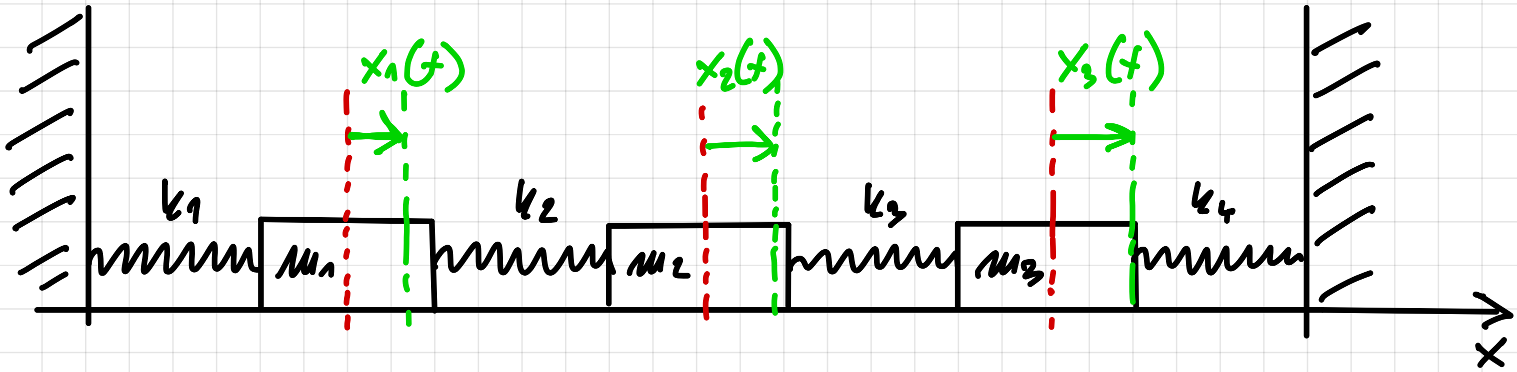
Lastne vrednosti simetričnega tenzorja

16. 3. 2022

Zgled: Določi lastna nihanja in lastne frekvence sistema, ki ga sestavljajo uteži z masami m_1, m_2, m_3, ki so s štirimi vzmetmi k_1, k_2, k_3, k_4 zaporedno pripete med dve nepremične stene in med seboj



Sklopljen sistem nihal opišemo s sistemom diferencialnih enačb za tri nihala:

$$\underbrace{m_1 \ddot{x}_1}^{ma_1} = -k_1 x_1 + k_2 (x_2 - x_1)$$

$$m_2 \ddot{x}_2 = -k_2 (x_2 - x_1) + k_3 (x_3 - x_2)$$

$$m_3 \ddot{x}_3 = -k_4 x_3 - k_3 (x_3 - x_2)$$

Oziroma.

$$m_1 \ddot{x}_1 = -(k_1 + k_2) x_1 + k_2 x_2$$

$$m_2 \ddot{x}_2 = -(k_2 + k_3) x_2 + k_3 x_3 + k_2 x_1$$

$$m_3 \ddot{x}_3 = k_3 x_2 - (k_3 + k_4) x_3$$

Rešitev sistema, ko bo nihal z neko frekvenco ω nastavimo kot:

amplituda posameznega nihala. (in predznak)

$$x_i(t) = y_i \sin(\omega t)$$

Če še upoštevamo, da je:

$$\ddot{x}_i(t) = -\omega^2 y_i \sin(\omega t),$$

lahko zapišemo:

$$
\left.
\begin{aligned}
+ m_1 y_1 \omega^2 &= +(k_1+k_2) y_1 \mp k_2 y_2 \\
+ m_2 y_2 \omega^2 &= +(k_2+k_3) y_2 \mp k_3 y_3 \mp k_2 x_1 \\
+ m_3 y_3 \omega^2 &= -k_3 y_2 + (k_3+k_4) y_3
\end{aligned}
\right| \cdot (-1)
$$

Sistem lahko prepišemo v matrično obliko kot:

$$
\omega^2 \cdot \underbrace{\begin{bmatrix} m_1 & 0 & 0 \\ 0 & m_2 & 0 \\ 0 & 0 & m_3 \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\vec{Y}} = \underbrace{\begin{bmatrix} k_1+k_2 & -k_2 & 0 \\ -k_2 & k_2+k_3 & -k_3 \\ 0 & -k_3 & k_3+k_4 \end{bmatrix}}_{K} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}}_{\vec{Y}}
$$

Dobimo:

$$\omega^2 M \vec{y} = K \cdot \vec{y} \,\Big/ \cdot M^{-1} \quad ; \quad M^{-1} = \begin{bmatrix} \frac{1}{m_1} & 0 & 0 \\ 0 & \frac{1}{m_2} & 0 \\ 0 & 0 & \frac{1}{m_3} \end{bmatrix}$$

Vpeljemo

$$\underbrace{\omega^2}_{=\lambda} \underbrace{M^{-1}M}_{I} \vec{y} = \underbrace{M^{-1}K}_{A} \vec{y} \quad ; \quad A = \begin{bmatrix} \frac{k_1+k_2}{m_1} & -\frac{k_2}{m_1} & 0 \\ -\frac{k_2}{m_2} & \frac{k_2+k_3}{m_2} & -\frac{k_3}{m_2} \\ 0 & -\frac{k_3}{m_3} & \frac{k_3+k_4}{m_3} \end{bmatrix}$$

Dobili smo sistem:

Če $m_1 = m_2 = m_3 = m$, potem lahko simetričen.

$$\boxed{A\vec{y} = \lambda \vec{y}} \quad ; \quad \vec{y} \neq 0$$

Dobljeni sistem imenujemo linearni problem lastnih vrednosti! Ta problem v fiziki pogosto srečamo in tokrat se bomo ukvarjali z reševanjem takšnih linearnih sistemov.

Naloga: Določiti želimo lastne vrednosti λ_i in pripadajoče lastne vektorje $\vec{y}_i$, ki rešijo ta sistem!

<u>V našem primeru:</u> Lastne vrednosti $\lambda = \omega^2$ bodo lastne frekvence sistema nihal. Lastni vektorji pa pripadajoče amplitude (do predznaka) posameznih nihal.

"Naivnejši" pot iskanja lastnih vrednosti:

Velja:

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

identiteta

$$A\vec{x} - \lambda\vec{x} = \underbrace{(A - \lambda I)}_{B}\vec{x} = 0 \quad (*)$$

Opomba:

$$B = \begin{bmatrix} b_{11} & b_{12} & b_{13} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & \dots & b_{2n} \\ & & & & \\ b_{n1} & b_{2n} & \dots & \dots & b_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \\ x_n \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Lahko zapišemo: $b_{i1}x_1 + b_{i2}x_2 + \dots + b_{in}x_n = 0$

Oziroma: $b_{i1} = b_{i2}\frac{x_2}{x_1} + \dots\dots b_{in}\frac{x_n}{x_1}$, $i = 1, \dots n$

Stolpci matrike niso neodvisni, zato je $\det(B) = 0$

Iz tega (*) velja, da je $\det(A - \lambda I) = 0$

Determinanta matrike $N \times N$ nam da t.i. karakteristični polinom N-tega reda v λ. Posledično lastne vrednosti sistema (*) ustrezajo ničlam karakterističnega polinoma:

Torej: $$p(\lambda) = \det(A - \lambda I) = 0$$

Ugotovitev: ker vemo, da ima polinom stopnje N vedno N ničel, bo imel sistem (*) N lastnih vrednosti. Lastne vrednosti ki ustrezajo večkratnim ničlam, imenujemo degenerirane lastne vrednosti.

Zgled: Izračunaj lastne vrednosti matrike

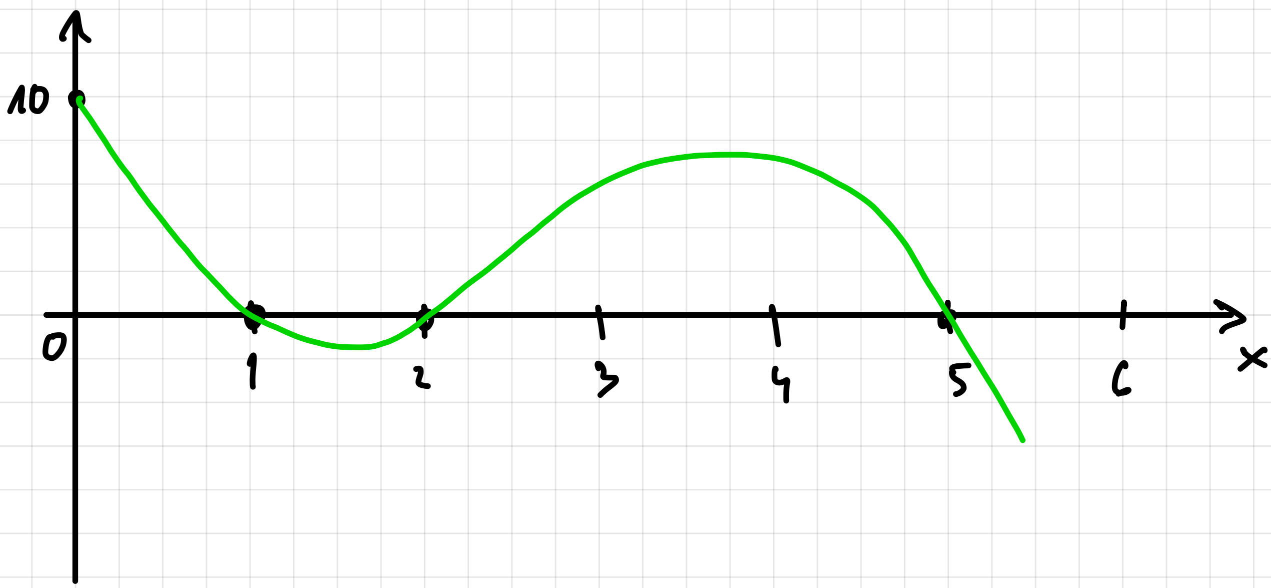
$$A = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & 1 \\ -1 & 1 & 4 \end{bmatrix}$$

Zapišimo determinanto:

$$\det(A - \lambda I) = \begin{vmatrix} 2-\lambda & -1 & -1 \\ -1 & 2-\lambda & 1 \\ -1 & 1 & 4-\lambda \end{vmatrix} \begin{matrix} 2-\lambda & -1 \\ -1 & 2-\lambda \\ -1 & 1 \end{matrix} =$$

$$= \underbrace{(2-\lambda)^2}_{\text{razdelimo}}(4-\lambda) + 2 - 2(2-\lambda) - (4-\lambda) = p(\lambda)$$

$$p(\lambda) = 10 - 17\lambda + 8\lambda^2 - \lambda^3 \quad (= 0)$$



Z iskanjem ničel dobimo: $\lambda_1 = 1,\ \lambda_2 = 2,\ \lambda_3 = 5.$

Izkaže se tudi, da so pripadajoči lastni vektorji

$$\vec{x}_1 = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\1\\0\end{bmatrix}, \quad \vec{x}_2 = \frac{1}{\sqrt{3}}\begin{bmatrix}1\\-1\\1\end{bmatrix}, \quad \vec{x}_3 = \frac{1}{\sqrt{6}}\begin{bmatrix}-1\\1\\2\end{bmatrix}$$

Komentar: Ničel polinoma stopnje $n \geq 5$ ne znamo več poiskati analitično. Postopati moramo numerično. Ta postopek je numerično nestabilen, zato v splošnem tega postopka ne uporabljamo za določanje lastnih vrednosti!

$\Rightarrow$ Obstajajo robustnejši triki.

Nekaj osnovnih pojmov: (Npr. Bronštejn, str. 219)

1.) Matrika je **simetrična**, če je enaka svoji transponirani matriki:

$$A = A^T \quad \text{oziroma} \quad a_{ij} = a_{ji}$$

2.) **Hermitska (sebi adjungirana)** matrika je kvadratna matrika, ki je enaka svoji adjungirani matriki:

$$A = A^+ \quad \text{ali} \quad a_{ij} = \overline{a_{ji}}$$

kompleksno konjugiran!

! Lastne vrednosti Hermitske (ali pa realne simetrične) matrike so vse realne!

3.) **Ortogonalna** matrika je matrika, katere inverz je enak njeni transponirani matriki. Velja

$$A \cdot A^T = A^T \cdot A = I \Rightarrow A^{-1} = A^T$$

Velja še: $\det(A) = \pm 1$

4.) Kompleksna matrika je **unitarna**, če velja $A^+ = A^{-1}$ oziroma $A \cdot A^+ = A^+ \cdot A = I$.

5.) Matrika je normalna, če velja:

$$A \cdot A^{+} = A^{+} \cdot A$$

Ortogonalne in unitarne matrike so normalne.

! Lastni vektorji normalne matrike, ki imajo nedegenerirane lastne vrednosti, so ortogonalni in napenjajo N-dimenzionalni vektorski prostor!

Opomba: Mi se bomo osredotočili na analizo realnih simetričnih matrik.

Diagonalizacija matrike

Matrika A se da diagonalizirati, če obstaja nesingularna matrika $X = [\vec{x}_1, \dots, \vec{x}_n]$ in diagonalna matrika $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$, da velja:

$$\det(X) \neq 0, \qquad X^{-1} A X = \Lambda \;\Big|\cdot X \qquad (*)$$

oziroma:

$$AX = X\Lambda \;\Big/\cdot X^{-1}$$

oziroma:

$$A = X \Lambda X^{-1}$$

Tedaj velja, da so diagonalni elementi matrike Λ lastne vrednosti matrike A, stolpci $\vec{x}_i$ matrike X pa njeni lastni vektorji.

Transponirajmo zgornjo zvezo:

$$A^T = (X\Lambda X^{-1})^T = (X^{-1})^T \Lambda^T X^T = (X^T)^{-1} \Lambda X^T$$

Če je A simetrična:

$$A^T = (X^T)^{-1} \Lambda X^T = A = X \Lambda X^{-1}$$

Sledi: $X^T = X^{-1}$ oziroma $X^T X = I$.

Ugotovili smo, da če je A simetrična, je transformacijska matrika X ortogonalna. !
Stolpci te matrike razpenjajo ortonormirano bazo. N dimenzionalnega evklidskega prostora in ohranjajo skalarni produkt

Zaključek: Realne simetrične matrike imajo realne lastne vrednosti in realne in ortonormirane lastne vektorje.

(*) Ker se ukvarjamo s problemom:

$$X^{-1} A X = \Lambda$$

problemu iskanja lastnih vrednosti rečemo tudi problem diagonalizacije matrike A. Ko matriko diagonaliziramo, imamo lastne vrednosti.

Podobnostna transformacija:

Imejmo transformacijsko matriko Z, $\det(Z) \neq 0$
Transformacija:

$$A \longrightarrow Z^{-1} A Z = B$$

Imenujemo podobnostna transformacija. Matriki A in B pravimo, da sta podobni! Pri podobnostni transformaciji se ohranjajo lastne vrednosti matrike.

$$\det(B - \lambda I) = \det(Z^{-1} A Z - \lambda I) =$$

$$= \det(Z^{-1} A Z - \lambda I Z^{-1} Z) = \det(Z^{-1}(A - \lambda I) Z)$$

$$= \det(Z^{-1}) \cdot \det(A - \lambda I) \cdot \det(Z) =$$

$$= \det(\underbrace{Z^{-1} Z}_{I}) \det(A - \lambda I) = \det(A - \lambda I)$$

Podobnostne relacije igrajo ključno vlogo pri računanju lastnih vrednosti. !

Osnovna ideja:

Matriko A z zaporednimi podobnostnimi transformacijami prevedemo proti diagonalni obliki.

$$A \longrightarrow Z_1^{-1} A Z_1 \longrightarrow Z_2^{-1} Z_1^{-1} A Z_1 Z_2$$
$$\longrightarrow Z_3^{-1} Z_2^{-1} Z_1^{-1} A Z_1 Z_2 Z_3 \longrightarrow \Lambda$$

Če nam postopek uspe priti do diagonalne matrike, potem so lastni vektorji stolpci celotne transformacije

$$X = Z_1 \cdot Z_2 \cdot Z_3 \cdot Z_4 \cdot \ldots \cdot Z_n$$

Opomba: Ni nam treba vedno iti vse do diagonalne matrike. Če nas zanimajo le lastne vrednosti, potem je dovolj, da matriko preoblikujemo v trikotno matriko.

To tudi izkoristimo, da se matrike ne da diagonalizirati, pa bi vseeno radi dobili lastne vrednosti.

Pomembno: Postopek diagonalizacije je stabilen, če so matrike Z_i unitarne!
Glej, Plestenjak, RUNM, str. 167.

Schurova forma: Za vsako matriko A obstaja unitarna matrika Q in zgornje trikotna matrika R, da je

$$\underbrace{Q^\dagger A Q}_{\text{Schurov razcep}} = \underbrace{R}_{\text{Schurova forma}}.$$

Zakaj tak razcep? Če je matrika R zgornje trikotna, so njene lastne vrednosti ravno vsi diagonalni elementi. Ker je R podoben A, so to tudi lastne vrednosti A!

Potenčna iterativna metoda:

za reševanje: $A\vec{x} = \lambda \vec{x}$

Postopek:

1. Izberemo začetni približek $\vec{x}^{(0)} = (x_{01}, \dots, x_{0n})^T$ tako, da je $\|\vec{x}^{(0)}\| = \sqrt{x_{01}^2 + \dots + x_{0n}^2} = 1$.

2. Izračunamo: $\vec{y}^{(i+1)} = A \cdot \vec{x}^{(i)}$; $i = 0, 1, 2, \dots$

3. Izračunamo: $a^{(i+1)} = \|\vec{y}^{(i+1)}\|$ → To konvergira k največji lastni vrednosti

To konvergira k lastnemu vektorju ← $\vec{x}^{(i+1)} = \dfrac{\vec{y}^{(i+1)}}{a}$

Pazimo, da je vse izračunano z istim redom! ← $\lambda^{(i+1)} = \dfrac{\vec{x}^{(i+1)} \cdot \left(A\vec{x}^{(i+1)}\right)}{\|x^{(i+1)}\|^2}$

4. Ponavljamo 2. in 3. tako dolgo, dokler

$$\|A\vec{x}^{(i+1)} - \lambda^{(i+1)}\vec{x}^{(i+1)}\| < \varepsilon$$

Izjava: Zaporedje vektorjev $\vec{x}^{(i)}$ konvergira k dominantnemu lastnemu vektorju, zaporedje $\lambda^{(i)}$ pa k dominantni lastni vrednosti.

Dokaz: Predpostavimo, da lahko A diagonaliziramo kot: $A = W \Lambda W^{-1}$, kjer sta

$$\underbrace{\Lambda = \begin{bmatrix} \lambda_1 & & & 0 \\ & \ddots & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{bmatrix}}_{\text{Diagonalna matrika lastnih vrednosti}}, \quad W = \underbrace{[\vec{w}_1, \vec{w}_2, \dots \dots \vec{w}_n]}_{\text{Matrika z lastnimi vektorji!}}$$

In naj velja: $\underbrace{|\lambda_1| > |\lambda_2| > |\lambda_3| > \dots > |\lambda_n|}_{\text{Urejene so po velikosti}}$

Razvijmo naš začetni vektor kot po lastnih vektorjih:

$$\vec{x}^{(0)} = \sum_{j=1}^{n} \alpha_j \cdot \vec{w_j}$$

V i-ti iteraciji našega postopka imamo:

✓ Takkolumat smo pravilo množenja.

$$\vec{x}^{(i)} = \frac{A^i \vec{x}^{(0)}}{\| A^i \vec{x}^{(0)} \|} = \frac{(W \Lambda^i W^{-1}) \sum_{j=1}^{m} \alpha_j \vec{w}_j}{\| \dots \dots \|} =$$

Upoštevajmo: $\underline{A^i} = (W \Lambda W^{-1})^i = \underbrace{(W \Lambda \overset{= I}{W^{-1})(W} \Lambda \overset{= I}{W^{-1})} \dots \dots}_{i}$

$$= \underline{W \Lambda^i W^{-1}}$$

$\leftarrow$ Lastni vektorji se ocene skalarjev

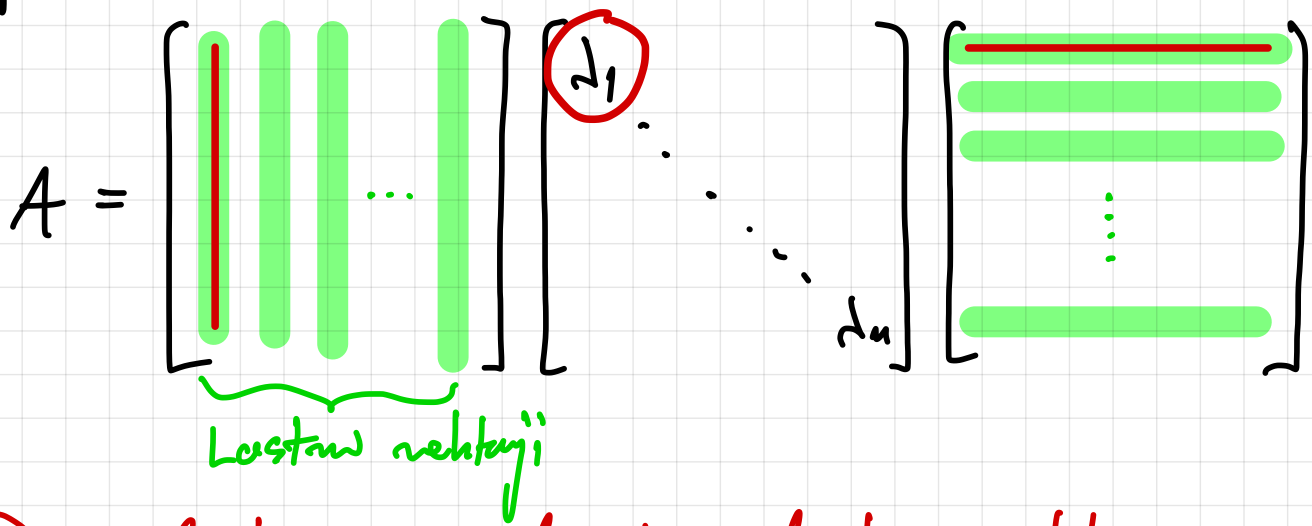
$$= \frac{\alpha_1 \lambda_1^i \vec{w}_1 + \dots + \alpha_m \lambda_m^i \vec{w}_m}{\| \alpha_1 \lambda_1^i \vec{w}_1 + \dots + \alpha_m \lambda_m^i \vec{w}_m \|} \begin{array}{l} / : \lambda_1^i \\ / : \lambda_1^i \end{array}$$

$$= \frac{\alpha_1 \vec{w}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^i \vec{w}_2 + \dots + \alpha_m \left(\frac{\lambda_m}{\lambda_1}\right)^i \vec{w}_m}{\| \alpha_1 \vec{w}_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^i \vec{w}_2 + \dots + \alpha_m \left(\frac{\lambda_m}{\lambda_1}\right)^i \vec{w}_m \|}$$

$$\xrightarrow{i \to \infty} \underline{\frac{\vec{w}_1}{\| \vec{w}_1 \|}} \quad \blacksquare$$

Kaj pa ostale lastne vrednosti?

Če A lahko diagonaliziramo, potem jo lahko zapišemo kot:



Prva lastna vrednost in lastni vektor že imamo. Ideja je, da ju odštejemo iz sistema in pogledamo s tem, kar ostane.

$$A_2 = A - \lambda_1 \cdot \underbrace{\left(\vec{x}_1 \vec{x}_1^T\right)}_{\text{Tenzorski produkt: } \vec{x}_1 \otimes \vec{x}_1} = A - \lambda_1 \begin{bmatrix} x_1^2 & x_1 x_2 & \cdots & x_1 x_n \\ x_1 x_2 & x_2^2 & \cdots & \cdots \\ & & & \\ \cdots & \cdots & \cdots & x_n^2 \end{bmatrix}$$

Za splošni korak:

$$A_i = A_{i-1} - \lambda_{(i-1)} \left(\vec{x}_{i-1} \vec{x}_{i-1}^T\right) \quad ; \quad i = 2, 3, \ldots, n$$

Hotellingova redukcija

Deluje za $A = A^T$

Inverzna potenčna metoda (ali Inverzna iteracija):

- Metoda je uporabna, kadar imamo približek za lastno vrednost in potrebujemo še lastni vektor.

- Ideja metode je reševanje sistema:

$$(A - \tau I)\vec{y} = \vec{b}$$

kjer je $\vec{b}$ naključni vektor in τ vrednost blizu iskane lastne λ_n. Tedaj je rešitev sistema $\vec{y}$ blizu lastnega vektorja $\vec{x}_n$ z lastno vrednostjo λ_n.

Algoritem:

1.) Izberemo τ blizu lastne vrednosti λ_n in normiran naključni vektor $\vec{b}_0$; $\|\vec{b}_0\| = 1$.

2.) Dobimo $\vec{y}_{i+1}$ z reševanjem sistema:

$$(A - \tau I)\vec{y}_{i+1} = \vec{b}_i \quad ; \; i = 0, 1, \dots n$$

To pomeni reševanje sistema linearnih enačb.

3.) Izračunamo nove približke:

$$\vec{b}_{i+1} = \frac{\vec{y}_{i+1}}{\|\vec{y}_{i+1}\|} \quad \longleftarrow \text{Približek lastnega vektorja}$$

$$\lambda_{i+1} = \vec{x}_{i+1}^{T} A \vec{x}_{i+1} \quad \longleftarrow \text{Približek lastne vrednosti}$$

4.) Ponavljamo 2.), 3.) dokler $\left|\lambda_{i+1} - \lambda_i\right| < \underline{\underline{\varepsilon}}$.

Predpisana natančnost.

Zakaj metoda deluje:

$$\underbrace{(A - \tau I)}_{\tilde{A}} \vec{y}_{i+1} = \vec{b}_i$$

$$\Big\Downarrow \ \text{Ekvivalentno}$$

$$\vec{y}_{i+1} = \tilde{A}^{-1} \vec{b}_i$$

Ugotovimo, da inverzna iteracija ustreza potenčni metodi za matriko $(A - \tau I)^{-1}$.

Metoda konvergira k dominantni lastni vrednosti za $\tilde{A}^{-1}$. Lastne vrednosti te matrike so

$$\tilde{\lambda}_i = \frac{1}{\lambda_i - \tau}$$

Dominantna vrednost je $\tilde{\lambda}_n$, ker je τ najbližje λ_n je $\tilde{\lambda}_n > \tilde{\lambda}_i$, $i \neq n$.

Komentar:

- S parametrom τ izbiramo, k kateremu lastnemu vektorju bo konvergiralo zaporedje $\vec{b}_{i+1}$.
- Če τ izberemo dovolj blizu λ_n, bo metoda zelo hitro konvergirala.
- V vsakem koraku moramo rešiti sistem enačb:

$$(A - \tau I)\vec{y}_{i+1} = \vec{b}_i$$

To rešimo z uporabo npr. razcepa LU ali drugih metod za reševanje linearnih sistemov enačb. Glej.: Plestenjak, RONM. poglavje 3.

Jacobijeva metoda:

Uporabna le za iskanje lastnih vrednosti in vektorjev simetričnih matrik!

- Jacobijeva metoda sestoji iz niza ortogonalnih podobnostnih transformacij — rotacij. Z vsako rotacijo poskušamo na 0 spraviti izbrani izvendiagonalni člen. Vsaka naslednja rotacija sicer "pokvari" delo prejšnje rotacije, a izven-diagonalni elementi kljub temu s številom iteracij postajajo vse manjši $\longrightarrow$ Vse dokler vsi izvendiagonalni elementi niso manjši od izbrane natančnosti ε.

V i-tem koraku imamo matriko:

$$\begin{array}{cc} & \begin{array}{ccccc} & p & & q & \end{array} \\ A^{(i)} = & \left[\begin{array}{ccccc} \ddots & & & & \\ \cdots & a_{rp} & \cdots & a_{rq} & \cdots \\ \cdots & a_{pp} & \cdots & a_{pq} & \cdots \\ & \vdots & \ddots & \vdots & \\ \cdots & a_{qp} & \cdots & a_{qq} & \cdots \\ & \vdots & & \vdots & \ddots \end{array}\right] \begin{array}{c} \\ r \\ p \\ \\ q \\ \\ \end{array} \end{array}$$

Ta matrični element želimo uničiti.

V $(i+1)$ koraku želimo uničiti element a_{pq}:

$$A^{(i+1)} = \underbrace{R_{pq}^T}_{\text{kateri element uničujemo}}(\theta)\, A^{(i)}\, R_{pq}(\theta)$$

Rotacijski kot.

$$R_{pq}(\theta) = \begin{array}{cc} \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos\theta & \cdots & -\sin\theta & & \\ & & \vdots & 1 & \vdots & & \\ & & \sin\theta & \cdots & \cos\theta & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix} & \begin{matrix} \\ \\ \leftarrow p \\ \\ \leftarrow q \\ \\ \\ \end{matrix} \\ \begin{matrix} \quad\quad\quad\quad \uparrow p & \quad\quad\quad \uparrow q \end{matrix} & \end{array}$$

c, s

Poglejmo samo dele matrik, ki jih nastavimo v 0:

$$A^{(i+1)} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a_{pp} & a_{pq} \\ a_{pq} & a_{qq} \end{bmatrix} \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$$

$$= \begin{bmatrix} c^2 a_{pp} + 2cs\,a_{pq} + s^2 a_{qq}, & \underbrace{-cs\,a_{pp} + c^2 a_{pq} - s^2 a_{pq} + cs\,a_{qq}}_{=0} \\ \underbrace{-cs\,a_{pp} + c^2 a_{pq} - s^2 a_{pq} + cs\,a_{qq}}_{=0}, & s^2 a_{pp} - 2sc\,a_{pq} + c^2 a_{qq} \end{bmatrix}$$

$$a_{pq}^{(i+1)} = -cs\,a_{pp} + c^2 a_{pq} - s^2 a_{pq} + sc\,a_{qq} =$$

$$= a_{pq}\cos 2\theta - \frac{1}{2}\sin 2\theta\,(a_{pp} - a_{qq}) = 0$$

Dobimo: $$\operatorname{ctg} 2\theta = \frac{a_{pp} - a_{qq}}{2\,a_{pq}} = \xi$$

Iz te enačbe sedaj določimo rotacijski kot θ. Da bo še lažje, zapišemo:

$$\operatorname{ctg} 2\theta = \frac{\cos^2\theta - \sin^2\theta}{2\sin\theta\cos\theta} = \frac{1 - \underbrace{\operatorname{tg}^2\theta}_{=t^2}}{2\underbrace{\operatorname{tg}\theta}_{=t}} = \xi$$

Dobimo kvadratno enačbo, ki jo rešimo:

$$t^2 + 2t\xi - 1 = 0$$

$$t = \frac{\operatorname{sgn}(\xi)}{|\xi| + \sqrt{\xi^2 + 1}}$$

$$c = \frac{1}{\sqrt{t^2 + 1}}, \qquad s = t \cdot c$$

Členi matrike v naslednji iteraciji bodo:

$$a_{pq}^{(i+1)} = a_{qp}^{(i+1)} = 0$$

$$a_{pp}^{(i+1)} = c^2 a_{pp} + s^2 a_{qq} + 2cs\, a_{pq}$$

$$a_{qq}^{(i+1)} = c^2 a_{qq} + s^2 a_{pp} - 2sc\, a_{pq}$$

Spremenijo pa se tudi drugi elementi v matriki a_{rp} in a_{rq}, kjer je $r \neq p$ in $r \neq q$.

$$a_{rp}^{(i+1)} = a_{rp}\, c + a_{rq}\, s$$

$$a_{rq}^{(i+1)} = a_{rq}\, c - a_{rp}\, s$$

Algoritem želimo oblikovati tako, da ne bomo matrike v celoti na novo računali, pač pa da bomo obstoječim elementom dodajali popravke.

$$a_{rp}^{(i+1)} = a_{rp}\, c + a_{rq}\, s = a_{rp}^{(i)} - a_{rp} \cdot (1-c) + a_{rq}\, s$$

$$= a_{rp} + s\left(a_{rq} - a_{rp}\left(\frac{1-c}{s}\right)\right)$$

$$\frac{1-c}{s} = \frac{1-\cos\theta}{\sin\theta} = \frac{\sin^2\frac{\theta}{2} + \cos^2\frac{\theta}{2} - \cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}}{2\sin\frac{\theta}{2}\cos\frac{\theta}{2}} =$$

$$= \operatorname{tg}\frac{\theta}{2} = \tau$$

Dakle:

$$a_{rp}^{(i+1)} = a_{rp} + s\left(a_{rq} - a_{rp}\tau\right)$$

$$a_{rq}^{(i+1)} = a_{rq} - s\left(a_{rp} + a_{rq}\tau\right) \longleftarrow$$

Izgleda mi da su obrnuti znakovi!

$$a_{pp}^{(i+1)} = c^2 a_{pp} + s^2 a_{qq} + 2cs\, a_{pq} =$$

$$= a_{pp} - (\underbrace{1-c^2}_{s^2})\, a_{pp} + s^2 a_{qq} + 2cs\, a_{pq}$$

$$= a_{pp} - s^2\Big[\underbrace{(a_{pp} - a_{qq})}_{\cancel{2} a_{pq}\cdot\frac{1-t^2}{\cancel{2}t}}\Big] + 2cs\, a_{pq}$$

$$= a_{pp} + a_{pq}\cdot\left(2\underset{=\frac{s^2}{t}}{cs} - \frac{s^2(1-t^2)}{t}\right) =$$

$$= a_{pp} + a_{pq}\, s^2\left(\frac{2-1+t^2}{t}\right) = \quad \Bigg| \quad \frac{1+t^2}{t} = \frac{\cancel{c}}{s\,c^{\cancel{2}}}$$

$$= a_{pp} + a_{pq}\cdot t$$

Ekvivalentno dobimo še:

$$a_{qq}^{(i+1)} = a_{qq} - (1-c^2)a_{qq} + s^2 a_{pp} - 2sc\, a_{pq} =$$
$$= a_{qq} + s^2(a_{pp} - a_{qq}) - 2sc\, a_{pq} =$$
$$= a_{qq} - a_{pq} t$$

<u>Kako se odločimo, katere elemente bomo uničevali:</u>

- Klasično: V vsakem koraku uničimo največji člen.
  Najhitrejša konvergenca, a zahteva veliko primerjanja
- Ciklična varianta: redno v enakem vrstnem redu
- Pragovna varianta: v danem vrstnem redu, a uničujemo le tiste elemente, ki imajo vrednosti nad pragom. Prag v vsakem ciklu zmanjšamo.

Algoritem:

1. Poiščemo največji izvendiagonalni element. $a_{p,q}$.
2. Izračunamo rotacijski kot θ oziroma s, c in t.
3. Izračunamo popravke k matriki A in z njimi popravimo matriko
4. Ponavljamo 1. – 3. toliko časa, dokler

$$\mathrm{off}(A) = \sum_{\substack{i,j=1 \\ i \neq j}}^{n} |a_{ij}|^2 < \varepsilon$$

Lastne vrednosti in lastni vektorji:

Po n iteracijah imamo:

Produkt ortogonalnih matrik je ortogonalna matrika.

$$\underline{A^{(n)}} = \underbrace{J_{(n-1)}^T J_{(n-2)}^T \cdots J_{(1)}^T}_{J^T} A \underbrace{\overbrace{J_{(1)} J_{(2)} \cdots J_{(n-1)}}}_{\underline{J}}$$

diagonalni elementi so lastne vrednosti

Stolpci matrike so lastni vektorji matrike!

Summary:

- Časovna zahtevnost algoritma je $O(n^3)$ z veliko konstanto.
- Uporaben za $N \leq 10$.
- Metoda je robustna.
- Dobro računa tudi majhne lastne vrednosti.

Algoritem QR:

- To je metoda, ki se pogosto uporablja v numeričnih paketih
- Simetrična matrika ni pogoj za delovanje algoritma.
- Metoda temelji na realnem Schurovem razcepu matrike A na ortogonalno matriko Q in zgornje trikotno matriko R.

Algoritem:

1. Določimo prvi približek $A_0 = A$
2. Izvedemo QR razcep matrike A_i ; $i = 0, 1, 2 \dots n-1$

$$A_{(i)} = Q_{(i)} R_{(i)}$$

3. Izračunamo naslednjo iteracijo matrike:

$$A_{(i+1)} = R_{(i)} Q_{(i)}$$

4. Koraka 2. in 3. ponavljamo toliko časa, dokler matrika $A_{(i)}$ ni dovolj podobna zgornje diagonalni:

$$\sum_{\substack{i,j=1 \\ j<i}}^{n} |a_{ij}|^2 < \underline{\varepsilon}$$

izberemo!

Zakaj algoritem deluje:

$$A^{(i+1)} = R^{(i)} Q^{(i)}$$

Vemo, da je $A^{(i)} = Q^{(i)} R^{(i)}$; ker je $Q^T = Q^{-1}$ velja tudi: $Q_{(i)}^T A^{(i)} = R^{(i)}$

$$A^{(i+1)} = Q_{(i)}^T A^{(i)} Q_{(i)}$$

Dobili smo znova podobnostno transformacijo. Ker je Q ortogonalna, je algoritem stabilen.

Velja: V limiti $i \to \infty$ matrika $A^{(i)}$ konvergira proti zgornje trikotni matriki. Lastne vrednosti ležijo na diagonalnih elementih matrike. Lastni vektorji so stolpci matrike

$$Q = Q_1 Q_2 \cdots\cdots Q_n$$

Razcep QR: (z uporabo Householderjevih zrcaljenj)

Izvesti želimo razcep: $\underbrace{Q}_{\text{Ortogonalna}}\ \underbrace{R}_{\text{Zgornje trikotna.}}$

$$A = QR$$

Cilj transformacije je najti linearno preslikavo, ki izbrani vektor $\vec{x}$ preslika v vektor z isto dolžino, a vzdolž izbranega baznega vektorja $\hat{e}_1$.

Naj bo $\vec{x}$ prvi stolpec matrike A in $\hat{e}_1 = (1, 0, 0, \dots 0)^T$ bazni vektor dimenzije n.
Vpeljemo:

$$\vec{u} = \vec{x} - \|\vec{x}\|\,\hat{e}_1$$

$$\vec{v} = \frac{\vec{u}}{\|\vec{u}\|}$$

Transformacija: $Q_1 = I - 2\vec{v}\,\vec{v}^T = P$

Matriko spremeni tako, da v 1. stolpcu matrike ostane le 1. vrednost. Ostale so nič!

Transformacijo Q_1 rečemo Householderjeva prezrcaljalna matrika.

Prezrcaljalna matrika ima naslednje lastnosti:

$$P^2 = (I - 2\vec{v}\vec{v}^T)(I - 2\vec{v}\vec{v}^T) =$$

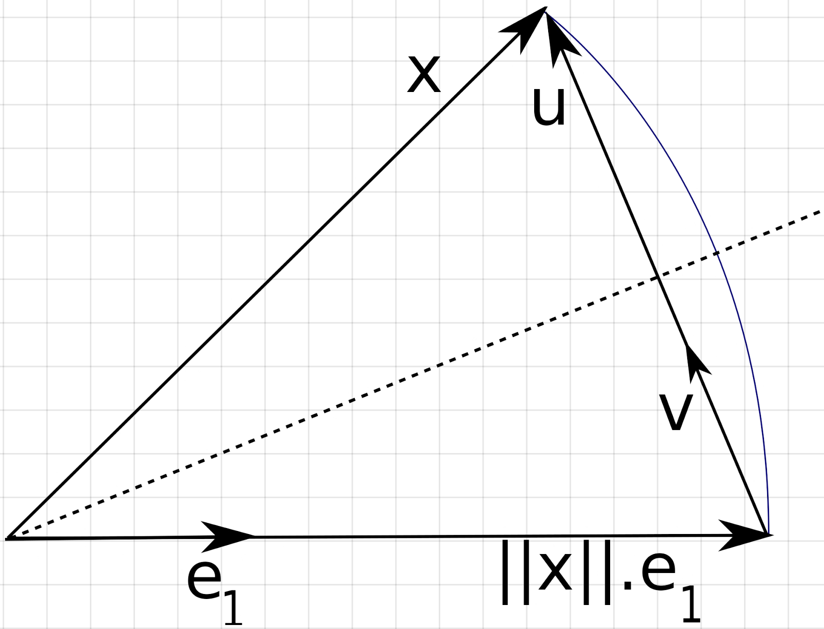
$$= I - 4\vec{v}\vec{v}^T + 4\vec{v}\underbrace{(\vec{v}^T\vec{v})}_{\substack{=1 \\ \text{ker normiran.}}}\vec{v}^T$$

$$= I - 4\vec{v}\vec{v}^T + 4\vec{v}\vec{v}^T = I = P \cdot P$$

Od tod še sledi: $P^{-1} = P$, ker velja $P^T = P$ je potem tudi $P^{-1} = P^T$.

$$P\vec{x} = (I - 2\vec{v}\vec{v}^T)\vec{x} = \vec{x} - 2\vec{v}\,\frac{(\vec{x}^T - \|\vec{x}\|\,\hat{e}_1^T)\,\vec{x}}{\|\vec{x} - \|\vec{x}\|\,\hat{e}_1\|}$$

$$= \vec{x} - 2\vec{v}\,\frac{(|x|^2 - |x| \cdot x_1)}{\underbrace{|\vec{x} - \|\vec{x}\|\,\hat{e}_1|^2}_{2|x|^2 - 2|x|x_1}} = \vec{x} - \vec{u} = \underline{\underline{\|x\|\,\hat{e}_1}}$$



Transformacija $Q_1 = P$ zrcali vektor $\vec{x}$ preko simetrale!

Kaj se zgodi z matriko:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & & a_{2n} \\ a_{31} & a_{32} & & a_{3n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & & a_{nn} \end{bmatrix}$$

Vzamemo 1. stolpec in za njega naredimo $\vec{x} \to Q_1$

$$Q_1 A = \begin{bmatrix} a_{11}' & a_{12}' & \cdots & a_{1n}' \\ 0 & a_{22}' & & a_{2n}' \\ 0 & a_{32}' & A' & a_{3n}' \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}' & & a_{nn}' \end{bmatrix}$$

Za matriko A' dimenzije $(N-1)\times(N-1)$ zopet določimo vektor $\vec{x}'$ dimenzije $(N-1)$, bazni vektor $\hat{e}_1' = (\underbrace{1, 0 \ldots\ldots 0}_{N-1})^T$ in določimo Q_2'

Celotna transformacijska matrika je potem:

$$Q_2 = \left[\begin{array}{c|ccc} 1 & 0 & 0 & \cdots \\ \hline 0 & & & \\ 0 & & \boxed{\quad Q_2' \quad} & \\ \vdots & & & \\ 0 & & & \end{array}\right]$$

$$Q_2 Q_1 A = \left[\begin{array}{cc|ccc} a_{11}' & a_{12}' & \cdots & \cdots & a_{1N}' \\ 0 & a_{22}'' & \cdots & \cdots & a_{2N}'' \\ \hline 0 & 0 & & & \\ \vdots & \vdots & & \boxed{\quad A'' \quad} & \\ 0 & 0 & & & \end{array}\right]$$

In potem tako naprej

Na koncu dobimo:

$$R = \underbrace{Q_{n-1} \cdots Q_2 Q_1}_{Q^T} A = Q^T A$$

$$QR = A \qquad \blacksquare$$

Opomba: QR razcep se da narediti tudi z drugačnimi transformacijami, npr. z Givensovo metodo.

Glej: NRC, str 469.
Plestenjak, Uvod v N. M., str. 136.

Summary metode QR:

- Če je matrika A simetrična matrika velikosti $n \times n$, potem je matrika R diagonalna matrika.

- Časovna zahtevnost algoritma je $O(n^3)$

- Osnovni algoritem se da pohitriti tako, da se matriko A najprej preoblikuje v tridiagonalno matriko A' (z uporabo npr. Householderjevih zrcaljenj), ali pa v Hessenbergovo obliko (zgornje trikotna matrika + ena obdiagonala)

  Glej, Plestenjak, Uvod v N.M., str. 183 in NRC, str. 470.