PA081: Programování numerických výpočtů

7. Rozklady matic

Aleš Křenek

jaro 2013

Motivace

- danou matici $\mathbf{A}$ vyjádříme jako součin

$$\mathbf{A} = \mathbf{A}_1 \mathbf{A}_2 \dots \mathbf{A}_n$$

- matice $\mathbf{A}_1, \mathbf{A}_2, \dots \mathbf{A}_n$ mají nějaké speciální vlastnosti
- zřejmější vlastnosti problému popsaného původní $\mathbf{A}$
  - kritické body vektorového pole
  - statisticky významné vlastnosti nějakého jevu
  - podmíněnost systému lineárních rovnic
  - ...
- vlastnosti rozkladu lze prakticky využít
  - řešení systému lineárních rovnic
- existují implementace algoritmů se známými vlastnostmi
  - zpravidla numericky stabilní
  - optimalizované na konkrétní CPU, maximálně efektivní

Přehled

- $\mathbf{A} = \mathbf{LU}$, resp. $\mathbf{PA} = \mathbf{LU}$
  - $\mathbf{L}$ a $\mathbf{U}$ jsou dolní a horní trojúhelníková
  - $\mathbf{P}$ je permutace řádků
- Choleského: $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
  - pro symetrickou pozitivně definitní $\mathbf{A}$
- $\mathbf{A} = \mathbf{QR}$
  - $\mathbf{Q}$ ortogonální, $\mathbf{R}$ horní trojúhelníková
  - symetrické varianty RQ, QL a LQ
- singulární hodnoty: $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}$
  - $\mathbf{U}, \mathbf{V}$ ortogonální, Σ diagonální
- spektrální: $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$
  - Λ diagonální, vlastní hodnoty
  - varianta Jordanova: blokově diagonální Λ, násobné vlastní hodnoty
- Shurova: $\mathbf{A} = \mathbf{V}\mathbf{S}\mathbf{V}^T$
  - $\mathbf{V}$ ortogonální
  - $\mathbf{S}$ horní trojúhelníková s vlastními hodnotami $\mathbf{A}$ na diagonále

LU dekompozice

Princip

- předpokládejme, že dokážeme rozložit $\mathbf{A} = \mathbf{LU}$ tak, že
  - $\mathbf{L}$ je spodní trojúhelníková matice (prvky nad diagonálou jsou nulové)
  - $\mathbf{U}$ je horní trojúhelníková matice (prvky pod diagonálou jsou nulové)
- potom lze psát

$$\mathbf{Ax} = (\mathbf{LU})\mathbf{x} = \mathbf{L}(\mathbf{Ux}) = \mathbf{b}$$

- původní systém lze vyřešit postupným řešením

$$\mathbf{Ly} = \mathbf{b} \qquad \text{a} \qquad \mathbf{Ux} = \mathbf{y}$$

- to je triviální dopřednou a zpětnou substitucí

LU dekompozice

Výpočet rozkladu

- prvky rozkladu $\mathbf{A} = \mathbf{LU}$ lze, s ohledem na nuly psát

$$
\begin{aligned}
i < j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ii}l_{ij} \\
i = j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ii}l_{jj} \\
i > j &: a_{ij} = u_{i1}l_{1j} + u_{i2}l_{2j} + \cdots + u_{ij}l_{jj}
\end{aligned}
$$

- je to systém N^2 rovnic v $N^2 + N$ neznámých
  - diagonála je pokryta u i l
  - můžeme tedy volit $l_{ii} = 1$

LU dekompozice

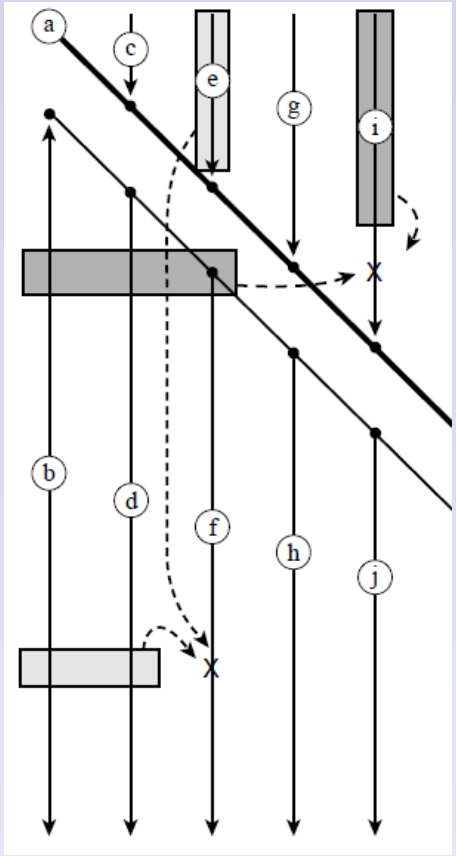
Croutův algoritmus

- postupně pro $j = 1, 2, \ldots, N$:
  - pro $i = 1, 2, \ldots j$ spočítáme na základě uvedených rovnic

$$u_{ij} = a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj}$$

  - pro $i = j+1, j+2, \ldots, N$

$$l_{ij} = \frac{1}{u_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \right)$$



- postup vždy využívá dříve spočtené prvky
- k numerické stabilitě je třeba navíc dodat pivoting l_{jj}

LU dekompozice

Shrnutí a použití

- řešení lineárních rovnic pro libovolný počet $\mathbf{b}$
  - Croutův algoritmus $O(N^3)$
  - dopředná a zpětná substituce $O(N^2)$
  - všechna $\mathbf{b}$ není třeba znát dopředu – hlavní přednost metody
- výpočet inverzní matice
  - řešení systému pro $\mathbf{b} =$ bázové vektory
  - potřebujeme-li počítat $\mathbf{A}^{-1}\mathbf{B}$, je lepší přímo pro sloupce $\mathbf{B}$ než explicitní vyjádření $\mathbf{A}^{-1}$

Choleského dekompozice

- předpokládáme $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
- po prvcích platí

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=0}^{i-1} l_{ik}^2} \qquad l_{ji} = \frac{1}{l_{ii}} \left(a_{ij} - \sum_{k=0}^{i-1} l_{ik} l_{jk} \right)$$

- podobně jako při LU dekompozici vždy využíváme dříve spočtené prvky
- odmocninu lze vždy spočítat pro symetrickou pozitivně definitní $\mathbf{A}$
  - omezenější, ale stále významná třída problémů
- algoritmus je efektivnější a numericky stabilní
  - cca. $2\times$ méně operací než LU, menší paměťová náročnost
  - má smysl používat speciální funkce z knihoven

QR dekompozice

- rozklad $\mathbf{A} = \mathbf{QR}$
  - $\mathbf{Q}$ ortogonální, $\mathbf{R}$ trojúhelníková
- systém $\mathbf{Ax} = \mathbf{b}$ lze psát

$$\mathbf{Rx} = \mathbf{Q}^T\mathbf{b}$$

- jednoduché řešení substitucí
- lepší numerické vlastnosti
- metody konstrukce – nulování prvků pod diagonálou
  - Gram-Schmidtův proces
  - Householderovy transformace
  - Givensovy rotace

QR dekompozice

Gram-Schmidtův proces

- Gram-Schmidtův ortogonalizační proces

$$\mathbf{u}_1 = \mathbf{a}_1 \qquad \mathbf{v}_1 = \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|}$$

$$\mathbf{u}_2 = \mathbf{a}_2 - \mathrm{proj}_{\mathbf{v}_1}\mathbf{a}_2 \qquad \mathbf{v}_2 = \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|}$$

$$\mathbf{u}_3 = \mathbf{a}_3 - \mathrm{proj}_{\mathbf{v}_1}\mathbf{a}_3 - \mathrm{proj}_{\mathbf{v}_2}\mathbf{a}_3 \qquad \mathbf{v}_3 = \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|}$$

$$\vdots$$

- potom

$$\mathbf{Q} = (\mathbf{v}_1 \dots \mathbf{v}_n) \qquad \mathbf{R} = \begin{pmatrix} \mathbf{v}_1 \cdot \mathbf{a}_1 & \mathbf{v}_1 \cdot \mathbf{a}_2 & \dots \\ 0 & \mathbf{v}_2 \cdot \mathbf{a}_2 & \dots \\ \dots & & \end{pmatrix}$$

- numerický problém, jsou-li některé $\mathbf{a}_i, \mathbf{a}_j$ skoro kolmé

QR dekompozice

Householderova transformace

- zrcadlení daného vektoru podle nadroviny
  - zarovná s bázovým vektorem, zachová velikost
- pro libovolné $\mathbf{x}$:

$$\mathbf{u} = \mathbf{x} + \alpha \mathbf{e}_1 \qquad \mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|} \qquad \mathbf{Q} = I - 2\mathbf{v}\mathbf{v}^T$$

- potom $\mathbf{Q}\mathbf{x} = (\alpha, 0, \ldots, 0)^T$

QR dekompozice

Householderova transformace

- zrcadlení daného vektoru podle nadroviny
  - zarovná s bázovým vektorem, zachová velikost
- pro libovolné $\mathbf{x}$:

$$\mathbf{u} = \mathbf{x} + \alpha \mathbf{e}_1 \qquad \mathbf{v} = \frac{\mathbf{u}}{\|\mathbf{u}\|} \qquad \mathbf{Q} = I - 2\mathbf{v}\mathbf{v}^T$$

- potom $\mathbf{Q}\mathbf{x} = (\alpha, 0, \ldots, 0)^T$
- triangulace $\mathbf{A}$

$$\mathbf{R} = \mathbf{Q}_{n-1} \ldots \mathbf{Q}_1 \mathbf{A} \qquad \text{a tedy} \qquad \mathbf{Q} = \mathbf{Q}_1^T \mathbf{Q}_2^T \ldots \mathbf{Q}_{n-1}^T$$

Základní dekompozice – shrnutí

- LU
  - ve variantě **PA** = **LU** existuje pro všechny čtvercové matice
  - odpovídá Gaussově eliminaci – trpí stejnými numerickými problémy
- Choleského
  - čtvercové, symetrické, pozitivně definitní matice
  - numericky stabilní (to je ale LU pro tyto matice také)
  - cca. $2\times$ méně operací
- QR
  - obecné $m \times n$ matice
  - existují numericky stabilní konstrukce (Householderova transformace)
  - výpočetně náročnější
- použití především k řešení systémů lineárních rovnic

Dekompozice na singulární hodnoty

Základní tvrzení

- libovolnou reálnou (i komplexní) matici lze rozložit

$$\mathbf{A}_{M\times N} = \mathbf{U}_{M\times N} \cdot \Sigma_{N\times N} \cdot \mathbf{V}^T_{N\times N} \quad (= \mathbf{U}'_{M\times M} \cdot \Sigma'_{M\times N} \cdot \mathbf{V'}^T_{N\times N})$$

- $\mathbf{U}$ je sloupcově ortogonální
  - až na nulové sloupce v případě $M \leq N$
- Σ diagonální a $\mathbf{V}$ ortogonální
- rozklad je unikátní až na
  - současnou perumutaci sloupců všech tří matic
  - lineární kombinaci sloupců $\mathbf{U}, \mathbf{V}$ odpovídajících nulovým σ_i

Dekompozice na singulární hodnoty

Geometrický význam

- $\mathbf{A}$ je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí $\mathbf{U}$

Dekompozice na singulární hodnoty

Geometrický význam

- **A** je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí **U**
- obor hodnot zobrazení **A**
  - sloupce **U** odpovídající nenulovým σ_i jsou jeho generátory

Dekompozice na singulární hodnoty

Geometrický význam

- $\mathbf{A}$ je složení transformací
  - rotace/zrcadlení $\mathbf{V}^{-1}$
  - zvětšení/zmenšení faktory σ_i ve směrech e_i, včetně degenerace ($\sigma_i = 0$)
  - rotace/zrcadlení a projekce do méně/více dimenzí $\mathbf{U}$
- obor hodnot zobrazení $\mathbf{A}$
  - sloupce $\mathbf{U}$ odpovídající nenulovým σ_i jsou jeho generátory
- nulový prostor $\mathcal{N}(\mathbf{A}) = \{\mathbf{x} \in \mathbb{R}^N : \mathbf{A}\mathbf{x} = 0\}$
  - řádky $\mathbf{V}^T$ odpovídající nulovým σ_i jsou jeho generátory

Dekompozice na singulární hodnoty

Numerický význam

- „oddělení zrna od plev“
- sloupce $\mathbf{U}$ a $\mathbf{V}$ jsou kolmé a normované
- veškeré potenciální degenerace soustředěny do Σ
  - singularity $\mathbf{A}$ odpovídají nulovým σ_i
  - včetně numerických ($\sigma_i \approx 0$)
- numericky velmi stabilní algoritmus dekompozice
- lze použít na řešení systémů lineárních rovnic
  - $M < N$ a $M = N$ singulární: reprezentant řešení + generátor prostoru
  - $M > N$: nejbližší řešení

Dekompozice na singulární hodnoty

Použití pro $M = N$

- řešení systému rovnic, resp. výpočet inverzní matice

$$\mathbf{A}^{-1} = \mathbf{V}[\mathrm{diag}(1/\sigma_i)]\mathbf{U}^T$$

- kdy to nejde
  - jedno nebo více σ_i je nulových – A byla singulární
  - $\sigma_{\min}/\sigma_{\max} < \epsilon$ (špatně podmíněná matice) – standardní metody řešení selhaly

Dekompozice na singulární hodnoty

Použití pro $M = N$

- řešení systému rovnic, resp. výpočet inverzní matice

$$\mathbf{A}^{-1} = \mathbf{V}[\mathrm{diag}(1/\sigma_i)]\mathbf{U}^T$$

- kdy to nejde
  - jedno nebo více σ_i je nulových – A byla singulární
  - $\sigma_{\min}/\sigma_{\max} < \epsilon$ (špatně podmíněná matice) – standardní metody řešení selhaly
- označme

$$\Sigma' = \left[\mathrm{diag}\begin{cases} 1/\sigma_i & \sigma_i \neq 0 \\ 0 & \sigma_i = 0 \end{cases}\right]$$

- rovnice $\mathbf{Ax} = \mathbf{b}$ nemusí mít řešení, přesto zkusíme $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$

Dekompozice na singulární hodnoty

Použití pro $M = N$

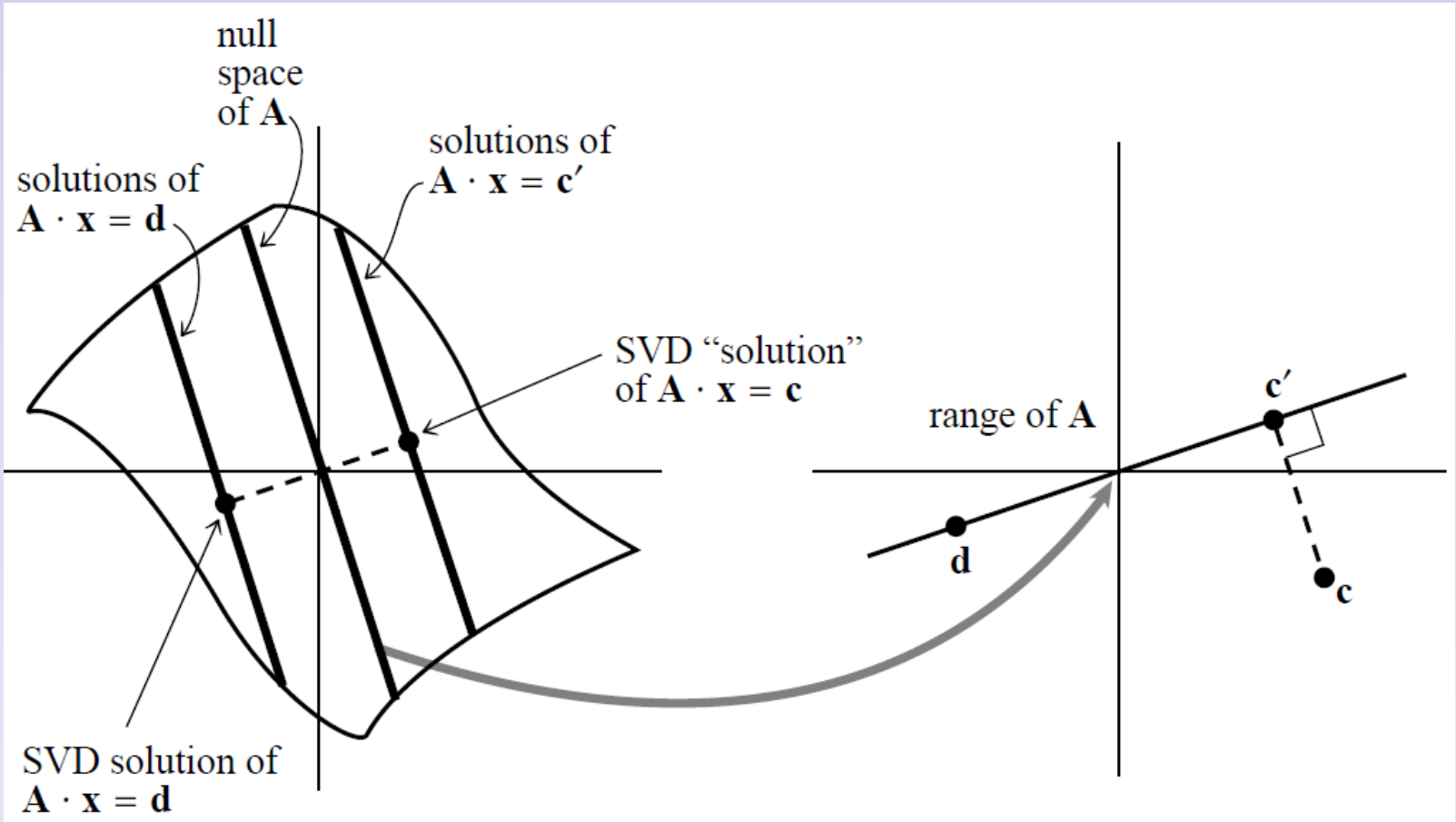
- hledáme nejbližší řešení, tj. minimalizujeme $|\mathbf{Ax} - \mathbf{b}|$
- pro libovolné $\mathbf{x}'$ je $\mathbf{A}(\mathbf{x}+\mathbf{x}') - \mathbf{b} = \mathbf{Ax} - \mathbf{b} + \mathbf{b}'$, kde $\mathbf{b}' = \mathbf{Ax}'$

$$\begin{aligned}
|\mathbf{Ax} - \mathbf{b} + \mathbf{b}'| &= |(\mathbf{U}\Sigma\mathbf{V}^T)(\mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}) - \mathbf{b} + \mathbf{b}'| \\
&= |(\mathbf{U}\Sigma\Sigma'\mathbf{U}^T - I)\mathbf{b} + \mathbf{b}'| \\
&= |\mathbf{U}((\Sigma\Sigma' - I)\mathbf{U}^T\mathbf{b} + \mathbf{U}^T\mathbf{b}')| \\
&= |(\Sigma\Sigma' - I)\mathbf{U}^T\mathbf{b} + \mathbf{U}^T\mathbf{b}'|
\end{aligned}$$

- $\Sigma\Sigma' - I$ je diagonální s nenulovými prvky pro $\sigma_i = 0$
- $\mathbf{b}'$ je v oboru hodnot $\mathbf{A}$, tedy $\mathbf{U}^T\mathbf{b}'$ má nenulové prvky právě pro $\sigma_i \neq 0$
- minimum právě pro $\mathbf{b}' = 0$ a tedy i $\mathbf{x}' = 0$

Dekompozice na singulární hodnoty

Použití pro $M = N$



Dekompozice na singulární hodnoty

Použití pro $M = N$ – prakticky

- singularitu $\mathbf{A}$ detekujeme podle $\sigma_i = 0$
- vypočteme nejbližší řešení jako $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$
- dosazením ověříme, zda je to přesné řešení
  - když ne, víme, že přesné řešení neexistuje
  - máme nejbližší aproximaci

Dekompozice na singulární hodnoty

Použití pro $M = N$ – prakticky

- singularitu $\mathbf{A}$ detekujeme podle $\sigma_i = 0$
- vypočteme nejbližší řešení jako $\mathbf{x} = \mathbf{V}\Sigma'\mathbf{U}^T\mathbf{b}$
- dosazením ověříme, zda je to přesné řešení
  - když ne, víme, že přesné řešení neexistuje
  - máme nejbližší aproximaci
- špatně podmíněná matice $|\sigma_{\max}| \gg |\sigma_{\min}|$
  - lépe v Σ' vynulovat i taková σ_i
  - paradoxní – zahazujeme část vstupní informace
  - v praxi dává lepší výsledky – právě tento vstup má tendenci škodit
  - stanovení prahu $\sigma_i \approx 0$ není triviální

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- méně rovnic, $M < N$
- nekonečně mnoho řešení
- rozklad na singulární hodnoty – $N - M$ nulových σ_i
  - nemusí být přesně nulové (numerické nepřesnosti)
  - může jich být více díky dalším singularitám
- Σ' vypočítáme vynulováním problematických σ_i
- přímo vypočteme reprezentativní řešení $\mathbf{x}$
  - včetně ověření, zda je skutečně řešením
- sloupce $\mathbf{V}$ odpovídající nulovaným σ_i generují prostor dalších řešení

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz
- (skoro) nulové singulární hodnoty
  - skrytá degenerace systému
  - může vést na jedno nebo i více přesných řešení

Dekompozice na singulární hodnoty

Použití pro $M \neq N$

- více rovnic, $M > N$
- neexistuje přesné řešení, hledáme nejbližší aproximaci
- rozklad na singulární hodnoty
  - obecně nemusí dát žádná nulová σ_i
  - získáme nejbližší aproximaci řešení, viz naznačený důkaz
- (skoro) nulové singulární hodnoty
  - skrytá degenerace systému
  - může vést na jedno nebo i více přesných řešení
- velmi malé singulární hodnoty
  - ukazují na nízkou citlivost problému
  - právě ve směrech odpovídajících sloupců $\mathbf{V}$
  - zpravidla lépe vynulovat v Σ'

Dekompozice na singulární hodnoty

Aproximace matic

- původní matici lze vyjádřit

$$A_{ij} = \sum_k \sigma_k U_{ik} V_{jk}$$

- je-li většina σ_i skoro nulových
- má smysl ukládat jen několik sloupců **U** a **V**
  - stále dostáváme poměrně přesnou aproximaci **A**
- násobení **Ax** je výrazně efektivnější – $K(M + N)$ operací

Dekompozice na singulární hodnoty

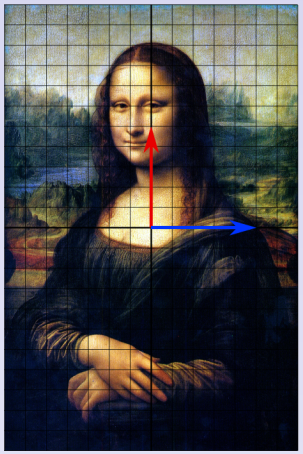
Algoritmus

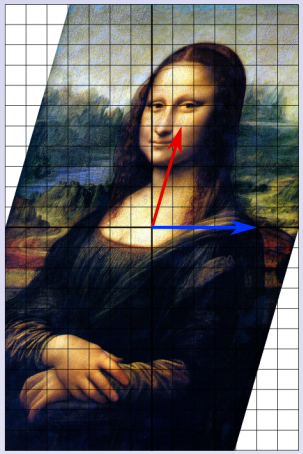
- první fáze – redukce na bidiagonální formu
  - využívá Householderovy transformace
- druhá fáze – iterační, varianta výpočtu vlastních hodnot
- výjimečně stabilní
  - zaměření na vytažení problematických vlastností $\mathbf{A}$ do Σ
- použijeme existující implementaci :-)
  - už to za nás jednou někdo udělal
  - další vylepšující triky dodavatelů knihoven
  - nezbavuje to odpovědnosti za interpretaci výsledku
- původní algoritmus Golub a Reinsch, *Singular value decomposition and least squares solutions*, 1970

Vlastní hodnoty a vektory

- vlastní hodnoty a vektory matice (transformace)

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





- obecná reálná matice nemusí mít reálné vlastní hodnoty
- více viz kursy lineární algebry

Vlastní hodnoty a vektory

Použití

- hledání kořenů polynomů
- výpočty vyšších mocnin matic

$$\mathbf{A}^n = \mathbf{V}\Lambda\mathbf{V}^{-1}\mathbf{V}\Lambda\mathbf{V}^{-1}\ldots\mathbf{V}\Lambda\mathbf{V}^{-1} = \mathbf{V}\Lambda^n\mathbf{V}^{-1}$$

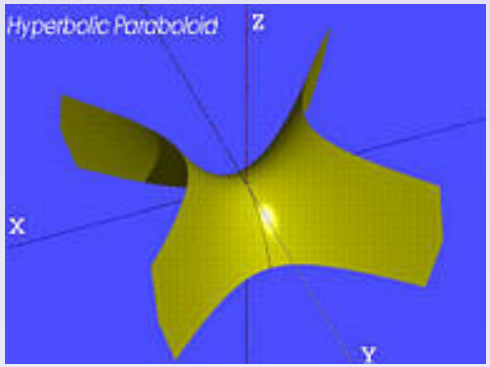
Vlastní hodnoty a vektory

Použití

- hledání kořenů polynomů
- výpočty vyšších mocnin matic

$$\mathbf{A}^n = \mathbf{V}\Lambda\mathbf{V}^{-1}\mathbf{V}\Lambda\mathbf{V}^{-1}\ldots\mathbf{V}\Lambda\mathbf{V}^{-1} = \mathbf{V}\Lambda^n\mathbf{V}^{-1}$$

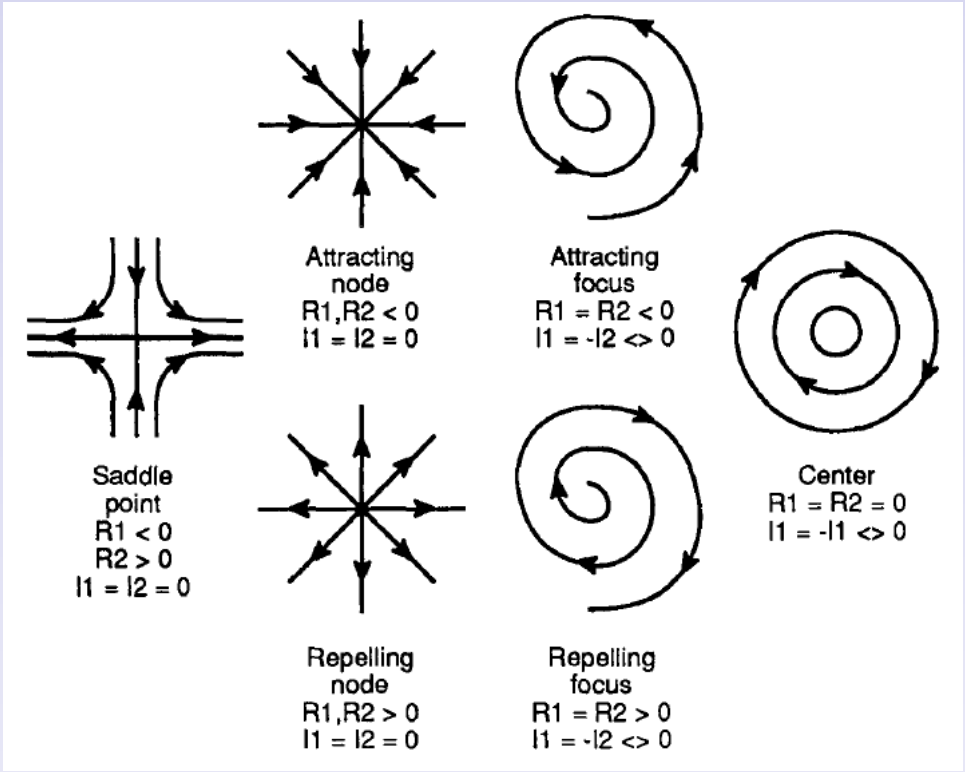
- kritické body funkce více proměnných
  - první parciální derivace jsou nulové
  - Hessián (matice druhých parciálních derivací) určuje aproximaci kvadrikou v daném bodě
  - symetrická matice – reálné hodnoty, ortonormální vektory
  - extrémy – všechny λ kladné, sedla – některé záporné
  - absolutní hodnoty λ určují tvar, vektory orientaci



Vlastní hodnoty a vektory

Použití

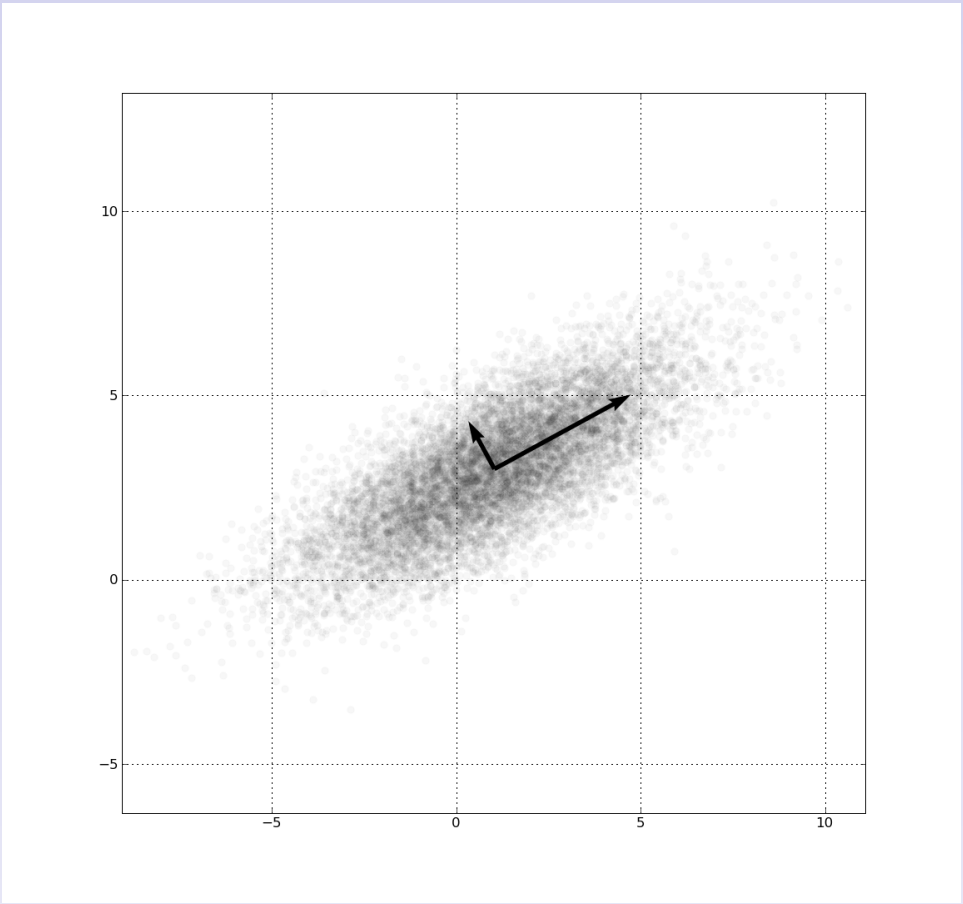
- kritické body vektorového pole
  - velikost vektoru je nulová
  - Jakobián (matice prvních parciálních derivací)



Vlastní hodnoty a vektory

Použití

- analýza hlavních komponent



- téma příští přednášky

Vlastní hodnoty a vektory

Algoritmy

- iterační algoritmus
  - $\mathbf{A}_0 := \mathbf{A}$
  - dekompozice $\mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k$
  - položíme $\mathbf{A}_{k+1} := \mathbf{R}_k \mathbf{Q}_k$
- platí $\mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^T \mathbf{Q}_k \mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^{-1} \mathbf{A}_k \mathbf{Q}_k$
- tedy iterační krok zachovává vlastní hodnoty
- lze ukázat, že za jistých okolností konverguje k Shurově formě

Vlastní hodnoty a vektory

Algoritmy

- numericky dost nepříjemný problém
- ještě jasnější případ, kdy sáhnout k hotovým řešením
- různé varianty pro různé případy
  - reálné a komplexní
  - vlastní hodnoty, vektory, obojí
  - různé speciální typy matic
- vztah k singulárním hodnotám
  - sloupce $\mathbf{U}$ v SVD jsou vlastní vektory $\mathbf{A}\mathbf{A}^T$
  - nenulové singulární hodnoty jsou odmocniny nenulových vlastních hodnot $\mathbf{A}\mathbf{A}^T$

Shrnutí

- základní rozklady matic
  - LU, Cholského, QR
  - použití především k řešení systémů lineárních rovnic
  - existují i další varianty
- SVD
  - náročnější postup, větší stabilita
  - špatně podmíněné systémy
  - větší náhled do problému – další aplikace
- vlastní hodnoty
  - rozklad matice $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$
  - přímé využití pro charakteristiku řady problémů
  - (více v příští přednášce)
- existující implementace
  - téměř vždy je použijeme
  - je nutné vědět, co děláme a co můžeme od dané implementace čekat