

## Egenvektorer, egenvärden och SVD

### Teoriövningar

1. Antag att en  $3 \times 3$ -matris  $A$  har egenvärdena  $\lambda_1 = 1$ ,  $\lambda_2 = 0.8$ ,  $\lambda_3 = 0.6$  med motsvarande egenvektorer

$$\mathbf{v}_1 = \begin{pmatrix} 1 \\ 0 \\ 2 \end{pmatrix}, \mathbf{v}_2 = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} \text{ och } \mathbf{v}_3 = \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}.$$

Låt  $\mathbf{v} = x_1\mathbf{v}_1 + x_2\mathbf{v}_2 + x_3\mathbf{v}_3$  vara en linjärkombination av de tre egenvektorerna.

- (a) Vilka 3-vektorer kan skrivas som en linjärkombination av de tre egenvektorerna.
  - (b) Vad är  $A^n\mathbf{v}_1$ ,  $A^n\mathbf{v}_2$ ,  $A^n\mathbf{v}_3$  respektive  $A^n\mathbf{v}$  där  $n$  är ett godtyckligt heltal.
  - (c) Vad händer med  $A^n\mathbf{v}$  då  $n \rightarrow \infty$ ? Se upp, det beror lite på koefficienterna  $x_1$ ,  $x_2$  och  $x_3$ .
  - (d) Antag nu att vi har en  $3 \times 3$ -matris med några egenvärden  $\lambda_1$ ,  $\lambda_2$  och  $\lambda_3$ . Svara på förra frågan igen för den här lite mer godtyckliga matrisen.
  - (e) Hur kan man bestämma en matris  $A$  som har de egenvärden och egenvektorer som vi föreskrev från början. Ni behöver inte utföra beräkningarna. Det räcker med att ni ger en 'formel'. Tips: Det har något med diagonalisering att göra.
2. Läs igenom och diskutera potensmetoden och inversiteration (avsnitt 7.5 i boken) så att alla förstår hur det fungerar.
  3. Bestäm alla möjliga ON-matriser av storlek  $2 \times 2$ . Vad för typ av avbildningar är de.

### Datorövningar

1. Vi återvänder till uppgift 2.58 i boken. Bestäm med hjälp av egenvärden och egenvektorer hur fördelningen mellan människor i storstad, tätort och landsbygd är efter LÅNG tid. (Man kan beräkna egenvärden och egenvektorer med kommandot 'eig'.)
2. Skriv Matlab-funktioner som:
  - (a) Givet en matris  $A$  och en precision  $p$  beräknar det största egenvärdet till  $A$  med hjälp av potensmetoden med sådan precision att skillnaden mellan två på varandra följande Rayleigh-kvoter i iterationen är mindre än  $10^{-p}$ . Tänk på att normalisera vektorerna. Kommandot 'norm(x)' ger längden av vektorn  $\mathbf{x}$ . Låt gärna precisionen vara ett valfritt argument och lägg in en standardprecision som används om man bara ger en matris. Antalet inparametrar kan man kontrollera med kommandot 'nargin'.

- (b) Samma sak fast istället det minsta egenvärdet med inversiteration. Observera att detta är ett bra tillfälle att utnyttja LU-faktoriseringen.
- (c) Givet samma som ovan plus ett tal  $r$  beräknar det egenvärde som är närmast  $r$ .

Förutom att returnera egenvärdet ska funktionerna också skriva ut hur många iterationer som krävdes. Testa era funktioner på några slumpmatriser (tag också gärna lite större sådana t ex  $500 \times 500$ ) genom att jämföra med resultatet från Matlabs rutin 'eig' som beräknar alla egenvärden (inklusive icke-reella) med hjälp av den så kallade QR-faktoriseringen. Vad händer med inversiterationen om man väljer ett värde att leta närmaste egenvärde kring som ligger långt ifrån egenvärdena?

3. Vi ska nu skriva Matlabkod för att illustrera SVD. Antag att vi har en (svartvit) bildfil i tex png, jpg eller gif-format (Matlab klarar ännu fler format). Det finns en att hämta på hemsidan, men ta gärna en egen (fast inte alltför stor). Man kan läsa in den till en matris i Matlab med hjälp av kommandot 'imread'. För att kunna göra beräkningar på den måste man dessutom konvertera den till 'normalt' format så `A=double(imread('bildfilen'))`; läser in bilden till  $A$ . Om man sedan vill visa den så använder man kommandona `imagesc(A)` och `colormap(gray)`. Antag att vi har en SVD-uppdelning  $A = U\Sigma V^t$  av  $A$ . Detta kan skrivas som (se föreläsningsanteckningar)

$$A = \sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^t.$$

Vi ser att  $A$  är skriven som en summa av  $r$  stycken matriser. Man får en approximation av  $A$  genom att ta en delsumma. Denna blir bra om de termer man inte tagit med har små singularvärden i förhållande till det största  $\sigma_1$ .

- (a) Skriv Matlabkod som ritar upp den ursprungliga bilden och sedan ritar upp en approximation av bilden genom att ta matrisen

$$\sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{v}_i^t,$$

för olika  $n$ . Gör en funktion som har SVD-uppdelningen (d v s matriserna  $U$ ,  $V$  och  $\Sigma$ ) samt  $n$  som inparametrar och som producerar motsvarande bild. (För att skapa ett nytt figurfönster använder man 'figure'.)

- (b) Hur många termer behövde ni för att bilden skulle bli acceptabel? Hur mycket behöver man lagra om man vill ha  $n$  termer i summan? Om vi har en bild som är  $100 \times 100$  pixlar, hur många termer kan man ta med i summan för att komprimera storleken till hälften?

Uppgift 1 bland teoriuppgifterna samt uppgift 3 bland datoruppgifterna ska redovisas skriftligt till Stefan. Sista inlämningsdag är måndagen den 3 mars. Instruktioner för redovisningen finns på hemsidan. Läs dessa noggrant!