SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

Maskininlärning med linjära system

av

Emanuel Hedberg

2025 - No K9

Maskininlärning med linjära system

Emanuel Hedberg

Självständigt arbete i matematik 15 högskolepoäng, grundnivå

Handledare: Yishao Zhou

2025

Abstract

This thesis explores some of the underlying mathematics behind machine learning, with a focus on linear algebra, especially Sylvester equations. These equations make appearances in areas such as control theory, but also even in machine learning contexts, such as weight-initialization, image processing, or multi-label learning. Research has shown how some optimization or labeling problems can be reduced to solving a Sylvester equation. Moreover, this thesis goes into both numerical methods of solving these equations as well as the necessary preconditions which guarantee a unique solution, with accompanying proof.

Sammanfattning

Denna uppsats undersöker en del av den matematiska grunden för maskininlärning med fokus på linjär algebra, särskilt Sylvesterekvationer. Dessa ekvationer förekommer bland annat inom styrteori, men används även i maskininlärningssammanhang såsom viktinitialisering, bildbehandling och multietikett-inlärning. Forskning har visat att vissa optimerings- och etiketteringsproblem kan reduceras till lösning av en Sylvesterekvation. Uppsatsen behandlar både numeriska metoder för att lösa sådana ekvationer och de nödvändiga villkor som garanterar en unik lösning, med tillhörande bevis.

Innehåll

1 Inledning

En maskin kan ses som ett slags verktyg som använder energi för att utföra ett arbete. Ända sedan hjulet uppfanns i Mesopotamien, cirka 3500 f.Kr., har människor uppfunnit och utvecklat maskiner för att göra våra liv enklare och mindre mödosamma. Den senaste tiden har idén om artificiell intelligens, att maskiner kan lära sig utföra mänskliga uppgifter till en sådan grad att de blir svåra att skilja från något som en människa gjort, blivit mer vanlig. Idag är ämnet mer populärt än någonsin, och det kommer troligen förbli så. Ändå finns det många frågor om exakt vad AI är och hur maskininlärning ser ut på den matematiska nivån. Det kommer fokuseras på en viss typ av matematik som understryker: linjär algebra och Sylvesterekvationer. Mer om det i senare avsnitt.

Maskininlärningsmodeller kan delas upp i fyra olika kategorier [Dev]:

1. Övervakad inlärning
2. Oövervakad inlärning
3. Förstärkningsinlärning
4. Generativ AI.

Övervakade inlärningsmodeller kan göra förutsägelser efter att ha observerat mycket data med de rätta svaren. De jämför sin förutsägelse med det rätta svaret, och justerar sina parametrar för att minimera felet. Processen upprepas tills modellen kan prestera "tillräckligt bra". En liknelse är en student som studerar inför en tenta genom att träna på många gamla tentor som också innehåller de rätta svaren. På så sätt lär sig studenten nytt material och gör sig redo för att ta den nya tentan med nya frågor utan tillgång till rätta svar. Exempel på sådana modeller är regressionsmodeller som predikterar ett numeriskt värde, och klassifikationsmodeller som klassificerar ett objekt genom att prediktera sannolikheten att objektet tillhör en viss kategori.

Oövervakade inlärningsmodeller gör prediktioner utifrån data som de får, men utan tillgång till de korrekta svaren. Syftet med en oövervakad modell är att upptäcka betydelsefulla mönster i data. En sådan modell kan använda sig av en teknik som kallas "klustring", där den grupperar data baserat på en eller flera olika faktorer.

Klustring är skilt från klassificering i och med att kategorierna inte är definierade av dig.

Förstärkningsinlärningsmodeller gör prediktioner genom att få belöningar eller straff baserat på deras handlingar. Målet är att få dem att följa en strategi som leder till så många belöningar som möjligt. Förstärkningsinlärning kan användas för att träna robotar att färdas i ett rum, eller mjukvaruprogram att spela spel.

Generativ AI utgörs av modeller som genererar media av någon form (exempelvis text, bild, eller video) baserat på en användares indata av viss form. En sådan modell har observerat mönster i data och lärt sig imitera data på vilken den tränats, genom en oövervakad metod. Den kan senare tränas övervakat eller genom förstärkning på specifik data som uppgifter den förväntas kunna utföra. Exempel på generativ AI är stora språkmodeller som ChatGPT.

I följande delavsnitt tas en djupare titt på en vanligt förekommande typ av modell, och matematik bakom den, för att skapa en djupare förståelse inför kommande avsnitt.

1.1 Djupa neuronnätverk

Djupa neuronnätverk är en slags modeller som kan förekomma i bland annat övervakad inlärning eller generativ AI. De kan användas för många olika sorters uppgifter, som bland annat bildigenkänning, språkbehandling, eller taligenkänning. Hur de fungerar matematiskt beskrivs av [San17c, San17d, San17b, San17a], och [Str19, Avsnitt VI.4, VII.1, VII.3].

Tag som exempel en svart-vit bild i låg upplösning av siffran “3” ritad av en människa. En annan människa skulle troligen inte ha några problem att identifiera vad bilden ska föreställa, men att få en maskin att korrekt identifiera symbolen baserat på bilden är inte lika lätt. Tanken med neuronnätverk är att de är inspirerade av den mänskliga hjärnan i hur de bör tolka invärdet och behandla det för att producera en rimlig prediktion, ett utvärde.

Ett nätverk består av lager: ett indatalager som representerar indatan, ett utdatalager som beskriver möjliga utdatan, och ett antal gömda lager emellan för hantering av data. Varje lager består av ett visst antal noder, där varje nod har ett visst numeriskt värde som kallas dess “aktivering”. För lager l definierar man vektorn $a^{(l)}$,

där $a_k^{(l)}$ är aktiveringen för nod k i lagret. Man börjar alltså med indatalagret $a^{(0)}$ som representerar bilden och dess pixlars aktiveringar. Då bilden är svart-vit kan man tänka sig att varje $a_k^{(l)} \in [0, 1]$, där 0 betyder svart och 1 betyder vit.

Framdataledning är algoritmen för hur ett lagers aktiveringar beräknas utifrån det föregående. Det ser ut som följande:

$$z^{(l)} := W^{(l)} a^{(l-1)} + b^{(l)}$$
$$a^{(l)} = \sigma^{(l)}(z^{(l)}).$$

Här är $W^{(l)}$ den så kallade viktmatrisen, och $b^{(l)}$ är biasvektorn, från lager $l-1$ till l. Funktionerna $\sigma^{(l)}$ är så kallade aktiveringsfunktioner vilka "aktiverar" lagren. Som förenkling kommer samma funktion att användas mellan varje lager i det här exemplet. Ett exempel på aktiveringsfunktion är sigmoidfunktionen $\sigma(x) = \frac{1}{1+e^{-x}}$.

Det är värt att poängtera att varje lager inte behöver vara lika stort. Om lager l består av n_l noder och $l-1$ består av n_{l-1} noder gäller $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ och $b^{(l)} \in \mathbb{R}^{n_l}$. Matriserna $W^{(l)}$ och vektorerna $b^{(l)}$ utgör nätverkets parametrar och dikterar hur väl optimerat det är. När du har kommit till utdatalagret har du en aktiveringsvektor $a^{(L)}$ som är nätverkets prediktion. I det här fallet kanske utdatalagret består av 10 noder, en för varje siffra $0-9$. Vektorn $a^{(L)}$ representerar hur starkt modellen "tror" att varje siffra befinner sig i bilden, där 0 betyder att siffran klart inte är korrekt och 1 betyder att siffran definitivt är korrekt.

Om nätverket är otränat kommer $a^{(L)}$ troligen verka slumpmässig och inte vara nära det korrekta svaret y. Bakdataledning handlar då om att mäta felet eller avståndet mellan $a^{(L)}$ och y, för att sedan uppdatera parametrarna för att minimera felet och således optimera nätverkets förmåga att prediktera korrekt utdata. För att mäta felet beräknas kostfunktionen J, vilket kan göras exempelvis genom medelkvadratfelet

$$J = \frac{1}{2} \left|\left| a^{(L)} - y \right|\right|_2^2.$$

Det är också möjligt att mer än ett exempel på indata testas på en gång, i vilket fall beräknas J som medelvärdet av varje exempels kostfunktion. Eftersom $a^{(L)}$ beräknades med avseende på alla viktmatriser och bias-vektorer kommer J vara en flervariabelfunktion med avseende på nämnda matrisers och vektorers värden. I vanliga fall löser man ekvationen $\nabla J = 0$ för att hitta invärdena vilka minimerar kostfunktionen, men då det är så många variabler involverade är den här metoden

orealistisk. En iterativ metod är gradientnedstigning.

En intuitition för gradientnedstigning är att man befinner sig på en yta med viss sluttning och följer sluttningen en bit ned mot dalens botten. Tag som exempel envariabelfunktionen $f : \mathbb{R} \to \mathbb{R}$ som $f(x) = x^4 - x + 1$ vilken har ett globalt minimum i $x' \in (0, 1)$. Om den initiella gissningen x_0 uppfyller $f'(x_0) > 0$ så gäller $x_0 > x'$. Uppdateras x_0 som $x_0 \leftarrow x_0 - \eta f'(x_0)$ för någon konstant $\eta > 0$ av godtycklig storlek får man ett nytt värde som är lite närmare x'. Om däremot $f'(x_0) < 0$ så gäller $x_0 < x'$, så samma uppdateringsformel ökar värdet på x_0 och för det närmare det sanna värdet. Upprepade sådana uppdateringar, eller gradientnedstigningar, för det numeriska värdet närmare det exakta värdet.

Samma princip kan appliceras på flervariabelfunktioner, då man använder funktionens gradienter. Så, uppdateringsformlerna ser ut som

$$W^{(l)} \leftarrow W^{(l)} - \eta \nabla_{W^{(l)}} J$$
$$b^{(l)} \leftarrow b^{(l)} - \eta \nabla_{b^{(l)}} J.$$

Vad som menas med ∇_X då X är en matris, är att de partiella derivatorna av den skalära funktionen med avseende på X_{ij} för varje par i, j sätts in i en matris av samma storlek som X. Alltså, om $X \in \mathbb{R}^{m \times n}$ och $f \colon \mathbb{R}^{m \times n} \to \mathbb{R}$ är en skalär funktion gäller

$$\nabla_X f = \begin{pmatrix} \frac{\partial f}{\partial X_{11}} \cdots \frac{\partial f}{\partial X_{1n}} \\ \cdots \\ \frac{\partial f}{\partial X_{m1}} \cdots \frac{\partial f}{\partial X_{mn}} \end{pmatrix} \in \mathbb{R}^{m \times n}.$$

Då x är en vektor i $\mathbb{R}^m$ gäller att gradienten är en vektor av samma dimension, alltså

$$\nabla_x f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \cdots \\ \frac{\partial f}{\partial x_m} \end{pmatrix} \in \mathbb{R}^m.$$

Det är något annorlunda i fallet då f är en vektorvärd funktion med komponenter $f_1, \cdots, f_m$. Om x är en vektor i $\mathbb{R}^n$ definieras Jacobi-matrisen

$$\frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} \cdots \frac{\partial f_1}{\partial x_n} \\ \cdots \\ \frac{\partial f_m}{\partial x_1} \cdots \frac{\partial f_m}{\partial x_n} \end{pmatrix}.$$

Om X är en matris i $\mathbb{R}^{p\times q}$ är $\frac{\partial f}{\partial X}$ strikt sett en tensor av storlek $m \times p \times q$, men en konvention är att representera den som en matris enligt

$$\frac{\partial f}{\partial X} := \frac{\partial f}{\partial \text{Vec}(X)} \in \mathbb{R}^{m\times pq}$$

där $\text{Vec}(X)$ är X uttryckt som en kolumnvektor.

I sammanhanget av neuronnätverk kallas η inlärningshastigheten, eftersom den dikterar hur mycket parametrarnas värden förändras. För att beräkna gradienterna används feltermer som är definierade enligt $\delta^{(l)} := \frac{\partial J}{\partial z^{(l)}}$ för varje lager l. Feltermerna kan beräknas enligt en rekursiv formel genom kedjeregeln:

$$\begin{aligned}
\delta^{(L)} &= \frac{\partial J}{\partial a^{(L)}} \cdot \frac{\partial a^{(L)}}{\partial z^{(L)}} = (a^{(L)} - y) \odot \sigma'(z^{(L)}) \\
\delta^{(l)} &= \frac{\partial z^{(l+1)}}{\partial a^{(l)}} \cdot \frac{\partial J}{\partial z^{(l+1)}} \cdot \frac{\partial a^{(l)}}{\partial z^{(l)}} = ((W^{(l+1)})^T \delta^{(l+1)}) \odot \sigma'(z^{(l)}).
\end{aligned}$$

Med $\odot$ menas elementvis multiplikation. Vi använder kedjeregeln för att beräkna gradienterna av J:

$$\begin{aligned}
\nabla_{W^{(l)}} J &= \frac{\partial J}{\partial z^{(l)}} \cdot \frac{\partial z^{(l)}}{\partial W^{(l)}} = \delta^{(l)} (a^{(l-1)})^T \\
\nabla_{b^{(l)}} J &= \left(\frac{\partial z^{(l)}}{\partial b^{(l)}}\right)^T \cdot \frac{\partial J}{\partial z^{(l)}} = \delta^{(l)}.
\end{aligned}$$

Det följer eftersom vi definierade $\delta^{(l)} := \frac{\partial J}{\partial z^{(l)}}$, samt $z^{(l)} := W^{(l)} a^{(l-1)} + b^{(l)}$. Vi får $\frac{\partial z^{(l)}}{\partial W^{(l)}} = (a^{(l-1)})^T$ enligt konventionen att vektorisera $W^{(l)}$. Transponering av $a^{(l-1)}$ sker för att matcha dimensionerna. Dessutom gäller $\frac{\partial z^{(l)}}{\partial b^{(l)}} = I$.

Vid många upprepade tester med framdataledning och uppdateringar med bakdataledning tränas nätverket och blir bättre på att förutsäga korrekt utdata, både för träningsdata och helt ny indata.

2 Definitioner och satser

Följande kapitel innehåller definitioner och satser med bevis som är relevanta i arbetet.

Sats 2.1. *Låt $X \in \mathbb{R}^{p\times r}$ och $Y \in \mathbb{R}^{r\times p}$. Då gäller*

$$\mathit{trace}(XY) = \mathit{trace}(YX).$$

Bevis. Vi har

$$\operatorname{trace}(XY) = \sum_{i=1}^{p}(XY)_{ii} = \sum_{i=1}^{p}\sum_{k=1}^{r} X_{ik}Y_{ki}.$$

Genom symmetri har vi

$$\operatorname{trace}(YX) = \sum_{i=1}^{r}\sum_{k=1}^{p} Y_{ik}X_{ki}.$$

Observera följande:

$$\begin{aligned}\operatorname{trace}(XY) &= \sum_{i=1}^{p}\sum_{k=1}^{r} X_{ik}Y_{ki}\\ &= \sum_{k=1}^{r}\sum_{i=1}^{p} Y_{ki}X_{ik}\\ &= \sum_{i=1}^{r}\sum_{k=1}^{p} Y_{ik}X_{ki}\\ &= \operatorname{trace}(YX).\end{aligned}$$

□

Korollarium 2.2 (Sats 2.1)**.** *Låt $A_1, \cdots, A_n$ vara matriser sådana att $A_1A_2\cdots A_n$ är kvadratisk. Då är spårfunktionen invariant under cykliska permutationer, det vill säga*

$$\mathit{trace}(A_1A_2\cdots A_n) = \mathit{trace}(A_2\cdots A_nA_1) = \cdots = \mathit{trace}(A_nA_1\cdots A_{n-1}).$$

Definition 2.3. För $m, n \in \mathbb{Z}^+$, låt $A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}$, och $C \in \mathbb{C}^{m\times n}$ vara givna. En kontinuerlig Sylvesterekvation är en matrisekvation av formen

$$AX - XB = C.$$

Definition 2.4. Låt $A \in \mathbb{C}^{m\times m}$. Vi noterar mängden av egenvärdena av A som $\Lambda(A)$.

Definition 2.5. Vi noterar det karaktäristiska polynomet för A som p_A, där $A \in \mathbb{C}^{m\times m}$. Det definieras som $p_A(t) = \det(tI - A)$, och dess rötter är egenvärdena av A.

Lemma 2.6. *Låt $T\colon V \to W$ vara en linjär avbildning och V, W vara två vektorrum. Då gäller* $\dim(\ker(T)) + \dim(Im(T)) = \dim(V)$.

Lemma 2.7. *Varje kvadratisk matris uppfyller sin egen karaktäristiska ekvation. Det vill säga, för $A \in \mathbb{C}^{m\times m}$ gäller $p_A(A) = 0$.*

Bevis. Antag först att A är diagonaliserbar. Då finns en inverterbar S och en matris $D = diag(\lambda_1, \cdots, \lambda_m)$ sådana att $A = SDS^{-1}$. Eftersom $A^k = SD^kS^{-1}$ där $D^k = diag(\lambda_1^k, \cdots, \lambda_m^k)$, betyder det att $p_A(A) = Sp_A(D)S^{-1}$ där $p_A(D) = diag(p_A(\lambda_1), \cdots, p_A(\lambda_m))$. Varje λ_i är en rot till p_A, vilket betyder att $p_A(D) = 0$ och därmed $p_A(A) = 0$.

I allmänhet kan vi använda det faktum att varje matris kan approximeras av diagonaliserbara matriser. Givet $A \in \mathbb{C}^{m\times m}$ kan vi finna en följd av matriser $\{A_k\colon k \in \mathbb{N}\}$ så att $A_k \to A$ då $k \to \infty$ och varje A_k har m entydiga egenvärden. Således är varje A_k diagonaliserbar. Det följer därmed att $p_k(A_k) = 0$ för varje k, där p_k är det karaktäristiska polynomet för A_k. Eftersom $\lim_{k\to\infty} A_k = A$ kan vi se att $\lim_{k\to\infty} p_k(A_k) = p_A(A)$. Då varje $p_k(A_k) = 0$ måste vi ha $p_A(A) = 0$. □

Lemma 2.8. *Låt $A \in \mathbb{C}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$, och $X \in \mathbb{C}^{m\times n}$. Om $AX - XB = 0$, gäller $g(A)X - Xg(B) = 0$ för alla polynom $g(t)$.*

Bevis. Ekvationen $AX - XB = 0$ kan skrivas om som $AX = XB$. Antag att $A^kX = XB^k$ för något $k > 1$, då basfallet $k = 1$ håller. Observera

$$A^{k+1}X = A(A^kX) = A(XB^k) = (AX)B^k = (XB)B^k = XB^{k+1}.$$

Således får man att $A^kX = XB^k$ för varje $k \in \mathbb{N}$. Betrakta polynomet

$$g(t) = \sum_{k=0}^{N} c_k t^k$$

givet skalärer $c_0, \cdots, c_N$. Notera att

$$g(A)X = \left(\sum_{k=0}^{N} c_k A^k\right) X = \sum_{k=0}^{N} c_k (A^kX) = \sum_{k=0}^{N} c_k XB^k = X\left(\sum_{k=0}^{N} c_k B^k\right) = Xg(B).$$

Lemmat följer. □

Sats 2.9. *Låt $A \in \mathbb{C}^{m\times m}$ och $B \in \mathbb{C}^{n\times n}$ vara givna. För varje $C \in \mathbb{C}^{m\times n}$, har ekvationen $AX - XB = C$ en unik lösning $X \in \mathbb{C}^{m\times n}$ om och endast om $\Lambda(A) \cap \Lambda(B) = \emptyset$.*

Bevis. [HJ13, s. 112] Låt $T\colon \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ vara en linjär avbildning definierad genom $T(X) = AX - XB$. Betrakta ekvationen $T(X) = 0$. Genom Lemma 2.8 fås att $p_B(A)X - Xp_B(B) = 0$. Lemma 2.7 garanterar att $p_B(B) = 0$, och därför gäller $p_B(A)X = 0$. Vi har alltså

$$p_B(A)X = \left(\prod_{\lambda\in\Lambda(B)} (A - \lambda I)\right) X = 0.$$

Först demonstreras att $\Lambda(A) \cap \Lambda(B) = \emptyset$ är ekvivalent med $\ker(T) = \{0\}$.

Om $\Lambda(A) \cap \Lambda(B) = \emptyset$, så är $A - \lambda I$ ickesingulär för varje $\lambda \in \Lambda(B)$ eftersom $(A - \lambda I)v = 0$ endast har lösningen $v = 0$. Det betyder att $p_B(A)$ är ickesingulär, alltså är $X = 0$ den enda lösningen till $p_B(A)X = 0$. Eftersom $AX - XB = 0$ medför $p_B(A)X = 0$ där den enda lösningen var den triviala, innebär det $\ker(T) = \{0\}$.

Om $\Lambda(A)\cap\Lambda(B) \neq \emptyset$, finns det $\lambda \in \Lambda(A)\cap\Lambda(B)$. Därmed finns nollskilda $u \in \mathbb{C}^m$ och $v \in \mathbb{C}^n$ sådana att

$$Au = \lambda u,$$

$$v^T B = \lambda v^T.$$

Observera att matrisen $uv^T \in \mathbb{C}^{m\times n}$ uppfyller följande:

$$\begin{aligned} A(uv^T) - (uv^T)B &= (Au)v^T - u(v^T B) \\ &= (\lambda u)v^T - u(\lambda v^T) \\ &= \lambda uv^T - \lambda uv^T \\ &= 0. \end{aligned}$$

Således existerar ett nollskilt $X \in \mathbb{C}^{m\times n}$ sådant att $AX - XB = 0$, och det innebär $\ker(T) \neq \{0\}$. Det har då visats att $\Lambda(A)\cap\Lambda(B) = \emptyset$ och $\ker(T) = \{0\}$ är ekvivalenta. Nu återstår att visa $\ker(T) = \{0\}$ är ekvivalent med att $T(X) = C$ har en entydig lösning för varje $C \in \mathbb{C}^{m\times n}$.

Antag $\ker(T) = \{0\}$. Enligt Lemma 2.6 gäller

$$\dim(\ker(T)) + \dim(\mathrm{Im}(T)) = \dim(\mathbb{C}^{m\times n}) = mn.$$

Eftersom $\ker(T) = \{0\}$ gäller $\dim(\ker(T)) = 0$, alltså $\dim(\mathrm{Im}(T)) = mn$. Men det innebär att $\mathrm{Im}(T) = \mathbb{C}^{m\times n}$, vilket betyder att T är surjektiv. Med andra ord, för varje $C \in \mathbb{C}^{m\times n}$ finns $X \in \mathbb{C}^{m\times n}$ sådant att $T(X) = C$.

Antag $X_1, X_2 \in \mathbb{C}^{m\times n}$, $X_1 \neq X_2$ och $T(X_1) = T(X_2)$. Det medföljer

$$T(X_1) - T(X_2) = T(X_1 - X_2) = 0$$

eftersom T är linjär. Eftersom $\ker(T) = \{0\}$ impliceras $X_1 - X_2 = 0$, alltså $X_1 = X_2$. Det motsäger det ursprungliga antagandet, alltså måste lösningen vara unik. Så, $\ker(T) = \{0\}$ medför existens och entydighet av lösning till $T(X) = C$.

Det är uppenbart att existens och entydighet av lösning medför $\ker(T) = \{0\}$. Sammanfattningen följer.[1] □

Sats 2.10. *För varje kvadratisk matris A gäller att $\Lambda(A) = \Lambda(A^T)$.*

Bevis. Satsen följer ifall $\det(tI - A)$ och $\det(tI - A^T)$ är samma polynom, eftersom polynomen har egenvärdena av A respektive A^T som rötter. Med andra ord måste det visas att $\det(M) = \det(M^T)$ för varje kvadratisk matris M. Det kan demonstreras genom induktion.

Då M är 1×1 är resultatet självklart. Tag basfallet då M är 2×2. Då gäller

$$\det(M) = M_{11}M_{22} - M_{21}M_{12} = M_{11}M_{22} - M_{12}M_{21} = \det(M^T).$$

Gör induktionsantagandet att $\det(M) = \det(M^T)$ för varje $k \times k$ matris M, för varje $k \leq n - 1$.

Betrakta $n \times n$ matrisen

$$M = \begin{pmatrix} \lambda & b^T \\ a & N \end{pmatrix}$$

[1] Beviset är inspirerat av det i Roger A. Horn och Charles R. Johnson, "Matrix Analysis," 2:a uppl. (Cambridge: Cambridge University Press, 2013), s. 112.

där λ är en skalär och N är $(n-1)\times(n-1)$. Då ser vi att

$$\det(M) = \lambda \det(N) + \sum_{i=1}^{n-1} (-1)^i a_i \det(N_i)$$

där N_i noterar matrisen i M som erhålls då rad $i+1$ och kolumn 1 tags bort.

Observera att

$$M^T = \begin{pmatrix} \lambda & a^T \\ b & N^T \end{pmatrix}.$$

Här får vi

$$\det(M^T) = \lambda \det(N^T) + \sum_{i=1}^{n-1} (-1)^i a_i \det(N_i').$$

Här är N_i' matrisen som erhålls då kolumn $i+1$ och rad 1 tags bort från M^T. Det gäller att $N_i' = N_i^T$ för varje i. Induktionsantagandet ger

$$\det(M^T) = \lambda \det(N) + \sum_{i=1}^{n-1} (-1)^i a_i \det(N_i) = \det(M).$$

Satsen följer. □

Definition 2.11. För $m, n \in \mathbb{Z}^+$, låt $A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}$, och $C \in \mathbb{C}^{m\times n}$ vara givna. En diskret Sylvesterekvation är en matrisekvation av formen

$$AXB - X = C.$$

Definition 2.12. Låt A vara $m \times n$ och B vara $p \times q$ över samma kropp. Kroneckerprodukten $A \otimes B$ definieras som en $(mp) \times (nq)$ matris enligt

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}.$$

Sats 2.13. *Låt $A \in \mathbb{C}^{m\times m}, B \in \mathbb{C}^{n\times n}$, och $C \in \mathbb{C}^{m\times n}$ vara givna. Ekvationen $AXB = C$ är ekvivalent med $(B^T \otimes A)\mathrm{Vec}(X) = \mathrm{Vec}(C)$.*

Bevis. Det är klart att $\mathrm{Vec}(AXB) = \mathrm{Vec}(C)$, så det som behöver härledas är $\mathrm{Vec}(AXB) = (B^T \otimes A)\mathrm{Vec}(X)$. Låt $X = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}$ där x_j är j:e kolumnen

av X. Produkten AXB kan uttryckas som

$$AXB = \begin{pmatrix} Ax_1 & \cdots & Ax_n \end{pmatrix} B.$$

Multiplikation med B på höger ger kolumn k av AXB som

$$(AXB)_k = \sum_{j=1}^{n} b_{jk}(Ax_j)$$

där b_{jk} är element (j,k) av B. Det betyder att

$$\mathrm{Vec}(AXB) = \begin{pmatrix} \sum_{j=1}^{n} b_{j1}(Ax_j) \\ \vdots \\ \sum_{j=1}^{n} b_{jn}(Ax_j) \end{pmatrix}.$$

Betrakta nu $(B^T \otimes A)$, där block (j,k) är given av $b_{kj}A$. Då $(B^T \otimes A)$ multipliceras med

$$\mathrm{Vec}(X) = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$$

ges produkten av

$$(B^T \otimes A)\mathrm{Vec}(X) = \begin{pmatrix} \sum_{j=1}^{n} b_{j1}(Ax_j) \\ \vdots \\ \sum_{j=1}^{n} b_{jn}(Ax_j) \end{pmatrix}.$$

Satsen följer. □

Lemma 2.14. *Om* $\Lambda(A) = \{\lambda_i\}_{i=1}^{m}$ *och* $\Lambda(B) = \{\mu_j\}_{j=1}^{n}$, *då är* $\Lambda(A \otimes B) = \{\lambda_i \mu_j\}_{i,j=1}^{m,n}$.

Bevis. Låt $\lambda \in \Lambda(A)$ och $\mu \in \Lambda(B)$, med egenvektorer x respektive y. Då erhålls följande:

$$\begin{aligned} (A \otimes B)(x \otimes y) &\overset{\text{egenskap av } \otimes}{=} (Ax) \otimes (By) \\ &= (\lambda x) \otimes (\mu y) \\ &= (\lambda\mu)(x \otimes y). \end{aligned}$$

□

Sats 2.15. *Givet $A \in \mathbb{C}^{m\times m}$ och $B \in \mathbb{C}^{n\times n}$ gäller*

$$\rho(A \otimes B) = \rho(A)\rho(B).$$

Bevis. Låt $\Lambda(A) = \{\lambda_i\}_{i=1}^m$ och $\Lambda(B) = \{\mu_j\}_{j=1}^n$. Utifrån lemma 2.14 har vi att $\Lambda(A \otimes B) = \{\lambda_i\mu_j : \lambda_i \in \Lambda(A), \mu_j \in \Lambda(B)\}$. Det medför

$$\rho(A \otimes B) = \max_{i,j} |\lambda_i\mu_j| = \max_i |\lambda_i| \max_j |\mu_j| = \rho(A)\rho(B).$$

□

Sats 2.16. *Låt $A \in \mathbb{C}^{m\times m}$ och $B \in \mathbb{C}^{n\times n}$ vara givna. För varje $C \in \mathbb{C}^{m\times n}$, har ekvationen $AXB - X = C$ en unik lösning $X \in \mathbb{C}^{m\times n}$ om och endast om $\lambda_A\lambda_B \neq 1$ för $\lambda_A \in \Lambda(A), \lambda_B \in \Lambda(B)$.*

Bevis. Låt avbildningen $S : \mathbb{C}^{m\times n} \to \mathbb{C}^{m\times n}$ definieras enligt $S(X) = AXB - X$. Avbildningen är klart linjär. Antag vidare att $\lambda_A \in \Lambda(A)$ och $\lambda_B \in \Lambda(B)$. Då finns nollskilda $u \in \mathbb{R}^m$ och $v \in \mathbb{C}^n$ sådana att

$$Au = \lambda u,$$

$$v^T B = \lambda v^T.$$

Observera, för $0 \neq uv^T \in \mathbb{R}^{m\times n}$:

$$\begin{aligned} S(uv^T) &= A(uv^T)B - uv^T \\ &= (Au)(v^T B) - uv^T \\ &= (\lambda_A u)(\lambda_B v^T) - uv^T \\ &= (\lambda_A\lambda_B - 1)uv^T. \end{aligned}$$

Därmed är egenvärdena av S av formen $\lambda_A\lambda_B - 1$. Om det finns λ_A, λ_B sådana att $\lambda_A\lambda_B = 1$ finns det $X \neq 0$ sådant att $S(X) = 0$. Men $S(0) = 0$, alltså gäller inte att det för varje C finns en unik lösning till $S(X) = C$. Å andra sidan, om $\lambda_A\lambda_B \neq 1$ för alla λ_A, λ_B har S endast nollskilda egenvärden, alltså är S inverterbar, och $T(X) = C$ har en unik lösning X för varje givet C. □

Lemma 2.17. *Om $T \in \mathbb{C}^{n\times n}$ är uppdelad som*

$$T = \begin{pmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{pmatrix} \begin{matrix} p \\ q \end{matrix}$$
$$\begin{matrix} p & q \end{matrix}$$

så gäller $\Lambda(T) = \Lambda(T_{11}) \cup \Lambda(T_{22})$.

Bevis. [GVL13, s. 350] Antag att

$$Tx = \begin{pmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \lambda \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

där $x_1 \in \mathbb{C}^p$ och $x_2 \in \mathbb{C}^q$. Om $x_2 \neq 0$ innebär det $T_{22}x_2 = \lambda x_2$, så $\lambda \in \Lambda(T_{22})$. Om däremot $x_2 = 0$ så gäller $T_{11}x_1 = \lambda x_1$, alltså $\lambda \in \Lambda(T_{11})$. Det medföljer $\Lambda(T) \subset \Lambda(T_{11}) \cup \Lambda(T_{22})$. Då $\Lambda(T)$ och $\Lambda(T_{11}) \cup \Lambda(T_{22})$ har samma kardinalitet måste de två mängderna vara identiska. □

Lemma 2.18. *Om $A \in \mathbb{C}^{n\times n}, B \in \mathbb{C}^{p\times p}$, och $X \in \mathbb{C}^{n\times p}$ uppfyller*

$$AX = XB, \quad rank(X) = p, \tag{1}$$

finns det en unitär matris $Q \in \mathbb{C}^{n\times n}$ sådan att

$$Q^H AQ = T = \begin{pmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{pmatrix} \begin{matrix} p \\ n-p \end{matrix}$$
$$\begin{matrix} p & n-p \end{matrix}$$

och $\Lambda(T_{11}) = \Lambda(A) \cap \Lambda(B)$.

Bevis. [GVL13, s. 350] Låt

$$X = Q \begin{pmatrix} R_1 \\ 0 \end{pmatrix}, \quad Q \in \mathbb{C}^{n\times n}, R_1 \in \mathbb{C}^{p\times p}$$

vara en QR faktorisering av X. Låt därefter

$$Q^H AQ = \begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix} \begin{matrix} p \\ n-p \end{matrix}.$$
$$\begin{matrix} p & n-p \end{matrix}$$

Genom att substituera detta i (1) och omordna termerna får man

$$AQ \begin{pmatrix} R_1 \\ 0 \end{pmatrix} = Q \begin{pmatrix} R_1 \\ 0 \end{pmatrix} B,$$

där vänstermultiplikation med Q^H ger

$$\begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix} \begin{pmatrix} R_1 \\ 0 \end{pmatrix} = \begin{pmatrix} R_1 \\ 0 \end{pmatrix} B.$$

Genom att R_1 är ickesingulär samt ekvationerna $T_{11}R_1 = R_1B$ och $T_{21}R_1 = 0$ får man $T_{21} = 0$ och $\Lambda(T_{11}) = \Lambda(B)$. Lemmat följer eftersom vi från Lemma 1 har $\Lambda(A) = \Lambda(T) = \Lambda(T_{11}) \cup \Lambda(T_{22})$. □

Sats 2.19. *Om $A \in \mathbb{C}^{n\times n}$ finns det en unitär matris $Q \in \mathbb{C}^{n\times n}$ sådan att*

$$Q^H AQ = D + N$$

där $D = diag(\lambda_1, \cdots, \lambda_n)$ och $N \in \mathbb{C}^{n\times n}$ är strängt övretriangulär. För övrigt can Q väljas så att egenvärdena λ_i uppkommer i vilken ordning som helst längs diagonalen.

Bevis. [GVL13, s. 351] Satsen håller klart om $n = 1$. Antag att den håller för alla matriser av dimension $n - 1$ eller mindre. Om $Ax = \lambda x$ och $x \neq 0$ så medförs det enligt Lemma 2.18 att det finns en unitär U som uppfyller

$$U^H AU = \begin{pmatrix} \lambda & \omega^H \\ 0 & C \end{pmatrix} \begin{matrix} 1 \\ n-1 \end{matrix}$$
$$\begin{matrix} \quad 1 & n-1 \end{matrix}$$

för något $\omega \in \mathbb{C}^{n-1}$ och $C \in \mathbb{C}^{(n-1)\times(n-1)}$. Genom induktion finns det en unitär $\tilde{U}$ så att $\tilde{U}^H C\tilde{U}$ är övre triangulär. Därför medförs att om $Q = U \cdot diag(1, \tilde{U})$, är $Q^H AQ$ övre triangulär. □

Sats 2.20. *Låt $A \in \mathbb{R}^{n\times n}$. Då har A en reell Schur-dekomposition. Med andra ord finns det en ortogonal $Q \in \mathbb{R}^{n\times n}$ sådan att*

$$Q^T AQ = \begin{pmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ 0 & R_{22} & \cdots & R_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & R_{mm} \end{pmatrix}$$

där varje R_{ii} är antingen 1×1 eller 2×2 vars egenvärden är komplexa konjugater.

Bevis. [GVL13, s. 377] De komplexa egenvärdena av A uppkommer i konjugatpar eftersom det karaktäristiska polynomet har reella koefficienter. Låt k vara antalet konjugatpar i $\Lambda(A)$. Vi bevisar satsen genom induktion på k. Observera att Lemma 2.18 och Sats 2.19 har reella motsvarigheter. Därför gäller satsen om $k = 0$. Antag nu att $k \geq 1$. Om $\lambda = \gamma + i\mu \in \Lambda(A)$ och $\mu \neq 0$, finns det vektorer y och $z \neq 0$ i $\mathbb{R}^n$ sådana att $A(y + iz) = (\gamma + i\mu)(y + iz)$, med andra ord

$$A \begin{pmatrix} y & z \end{pmatrix} = \begin{pmatrix} y & z \end{pmatrix} \begin{pmatrix} \gamma & \mu \\ -\mu & \gamma \end{pmatrix}.$$

Antagandet $\mu \neq 0$ implicerar att y och z spänner ett tvådimensionellt, reellt invariant delrum W av A. Det betyder att $W \subseteq \mathbb{R}^n$ och $A(W) \subseteq W$. Det följer från Lemma 2.18 att det finns en ortogonal $U \in \mathbb{R}^{n \times n}$ så att

$$U^T A U = \begin{matrix} \begin{pmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{pmatrix} \begin{matrix} 2 \\ n-2 \end{matrix} \\ \begin{matrix} 2 & n-2 \end{matrix} \end{matrix}$$

där $\Lambda(T_{11}) = \{\lambda, \bar{\lambda}\}$. Genom induktion finns det en ortogonal $\tilde{U}$ så att $\tilde{U}^T T_{22} \tilde{U}$ har den nödvändiga strukturen. Satsen följer genom att sätta $Q = U \cdot diag(I_2, \tilde{U})$. □

Sats 2.21. *Låt M vara en kvadratisk matris. Delsummorna*

$$S_k = \sum_{i=0}^{k} M^i$$

konvergerar då $k \to \infty$ om och endast om $\rho(M) < 1$.

Bevis. Antag att $S_k = \sum_{i=0}^{k} M^i$ konvergerar. Tag $\lambda \in \Lambda(M)$ med korresponderande egenvektor v och observera

$$S_k v = \sum_{i=0}^{\infty} M^i v = \sum_{i=0}^{\infty} \lambda^i v.$$

Eftersom S_k konvergerar, så konvergerar $S_k v$ i vektorrummet. Då behöver $\sum_{i=0}^{\infty} \lambda^i$ konvergera, vilket sker om och endast om $|\lambda| < 1$. Då det här gäller för varje $\lambda \in \Lambda(M)$ innebär det att $\max_{\lambda \in \Lambda(M)} |\lambda| = \rho(M) < 1$.

Antag nu istället att $\rho(M) < 1$. Gelfrands formel säger att

$$\rho(M) = \lim_{i \to \infty} ||M^i||^{\frac{1}{i}}$$

för varje matrisnorm $||.||$. Då $\rho(M) < 1$ finns det ett konstant r sådant att $\rho(M) < r < 1$. Det medföljer $\lim_{i \to \infty} ||M^i||^{\frac{1}{i}} < r$, med andra ord finns det ett K sådant att $||M^i|| < r^i$ för varje $i > K$. Betrakta nu serien

$$\sum_{i=0}^{\infty} ||M^i|| = \sum_{i=0}^{K} ||M^i|| + \sum_{i=K+1}^{\infty} ||M^i||.$$

Delsumman $\sum_{i=0}^{K} ||M^i||$ existerar då det involverar ändligt många termer, samtidigt som resten uppfyller

$$\sum_{i=K+1}^{\infty} ||M^i|| < \sum_{i=K+1}^{\infty} r^i$$

vilket konvergerar eftersom $|r| < 1$. Således konvergerar $\sum_{i=0}^{\infty} ||M^i||$ för varje norm, vilket medför att $\sum_{i=0}^{\infty} M^i$ konvergerar. □

3 Sylvesterekvationer och användningsområden

Arbetet kommer att behandla två sorters Sylvesterekvationer: kontinuerliga och diskreta. De är två sorters matrisekvationer. Se definitioner 2.3 (s. 15) och 2.11 (s. 17) för hur dessa ekvationer definieras. Några vanliga användningsområden för Sylvesterekvationer är kontrollteori och dynamiska system, där namnen "kontinuerliga" och "diskreta" säger huruvida ekvationen dyker upp i ett kontinuerligt eller diskret system. Ovannämnda ekvationer uppkommer också inom maskininlärningssammanhang, bland annat viktinitialisering [DBP21], semi-övervakad multi-etikettering [CSWZ08], och bildbehandling [HNO06]. Följande två delavsnitt täcker två exempel i mer detalj, en för kontinuerliga och en annan för diskreta ekvationer.

3.1 Datadriven viktinitialisering

Som beskrivet i avsnitt 1.1 beror ett neuronnätverks optimering på dess parametrar, så kallade vikter W och biasvektorer b. I det avsnittet täcktes inte bara hur ett lager påverkar det nästa, men också hur nätverkets parametrar uppdateras för att minska felet. Man kan tänka sig en ännu mer optimerad lösning, där vikterna från början inte genereras slumpmässigt enligt en fördelning utan är försiktigt valda utifrån träningsdatan för att producera mer precisa resultat.

Syftet är att utifrån träningsdata $D = \{(x_i, y_i)\}_{i=1}^{N}$ använda en liten del $\tilde{D} \subset D$ för att konstruera en viktmatris vilken minimerar kodnings- och avkodningsförlusten [DBP21, s. 2]. Indata X respektive vikter W definieras först för ett nätverk med konvolutionella lager, vilka har formen av 4D tensorer snarare än 2D matriser. Däremot förklarar de att deras metod kan översättas till fullkopplade nätverk ("fully connected" på engelska), alltså det typ av nätverk som togs upp i avsnitt 1.1.

Låt indata respektive vikter representeras av $X \in \mathbb{R}^{d_i \times n}$ och $W \in \mathbb{R}^{d_0 \times d_i}$, där d_i är dimensionen av indatalagret, d_0 är dimensionen av utdatalagret och n är antalet exempel av indata. Det menas på att en bra viktmatris bör koda X till en latent kod $S \in \mathbb{R}^{d_0 \times n}$, vilken sedan kan avkodas tillbaka till X. Här låts W^T vara avkodningsmatrisen. Teoretiskt hade man kunnat låta avkodaren vara en separat matris V, men då hade optimeringsproblemet blivit betydligt mer komplicerat. Att använda W^T är ett medvetet designval som reducerar antalet fria variabler och möjliggör en elegant lösning. Latent kod kan uttrycka komplicerad indata på ett meningsfullt sätt, vilket stärker modellens förmåga att förstå och manipulera det [Ber25].

Ovan beskrivning av problemet kan översättas till ett regulariseringsproblem [DBP21, s. 3]: att hitta matrisen W vilken minimerar

$$f(W) := ||X - W^T S||_F^2 + \lambda ||WX - S||_F^2 \tag{2}$$

där $||\cdot||_F$ är Frobenius-normen för matriser. Frobenius-normen för en matris A är definierad som

$$||A||_F = \sqrt{\sum_{i,j} |a_{ij}|^2} = \sqrt{\text{trace}(A^T A)}.$$

I (2) mäter $||X - W^T S||_F^2$ avkodningsförlusten, $||WX - S||_F^2$ mäter kodningsförlusten och $\lambda > 0$ är en skalär vilken väger kodningsförlusten.

För att lösa optimeringsproblemet skrivs högerledet HL i (2) om enligt

$$HL = \text{trace}((X - W^T S)^T (X - W^T S)) + \lambda \text{trace}((WX - S)^T (WX - S))$$

eftersom $||A||_F^2 = \text{trace}(A^T A)$. Högerledet kan därefter expanderas som

$$\begin{aligned}
HL &= \text{trace}((S^T W - X^T)(X - W^T S)) + \lambda \text{trace}((S^T - X^T W^T)(WX - S)) \\
&= \text{trace}(S^T WX - S^T W W^T S - X^T X + X^T W^T S) \\
&+ \lambda \text{trace}(S^T WX - S^T S - X^T W^T WX + X^T W^T S) \\
&= (1 + \lambda)(\text{trace}(S^T WX) + \text{trace}(X^T W^T S)) - \text{trace}(S^T W W^T S) \\
&- \lambda \text{trace}(X^T W^T WX) + k \\
&= (1 + \lambda)(\text{trace}(S^T WX) + \text{trace}(S^T WX)^T) - \text{trace}(S^T W W^T S) \\
&- \lambda \text{trace}(X^T W^T WX) + k \\
&= 2(1 + \lambda)\text{trace}(S^T WX) - \text{trace}(S^T W W^T S) - \lambda \text{trace}(X^T W^T WX) + k
\end{aligned}$$

där k är en konstant med avseende på W. För att hitta minimum beräknas gradienten av f med avseende på W. Termen $\text{trace}(S^T WX)$ kan skrivas om med indexnotation

$$\text{trace}(S^T WX) = \sum_i [S^T WX]_i = \sum_i \sum_j S_{ij}^T [WX]_{ji} = \sum_i \sum_j \sum_k S_{ij}^T W_{jk} X_{ki}.$$

Således är den partiella derivatan med avseende på W_{jk} lika med

$$\frac{\partial \text{trace}(S^T WX)}{\partial W_{jk}} = \sum_i S_{ij}^T X_{ki} = [XS^T]_{kj}.$$

Det implicerar

$$\nabla_W \text{trace}(S^T W X) = (XS^T)^T = SX^T.$$

Utifrån sats 2.1 kan det härledas att $\text{trace}(S^T W W^T S) = \text{trace}(W^T S S^T W)$ samt $\text{trace}(X^T W^T W X) = \text{trace}(W X X^T W^T)$, se bevis (s. 15).

Betrakta funktionen $\text{trace}(W^T M W)$ för en symmetrisk matris M. Med indexnotation ges $\text{trace}(W^T M W)$ av

$$\text{trace}(W^T M W) = \sum_i \sum_j \sum_k W_{ij}^T M_{jk} W_{ki} = \sum_i \sum_j \sum_k W_{ji} M_{jk} W_{ki}.$$

Här vill man beräkna $\frac{\partial \text{trace}(W^T M W)}{\partial W_{pq}}$. De termer i $\text{trace}(W^T M W)$ som innehåller W_{pq} är då $j = p$ och $i = q$, eller $k = p$ och $i = q$. I första fallet ser termerna ut som $W_{pq} M_{pk} W_{kq}$, och i derivatan $M_{pk} W_{kq}$ där de summeras över k. I andra fallet ser termerna i derivatan ut som $W_{jp} M_{jp}$ vilka summeras över j. Det vill säga

$$\frac{\partial \text{trace}(W^T M W)}{\partial W_{pq}} = \sum_k M_{pk} W_{kq} + \sum_j W_{jq} M_{jp} = [MW]_{pq} + [M^T W]_{pq}$$

Här får man

$$\nabla_W \text{trace}(W^T M W) = MW + M^T W = 2MW$$

då M är symmetrisk.

På mycket liknande sätt kan man härleda att

$$\nabla_W \text{trace}(W M W^T) = 2WM.$$

Eftersom både SS^T och XX^T är symmetriska gäller

$$\nabla_W \text{trace}(W^T S S^T W) = 2SS^T W,$$

$$\nabla_W \text{trace}(W X X^T W^T) = 2WXX^T.$$

Med allt det sagt får man att gradienten av $f(W)$ är

$$\nabla_W f(W) = 2(1+\lambda)SX^T - 2SS^T W - 2\lambda W X X^T.$$

Gradienten sätts lika med 0, och efter förenkling fås ekvationen

$$SS^TW + W\lambda XX^T = (1+\lambda)SX^T.$$

Det har alltså visats att sökning efter en optimal vikt i ett nätverk reduceras till att lösa en kontinuerlig Sylvesterekvation.

Ett specialfall i (2) är då $\lambda = 0$. Då förenklas Sylvesterekvationen till $SS^TW = SX^T$, villket innebär att W är en minstakvadratlösning till $S^TW \approx X^T$, eller ekvivalent $W^TS \approx X$. Regulariseringsparametern λ är ofta mycket liten, villket medför att lösningen även för $\lambda > 0$ resulterar i en matris W där $W^TS \approx X$.

3.2 Bildbehandling och avskärpa

I avsnitt 1.1 användes ett exempel där en svartvit bild på symbolen "3" representerades av en vektor. En bild kan däremot representeras på flera olika sätt, bland annat som en matris. Givet en ursprungsbild X kan en suddig bild B skapas genom ett bilinjärt förhållande

$$B = A_cXA_r^T$$

där A_c är en kolumnsuddande matris och A_r är en radsuddande matris [HNO06, s. 4]. Namnen kommer ifrån hur matriserna agerar på X: A_c multipliceras till varje kolumn av X, och A_r^T till varje rad.

En naiv metod skulle vara att lösa för X direkt genom $X = A_c^{-1}B(A_r^T)^{-1}$ [HNO06, ss. 5–6]. Det här leder dock till ett dåligt resultat på grund av att den suddiga bilden innehåller så kallat additivt Gaussiskt brus. Modellen för den suddiga bilden ges egentligen av $B = A_cXA_r^T + E$, där $E \sim \mathcal{N}(0, \sigma^2)$ är brusmatrisen. Den naiva metoden ger alltså inte X, utan $X + A_c^{-1}E(A_r^T)^{-1}$. Det som då oftast händer är att direkt inversion amplifierar bruskomponenterna och ger en mycket brusig återställd bild. Därför anses det inte som en lämplig metod för att lösa problemet.

En återställningsmetod [HNO06, s. 72] som fungerar bättre involverar att lösa

$$\min_x ||b - Ax||_2^2 + \alpha^2||x||_2^2$$

där $x = \text{Vec}(X)$ och $b - Ax = \text{Vec}(B - A_cXA_r^T)$. Om $C \in \mathbb{C}^{m\times n}$ så har vi

$$\text{Vec}(C) = (C_{11}, \cdots C_{m1}, C_{12}, \cdots, C_{m2}, \cdots, C_{1n}, \cdots, C_{mn}),$$

vilket betyder

$$||\text{Vec}(C)||_2 = \sqrt{\sum_{i,j} |C_{ij}|^2} = ||C||_F.$$

Det innebär att regulariseringsproblemet är ekvivalent med

$$\min_X ||B - A_c X A_r^T||_F^2 + \alpha^2 ||X||_F^2.$$

Genom att uttrycka Frobenius-normen enligt spårfunktionen fås

$$\begin{aligned}
&\min_X ||B - A_c X A_r^T||_F^2 + \alpha^2 ||X||_F^2 \\
&= \min_X \text{trace}((B - A_c X A_r^T)^T (B - A_c X A_r^T)) + \alpha^2 \text{trace}(X^T X) \\
&= \min_X \text{trace}((B^T - A_r X^T A_c^T)(B - A_c X A_r^T)) + \alpha^2 \text{trace}(X^T X) \\
&= \min_X -\text{trace}(B^T A_c X A_r^T) - \text{trace}(A_r X^T A_c^T B) + \text{trace}(A_r X^T A_c^T A_c X A_r^T) \\
&+ \alpha^2 \text{trace}(X^T X) + k \\
&= \min_X -\text{trace}(B^T A_c X A_r^T) - \text{trace}((B^T A_c X A_r^T)^T) + \text{trace}(A_r X^T A_c^T A_c X A_r^T) \\
&+ \alpha^2 \text{trace}(X^T X) + k \\
&= \min_X -2\text{trace}(B^T A_c X A_r^T) + \text{trace}(A_r X^T A_c^T A_c X A_r^T) + \alpha^2 \text{trace}(X^T X) + k \\
&= \min_X -2\text{trace}((A_c^T B A_r)^T X) + \text{trace}((A_c X A_r^T)^T (A_c X A_r^T)) + \alpha^2 \text{trace}(X^T X) + k
\end{aligned}$$

Problemet löses genom att beräkna gradienten och sätta den lika med 0. Gör man det får man ekvationen

$$A_c^T A_c X A_r^T A_r + \alpha^2 X = A_c^T B A_r,$$

vilken är en typ av diskret Sylvesterekvation. Man kan alltså se hur dessa ekvationer kan förekomma i modeller som är avsedda för bildbehandling.

4 Att lösa Sylvesterekvationer

Det finns särskilda krav som en Sylvesterekvation behöver uppfylla för att ha en unik lösning, se satser 2.9 (s. 17) och 2.16 (s. 19). I verkliga scenarion där Sylvesterekvationer uppkommer i sammanhang av maskininlärning, handlar det om mycket stora matriser som inte går att lösa för hand. Följande delavsnitt handlar om att ta reda på hur dessa matrisekvationer kan lösas.

4.1 Kontinuerliga Sylvesterekvationer

4.1.1 Bartels-Stewart algoritmen

En metod som ofta nämns är Bartels-Stewart algoritmen [SM19, ss. 1–3]. Den går ut på att ta en allmän ekvation med täta matriser och reducera den till en mycket förenklad version med glesare matriser. .

Givet $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{n\times n}$ och $C \in \mathbb{R}^{m\times n}$ där $\Lambda(A) \cap \Lambda(-B) = \emptyset$ (sats och bevis på ss. 17–18), betrakta ekvationen

$$AX + XB = C. \tag{3}$$

Då A och B är kvadratiska matriser kan reell Schur-dekomposition utföras, se sats 2.20 med bevis 2 på s. 22. Det vill säga

$$A = U\tilde{A}U^T,$$
$$B = V\tilde{B}V^T$$

där U, V är ortogonala och $\tilde{A}, \tilde{B}$ är övre kvasitriangulära. Att vara övre kvasitriangulär innebär att vara blockvis övre triangulär, där ett block på diagonalen kan vara 1×1 eller 2×2. Med användning av dekompositionen i (3) fås

$$U\tilde{A}U^TX + XV\tilde{B}V^T = C.$$

Båda sidor multipliceras till vänster med U^T och till höger med V för att få

$$\tilde{A}(U^TXV) + (U^TXV)\tilde{B} = U^TCV.$$

Låt därefter $Y = U^T XV$ samt $\tilde{C} = U^T CV$, och då har (3) reducerats till

$$\tilde{A}Y + Y\tilde{B} = \tilde{C}. \tag{4}$$

Då $\tilde{A}$ och $\tilde{B}$ är övre kvasitriangulära är matriserna i (4) glesare jämfört med dem i (3). Det är tydligt att $\tilde{A} \in \mathbb{R}^{m\times m}$, $\tilde{B} \in \mathbb{R}^{n\times n}$, och $\tilde{C}, Y \in \mathbb{R}^{m\times n}$.

Låt $\tilde{A}, \tilde{B}$ uttryckas som övre triangulära blockmatriser. Det vill säga

$$\tilde{A} = \left[\tilde{A}_{ij}\right]_{i,j=1}^{p}$$

och

$$\tilde{B} = \left[\tilde{B}_{ij}\right]_{i,j=1}^{q}$$

där diagonalblocken är skalärer eller 2×2, och $\tilde{A}_{ij} = 0$ samt $\tilde{B}_{ij} = 0$ om $i > j$. Dessutom gäller att $\tilde{A}_{ij}$ har lika många rader som $\tilde{A}_{ii}$ och lika många kolumner som $\tilde{A}_{jj}$, det för att $\tilde{A}_{ij}$ befinns på samma blockrad som $\tilde{A}_{ii}$ och samma blockkolumn som $\tilde{A}_{jj}$. Detsamma för blocken i $\tilde{B}$. På så sätt gäller $\tilde{C} = \left[\tilde{C}_{ij}\right]_{i=1,\,j=1}^{p,\,q}$ och $Y = [Y_{ij}]_{i=1,\,j=1}^{p,\,q}$ som blockmatriser. Inom $\tilde{A}Y$ och $Y\tilde{B}$ multipliceras Y_{ij} till vänster med $\tilde{A}_{ki}$ för $k = 1, \ldots, p$ och till höger med $\tilde{B}_{jl}$ för $l = 1, \ldots, q$. Således måste Y_{ij} ha samma antal rader som $\tilde{A}_{ki}$ har kolumner, vilket är samma som i $\tilde{A}_{ii}$, och samma antal kolumner som $\tilde{B}_{jl}$ har rader, vilket är samma som i $\tilde{B}_{jj}$. Slutligen behöver $\tilde{C}_{ij}$ vara av samma storlek som Y_{ij}. Med detta i åtanke kommer metoden som följer att ge mening.

Varje blockelement i vänsterledet i (4) måste vara lika med motsvarande blockelement i högerled. Man kan se att

$$(\tilde{A}Y + Y\tilde{B})_{ij} = \sum_{k=1}^{p} \tilde{A}_{ik}Y_{kj} + \sum_{l=1}^{q} Y_{il}\tilde{B}_{lj} = \sum_{k=i}^{p} \tilde{A}_{ik}Y_{kj} + \sum_{l=1}^{j} Y_{il}\tilde{B}_{lj}$$

eftersom $\tilde{A}_{ik} = 0$ då $k < i$ och $\tilde{B}_{lj} = 0$ då $l > j$. Man får

$$\sum_{k=i}^{p} \tilde{A}_{ik}Y_{kj} + \sum_{l=1}^{j} Y_{il}\tilde{B}_{lj} = \tilde{C}_{ij}.$$

Isolera alla block vilka innehåller Y_{ij} på en sida för att få

$$\tilde{A}_{ii}Y_{ij} + Y_{ij}\tilde{B}_{jj} = \tilde{C}_{ij} - \left(\sum_{k=i+1}^{p} \tilde{A}_{ik}Y_{kj} + \sum_{l=1}^{j-1} Y_{il}\tilde{B}_{lj} \right).$$

Man får alltså en samling av pq stycken Sylvesterekvationer som är mycket små och enkla att lösa, men som måste lösas i rätt ordning. Man börjar med $i = p$, $j = 1$ då båda summorna i högerledet blir triviala. Sedan löser man för $j = 2$, därefter $j = 3$ och så vidare ända till $j = q$. Efter att ha löst för rad p gör man på samma sätt för rad $p-1$, sedan $p-2$ och så vidare till rad 1. Till slut har Y beräknats, och då utför man en sista substitution tillbaka till $X = UYV^T$, och har då löst (3).

4.1.2 ADI-metoden

Alternating-Directional-Implicit (ADI) metoden är en iterativ metod för att numeriskt lösa Sylvesterekvationer. Med stora och glesa system kan iterativa metoder ha låg beräkningskostnad per iteration. Det finns dessutom mycket frihet då man kan bestämma hur många iterationer som skall utföras, samt eventuella initialvärden och parametrar, vilka påverkar lösningens konvergens. Först appliceras metoden på Lyapunovekvationer [Pen00, ss. 1401–1402]. En Lyapunovekvation är en typ av Sylvesterekvation och ser ut på följande vis:

$$A^T X + XA = -BB^T. \tag{5}$$

Något annat som skiljer (5) från en standard Sylvesterekvation är att varje inblandad matris är kvadratisk och av samma storlek. Matrisen A antags dessutom vara stabil, vilket betyder att systemet $\dot{x}(t) = Ax(t)$ uppfyller $\lim_{t\to\infty} x(t) = 0$ för alla initialvärden $x(0)$.

En konsekvens av att A är stabil är att alla dess egenvärden är element av $\mathbb{C}_-$, de komplexa talen med negativ reell del. Antag att $\Re(\lambda) \geq 0$ för varje egenvärde λ, och låt $v \neq 0$ vara en tillhörande egenvektor. En partikulärlösning till $\dot{x}(t) = Ax(t)$ är

$x(t) = e^{At}x(0)$. Välj initialvärdet $x(0) = v$ och observera

$$\begin{aligned} x(t) &= e^{At}v \\ &= \sum_{k=0}^{\infty} \frac{t^k}{k!} A^k v \\ &= \sum_{k=0}^{\infty} \frac{t^k}{k!} \lambda^k v \\ &= e^{\lambda t} v. \end{aligned}$$

Om $\lambda > 0$ växer $|x(t)|$ exponentiellt då t växer linjärt. Om $\lambda = 0$ är $x(t)$ konstant. I båda fallen avtar lösningen inte mot 0, vilket strider mot antagandet av stabilitet. Därför måste egenvärdena av A vara i $\mathbb{C}_-$.

Det innebär att $-A^T$ har sina egenvärden i $\mathbb{C}_+$, eftersom A och A^T har samma egenvärden (se sats 2.10 på s. 18). Därför har A och $-A^T$ disjunkta egenvärden och det försäkrar att (5) har en unik lösning.

Tanken med metoden är att använda så kallade skiftparametrar $\{p_1, p_2, \cdots, p_k\} \subset \mathbb{C}_-$ för att "skifta" A och A^T i ekvationen. Den initiella matrisen $X_0 = 0$ och iterationen utförs i två steg enligt följande:

$$\begin{aligned} (A^T + p_i I)X_{i-\frac{1}{2}} + X_{i-1}(A - p_i I) &= -BB^T, \\ (A^T - p_i I)X_{i-\frac{1}{2}} + X_i(A + p_i I) &= -BB^T. \end{aligned}$$

Tvåstegsiterationen kan omskrivas som en enkelstegsiteration genom att först lösa för $X_{i-\frac{1}{2}}$ i första steget. Antaget att $(A + p_i I)$ är ickesingulär erhålls

$$X_{i-\frac{1}{2}} = -(A^T + p_i I)^{-1}BB^T - (A^T + p_i I)^{-1}X_{i-1}(A - p_i I).$$

Därefter löses för X_i enligt

$$X_i = -BB^T(A + p_i I)^{-1} - (A^T - p_i I)X_{i-\frac{1}{2}}(A + p_i I)^{-1}.$$

Substituera uttrycket för $X_{i-\frac{1}{2}}$ för att få

$$\begin{aligned}
X_i &= -BB^T(A+p_iI)^{-1} \\
&\quad - (A^T - p_iI)\left(-(A^T+p_iI)^{-1}BB^T - (A^T+p_iI)^{-1}X_{i-1}(A-p_iI)\right)(A+p_iI)^{-1} \\
&= (A^T - p_iI)(A^T+p_iI)^{-1}X_{i-1}(A-p_iI)(A+p_iI)^{-1} \\
&\quad + (A^T - p_iI)(A^T+p_iI)^{-1}BB^T(A+p_iI)^{-1} - BB^T(A+p_iI)^{-1}.
\end{aligned}$$

Tvåstegsiterationen är alltså ekvivalent med

$$X_i = U_iX_{i-1}V_i + W_i$$

där

$$\begin{aligned}
U_i &= (A^T - p_iI)(A^T+p_iI)^{-1}, \\
V_i &= (A - p_iI)(A+p_iI)^{-1}, \\
W_i &= (A^T - p_iI)(A^T+p_iI)^{-1}BB^T(A+p_iI)^{-1} - BB^T(A+p_iI)^{-1}.
\end{aligned}$$

Här kan ett konvergensvillkor härledas. Spektralradien $\rho(M)$ för en matris M är det största absolutbeloppet av dess egenvärden. Hur egenvärdena av U_i och V_i ser ut behöver härledas.

Låt λ vara ett egenvärde av A med egenvektor v, och antag att $A+pI$ är inverterbar. Då gäller $(A+pI)v = (\lambda+p)v$. Man får därefter

$$v = (A+pI)^{-1}(A+pI)v = (\lambda+p)(A+pI)^{-1}v$$

vilket säger

$$(A+pI)^{-1}v = \frac{1}{\lambda+p}v.$$

Således gäller

$$(A-pI)(A+pI)^{-1}v = \frac{\lambda-p}{\lambda+p}v.$$

Då A och A^T har samma egenvärden (sats 2.10 s. 18) erhålls på liknande sätt

$$(A^T-pI)(A^T+pI)^{-1}v = \frac{\lambda-p}{\lambda+p}v.$$

Således innebär det att

$$\rho(U_i) = \max_{\lambda \in \Lambda(A)} \left| \frac{\lambda - p_i}{\lambda + p_i} \right|,$$
$$\rho(V_i) = \max_{\lambda \in \Lambda(A)} \left| \frac{\lambda - p_i}{\lambda + p_i} \right|.$$

där $\Lambda(A)$ är mängden egenvärden till A (definition 2.4).

Låt $E_i := X - X_i$, där $E_0 = X$ och X är den unika lösningen till (5) och som dessutom uppfyller $X = U_i X V_i + W_i$ för alla i. Följande fås:

$$\begin{aligned} E_i &= X - X_i \\ &= U_i X V_i + W_i - U_i X_{i-1} V_i - W_i \\ &= U_i (X - X_{i-1}) V_i \\ &= U_i E_{i-1} V_i. \end{aligned}$$

För den repeterade iterationen gäller $E_k = P_k X Q_k$, där $P_k = U_k U_{k-1} \cdots U_1$ och $Q_k = V_1 V_2 \cdots V_k$. För att analysera konvergensen mer precist vektoriserar vi felet enligt

$$\text{Vec}(E_k) = (Q_k^T \otimes P_k)\text{Vec}(X).$$

Se definition 2.12 på s. 19 för hur operatorn $\otimes$ är definierad. För att se varför ovan likhet stämmer, hänvisa till sats 2.13 på s. 19. Spektralradien för Kroneckerprodukten är

$$\rho(Q_k^T \otimes P_k) = \rho(Q_k^T)\rho(P_k) = \rho(Q_k)\rho(P_k)$$

enligt sats 2.15 på s. 20. Likheten $\rho(Q_k^T) = \rho(Q_k)$ gäller då $\Lambda(Q_k^T) = \Lambda(Q_k)$.

Eftersom spektralradien är submultiplikativ erhålls

$$\rho(P_k) \leq \prod_{i=1}^{k} \rho(U_i), \quad \rho(Q_k) \leq \prod_{i=1}^{k} \rho(V_i) = \prod_{i=1}^{k} \rho(U_i).$$

Det ger

$$\rho(Q_k^T \otimes P_k) = \rho(P_k)\rho(Q_k) \leq \left(\prod_{i=1}^{k} \rho(U_i) \right)^2.$$

Om produkten $\prod_{i=1}^{k} \rho(U_i)$ tenderar mot 0 då $k \to \infty$ medföljs att $\rho(Q_k^T \otimes P_k) \to 0$. Det leder till att iterationsoperatorn $Q_k^T \otimes P_k \to 0$ i varje matrisnorm, och därmed

$E_k \to 0$, det vill säga $X_k \to X$. Optimala valet av parametrar $\{p_1, \cdots, p_k\}$ gör $\prod_{i=1}^{k} \rho(U_i)$ minimal.

Metoden kan förlängas till allmänna Sylvesterekvationer av formen

$$AX + XB = C \tag{6}$$

genom att använda två konstanta parametrar α och β [LZZ20, s. 2]. Här antags A och B vara positivt semidefinita samt att minst en av dem är positivt definit. Om så gäller medför det $\Lambda(A) \subset [0, \infty)$ och $\Lambda(B) \subset [0, \infty)$, vilket vidare medför $\Lambda(-B) \subset (-\infty, 0]$. Om dessutom A eller B är positivt definita innebär det $0 \notin \Lambda(A)$ och $0 \notin \Lambda(B)$, och därmed kan A och $-B$ inte ha något egenvärde gemensamt. På så sätt säkerställs att (6) har en unik lösning.

Iterationen utförs då i två steg enligt

$$\begin{aligned}(A + \alpha I)X_{i-\frac{1}{2}} + X_{i-1}(B - \alpha I) &= C,\\ (A - \beta I)X_{i-\frac{1}{2}} + X_i(B + \beta I) &= C.\end{aligned}$$

Genom liknande argument som för Lyapunov-fallet erhålls en rekursiv felrelation på

formen $E_i = U_i E_{i-1} V_i$, där spektralradierna $\rho(U_i)$ och $\rho(V_i)$ begränsas av

$$\rho(U_i) \leq \max_{x \in \Lambda(A)} \left| \frac{x - \beta}{x + \alpha} \right|, \qquad \rho(V_i) \leq \max_{y \in \Lambda(B)} \left| \frac{y - \alpha}{y + \beta} \right|.$$

Vektorisering av felet ger att spektralradien är högst

$$\prod_{k=1}^{i} \rho(U_k)\rho(V_k) \leq \max_{x \in \Lambda(A), y \in \Lambda(B)} \prod_{k=1}^{i} \left| \frac{(x - \beta)(y - \alpha)}{(x + \alpha)(y + \beta)} \right|.$$

Konvergens av X_i mot den unika lösningen X är därför garanterad om

$$\max_{x \in \Lambda(A), y \in \Lambda(B)} \prod_{k=1}^{i} \left| \frac{(x - \beta)(y - \alpha)}{(x + \alpha)(y + \beta)} \right| \to 0 \quad \text{när} \quad i \to \infty.$$

Skiftparametrarna $\{\alpha, \beta\}$ sägs därmed vara optimala om de minimerar ovan-

nämnda produkt, det vill säga om de löser min-max-problemet

$$\min_{\{\alpha,\beta\}} \max_{x\in\Lambda(A),y\in\Lambda(B)} \prod_{k=1}^{i} \left| \frac{(x-\beta)(y-\alpha)}{(x+\alpha)(y+\beta)} \right|.$$

4.2 Diskreta Sylvesterekvationer

Det är också värt att diskutera en lösningsmetod för diskreta Sylvesterekvationer [LMK16, s. 584]. Givet matriser A, B och C av lämpliga dimensioner definieras

$$AXB + X = C \tag{7}$$

vilken antags ha en unik lösning. Ovan ekvation kan tänkas lösas genom en slags Bartels-Stewart algoritm och reell Schur-dekomposition som med den kontinuerliga motsvarigheten. Ekvationen (7) skulle då se ut som

$$U\tilde{A}U^T XV\tilde{B}V^T + X = C$$

och vidare

$$\tilde{A}U^T XV\tilde{B} + U^T XV = U^T CV$$

vilket kan skrivas som

$$\tilde{A}Y\tilde{B} + Y = \tilde{C}$$

där $Y = U^T XV$ och $\tilde{C} = U^T CV$, etcetera. Dock för att undvika repetition tags istället upp en unik metod. Observera från (7) att

$$X = C - AXB.$$

Insätt $X = C - AXB$ i högerledets X för att få

$$X = C - ACB + A^2XB^2.$$

Insättningen $X = C - AXB$ i högerledet kan göras arbiträrt många gånger. Det erhålls

$$X = \sum_{i=0}^{\infty} (-1)^i A^i C B^i.$$

Mer om konvergenskravet kommer senare. Här kan en delsumma definieras enligt

$$X_k = \sum_{i=0}^{k} (-1)^i A^i C B^i$$

för $k \geq 0$, men det bör noteras att den summan konvergerar långsamt. Istället kan en mer effektiv sådan definieras enligt

$$\begin{cases} X_0 & = C \\ X_{k+1} & = X_k + (-1)^{2^k} A^{2^k} X_k B^{2^k} \quad (k \geq 0). \end{cases}$$

Genom induktion kan det visas att den slutna formeln är

$$X_k = \sum_{i=0}^{2^k-1} (-1)^i A^i C B^i.$$

Observera att basfallet gäller, ty

$$X_0 = \sum_{i=0}^{0} (-1)^i A^i C B^i = C.$$

Antag att den slutna formeln gäller för något $k \geq 0$. Då gäller

$$\begin{aligned} \sum_{i=0}^{2^{k+1}-1} (-1)^i A^i C B^i &= \sum_{i=0}^{2^k-1} (-1)^i A^i C B^i + \sum_{i=2^k}^{2^{k+1}-1} (-1)^i A^i C B^i \\ &= \sum_{i=0}^{2^k-1} (-1)^i A^i C B^i + (-1)^{2^k} A^{2^k} \left(\sum_{i=0}^{2^k-1} (-1)^i A^i C B^i \right) B^{2^k} \\ &= X_k + (-1)^{2^k} A^{2^k} X_k B^{2^k} \\ &= X_{k+1}. \end{aligned}$$

Nu när den slutna formeln har visats stämma gäller det att härleda kravet för att den partiella summan ska konvergera. Låt $T_i = (-1)^i A^i C B^i$ för $i \geq 0$. Vektorisera sedan båda led enligt sats 2.13 (s. 19) och få

$$\begin{aligned} \text{Vec}(T_i) &= (-1)^i ((B^i)^T \otimes A^i) \text{Vec}(C) \\ &= (-1)^i ((B^T)^i \otimes A^i) \text{Vec}(C) \\ &= (-(B^T \otimes A))^i \text{Vec}(C). \end{aligned}$$

Att $(B^T)^i \otimes A^i = (B^T \otimes A)^i$ är en direkt konsekvens av att Kronecker-produkten uppfyller $(M \otimes N)(P \otimes Q) = (MP) \otimes (NQ)$. Från ovan vektorisering är det klart att

$$\mathrm{Vec}(X_k) = \sum_{i=0}^{2^k-1} \mathrm{Vec}(T_i) = \left(\sum_{i=0}^{2^k-1} (-(B^T \otimes A))^i \right) \mathrm{Vec}(C).$$

Konvergens av $\mathrm{Vec}(X_k)$ mot $\mathrm{Vec}(X)$ sker om och endast om $\sum_{i=0}^{\infty} M^i$ konvergerar elementvis (eller normvis). Sats 2.21 (s. 24) säger att delsummorna $S_k = \sum_{i=0}^{k} (-(B^T \otimes A))^i$ konvergerar om och endast om $\rho(-(B^T \otimes A)) < 1$.

Enligt sats 2.15 (s. 21) är $\rho(-(B^T \otimes A)) = \rho(B)\rho(A)$. Sammanfattningsvis konvergerar alltså X_k mot X om och endast om $\rho(A)\rho(B) < 1$.

5 Sammanfattning

Det här arbetet har undersökt Sylvesterekvationer, kontinuerliga och diskreta, samt två exempel på användningsområden inom maskininlärning. Det har dessutom täckt villkoren för existens och entydighet av lösningar för dessa ekvationer, tillsammans med bevis och ett par exempel på lösningsmetoder. Det må vara en liten del av helheten, men förhoppningsvis har det skapat lite mer förståelse för vad som kan ske under ytan. AI grundar sig på matematik, kanske mest av allt den linjära algebran.

AI-modeller kommer sannolikt fortsätta att utvecklas för att bli mer effektiva och kräva mindre mängder träningsdata än de gör idag. Inom de kommande åren kommer fler företag att integrera AI-verktyg på arbetsplatser, i fysiska butiker och nätbutiker, och till och med i hemmen. Det finns dessutom stor potential för en bredare användning av dessa självlärande maskiner. I sjukvården skulle optimerade linjära modeller kunna ställa snabbare diagnoser från medicinska bilder. I utvecklingsländer kan resurseffektiva algoritmer möjliggöra tillgång till utbildning, jordbruksoptimering och tidiga varningssystem för naturkatastrofer.

Där det finns stor potential finns dock också stor risk. Utvecklingen av AI kommer att fortsätta i många år framöver, och nyckeln är att matematiken inte bara driver effektivitet, utan också rättvisa, transparens, och etiskt ansvar.

Referenser

[Ber25] Dave Bergmann, *What Is Latent Space?*, January 2025, Hämtad 20 Oktober 2025.

[CSWZ08] Gang Chen, Yangqiu Song, Fei Wang, and Changshui Zhang, *Semi-supervised Multi-label Learning by Solving a Sylvester Equation*, Proceedings of the 2008 SIAM International Conference on Data Mining, SIAM, 2008, pp. 410–419.

[DBP21] Debasmit Das, Yash Bhalgat, and Fatih Porikli, *Data-Driven Weight Initialization with Sylvester Solvers*, Practical ML for Developing Countries Workshop, ICLR 2021, May 2021, arXiv:2105.10335 [cs.NE].

[Dev] Google Developers, *What is Machine Learning?*, https://developers.google.com/machine-learning/intro-to-ml/what-is-ml, Hämtad 12 September 2025.

[GVL13] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, 4th ed., Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, 2013.

[HJ13] Roger A. Horn and Charles R. Johnson, *Matrix analysis*, 2nd ed., Cambridge University Press, 2013.

[HNO06] Per Christian Hansen, James G. Nagy, and Dianne P. O'Leary, *Deblurring Images*, 1st ed., Society for Industrial and Applied Mathematics, 2006.

[LMK16] Jicheng Li, Ziya Mei, and Xu Kong, *A New Version of the Smith Method for Solving Sylvester Equation and discrete-time Sylvester Equation*, Journal of Applied Analysis and Computation **6** (2016), no. 2, 582–599.

[LZZ20] Zhong-Yun Liu, Yang Zhou, and Yulin Zhang, *On Inexact Alternating Direction Implicit Iteration for Continuous Sylvester Equations*, Preprint, Centro de Matemática, Universidade do Minho, 2020.

[Pen00] Thilo Penzl, *A Cyclic Low-Rank Smith Method for Large Sparse Lyapunov Equations*, SIAM Journal on Scientific Computing **21** (2000), no. 4, 1401–1418.

[San17a] Grant Sanderson, *Backpropagation calculus | Deep Learning Chapter 4*, https://www.youtube.com/watch?v=tIeHLnjs5U8, November 2017, YouTube video, publicerad 3 November 2017.

[San17b] ———, *Backpropagation, intuitively | Deep Learning Chapter 3*, https://www.youtube.com/watch?v=Ilg3gGewQ5U, November 2017, YouTube video, publicerad 3 November 2017.

[San17c] ———, *But what is a Neural Network? | Deep Learning Chapter 1*, https://www.youtube.com/watch?v=aircAruvnKk, October 2017, YouTube video, publicerad 5 Oktober 2017.

[San17d] ———, *Gradient descent, how neural networks learn | Deep Learning Chapter 2*, https://www.youtube.com/watch?v=IHZwWFHWa-w, October 2017, YouTube video, publicerad 16 Oktober 2017.

[SM19] Angelika Schwarz and Carl Christian Kjelgaard Mikkelsen, *Robust Task-Parallel Solution of the Triangular Sylvester Equation*, Tech. report, Department of Computing Science, Umeå University, May 2019, arXiv:1905.10574 [cs.MS].

[Str19] Gilbert Strang, *Linear Algebra and Learning from Data*, 1st ed., Wellesley-Cambridge Press, 2019.

Rättningar till maskininlärning med linjära system

Emanuel Hedberg

March 2026

s. 36

Det räcker att välja endast en skiftparameter $p \in \mathbb{C}_-$ istället för flera distinkta p_i värden. Följden är en förenklad iteration

$$(A^T + pI)X_{k+\frac{1}{2}} + X_k(A - pI) = -BB^T,$$

$$(A^T - pI)X_{k+\frac{1}{2}} + X_{k+1}(A + pI)^{-1} = -BB^T.$$

ss. 37 - 39

Deriveringen av konvergenskriteriet är onödigt komplicerad och inkorrekt. Uttryck tvåstegsiterationen som enkel iteration och få

$$X_{k+1} = PX_kQ + R$$

där

$$P = (A^T - pI)(A^T + pI)^{-1},$$

$$Q = (A - pI)(A + pI)^{-1},$$

$$R = (A^T - pI)(A^T + pI)^{-1}BB^T(A + pI)^{-1} - BB^T(A + pI)^{-1},$$

antaget $(A + pI)$ är inverterbar. Observera att X är den unika matrisen som uppfyller $X = PXQ + R$. Det innebär

$$X - X_k = PXQ + R - PX_{k-1}Q - R = P(X - X_{k-1})Q.$$

Repeterad applicering av rekursiv formel ger

$$X - X_k = P^k(X - X_0)Q^k = P^kXQ^k.$$

Vektorisera systemet enligt Sats 2.13 (s. 19):

$$Vec(X - X_k) = ((Q^k)^T \otimes P^k)Vec(X) = (Q^T \otimes P)^kVec(X).$$

Vektorn $Vec(X - X_k)$ konvergerar om och endast om $(Q^T \otimes P)^k \to 0$ då $k \to \infty$. Det är ekvivalent med

$$\rho(Q^T \otimes P) \overset{Sats2.15}{=} \rho(Q^T)\rho(P) \overset{Sats2.10}{=} \rho(P)\rho(Q) < 1.$$

Observera:

$$\begin{aligned}\rho(P) &= \rho((A^T - pI)(A^T + pI)^{-1}) \\ &= \rho((A - pI)^T(A + pI)^{-T}) \\ &= \rho(((A + pI)^{-1}(A - pI))^T) \\ &\overset{Sats2.10}{=} \rho((A + pI)^{-1}(A - pI)).\end{aligned}$$

Notera att $\rho(P) = \rho((A+pI)^{-1}(A-pI))$ och $\rho(Q) = \rho((A-pI)(A+pI)^{-1})$. Det går att visa $\rho(P) = \rho(Q)$ då det följer från $\rho(MN) = \rho(NM)$. För $\lambda \in \Lambda(MN)$ och motsvarande egenvektor v gäller följande:

$$NM(Nv) = N(MNv) = N(\lambda v) = \lambda(Nv)$$

och därmed $\lambda \in \Lambda(NM)$. På så sätt är $\rho(P)\rho(Q) = \rho(Q)^2$ och konvergens sker om och endast om $\rho(Q)^2 < 1$, alltså $\rho(Q) < 1$. Ett optimalt p bör därför så långt som möjligt minimera

$$\rho((A - pI)(A + pI)^{-1}).$$

ss. 39 - 40

För allmänna kontinuerliga Sylvesterekvationen utförs en liknande derivering. Enkeliterationen blir

$$X_{k+1} = UX_kV + W$$

där

$$\begin{aligned}U &= (A - \beta I)(A + \alpha I)^{-1}, \\ V &= (B - \alpha I)(B + \beta I)^{-1},\end{aligned}$$

och W en matris som kommer att försvinna i nästa steg. På samma sätt som tidigare fås $X - X_k = U^k X V^k$, där konvergens sker om och endast om $\rho(U)\rho(V) < 1$. Därför bör α och β så långt som möjligt minimera

$$\rho((A - \beta I)(A + \alpha I)^{-1}) \cdot \rho((B - \alpha I)(B + \beta I)^{-1}).$$

Rättningar till maskininlärning med linjära system

Emanuel Hedberg

March 2026

s. 36

Det räcker att välja endast en skiftparameter $p \in \mathbb{C}_-$ istället för flera distinkta p_i värden. Följden är en förenklad iteration

$$(A^T + pI)X_{k+\frac{1}{2}} + X_k(A - pI) = -BB^T,$$

$$(A^T - pI)X_{k+\frac{1}{2}} + X_{k+1}(A + pI)^{-1} = -BB^T.$$

ss. 37 - 39

Deriveringen av konvergenskriteriet är onödigt komplicerad och inkorrekt. Uttryck tvåstegsiterationen som enkel iteration och få

$$X_{k+1} = PX_kQ + R$$

där

$$P = (A^T - pI)(A^T + pI)^{-1},$$

$$Q = (A - pI)(A + pI)^{-1},$$

$$R = (A^T - pI)(A^T + pI)^{-1}BB^T(A + pI)^{-1} - BB^T(A + pI)^{-1},$$

antaget $(A + pI)$ är inverterbar. Observera att X är den unika matrisen som uppfyller $X = PXQ + R$. Det innebär

$$X - X_k = PXQ + R - PX_{k-1}Q - R = P(X - X_{k-1})Q.$$

Repeterad applicering av rekursiv formel ger

$$X - X_k = P^k(X - X_0)Q^k = P^kXQ^k.$$

Vektorisera systemet enligt Sats 2.13 (s. 19):

$$Vec(X - X_k) = ((Q^k)^T \otimes P^k)Vec(X) = (Q^T \otimes P)^kVec(X).$$

Vektorn $Vec(X - X_k)$ konvergerar om och endast om $(Q^T \otimes P)^k \to 0$ då $k \to \infty$. Det är ekvivalent med

$$\rho(Q^T \otimes P) \overset{Sats2.15}{=} \rho(Q^T)\rho(P) \overset{Sats2.10}{=} \rho(P)\rho(Q) < 1.$$

Observera:

$$\begin{aligned}\rho(P) &= \rho((A^T - pI)(A^T + pI)^{-1}) \\ &= \rho((A - pI)^T(A + pI)^{-T}) \\ &= \rho(((A + pI)^{-1}(A - pI))^T) \\ &\overset{Sats2.10}{=} \rho((A + pI)^{-1}(A - pI)).\end{aligned}$$

Notera att $\rho(P) = \rho((A+pI)^{-1}(A-pI))$ och $\rho(Q) = \rho((A-pI)(A+pI)^{-1})$. Det går att visa $\rho(P) = \rho(Q)$ då det följer från $\rho(MN) = \rho(NM)$. För $\lambda \in \Lambda(MN)$ och motsvarande egenvektor v gäller följande:

$$NM(Nv) = N(MNv) = N(\lambda v) = \lambda(Nv)$$

och därmed $\lambda \in \Lambda(NM)$. På så sätt är $\rho(P)\rho(Q) = \rho(Q)^2$ och konvergens sker om och endast om $\rho(Q)^2 < 1$, alltså $\rho(Q) < 1$. Ett optimalt p bör därför så långt som möjligt minimera

$$\rho((A - pI)(A + pI)^{-1}).$$

ss. 39 - 40

För allmänna kontinuerliga Sylvesterekvationen utförs en liknande derivering. Enkeliterationen blir

$$X_{k+1} = UX_kV + W$$

där

$$\begin{aligned}U &= (A - \beta I)(A + \alpha I)^{-1}, \\ V &= (B - \alpha I)(B + \beta I)^{-1},\end{aligned}$$

och W en matris som kommer att försvinna i nästa steg. På samma sätt som tidigare fås $X - X_k = U^k X V^k$, där konvergens sker om och endast om $\rho(U)\rho(V) < 1$. Därför bör α och β så långt som möjligt minimera

$$\rho((A - \beta I)(A + \alpha I)^{-1}) \cdot \rho((B - \alpha I)(B + \beta I)^{-1}).$$