler og vurderingene o_i og d_j vet vi nå at vi kan finne en offensiv og defensiv rangering av lagene ved å se på størrelsene til o_i for alle i og d_j for alle j. Nå skal vi se hvordan disse vurderingene kan kombineres på en naturlig måte slik at vi kan få ut en total vurdering av lagene som også gir oss en total rangering. Den totale vurderingen r_i av lag i ut i fra o_i og d_i er:

$$r_i = \frac{o_i}{d_i}$$

Det vil altså være slik at laget med den høyeste verdien på r_i vil være det laget som ODM rangerer som nr 1. Dette vil da være det beste laget fordi de har det største forholdet mellom offensiv og defensiv vurdering. Rangeringen fra o_i eller d_i vil ikke nødvendigvis være lik den vi får fra r_i. Vi ser til slutt på et lite eksempel.

Eksempel 2.3.3
Vi ser på de offensive og defensive vurderingene av to lag for så å finne en total vurdering og rangering av disse to lagene.

Lag 1	*Lag 2*
$o_1 = 10$	$o_2 = 6$
$d_1 = 4$	$d_2 = 2$

Vi ser ut ifra dette at lag 1 har best vurdering offensivt fordi $o_1 > o_2$ og lag 2 har den beste vurderingen defensivt fordi $d_1 > d_2$, den totale vurderingen av de to lagene blir dermed.

Lag 1	*Lag 2*
$r_1 = 10/4 = 2.5$	$r_2 = 6/2 = 3$

Vi ser altså at lag 2 har en høyere total vurdering enn lag 1 og derfor rangeres lag 2 som et bedre lag enn lag 1.

Et lag som har en bra offensiv vurdering og en dårlig defensiv vurdering kan fort ende opp med å ha en dårlig total vurdering, slike tilfeller skal vi se nærmere på når vi nå skal anvende ODM på egenproduserte scorematriser.

2.4 ODM anvendt på scorematriser fra Get-ligaen

For at vi skal kunne anvende ODM på en scorematrise må matrisen enten være positiv eller ikke-negativ med total support. Dersom matrisen ikke tilfredsstiller dette vil ikke ODM konvergere og vi ender ikke opp med noen fornuftige vurderinger. Vi husker fra kapittel 1.3 at om en matrise har total support eller ikke til syvende og sist avhenger av hvor matrisen har null-elementer og hvor mange nullelementer den har. De scorematrisene vi skal se på nå inneholder naturlig nok nuller langs diagonalen, men de resterende elementene vil inneholde få nuller. På bakgrunn av at alle vurderingene konvergerer så vil dette bety at alle scorematrisene har total support.

Hva betyr det praktisk dersom en av scorematrisene ikke har total support? Hvis vi havner i en situasjon hvor et lag har spilt en hel runde i Get-

ligaen uten å score eller uten å slippe inn mål så vil scorematrisen ha en nullrad eller nullkolonne. Da vil ikke matrisen ha total support. Jeg har aldri hørt om et slikt tilfelle så vi behøver ikke bekymre oss for dette. Dersom en scorematrise faktisk ikke har total support kan vi benytte oss av et vanlig triks som innebærer og addere et lite positivt tall overalt, dermed er matrisen positiv og har total support. Dette tallet er veldig lite og påvirker ikke vurderingene av lagene noe særlig.

Vi skal straks se på vurderingene og rangeringene vi får fra de ulike scorematrisene i vedlegg A.3 som alle dreier seg om Get-ligasesongen 15/16. Jeg vurderer å rangerer disse ved å bruke programmet i vedlegg A.2. I tolkningen av rangeringene vi får må vi alltid huske hva rangeringen baserer seg på. Et lag er ikke nødvendigvis bedre enn et annet selv om én av rangeringene sier det, vi kommer tilbake til dette i diskusjonen av rangeringene litt senere. Når jeg rangerer lagene bruker jeg tall til å representere lagene, tallene følger av den offesielle rangeringen, altså tabell 2.1. Det er også denne rekkefølgen på lagene jeg sammenligner med når vi diskuterer rangeringene. For å være helt sikker på at du forstår, er lagene og tallene satt sammen på følgende måte, de har også den samme rekkefølgen fra venstre i **o** og **d**:

1	=	Stavanger Oilers
2	=	Løreskog IK
3	=	Frisk Asker
4	=	Sparta Sarpsborg
5	=	Vålerenga Ishockey
6	=	Storhamar Hockey
7	=	Stjernen Hockey
8	=	Manglerud Stars
9	=	Lillehammer IK
10	=	Kongsvinger Knights

Antall mål scort og sluppet inn i 1. runde, scorematrise A.1

$$\mathbf{o} = \begin{bmatrix} 46.9 & 30.8 & 30.8 & 34.0 & 22.9 & 23.8 & 32.9 & 22.5 & 17.2 & 14.9 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 1 & 4 & 7 & 2 & 3 & 6 & 5 & 8 & 9 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.60 & 0.75 & 1.0 & 0.72 & 0.65 & 0.79 & 1.1 & 1.2 & 1.3 & 1.8 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 1 & 5 & 4 & 2 & 6 & 3 & 7 & 8 & 9 & 10 \end{bmatrix}$$
$$\mathbf{r}_1 = \begin{bmatrix} 78.0 & 41.0 & 30.8 & 47.1 & 34.7 & 29.9 & 28.7 & 18.4 & 13.2 & 8.2 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 1 & 4 & 2 & 5 & 3 & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$$

Antall mål scort og sluppet inn i 2. runde, scorematrise A.2

$$\mathbf{o} = \begin{bmatrix} 30.0 & 44.8 & 23.5 & 31.6 & 23.3 & 20.0 & 16.4 & 19.3 & 26.6 & 16.2 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 2 & 4 & 1 & 9 & 3 & 5 & 6 & 8 & 7 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.55 & 0.63 & 0.62 & 1.19 & 0.99 & 0.74 & 1.27 & 1.281.39 & 1.33 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 1 & 3 & 2 & 6 & 5 & 4 & 7 & 8 & 10 & 9 \end{bmatrix}$$
$$\mathbf{r}_2 = \begin{bmatrix} 54.1 & 71.0 & 37.7 & 26.4 & 23.4 & 26.7 & 12.8 & 14.9 & 19.0 & 12.1 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 2 & 1 & 3 & 6 & 4 & 5 & 9 & 8 & 7 & 10 \end{bmatrix}$$

Antall mål scort og sluppet inn i 3. runde, scorematrise A.3

$$\mathbf{o} = \begin{bmatrix} 45.5 & 31.2 & 29.6 & 33.3 & 23.8 & 26.6 & 36.2 & 24.6 & 28.2 & 17.5 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 1 & 7 & 4 & 2 & 3 & 9 & 6 & 8 & 5 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.93 & 0.88 & 0.75 & 0.90 & 0.65 & 0.72 & 1.48 & 1.21 & 1.21 & 1.23 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 5 & 6 & 3 & 2 & 4 & 1 & 9 & 8 & 10 & 7 \end{bmatrix}$$
$$\mathbf{r}_3 = \begin{bmatrix} 48.5 & 35.4 & 39.0 & 36.6 & 36.2 & 36.7 & 24.3 & 20.2 & 23.3 & 14.2 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 1 & 3 & 6 & 4 & 5 & 2 & 7 & 9 & 8 & 10 \end{bmatrix}$$

Antall mål scort og sluppet inn i 4. runde, scorematrise A.4

$$\mathbf{o} = \begin{bmatrix} 26.7 & 24.0 & 25.0 & 47.0 & 26.6 & 41.5 & 35.8 & 18.3 & 22.0 & 10.3 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 4 & 6 & 7 & 1 & 5 & 3 & 2 & 9 & 8 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.34 & 0.81 & 1.00 & 1.01 & 0.56 & 0.70 & 1.13 & 1.78 & 1.04 & 1.64 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 1 & 5 & 6 & 2 & 3 & 4 & 9 & 7 & 8 & 10 \end{bmatrix}$$
$$\mathbf{r}_4 = \begin{bmatrix} 78.2 & 29.4 & 24.8 & 46.5 & 46.9 & 58.7 & 31.5 & 10.2 & 21.1 & 6.3 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 1 & 6 & 5 & 4 & 7 & 2 & 3 & 9 & 8 & 10 \end{bmatrix}$$

Antall mål scort og sluppet inn i 5. runde, scorematrise A.5

$$\mathbf{o} = \begin{bmatrix} 25.7 & 35.7 & 31.0 & 34.6 & 19.2 & 22.5 & 26.8 & 21.4 & 28.8 & 12.0 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 2 & 4 & 3 & 9 & 7 & 1 & 6 & 8 & 5 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.54 & 0.75 & 0.53 & 1.36 & 0.83 & 0.97 & 1.31 & 0.83 & 1.00 & 1.93 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 3 & 1 & 2 & 5 & 8 & 6 & 9 & 7 & 4 & 10 \end{bmatrix}$$
$$\mathbf{r}_5 = \begin{bmatrix} 47.4 & 47.0 & 57.7 & 25.4 & 23.1 & 23.2 & 20.3 & 25.7 & 28.8 & 6.2 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 3 & 1 & 2 & 9 & 8 & 4 & 6 & 5 & 7 & 10 \end{bmatrix}$$

Totalt antall mål scort og sluppet inn, scorematrise A.6

$$\mathbf{o} = \begin{bmatrix} 172 & 162 & 139 & 175 & 121 & 138 & 146 & 107 & 119 & 71 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 1 & 4 & 2 & 7 & 3 & 6 & 5 & 9 & 8 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.60 & 0.78 & 0.79 & 1.03 & 0.73 & 0.80 & 1.25 & 1.25 & 1.16 & 1.56 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 1 & 5 & 2 & 3 & 6 & 4 & 9 & 7 & 8 & 10 \end{bmatrix}$$
$$\mathbf{r}_6 = \begin{bmatrix} 283 & 207 & 175 & 169 & 165 & 172 & 116 & 85 & 102 & 45 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 1 & 2 & 3 & 6 & 4 & 5 & 7 & 9 & 8 & 10 \end{bmatrix}$$

Her ser vi altså hvordan lagenes offendsive vurderinger **o**, defensive vurderinger **d** og totale vurderinger **r** fører til rangeringer. Vi må huske at disse vurderingene og rangeringene kun baserer seg på målene som er scort og sluppet inn. Et lag vil ikke nødvendigvis være bedre enn et annet selv om det rangeres bedre, de vil bare ha et bedre forhold mellom scorte mål og innslupne mål. Vi skal diskutere disse vurderingene og rangeringen mer i neste

kapittel. Som nevnt tidligere kan vi også sette opp scorematriser hvor a_{ij} betyr andre ting enn scorte mål. I vedlegg A.3 finnes det et par scorematriser som er litt annerledes.

Scorematrise A.7 er en scorematrise hvor a_{ij} forteller oss hvor mange ganger lag i har vunnet mot lag j. Alle lagene møter hverandre som sagt fem ganger og da kan det være interessant å se på hvor mange av disse oppgjørene hvert av lagene har vunnet. Det er nyttig å se på denne vurderingen av lagene fordi vi hittil bare har basert oss på antall mål, men hvor mange mål et lag har scort og sluppet inn forteller oss ingenting om hvor mange av de fem kampene de har vunnet. Resultatene mellom lag i og lag j kan i løpet av de fem kampene være:

$$\begin{array}{c} \text{Lag } i - \text{Lag } j \\ 10 - 0 \\ 0 - 3 \\ 0 - 2 \\ 0 - 1 \\ 0 - 1 \end{array}$$

Dermed vil lag i rangeres høyere en lag j dersom vi ser på antall mål, men motsatt dersom vi ser på antall seiere.

Scorematrise A.8 er en matrise hvor a_{ij} forteller oss antallet poeng lag i har tatt i løpet av de fem kampen mot lag j. Denne matrisen forteller oss litt om hvor jevne kampene har vært fremfor hvem som vant. Scorematrisen A.7 gir oss kun en pekepinn på hvem som vinner, altså ikke om kampene mellom lagne har gått til ekstraperiode eller straffer. Derfor vil scorematrise A.8 være en bedre beskrivelse av hvordan lagene har spilt mot hverandre. Rangeringen av lagene ut ifra denne matrisen gir oss en pekepinn på hvor sterk prestasjon det er å ta poeng mot hvert av de andre lagene. Dersom vi ser på poengfordelingen mellom lag i og lag j sammenlignet med lag l og lag m:

Lag i – Lag j	Lag l – Lag m
$3-0$	$2-1$
$3-0$	$2-1$
$3-0$	$2-1$
$3-0$	$2-1$
$3-0$	$2-1$

Her ser vi at både lag i og lag l har vunnet alle sine kamper, men kampene mellom lag l og lag m ser ut til å ha vært mye jevnere enn kampene mellom lag i og j i alle fall poengmessig. Vi skal nå se på rangeringene vi får fra disse to scorematrisene.

Antall seiere og tap totalt, scorematrise A.7

$$\mathbf{o} = \begin{bmatrix} 32.1 & 32.7 & 28.4 & 29.0 & 29.6 & 23.5 & 19.5 & 12.6 & 16.4 & 4.2 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 2 & 1 & 5 & 4 & 3 & 6 & 7 & 9 & 8 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.59 & 0.54 & 0.63 & 0.59 & 1.10 & 0.97 & 1.08 & 1.35 & 1.42 & 1.74 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 2 & 1 & 4 & 3 & 6 & 7 & 5 & 8 & 9 & 10 \end{bmatrix}$$
$$\mathbf{r}_7 = \begin{bmatrix} 53.8 & 60.5 & 44.7 & 49.1 & 26.8 & 24.2 & 18.0 & 9.3 & 11.5 & 2.4 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 2 & 1 & 4 & 3 & 5 & 6 & 7 & 9 & 8 & 10 \end{bmatrix}$$

Antall poeng plukket mot hvert lag totalt, scorematrise A.8

$$\mathbf{o} = \begin{bmatrix} 101.5 & 98.8 & 85.7 & 86.6 & 84.6 & 70.0 & 64.0 & 39.3 & 43.3 & 12.5 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 1 & 2 & 4 & 3 & 5 & 6 & 7 & 9 & 8 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 0.49 & 0.55 & 0.66 & 0.61 & 1.17 & 0.86 & 1.15 & 1.36 & 1.44 & 1.73 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 1 & 2 & 4 & 3 & 6 & 7 & 5 & 8 & 9 & 10 \end{bmatrix}$$
$$\mathbf{r}_8 = \begin{bmatrix} 206 & 176 & 129 & 141 & 72 & 80 & 55 & 28 & 30 & 7 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 1 & 2 & 4 & 3 & 6 & 5 & 7 & 9 & 8 & 10 \end{bmatrix}$$

Til slutt i dette kapittelet ønsker jeg bare å se på en litt artig vurdering og rangering av lagene. Vi vet jo som sagt at alle lagene spiller 45 kamper hvor 5 er mot hvert av de 9 lagene. Dette betyr at det ikke vil være balanse

mellom antallet hjemme- og bortekamper, og noen lag vil kanskje ha en liten fordel av oppsettet mellom lagene. I scorematrise A.9 vil a_{ij} være antallet hjemmekamper lag i har spilt mot lag j, og vi ender opp med følgende rangering av lagene:

Antall hjemmekamper spilt mot hvert lag, scorematrise A.9

$$\mathbf{o} = \begin{bmatrix} 22.04 & 22.99 & 22.95 & 22.99 & 22.04 & 22.86 & 22.04 & 22.95 & 22.04 & 22.04 \end{bmatrix}$$
$$\text{Offensiv rangering} = \begin{bmatrix} 2 & 4 & 3 & 8 & 6 & 1 & 5 & 7 & 9 & 10 \end{bmatrix}$$
$$\mathbf{d} = \begin{bmatrix} 1.02 & 0.98 & 0.97 & 0.98 & 1.02 & 0.97 & 1.02 & 0.97 & 1.02 & 1.02 \end{bmatrix}$$
$$\text{Defensiv rangering} = \begin{bmatrix} 3 & 6 & 8 & 2 & 4 & 1 & 5 & 7 & 9 & 10 \end{bmatrix}$$
$$\mathbf{r}_9 = \begin{bmatrix} 21.60 & 23.42 & 23.42 & 23.41 & 21.60 & 23.42 & 21.61 & 23.42 & 21.60 & 21.61 \end{bmatrix}$$
$$\text{Total rangering} = \begin{bmatrix} 2 & 3 & 6 & 8 & 4 & 7 & 10 & 1 & 5 & 9 \end{bmatrix}$$

2.5 Tolkning av rangeringene

Som vi ser får vi ut vurderinger og rangeringer fra alle scorematrisene i vedlegg A.3, men hvordan kan vi tolke denne informasjonen om lagene og ikke minst hva kan vi bruke den til? Det vi generelt kan se fra kapittel 2.4 er hvordan de forskjellige offensive, defensive og totale vurderingene og rangeringene til hvert lag varierer avhengig av hva vi baserer scorematrisen på.

Det er fort gjort å kun se på de ulike rangeringene av lagene og trekke konklusjoner ut ifra disse, det må vi være litt forsiktige med å gjøre. Rangeringen forteller oss bare om rekkefølgen på størrelsene til de ulike vurderingene, den sier oss ingenting om hvor like eller ulike vurderingene til flere lag er. Dersom vi ser på den defensive vurderingen i tredje runde, fra scorematrise A.3, kan vi se at både lag 8 og 9 har svært like vurderinger, men vi rangerer lag 9 foran pga desimalene lenger bak komma. Vi må derfor alltid se på vurderingene samtidig når vi diskuterer rangeringene, som vi sa tidligere så er det vurderingene som inneholder all informasjonen. Nå skal vi se på hva de ulike vurderingene og rangeringene forteller oss og om vi kan trekke noen håndfaste konklusjoner ut i fra de resultatene vi så i kapittel 2.4.

Rangeringer fra målmatrisene, A.1 til A.6:
Totalt har vi kommet frem til 18 vurderinger og rangeringer som baserer

seg på antall mål scort og sluppet inn for de forskjellige lagene. De lagene som rangeres høyt i disse rangeringen vil ha evnen til å score mange mål, selv mot gode lag, og slippe inn få mål. Dersom et lag rangeres lavt i denne rangeringen vil det bety at laget slipper inn mange mål mot de fleste og har store problemer med å score mål dersom motstanden blir for god. Vi ser at de lagene som rangeres høyt ifølge tabell 2.1 også rangeres høyt i disse rangeringene, så rangeringne er ikke helt håpløse selv om de kun baserer seg på antall mål scort og sluppet inn. Det er veldig mye man kan se og mene ut ifra disse rangeringene og jeg skal ikke gå igjennom alt nå, men noen ting er verdt å legge merke til. Vi kan blant annet se at lag 1 rangeres som nr i totalt 11 av de 18 rangeringene, så dette laget er klart best når det gjelder antall mål scort og sluppet inn. Lag 10 rangeres som nr 10 i 16 av de 18 rangeringene og er tydelig dårligst. Vi ser at lag 9 rangeres foran lag 8 i rangeringen fra det totale antall scorede mål, det er ikke så rart ettersom lag 9 rangerers foran lag 8 i 13 av de 18 rangeringene. Vi ser også at lag 4 rangeres veldig høyt i alle de offensive rangeringene, men på grunn av dårlige defensive rangeringer så rangeres det som nr 5 i den totale rangeringen. Lag 6 er et lag som rangeres overaskende høyt i den totale rangeringen, de rangeres som nr 4, dette skyldes enkelt og greit at de har vært rangert i topp fire 8 av de 18 rangeringene, samtidig er de rangert i bunn fire i 4 av de 18 rangeringene. Så som en liten konklusjon av disse rangeringene vil jeg si at lag 6 og lag 9 har gjort gode sesonger målmessig, men har ikke klart å skaffe seg de avgjørende poengene slik at de kan klatre oppover på den offesielle tabellen fra tabell 2.1.

Rangeringer fra seiersmatrisen, A.7:
Jeg nevnte i kapittel 2.4 at det også kunne være lurt å rangere lagene med hensyn til hvor mange kamper de har vunnet og tapt. Når vi studerer den totale rangeringen fra matrise A.7 var det ingen kjempe store overaskelser sammenlignet med den offesielle rekkefølgen fra tabell 2.1. Vi ser altså at de "beste" lagene er de som også vinner flest kamper, noe som er naturlig, og de "dårligste" vinner færrest kamper. Lag nr 10 er også her desidert sist, og lag 9 rangerers igjen foran lag 8. Vi ser også at lag 2 rangeres foran lag 1, og lag 4 foran lag 3, fra tabell 2.1 vet vi at det var tett mellom lagene, men ikke at lagene ville rangerers slik. Denne rangeringen må bety at lag 2 har vunnet flere kamper mot anntatt bra motstand enn det lag 1 har gjort, og tapt færre kamper mot anntatt dårligere motstand, på samme måte er det med lag 4 i forhold til lag 3. Lag 2 og 4 har dessverre ikke klart å få med seg nok poeng i de kampene de har vunnet til å gå forbi på den offesielle tabellen. Den store overaskelsen fra denne rangeringen er den offensive og defensive rangeringen til lag 5, deres offensive vurdering er bra mens deres defensive er dårlig, dette betyr at de har vunnet mange kamper mot bra motstandere

og den offensive vurderingen har fått et løft, men så har de også tapt flere kamper mot relativt dårlig motstand som betyr at deres defensive vurdering øker, så den defensive rangeringen synker. Dermed ender de totalt sett opp midt på treet, der de "hører hjemme" ifølge tabell 2.1.

Rangeringer fra poengmatrisen, A.8:
Den offensive rangeringen viser oss om det er lag som har tatt mange poeng mot relativt bra motstand, mens den defensive rangeringen viser oss om det er lag som har tapt poeng mot relaitvt dårlig motstand. Vi ser at denne totalrangeringen er veldig lik rangeringen fra tabell 2.1, noe som ikke er så merkelig siden begge baserer seg på poeng. Lag 4 og lag 3 hadde like mange poeng ifølge tabell 2.1 og det er jevnt mellom disse lagene i denne rangeringen også. Lag 4 rangeres høyere ifølge ODM fordi de har gjort det bedre poengmessig, altså tatt flere av sine poeng mot relativt bra motstand sammenlignet med lag 3. Vurderingene til lag 5, 6 og 7 er forholdsvis gjevne som også kan gjenspeiles av tabell 2.1, og igjen så rangeres lag 9 så vidt foran lag 8, med lag 10 på en klar sisteplass.

Rangeringer fra hjemmekampmatrisen, A.9:
Den siste rangeringen vi gjorde i kapittel 2.4 rangerer lagene ut ifra antall hjemme- og bortekamper. Når vi skal tolke disse rangeringene må vi gå ut ifra at alle lagene er like gode offensivt og alle lagene er like gode deffensivt, men at alle lagene vil ha en liten fordel når de spiller hjemmekamper. De fem lagene som rangeres først har liten forskjell i vurdering, men fra de fem første ned til de fem siste er det en reell forskjell. Dette betyr dermed at fem lag har hatt en liten fordel i Get-ligasesongen 15/16 på grunn av hvor kampene har blitt spilt. Lag 2, 3, 4, 6 og 8 har spilt én hjemmekamp mer enn de resterende lagene og hadde ut ifra ODM en antatt liten fordel av dette.

2.6 Kan en kombinasjonsranging forutse utfallet av en kamp?

Vi har til nå sett litt på flere rangeringer av lagene i Get-ligaen på bakgrunn av forskjellige kriterier, så spørsmålet blir: Hva kan disse rangeringene brukes til?

En naturlig måte å anvende en rangeringsmodell innenfor sport er ved å forsøke å forutse utfallet av kommende kamper. Vi kan bruke vurderingene og

rangeringene og sammenligne de to lagene som møtes, deretter må vi gjøre et par antagelser og kommer frem til at det best rangerte laget mest sannsynlig vil vinne kampen. Vi må jo også huske at selv om et lag rangeres bedre enn et annet vil det ikke være sikkert at det faktisk er så veldig mye bedre, det er litt avhengig av hva vi rangerer lagene ut ifra og størrelsesforskjellen mellom vurderingene.

De forskjellige rangeringene gir oss forkjellig informasjon om lagene. Baserer rangeringen seg på antall scorte mål i løpet av sesongen vil den ikke fortelle oss noen ting om hvordan enkeltkampene har gått eller hvem som vinner over hvem. For å få en forståelse av dette må vi se på rangeringen som baserer seg på antall seiere, eller antall poeng. Disse vil igjen ikke si oss om et lag er mye bedre enn et annet eller om de bare er litt bedre, da må vi se på rangeringen med hensyn på antall mål igjen.

Jeg mener på bakgrunn av dette at det er vanskelig å forutse utfallet av en kamp, dersom vi kun tar hensyn til én vurdering eller rangering. Jeg kommer nå med en alternativ rangeringsmetode av lagene som baserer seg på alle vurderingene vi har funnet ved hjelp av ODM. Jeg ønsker å kombinere de ulike vurderingene og sitte igjen med en total vurdering som gir oss en total rangering av lagene. Jeg kaller denne rangeringsmetoden for en *kombinasjonsrangering*. Kombinasjonsrangeringens algoritmen vil være å ta vurderinger vi får ODM og komponentvis multiplisere alle med hverandre slik at alle vurderingene blir påvirket av hverandre. Vi sitter da igjen med en total vurdering som baserer seg på flere andre vurderinger. Kombinasjonsrangeringen anvendt på vurderingene fra kapittel 2.4 blir:

$$\mathbf{R} = \mathbf{r}_1 \cdot \mathbf{r}_2 \cdot \mathbf{r}_3 \cdot \mathbf{r}_4 \cdot \mathbf{r}_5 \cdot \mathbf{r}_6 \cdot \mathbf{r}_7 \cdot \mathrm{r}_8 \cdot \mathbf{r}_9$$

$$\mathbf{R} = \begin{bmatrix} 5.1884628 & 0.7466837 & 0.1560201 & & \\ & 0.1505187 & 0.0224020 & 0.0317932 & \\ & 0.0014617 & 0.0000795 & 0.0002753 & 0.0000001 \end{bmatrix} \cdot 10^{16}$$

$$\textbf{Total kombinasjonsrangering} = \begin{bmatrix} 1 & 2 & 3 & 4 & 6 & 5 & 7 & 9 & 8 & 10 \end{bmatrix}$$

Det er flere av disse rangeringene som baserer seg på det samme, slik som at $\mathbf{r}_6$ på en måte er en sum av $\mathbf{r}_i$ for i fra 1 til 5. Jeg tenker at dette bare er bra for resultatet fordi alle rangeringene vurderer lagene på bakgrunn av litt forskjellige ting som betyr at ingen er helt like og jeg vil ha med så mye data som mulig. $\mathbf{r}_1$ til $\mathbf{r}_5$ baserer seg på antall mål, men er litt påvirket av hvem som vant de forskjellige kampene fordi alle tallene i de opprinnelige scorematrisene baserer seg på mål scoret i én kamp. $\mathbf{r}_6$ baserer seg på mål scort totalt som også kan være en fornuftig måte å rangere lagene etter.

Dersom vi nå studerer denne vurderingen vi fikk av lagene ser vi at det er svært lite som skiller lag 3 og 4, akkurat slik var det også ifølge tabell 2.1. Det andre vi kan se er at det virker som lag 6 og lag 9 gjorde bedre sesonger enn det tabell 2.1 viser. Bortsettfra disse små justeringene er jeg fornøyd med rangeringen jeg fikk, selv om det fortsatt er vanskelig å bruke denne til å forutse utfallet av fremtidige kamper. Det er forskjellen mellom vurderingene til hvert av lagene som blir avgjørende når det kommer til å forutse utfallet av kamper. Dersom denne forskjellen er liten vil det mest sannsynlig bli en jevn kamp, men dersom forskjellen er stort er det sannsynlig å tro at laget med den høyeste vurderingen vil vinne kampen.

Som en siste notis, ønsker jeg bare å påpeke at vi i ikke i noen av rangeringene har tatt hensyn til viktigheten av kampene. Dersom et dårlig rangert lag må vinne en kamp mot et godt lag for å ikke rykke ned, samtidig som det gode laget ikke har noe å spille for, kan det tenkes av det antatt svakeste laget vinner selv om alle rangeringer sier noe annet. Det er også mange andre aspekter vi ikke har tatt hensyn til, slik som om nøkkelspillere er skadet eller ikke spiller osv. Noen antagelser må man bare ta når vi skal prøve å forutse utfallet av kamper.

2.7 Tolkning av elementene i en dobbeltstokastisk matrise

I denne oppgaven har vi sett på rangeringer og vurderinger av lag ut ifra ulike scorematriser. Dersom scorematrisen multipliseres med to diagonalmatriser hvor elementene langs diagonalen er de inverse til den offensive og den defensive vurderingen fra ODM vil vi få ut en dobbeltstokastisk matrise, dette så vi på i kapittel 2.3. Nå ønsker jeg å se litt på hva elementene i disse ds-matrisene faktisk betyr og om de kan tolkes fornuftig.

Elementene i scorematrise A.6 viser oss det totale antallet mål lag i har scort på lag j, den dobbeltstokastiske matrisen vi får dersom vi bruker Sinkhorn-Knopp algoritmen på denne matrisen er:

$$A = \begin{bmatrix} 0 & 0.16 & 0.08 & 0.12 & 0.10 & 0.10 & 0.07 & 0.11 & 0.08 & 0.14 \\ 0.16 & 0 & 0.13 & 0.10 & 0.08 & 0.12 & 0.08 & 0.07 & 0.11 & 0.12 \\ 0.14 & 0.03 & 0 & 0.11 & 0.17 & 0.12 & 0.10 & 0.09 & 0.09 & 0.10 \\ 0.12 & 0.13 & 0.09 & 0 & 0.09 & 0.14 & 0.09 & 0.14 & 0.08 & 0.09 \\ 0.12 & 0.08 & 0.14 & 0.12 & 0 & 0.11 & 0.11 & 0.13 & 0.10 & 0.06 \\ 0.04 & 0.13 & 0.12 & 0.12 & 0.07 & 0 & 0.12 & 0.10 & 0.16 & 0.10 \\ 0.11 & 0.11 & 0.06 & 0.12 & 0.10 & 0.12 & 0 & 0.13 & 0.11 & 0.11 \\ 0.09 & 0.12 & 0.08 & 0.07 & 0.12 & 0.08 & 0.15 & 0 & 0.12 & 0.14 \\ 0.11 & 0.10 & 0.14 & 0.09 & 0.10 & 0.08 & 0.13 & 0.11 & 0 & 0.10 \\ 0.09 & 0.10 & 0.12 & 0.10 & 0.13 & 0.10 & 0.11 & 0.10 & 0.12 & 0 \end{bmatrix}$$

Vi vet at elementene i en stokastisk matrise representerer sannsynligheter for at et utfall skal skje, avhengig av hvilken rad og kolonne elementet er plassert i. Når det er snakk om en ds-matrise vil ett element i matrisen påvirke alle de andre elementene, alle elementene er altså avhengig av hverandre. Vi vet at alle rad- og kolonnesummer er lik 1, dermed kan det tenkes at elementene representerer ulike sannsynligheter for bestemte utfall.

Dersom vi forstår hva ett element forteller oss så forstår vi hva alle forteller oss, derfor ønsker jeg å finne ut hva elementet $a_{1,10} = 0.14$ betyr. Dette er plassen hvor scorematrisen har det antallet mål Stavanger har scort mot Kongsvinger totalt denne sesongen. Stavanger scorte 181 mål totalt i år og Kongsvinger slapp inn 225 mål. Totalt i år scorte Stavanger 40 mål mot Kongsvinger. Prosentandelen av de målene Stavanger har scort på Kongsvinger utgjør:

$$\frac{\text{Antall mål Stavanger har scort mot Kongsvinger}}{\text{Antall mål Kongsvinger har sluppet inn totalt}} = \frac{40}{225} = 0.1778$$

$$\frac{\text{Antall mål Stavanger har scort mot Kongsvinger}}{\text{Antall mål Stavanger har scort totalt}} = \frac{40}{181} = 0.2209$$

Ingen av disse tallene er 0.14, som betyr at $a_{1,10}$ må representere noe annet en akkurat disse forholdene. Det eneste vi vet om elementene i ds-matrisen er at de må handle om scorede mål og inslupne mål og at alle tallene påvirker hverandre. Etter å ha studert og sammenlignet ds-matrisen og den opprinnelige scorematrisen A.6 i vedlegg A.3, kan man se at der hvor det er stor forskjell i elementene a_{ij} og a_{ji} i scorematrisen vil det også være en tydelig forskjell mellom a_{ij} og a_{ji} i ds-matrisen. På bakgrunn av dette tenker jeg at et element i matrisen er et tall som forteller oss noe om hvor sannsynlig det er at lag i skal score på lag j. Det vil ikke være en direkte sannsynlighet fordi tallet vil være preget av hvor sannsynlig det er at lag i skal score

mot de andre lagene samtidig som det er preget av hvor sannsynlig det er at lag j slipper inn mål mot de andre lagene. Med dette i bakhodet velger jeg å runde av med den tanken at elementet a_{ij} representerer den vektede sannsynligheten for at lag i skal score på lag j. I tilfelle med Stavanger og Kongsvinger hvor $a_{1,10} = 0.14$ og $a_{10,1} = 0.09$ vil jeg påstå at det er sannsynlig å tro at Stavanger kommer til å score flere mål enn Kongsvinger i en tilfeldig kamp. På samme måte vil det være dersom vi ser på en scorematrise som baserer seg på poeng, eller seiere. Helt til slutt vil jeg bare nevne at dette dessverre ikke stemmer helt for alle lag. Dette mener jeg er på grunn av at alle elementene er avhengig av hverandre. Derfor ville jeg vært forsiktig med å forsøke å bruke en ds-matrise skalert fra en scorematrise for å forutse utfallet av fremtidige kamper. Dersom jeg skulle forsøke å forutse utfallet av en kamp ville jeg heller basert tankegangen min på de verdiene vi fikk fra kombinasjonsrangeringen etter å ha vurdert lagene ved hjelp av ODM. Er det stor forskjell mellom vurderingene to lag har kan det være sannsynlig å tro at laget med best vurdering vinner, men igjen så baserer dette seg på om vi har rangert ut ifra et sett med fornuftige scorematriser.

Tillegg A

Programmer og Scorematriser

A.1 Skalering av en matrise A til en dobbelt-stokastisk matrise

```
clc
clear all

K = 20;
n = 5;
A = randi(100,n)
e = ones(1,n);
eps = 10^-11;

for k=1:K
        k
        R = sum(A');
        D_1 = diag(R.^-1);
        A = D_1*A;
        S = sum(A)
        D_2 = diag(S.^-1);
        A = A*D_2;
        R = sum(A')
        S = sum(A);

        R_test = norm(R - e);
        S_test = norm(S - e);
                if R_test < eps & S_test < eps
                        A
                        break
                end
end
```

A.2 ODM

```
clc
clear all

K = 100;
n = 5;
A = randi(100,n)
e = ones(1,n);
eps = 10^-11;
d = ones(n,1);

for k=1:K
        k;
        D_2 = diag(d.^-1);
        o = sum((A*D_2)');
        D_1 = diag(o.^-1);
        d = sum(D_1*A);
        B = D_1*A*D_2;
        R = sum(B');
        S = sum(B);

        R_test = norm(R - e);
        S_test = norm(S - e);
                if R_test < eps & S_test < eps
                        B;
                        break
                end
end
k;
B;
d;
o;
r = o./d
```

A.3 Scorematrisene tilhørende kapittel 2.4

Alle scorematrisene under er satt opp med samme struktur, det vil alltid være snakk om at a_{ij} er et antall som lag i gjorde eller hadde mot lag j. Vi kan forstå det bedre ved å se på det lille eksemplet i teksten til den første matrisen A.1.

Matrise A.1 viser det antallet mål som ble scort og sluppet inn i den første runden av Get-ligaen 15/16. Matrisen viser også resultatene fra denne runden. Vi kan altså se at Lørenskog slo Kongsvinger 13 - 1 fordi $a_{2,10} = 13$ og $a_{10,2} = 1$

$$A = \begin{bmatrix} 0 & 6 & 4 & 10 & 3 & 2 & 2 & 7 & 5 & 5 \\ 2 & 0 & 2 & 2 & 4 & 3 & 4 & 1 & 2 & 13 \\ 3 & 1 & 0 & 2 & 5 & 3 & 3 & 3 & 2 & 7 \\ 2 & 4 & 1 & 0 & 1 & 5 & 5 & 10 & 1 & 6 \\ 2 & 0 & 4 & 2 & 0 & 2 & 3 & 3 & 4 & 4 \\ 1 & 2 & 4 & 0 & 0 & 0 & 5 & 3 & 10 & 2 \\ 3 & 2 & 2 & 3 & 2 & 3 & 0 & 7 & 3 & 8 \\ 1 & 2 & 1 & 0 & 2 & 4 & 3 & 0 & 5 & 5 \\ 2 & 3 & 3 & 0 & 1 & 1 & 1 & 2 & 0 & 3 \\ 0 & 1 & 3 & 3 & 1 & 0 & 3 & 1 & 2 & 0 \end{bmatrix} \tag{A.1}$$

Matrise A.2 viser det antallet mål som ble scort og sluppet inn i den andre runden av Get-ligaen 15/16.

$$A = \begin{bmatrix} 0 & 3 & 0 & 3 & 3 & 3 & 1 & 7 & 5 & 8 \\ 6 & 0 & 6 & 5 & 2 & 3 & 3 & 3 & 6 & 7 \\ 1 & 2 & 0 & 3 & 5 & 2 & 4 & 2 & 1 & 4 \\ 2 & 2 & 2 & 0 & 2 & 4 & 7 & 3 & 7 & 2 \\ 1 & 1 & 3 & 4 & 0 & 2 & 3 & 5 & 4 & 0 \\ 0 & 2 & 1 & 3 & 3 & 0 & 2 & 3 & 5 & 3 \\ 2 & 0 & 1 & 3 & 2 & 0 & 0 & 3 & 3 & 3 \\ 1 & 2 & 1 & 2 & 2 & 0 & 4 & 0 & 2 & 6 \\ 1 & 2 & 2 & 5 & 2 & 1 & 6 & 7 & 0 & 1 \\ 1 & 1 & 1 & 1 & 2 & 4 & 2 & 0 & 2 & 0 \end{bmatrix} \tag{A.2}$$

Matrise A.3 viser det antallet mål som ble scort og sluppet inn i den tredje runden av Get-ligaen 15/16.

$$A = \begin{bmatrix} 0 & 7 & 3 & 4 & 3 & 5 & 7 & 2 & 2 & 12 \\ 2 & 0 & 4 & 3 & 1 & 3 & 5 & 4 & 5 & 5 \\ 4 & 0 & 0 & 3 & 3 & 4 & 2 & 3 & 6 & 4 \\ 5 & 4 & 4 & 0 & 2 & 2 & 5 & 6 & 1 & 4 \\ 4 & 4 & 2 & 3 & 0 & 0 & 5 & 4 & 3 & 0 \\ 2 & 4 & 2 & 3 & 1 & 0 & 4 & 4 & 4 & 4 \\ 1 & 4 & 3 & 4 & 2 & 5 & 0 & 4 & 6 & 5 \\ 1 & 3 & 1 & 1 & 3 & 2 & 7 & 0 & 4 & 3 \\ 3 & 0 & 3 & 5 & 2 & 3 & 4 & 3 & 0 & 4 \\ 3 & 1 & 1 & 1 & 2 & 0 & 3 & 4 & 3 & 0 \end{bmatrix} \tag{A.3}$$

Matrise A.4 viser det antallet mål som ble scort og sluppet inn i den fjerde runden av Get-ligaen 15/16.

$$A = \begin{bmatrix} 0 & 3 & 4 & 3 & 0 & 3 & 5 & 6 & 3 & 2 \\ 1 & 0 & 2 & 2 & 1 & 2 & 5 & 5 & 3 & 4 \\ 2 & 0 & 0 & 3 & 1 & 2 & 4 & 6 & 3 & 3 \\ 4 & 7 & 4 & 0 & 4 & 2 & 3 & 5 & 4 & 6 \\ 1 & 2 & 5 & 2 & 0 & 3 & 2 & 7 & 2 & 4 \\ 1 & 6 & 5 & 9 & 1 & 0 & 4 & 6 & 4 & 8 \\ 1 & 3 & 1 & 2 & 3 & 5 & 0 & 8 & 6 & 6 \\ 0 & 2 & 3 & 2 & 1 & 0 & 3 & 0 & 3 & 6 \\ 1 & 1 & 2 & 1 & 1 & 3 & 4 & 2 & 0 & 7 \\ 0 & 1 & 1 & 2 & 2 & 0 & 0 & 3 & 1 & 0 \end{bmatrix} \tag{A.4}$$

Matrise A.5 viser det antallet mål som ble scort og sluppet inn i den femte og siste runden av Get-ligaen 15/16.

$$A = \begin{bmatrix} 0 & 3 & 1 & 3 & 4 & 1 & 1 & 2 & 2 & 13 \\ 5 & 0 & 3 & 5 & 2 & 5 & 1 & 2 & 5 & 3 \\ 2 & 1 & 0 & 6 & 4 & 3 & 6 & 3 & 3 & 5 \\ 0 & 2 & 2 & 0 & 3 & 7 & 2 & 7 & 4 & 7 \\ 1 & 1 & 0 & 5 & 0 & 4 & 4 & 1 & 2 & 4 \\ 0 & 0 & 2 & 3 & 3 & 0 & 6 & 2 & 3 & 6 \\ 3 & 4 & 0 & 7 & 2 & 2 & 0 & 2 & 2 & 4 \\ 3 & 1 & 1 & 3 & 2 & 1 & 4 & 0 & 2 & 4 \\ 1 & 4 & 4 & 1 & 3 & 0 & 5 & 3 & 0 & 5 \\ 0 & 2 & 1 & 1 & 0 & 2 & 2 & 1 & 2 & 0 \end{bmatrix} \tag{A.5}$$

Matrise A.6 viser det antallet mål som ble scort og sluppet inn totalt i løpet av alle de fem rundene i Get-ligaen.

$$A = \begin{bmatrix} 0 & 22 & 12 & 23 & 13 & 14 & 16 & 24 & 17 & 40 \\ 16 & 0 & 17 & 17 & 10 & 16 & 18 & 15 & 21 & 32 \\ 12 & 4 & 0 & 17 & 18 & 14 & 19 & 17 & 15 & 23 \\ 13 & 19 & 13 & 0 & 12 & 20 & 22 & 31 & 17 & 25 \\ 9 & 8 & 14 & 16 & 0 & 11 & 17 & 20 & 15 & 12 \\ 4 & 14 & 14 & 18 & 8 & 0 & 21 & 18 & 26 & 23 \\ 10 & 13 & 7 & 19 & 11 & 15 & 0 & 24 & 20 & 26 \\ 6 & 10 & 7 & 8 & 10 & 7 & 21 & 0 & 16 & 24 \\ 8 & 10 & 14 & 12 & 9 & 8 & 20 & 17 & 0 & 20 \\ 4 & 6 & 7 & 8 & 7 & 6 & 10 & 9 & 10 & 0 \end{bmatrix} \tag{A.6}$$

Matrise A.7 viser en oversikt over antallet kamper som er vunnet og tapt mellom lagene i løpet av Get-ligaen 15/16. Den er bygget opp slik at element a_{ij} er antall seiere lag i har over lag j.

$$A = \begin{bmatrix} 0 & 3 & 2 & 3 & 3 & 5 & 2 & 4 & 4 & 5 \\ 2 & 0 & 5 & 2 & 3 & 3 & 4 & 4 & 4 & 5 \\ 3 & 0 & 0 & 3 & 4 & 3 & 4 & 5 & 2 & 5 \\ 2 & 3 & 2 & 0 & 1 & 3 & 4 & 5 & 4 & 5 \\ 2 & 2 & 1 & 4 & 0 & 3 & 4 & 4 & 4 & 3 \\ 0 & 2 & 2 & 2 & 2 & 0 & 3 & 4 & 5 & 4 \\ 3 & 1 & 1 & 1 & 1 & 2 & 0 & 2 & 3 & 5 \\ 1 & 1 & 0 & 0 & 1 & 1 & 3 & 0 & 3 & 4 \\ 1 & 1 & 3 & 1 & 1 & 0 & 2 & 2 & 0 & 4 \\ 0 & 0 & 0 & 0 & 2 & 1 & 0 & 1 & 1 & 0 \end{bmatrix} \tag{A.7}$$

Matrise A.8 er bygget opp slik at a_{ij} viser antallet poeng lag i har tatt i kamper mot lag j i løpet av de 5 rundene i Get-ligaen.

$$\begin{bmatrix} 0 & 9 & 6 & 10 & 11 & 15 & 7 & 13 & 13 & 15 \\ 6 & 0 & 15 & 7 & 8 & 9 & 11 & 11 & 13 & 15 \\ 9 & 0 & 0 & 8 & 12 & 8 & 12 & 14 & 8 & 15 \\ 5 & 8 & 7 & 0 & 4 & 10 & 10 & 15 & 12 & 15 \\ 4 & 7 & 3 & 11 & 0 & 9 & 11 & 11 & 12 & 9 \\ 0 & 6 & 7 & 5 & 6 & 0 & 10 & 13 & 14 & 13 \\ 8 & 4 & 3 & 5 & 4 & 5 & 0 & 6 & 9 & 14 \\ 2 & 4 & 1 & 0 & 4 & 2 & 9 & 0 & 9 & 12 \\ 2 & 2 & 7 & 3 & 3 & 1 & 6 & 6 & 0 & 12 \\ 0 & 0 & 0 & 0 & 6 & 2 & 1 & 3 & 3 & 0 \end{bmatrix} \tag{A.8}$$

Matrise A.9 viser en oversikt over antallet hjemme- og bortekamper i løpet av Get-ligaen 15/16. Den er bygget opp slik at element a_{ij} er antall hjemmekamper lag i hadde mot lag j.

$$A = \begin{bmatrix} 0 & 3 & 2 & 3 & 2 & 2 & 2 & 2 & 3 & 3 \\ 2 & 0 & 3 & 2 & 2 & 3 & 3 & 3 & 2 & 3 \\ 3 & 2 & 0 & 3 & 3 & 3 & 2 & 2 & 3 & 2 \\ 2 & 3 & 2 & 0 & 2 & 3 & 3 & 3 & 2 & 3 \\ 3 & 3 & 2 & 3 & 0 & 2 & 2 & 2 & 3 & 2 \\ 3 & 2 & 2 & 2 & 3 & 0 & 3 & 2 & 3 & 3 \\ 3 & 2 & 3 & 2 & 3 & 2 & 0 & 3 & 2 & 2 \\ 3 & 2 & 3 & 2 & 3 & 3 & 2 & 0 & 3 & 2 \\ 2 & 3 & 2 & 3 & 2 & 2 & 3 & 2 & 0 & 3 \\ 2 & 2 & 3 & 2 & 3 & 2 & 3 & 3 & 2 & 0 \end{bmatrix} \tag{A.9}$$

Bibliografi

[1] G. Birkhoff, Tres observaciones sobre el algebra lineal, National University of Tucumán, (1946), 147-150

[2] R.A. Brualdi, *Combinatorial Matrix Classes*, Cambridge University Press, Cambridge, 2006.

[3] G. Dahl *Network flows and combinatorial matrix theory*, University of Oslo, 2013

[4] A.Y. Govan, A.N. Langville og C.D. Meyer, *Offense-defense approach to ranking team sports* , Journal of Quantitative Analysis in Sports, Article 4. 2009, 5(1),

[5] A.N. Langville og C.D. Meyer, *Google's PageRank and Beyond: The Science of Search Engine Rankings*, Princeton University Press, June 2006.

[6] A.N. Langville og C.D. Meyer, *Who's ♯1: The Science of Rating and Ranking.*, Princeton University Press, 2012.

[7] Norges ishockeyforbund, `http://get-ligaen.stats.pointstreak.com/scoreboard.html`