University of Ljubljana
Faculty of Mathematics and Physics

3. Naloga: Diagonalizacija matrik, lastne vrednosti in vektorji

Ma-Fi Praktikum 2020/21

Lastne vrednosti in vektorji simetrične matrike

- Že mnogokrat viden problem pri matematiki, v tej nalogi vezan na kvantno mehaniko (o tem več kasneje).
- Ne gre pozabiti, da praktično vedno iščemo lastne vrednosti **IN** lastne vektorje.
- Uporaba determinante (karakterističnega polinoma) dobra kvečjemu do dim=4.

$$\mathbf{A} \cdot \mathbf{x} = \lambda \mathbf{x} \qquad\qquad \det |\mathbf{A} - \lambda \mathbf{I}| = 0$$

- Za **simetrično matriko** (z nedegen. LV) so lastne vrednosti **realne**, diagonalizira se jo z ortogonalno matriko **Z ($\mathbf{Z}^{-1} = \mathbf{Z}^{T}$).**
- V **preprostih metodah** iščemo postopek, kjer **po vrsti** dobimo lastne vektorje **v,** in jih na koncu zložimo v stolpcih skupaj v to matriko **Z.**

$$\mathbf{Z} = \left[\begin{array}{c|c|c|c} & & & \\ \mathbf{v_1} & \mathbf{v_1} & \cdots & \mathbf{v_N} \\ & & & \end{array} \right]$$

Potenčna iterativna metoda

- **Najpreprostejši** pristop, v praksi 'še kar' deluje, je pa treba preveriti, če so rezultati smiselni…
  - Izberemo začetni $\mathbf{x}^{(0)}$ približek za lastni vektor (če ne vemo ničesar, je pač naključni nastavek).
  - Za začetek ta vektor normiramo in ga nato vključimo v iterativno proceduro:

$$\boldsymbol{y}^{(i+1)} = \underline{A}\boldsymbol{x}^{(i)}, \quad \boldsymbol{x}^{(i+1)} = \frac{1}{a}\boldsymbol{y}^{(i+1)}$$

- Ko se začne v postopku vektor **x** 'ponavljati' (se malo spreminja), je **a** dober približek za lastno vrednost, **x** pa za ustrezni lastni vektor.
  - Včasih imamo smolo z izbiro $\mathbf{x}^{(0)}$**,** (dobimo npr. rezultat nič), zato ga zamenjamo in ponovimo…
  - Podobno lahko poiščemo tudi najmanjšo lastno vrednost, ki je enaka največji lastni vrednosti inverzne matrike **A.**
- Postopek zaustavimo, ko je **lastna vrednost stabilna znotraj neke natančnosti**, s formulo:

$$\left\| \underline{A}\,\mathbf{x}^{(k)} - \rho_k\,\mathbf{x}^{(k)} \right\| \le \varepsilon, \quad \rho_k = \frac{\mathbf{x}^{(k)} \cdot (\underline{A}\,\mathbf{x}^{(k)})}{\left\| \mathbf{x}^{(k)} \right\|^2}$$

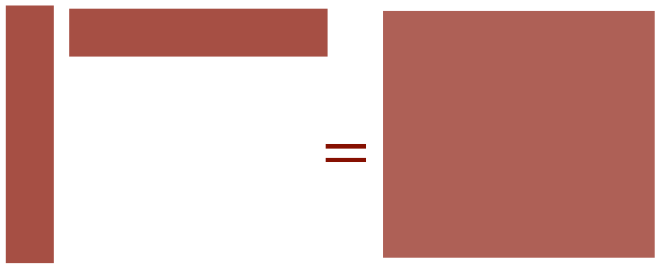
Potenčna iterativna metoda

- **Najpreprostejši** pristop, v praksi 'še kar' deluje, je pa treba preveriti, če so rezultati smiselni…
  - Izberemo začetni $\mathbf{x}^{(0)}$ približek za lastni vektor (če ne vemo ničesar, je pač naključni nastavek).
  - Za začetek ta vektor normiramo in ga nato vključimo v iterativno proceduro:

$$\boldsymbol{y}_{(i+1)} = \underline{A}\boldsymbol{x}_{(i)}, \quad \boldsymbol{x}_{(i+1)} = \frac{1}{a}\boldsymbol{y}_{(i+1)}$$

- Ko se začne v postopku vektor **x** 'ponavljati' (se malo spreminja), je $a \sim \lambda$ dober približek za lastno vrednost, **x** pa za ustrezni lastni vektor.
  - Včasih imamo smolo z izbiro $\mathbf{x}^{(0)}$, (dobimo rezultat nič), zato ga zamenjamo in ponovimo…
  - Podobno lahko poiščemo tudi najmanjšo lastno vrednost, ki je enaka največji lastni vrednosti inverzne matrike **A.**
- Ko imamo dober približek lastnega vektorja in vrednosti, **jo odstranimo iz računa z Wielandt/Hotellingovo redukcijo** (iz lastnega vektorja naredimo matriko = vnanji produkt!) in ponovimo postopek….

$$\underline{A}' = \underline{A} - \lambda_i\, \boldsymbol{x}_{(i)} \otimes \boldsymbol{x}_{(i)} = \underline{A} - \lambda_i\, \boldsymbol{x}_{(i)}\boldsymbol{x}_{(i)}^\top$$



Vnanji produkt: $\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{a}\boldsymbol{b}^\top$

Naprednejši pristop

- Za **simetrično matriko** (z nedegen. LV) so lastne vrednosti **realne**, diagonalizira se jo z **ortogonalno matriko Z ($\mathbf{Z}^{-1} = \mathbf{Z}^{T}$)**:

$$\mathbf{A} \cdot \mathbf{x} = \lambda \mathbf{x} \qquad \det |\mathbf{A} - \lambda \mathbf{I}| = 0$$

- **Rešimo raje sistem**, kjer v matriki **Z** **hkrati** dobimo (ortogonalne, normirane) lastne vektorje **v**:

$$\mathbf{A} \cdot \mathbf{Z} = \mathbf{Z} \cdot \mathrm{diag}\left(\lambda_1 \ldots \lambda_N\right)$$

$$\mathbf{Z}^{-1} \cdot \mathbf{A} \cdot \mathbf{Z} = \mathrm{diag}\left(\lambda_1 \ldots \lambda_N\right)$$

$$\mathbf{Z} = \left[\begin{array}{c|c|c|c} \mathbf{v_1} & \mathbf{v_1} & \cdots & \mathbf{v_N} \end{array} \right]$$

vsak stolpec je normiran lastni vektor!

Sledi numerično pomembno dejstvo: elementi Z nikoli niso veliki!

Konstrukcija transformacije Z

$$\mathbf{A} \quad \rightarrow \quad \mathbf{Z}^T \cdot \mathbf{A} \cdot \mathbf{Z}$$

- **Celoten problem se torej prevede na iskanje ustrezne matrike Z.**
  - Ker gre za numerične metode, jo poskušamo skonstruirati z zaporednimi približki, dokler matrika ni **'dovolj' diagonalna (v okviru željene natančnosti...).**
- Imamo torej zaporedje ortogonalnih transformacij:

$$\begin{aligned} \mathbf{A} \quad &\rightarrow \quad \mathbf{P}_1^{-1} \cdot \mathbf{A} \cdot \mathbf{P}_1 \quad \rightarrow \quad \mathbf{P}_2^{-1} \cdot \mathbf{P}_1^{-1} \cdot \mathbf{A} \cdot \mathbf{P}_1 \cdot \mathbf{P}_2 \\ &\rightarrow \quad \mathbf{P}_3^{-1} \cdot \mathbf{P}_2^{-1} \cdot \mathbf{P}_1^{-1} \cdot \mathbf{A} \cdot \mathbf{P}_1 \cdot \mathbf{P}_2 \cdot \mathbf{P}_3 \quad \rightarrow \quad \text{etc.} \end{aligned}$$

$$\mathbf{Z} = \mathbf{P}_1 \cdot \mathbf{P}_2 \cdot \mathbf{P}_3 \cdot \ldots$$

- To se torej prevede na smiselno izbiro matrik $\mathbf{P}_j$.
  - Nekaj tipičnih pristopov: QR,Jacobi, Givens, Householder…

QR iterativna metoda (ponovitev..)

- V principu enostavna metoda, ki pa se izkaze, da je zelo robustna/ učinkovita, verjetno tudi **najboljša numerična metoda za izračun vseh lastnih vrednosti splošne nesimetrične matrike.**
- Postopek je dovolj preprost v osnovi (o praktični izvedbi kasneje…)

$$
\begin{aligned}
A_0 &= A \\
A_k &= Q_k R_k (\text{ QR razcep }) \\
A_{k+1} &= R_k Q_k \\
k &= 0, 1, \ldots
\end{aligned}
$$

Schurova faktorizacija:
A = Z T Z^{-1}
(T trikotna…)

QR razcep = razcep na ortogonalno matriko Q in (zg.) trikotno matriko R

- Ta postopek naj bi konvergiral k trikotni matriki (za simetrične matrike k diagonalni … razen v patoloških primerih).
- V praktičnih izvedbah pa ga tipično nekoliko popestrijo za hitrejšo konvergenco, vpeljejo premike in najprej spravijo na tridiagonalno obliko…

$$A_{k+1} = R_k Q_k = Q_k^{-1} Q_k R_k Q_k = Q_k^{-1} A_k Q_k = Q_k^\top A_k Q_k$$

$$\mathbf{Z} = \mathbf{Q}_1 \cdot \mathbf{Q}_2 \cdot \mathbf{Q}_3 \cdot \ldots$$

Konstruiramo končno Z

QR iterativna metoda (ponovitev..)

- Ta postopek naj bi konvergiral k trikotni matriki (za simetrične matrike k diagonalni … razen v patoloških primerih).

Patološki primer

Example 3 (The QR iteration spinning its wheels) Let $D = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ and let $V = \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix}$, and let

$$A = VDV^{-1} = \begin{bmatrix} 5 & -4 \\ 6 & -5 \end{bmatrix} .$$

Now let's run the QR iteration we find

$$A^{(1)} = \frac{1}{61} \begin{bmatrix} 5 & 616 \\ 6 & -5 \end{bmatrix} .$$

At the next stage, something remarkable happens:

$$A^{(2)} = \begin{bmatrix} 5 & -4 \\ 6 & -5 \end{bmatrix} .$$

We are back to where we started! It is clear that for this matrix, the QR iteration will not ever converge. Instead, it will just cycle back and forth between the two matrices $\begin{bmatrix} 5 & -4 \\ 6 & -5 \end{bmatrix}$ and $\frac{1}{61}\begin{bmatrix} 5 & 616 \\ 6 & -5 \end{bmatrix}$.

Why did this happen? As we shall see, it happened because the eigenvalues of A, 1 and -1, have the same absolute value.

Jacobijeva metoda

- Najpreprostejši pristop je kar z rotacijami, a zelo učinkovit…
  - Arbitrarna rotacija okrog neke osi v N dimenzijah:

$$\mathbf{P}_{pq} = \begin{bmatrix} 1 & & & & & & \\ & \cdots & & & & & \\ & & c & \cdots & s & & \\ & & \vdots & 1 & \vdots & & \\ & & -s & \cdots & c & & \\ & & & & & \cdots & \\ & & & & & & 1 \end{bmatrix}$$

$$c = \cos(\varphi)$$
$$c = \sin(\varphi)$$

Razen v vrsticah in stolpcih p,q elementi enaki 0!

- To nam da transformacijo matrike A v vrsticah in stolpcih p in q.
  - **Ostale vrednosti v novi matriki A' nespremenjene!**

$$\mathbf{A}' = \mathbf{P}^T_{pq} \cdot \mathbf{A} \cdot \mathbf{P}_{pq}$$

$$\mathbf{A}' = \begin{bmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{bmatrix}$$

$$\begin{aligned} a'_{rp} &= c a_{rp} - s a_{rq} \\ a'_{rq} &= c a_{rq} + s a_{rp} \\ a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2sc a_{pq} \\ a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2sc a_{pq} \\ a'_{pq} &= (c^2 - s^2) a_{pq} + sc(a_{pp} - a_{qq}) \end{aligned}$$

$$r \neq p,\ r \neq q$$

Jacobijeva metoda

- V Jacobijevi metodi postavimo člene a'_{pq} na nič z izbiro kota:

$$\theta \equiv \cot(2\varphi) \equiv \frac{c^2 - s^2}{2sc} = \frac{a_{qq} - a_{pp}}{2a_{pq}}$$

$$\begin{aligned} a'_{rp} &= c a_{rp} - s a_{rq} \qquad r \neq p,\ r \neq q \\ a'_{rq} &= c a_{rq} + s a_{rp} \\ a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2sc a_{pq} \\ a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2sc a_{pq} \\ a'_{pq} &= (c^2 - s^2) a_{pq} + sc(a_{pp} - a_{qq}) \end{aligned}$$

- Rešujemo enačbo:

$$t = \frac{s}{c}; \qquad t^2 + 2t\theta - 1 = 0$$

- Z rešitvijo (ekvivalent običajnega zapisa kvadratne enačbe!):

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

$$c = \frac{1}{\sqrt{t^2 + 1}}$$

$$s = tc$$

Numerično stabilnejši recept!

$$\mathbf{A}' = \begin{bmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{bmatrix}$$

Jacobijeva metoda

- **Dosegli smo svoj cilj, a kaj nam to pomaga, če naslednji korak vse spet poruši?**

$$a'_{pq} = 0$$

$$\mathbf{A}' = \begin{bmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{bmatrix}$$

$$\begin{aligned} a'_{rp} &= c a_{rp} - s a_{rq} \\ a'_{rq} &= c a_{rq} + s a_{rp} \\ a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2 s c a_{pq} \\ a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2 s c a_{pq} \\ a'_{pq} &= (c^2 - s^2) a_{pq} + s c (a_{pp} - a_{qq}) \end{aligned}$$

$$r \neq p,\ r \neq q$$

- **Metoda konvergira**, ker se **vsota izvendiag. členov monotono manjša!**
  - Vsota diag. členov pa ustrezno veča - gre pač za ortogonalno transformacijo , ki ohranja normo!

$$S = \sum_{r \neq s} |a_{rs}|^2 \longrightarrow S' = S - 2\,|a_{pq}|^2$$

- Jacobijeva metoda tako definira **sweep:** vsi indeksi tečejo po zgornjih trikotnih vrednostih (pq= 12,..,1N, 23, 24…2N,).
  - Sweep ponavljamo do želene natančnosti (recimo velikost S zgoraj!)

Givensova metoda

- V Givensovi metodi postavimo na nič člen matrike A', ki **NI v ogljiščih *p,q*.** Postavimo *a'*$_{rq}$ na nič z ustrezno izbiro kota:

$$a'_{rq} = 0, \quad r \neq p, \quad r \neq q$$

$$\mathbf{A}' = \begin{bmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \vdots & & \vdots & & \vdots & & \vdots \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{bmatrix}$$

$$\begin{aligned} a'_{rp} &= c a_{rp} - s a_{rq} \\ a'_{rq} &= c a_{rq} + s a_{rp} \end{aligned} \qquad r \neq p,\ r \neq q$$

$$\begin{aligned} a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2sc a_{pq} \\ a'_{qq} &= s^2 a_{pp} + c^2 a_{qq} + 2sc a_{pq} \\ a'_{pq} &= (c^2 - s^2) a_{pq} + sc(a_{pp} - a_{qq}) \end{aligned}$$

- Izkaže se, da lahko z ustrezno sekvenco transformacij spravimo matriko **A** v **tridiagonalno obliko s končnim številom transformacij!**

$$\mathbf{P}_{23}, \mathbf{P}_{24}, \ldots, \mathbf{P}_{2n}; \mathbf{P}_{34}, \ldots, \mathbf{P}_{3n}; \ldots; \mathbf{P}_{n-1,n}$$

$$\boldsymbol{P}_{jk} \quad \text{izniči} \quad a_{k,j-1}$$

$$A'_{\text{tridiag}} = \begin{bmatrix} \beta_1 & \gamma_1 & & & & \\ \alpha_2 & \beta_2 & \gamma_2 & & & \\ & & \vdots & & & \\ & & & \alpha_{n-1} & \beta_{n-1} & \gamma_{n-1} \\ & & & & \alpha_n & \beta_n \end{bmatrix}$$

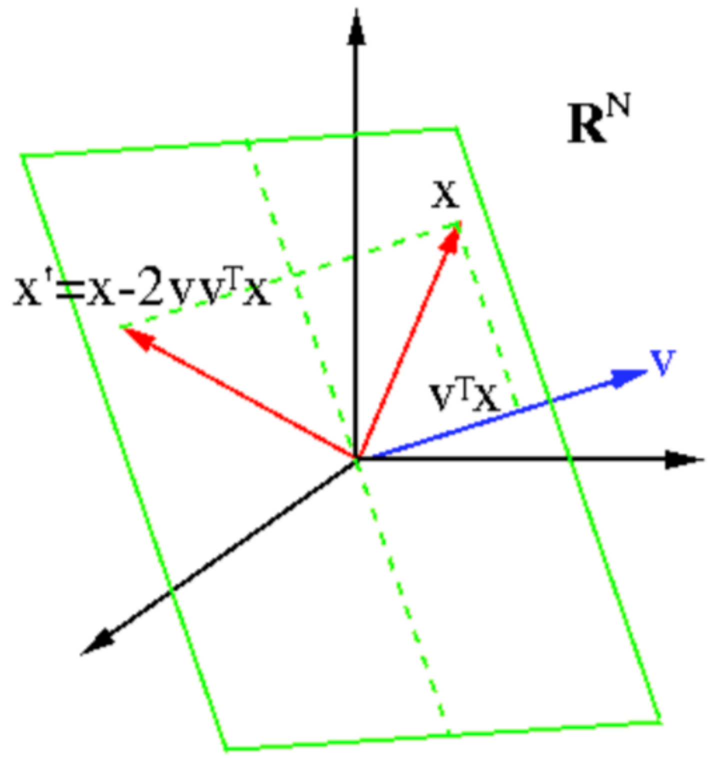
Householderjeve metode

- Preden se lotimo tridiagonalne matrike, se spoznajmo se z enim drugačnim pristopom na poti k diagonalizaciji, to je z družino Householderjevih metod.
  - **Na kratko, namesto rotacij uporabljajo zrcaljenja = projekcije!**

Vnanji produkt: $\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{a}\boldsymbol{b}^\top$

$$\mathbf{x}' = \mathbf{x} - 2\frac{\mathbf{u}}{\|\mathbf{u}\|}\frac{\mathbf{u}^T}{\|\mathbf{u}\|}\mathbf{x} = \mathbf{x} - 2\mathbf{v}\mathbf{v}^T\mathbf{x} = \left(\mathbf{I} - 2\mathbf{v}\mathbf{v}^T\right)\mathbf{x} = \mathbf{P}\mathbf{x}$$

Skalarni produkt: $\boldsymbol{a} \cdot \boldsymbol{b} = \boldsymbol{a}^\top\boldsymbol{b}$



- projekcija P čez ravnino, definirano z (normirano) normalo **v!**

$$\mathbf{P} = \mathbf{P}^{-1} = \mathbf{P}^T$$

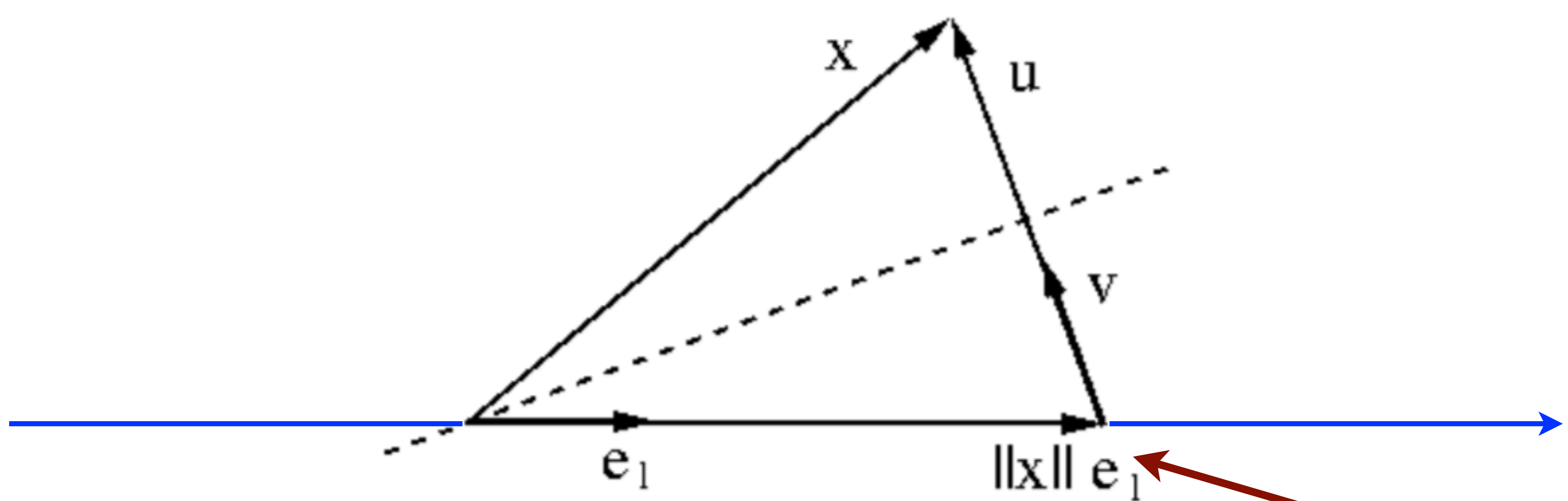
$$\mathbf{P} \cdot \mathbf{P} = \mathbf{I}$$

Značilna lastnost projekcij…

Householderjeve metode

- Izberemo si posebno normalo tako, da sprojeciramo vektor na določeno os!
    - **V enem koraku pospravimo vse ostale komponente!**

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



x dobi komponente ($\|x\|$,0,0,..)

- Če to apliciramo na matrike, je to izhodišče za **QR dekompozicijo ali transformacijo v tridiag. formo!**

Householderjeve metode

- Začnimo z zgledom za QR dekompozicijo:
  - **Q** je ortogonalna matrika, **R** pa zgornje trikotna.
  - **Kot že rečeno, v bistvu iterativna metoda:**

Zanimivost: **Jordanova forma ni numerično stabilna (ni uporabna..)**

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{R}$$

$$\mathbf{A} \longrightarrow \mathbf{A}' = \mathbf{R} \cdot \mathbf{Q} = \mathbf{Q}^{-1} \cdot \mathbf{A} \cdot \mathbf{Q}$$

- S ponavljanjem te transformacije **A** konvergira k zgornje trikotni matriki (Schurova forma): lastne vrednosti na diagonali, matrika **Z** konstruirana z zaporedjem transformacij $\mathbf{Q_j}$.
- **Uporabimo Householderjevo projekcijo na A (obstaja tudi Givensov pristop!):**

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ a_{31} & a_{32} & \cdots & a_{3N} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N1} & a_{32} & \cdots & a_{NN} \end{bmatrix} = \left[\begin{array}{c|c|c|c} \mathbf{c}_1 & \mathbf{c}_2 & \cdots & \mathbf{c}_N \end{array} \right]$$

obravnavajmo stolpce…

Householderjeve metode

- **Uporabimo Householderjevo projekcijo na A:**

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ a_{31} & a_{32} & \cdots & a_{3N} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} = \left[\begin{array}{c|c|c|c} \boxed{\mathbf{c}_1} & \mathbf{c}_2 & \cdots & \mathbf{c}_N \end{array} \right]$$

- Vpeljemo projekcijski vektor za prvi stopec, ki nam pusti samo prvo vrednost:

$$\mathbf{u} = \mathbf{c}_1 - \left\|\mathbf{c}_1\right\| \mathbf{e}_1 \qquad \mathbf{Q}_1 = \mathbf{P} = \mathbf{I} - 2\frac{\mathbf{u}\mathbf{u}^T}{\|\mathbf{u}\|^2} = \mathbf{I} - 2\frac{\mathbf{c}_1\mathbf{c}_1^T}{\|\mathbf{c}_1\|^2}$$

$$\mathrm{Q}_1\mathrm{A} = \left[\begin{array}{c|ccc} a'_{11} & a'_{12} & \cdots & a'_{1N} \\ \hline 0 & & & \\ \vdots & & \boxed{\mathrm{A}'} & \\ 0 & & & \end{array} \right]$$

Householderjeve metode

- Potem uporabimo Householderjevo projekcijo na A’, ki je za ima dimenzijo N-1 !

$$\mathbf{A}' = \begin{bmatrix} a'_{11} & a'_{12} & \cdots & a'_{1(N-1)} \\ a'_{21} & a'_{22} & \cdots & a'_{2(N-1)} \\ a'_{31} & a'_{32} & \cdots & a'_{3(N-1)} \\ \vdots & \vdots & \cdots & \vdots \\ a'_{(N-1)1} & a'_{(N-1)2} & \cdots & a'_{(N-1)(N-1)} \end{bmatrix} = \left[\boxed{\mathbf{c}'_1} \,\middle|\, \mathbf{c}'_2 \,\middle|\, \cdots \,\middle|\, \mathbf{c}'_{(N-1)} \right]$$

- Spet projekcijski vektor za prvi stopec in transformacijo:

$$\mathbf{Q}'_2 = \mathbf{I}_{(N-1)} - 2\frac{\mathbf{c}'_1 \mathbf{c}'^T_1}{\|\mathbf{c}'_1\|^2} \qquad \mathrm{Q}_2 = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \boxed{\mathrm{Q}'_2} & \\ 0 & & & \end{array}\right]$$

$$\mathrm{Q}_2\mathrm{A}' = \mathrm{Q}_2\mathrm{Q}_1\mathrm{A} = \left[\begin{array}{cc|ccc} a'_{11} & a'_{12} & a'_{13} & \cdots & a'_{1N} \\ 0 & a''_{22} & a''_{23} & \cdots & a''_{2N} \\ \hline 0 & 0 & & & \\ \vdots & \vdots & & \boxed{\mathrm{A}''} & \\ 0 & 0 & & & \end{array}\right]$$

- … in tako naprej …

$$\mathbf{R} = \mathbf{Q}_{N-1} \cdots \mathbf{Q}_1 \mathbf{A}$$

Householderjeve metode

- Zdaj pa se tridiagonalizacija:
  - **A** je spet N x N (simetricna) matrika. Vpeljemo Householderjevo projekcijo:

$$
\mathrm{P}_1 = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \mathrm{P}^{(N-1)} & \\ 0 & & & \end{array}\right] \quad \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ a_{31} & a_{32} & \cdots & a_{3N} \\ \vdots & \vdots & \cdots & \vdots \\ a_{N1} & a_{32} & \cdots & a_{NN} \end{bmatrix}
$$

$$
\mathbf{P}^{(N-1)} = \mathbf{I}_{(N-1)} - 2\frac{\mathbf{d}_1 \mathbf{d}_1^T}{\|\mathbf{d}_1\|^2}
$$

$\mathbf{d_1}$

takoj spustimo eno vrednost!
(gremo v N-1 dim.)

$$
\mathbf{P}_1\mathbf{A} = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbf{P}^{(N-1)} & \\ 0 & & & \end{array}\right] \cdot \left[\begin{array}{c|ccc} a_{11} & a_{12} & \cdots & a_{1N} \\ \hline a_{21} & & & \\ \vdots & & \mathbf{A}^{(N-1)} & \\ a_{N1} & & & \end{array}\right]
$$

Householderjeve metode

- Naredimo ortogonalno transformacijo in dobimo znake tridiagonalne matrike…

$$\mathbf{P}_1\mathbf{A} = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ \vdots & & \mathbf{P}^{(N-1)} & \\ 0 & & & \end{array}\right] \cdot \left[\begin{array}{c|ccc} a_{11} & a_{12} & \cdots & a_{1N} \\ \hline a_{21} & & & \\ \vdots & & \mathbf{A}^{(N-1)} & \\ a_{N1} & & & \end{array}\right] = \left[\begin{array}{c|ccc} a_{11} & a_{12} & \cdots & a_{1N} \\ \hline a'_{21} & & & \\ 0 & & & \\ \vdots & & \mathbf{P}^{(N-1)} \cdot \mathbf{A}^{(N-1)} & \\ 0 & & & \end{array}\right]$$

$$\mathbf{A}' = \mathbf{P}_1\mathbf{A}\mathbf{P}_1 = \left[\begin{array}{c|ccc} a_{11} & a'_{12} & 0 & \cdots \; 0 \\ \hline a'_{21} & & & \\ 0 & & & \\ \vdots & & \mathbf{P}^{(N-1)} \cdot \mathbf{A}^{(N-1)} \cdot \mathbf{P}^{(N-1)} & \\ 0 & & & \end{array}\right]$$

- … in tako naprej do tridiagonalne forme …

$$\mathbf{Z} = \mathbf{P}_1 \cdot \mathbf{P}_2 \cdot \ldots \cdot \mathbf{P}_{N-2}$$

Diagonalizacija tridiagonalne forme

- Seveda moramo tudi tridiagonalno formo diagonalizirati:
  - Uporabimo QR dekompozicijo (z izboljšavami…):
    - **Bistvena informacija je, da so te metode za tridiagonalno matriko bistveno hitrejše!**
    - **Iz Num. Rec. 'biblije':**

The workload in the QL algorithm is $O(n^3)$ per iteration for a general matrix, which is prohibitive. However, the workload is only $O(n)$ per iteration for a tridiagonal matrix and $O(n^2)$ for a Hessenberg matrix, which makes it highly efficient on these forms.

- Torej, **najprej** do tridiagonalne forme (Householder ali Givens), **potem** QR iteracije do diagonalizacije (ali kaj podobnega..)

- Obstaja tudi družina metod 'deli in vladaj' (**Divide and conquer**) kot alternativa za diagonalizacijo tridiag. matrik (LAPACK, Wikipedia…).
  - Uporabljene interno v Numpy eigh funkciji…

Splošna diagonalizacija

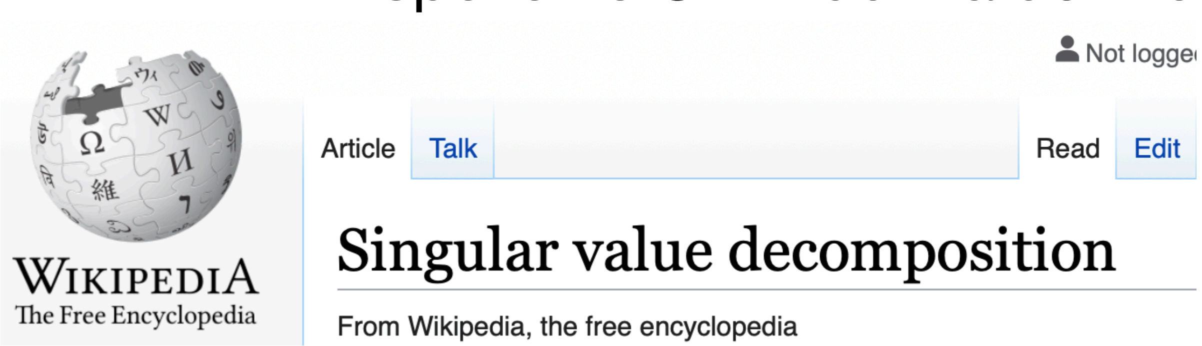
- V praksi redno uporabljena **QR** dekompozicija (deluje za m x n matrike…)
- Splošno matriko lahko (skoraj) diagonaliziramo s pomočjo postopka **Singular Value Decomposition (SVD).**
  - Velja tudi za kompleksne matrike (kompaktna verzija SVD)

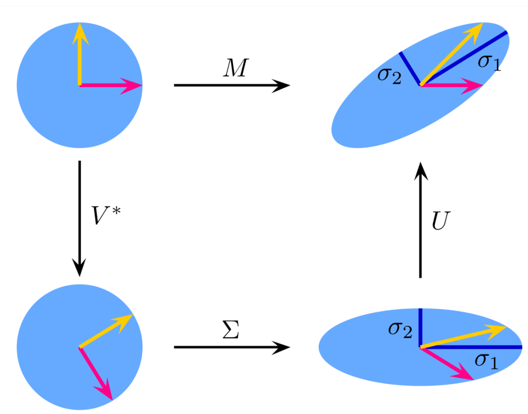
$$A_{m\times n} = U_{m\times r}\cdot\Sigma_{r\times r}\cdot V^{\dagger}_{r\times n}$$

$$U_{m\times r}, V_{n\times r} \longleftarrow \text{(semi) unitarni}$$

$$r < \min(m,n) \longleftarrow \text{rang matrike A}$$

  - Σ je diagonalna matrika.
  - Za realne matrike A sta U in V (semi) ortogonalni.
    - Semi-ortogonalna matrika: Za št. vrstic > št. stolpcev so stolpci ortonormalni vektorji in obratno.
    - V našem primeru r ortonormalnih vektorjev in r lastnih vrednosti.
- Zelo učinkoviti (stabilni, robustni, hitri) numerični algoritmi!
- Prispevek o SVD tudi na učilnici.





$M = U\cdot\Sigma\cdot V^*$

rotaciji + deformacija…

Diagonalizacija za iskanje ničel polinomov

- Diagonalizacija kvadratne matrike je splošno uporabna tudi za iskanje ničel polinomov.

$$p(z) = c_0 + c_1 z + \cdots + c_{n-1} z^{n-1} + z^n$$

- Uporabi se t.i. **Companion Matrix (C):**

$$C = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 & 1 \\ -c_0 & -c_1 & \dots & -c_{n-2} & -c_{n-1} \end{pmatrix}.$$

- Karakteristični polinom det(C - λI) je enak našemu polinomu p(λ) .
- Dokaz: recimo, da je *z* ničla našega polinoma p(z) - je tudi lastna vrednost matrike… Če zmnožimo:

$$C \begin{pmatrix} 1 \\ z \\ z^2 \\ \vdots \\ z^{n-1} \end{pmatrix} = \begin{pmatrix} z \\ z^2 \\ \vdots \\ z^{n-1} \\ -c_0 - c_1 z - \cdots - c_{n-1} z^{n-1} \end{pmatrix} = \begin{pmatrix} z \\ z^2 \\ \vdots \\ z^{n-1} \\ z^n \end{pmatrix} = z \begin{pmatrix} 1 \\ z \\ z^2 \\ \vdots \\ z^{n-1} \end{pmatrix}$$

ker je z ničla velja

λ**v -** lastna vrednost in vektor

QR dekompozicija nam da te 'ničle'…

University of Ljubljana
Faculty of Mathematics and Physics

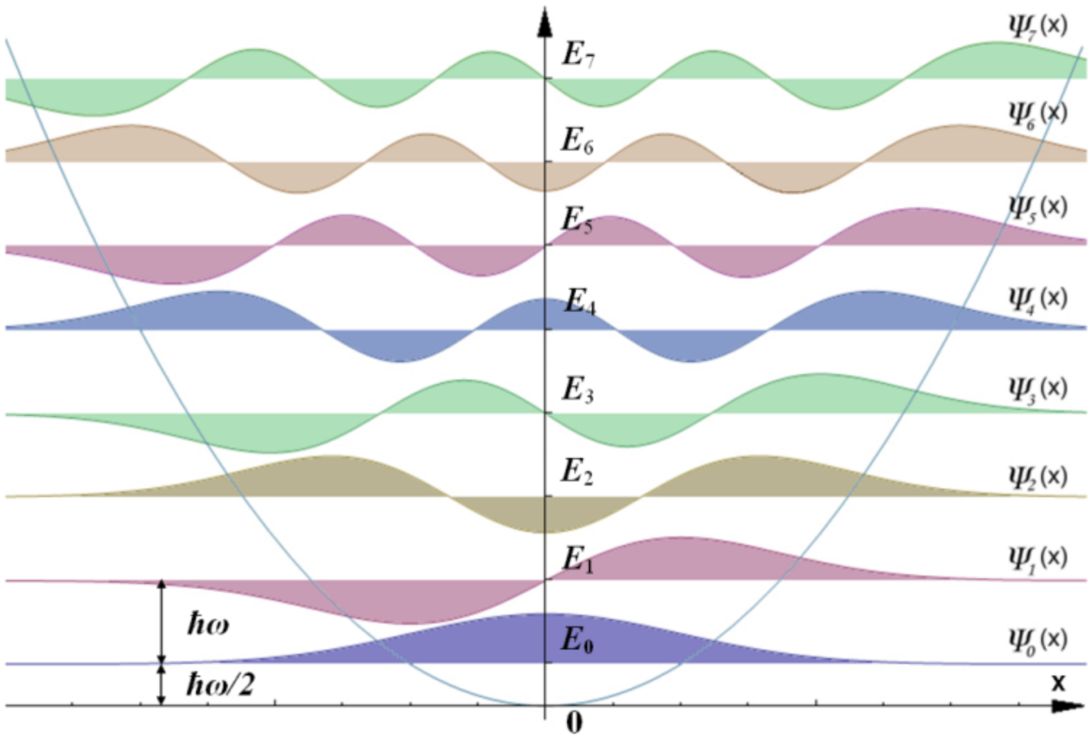
Harmonski oscilator

- Osnove kvantne mehanike - zadnji člen brezdim.:

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}k\hat{x}^2 = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega^2\hat{x}^2 = \frac{1}{2}\left(p^2 + q^2\right)$$

- Iskanje lastnih stanj kot rešitve časovno neodvisne enačbe:

$$\hat{H}\,|\psi\rangle = E\,|\psi\rangle$$



- Rešitve sorazmerno preproste (vsebujejo hermitove polinome H)

$$\psi_n(x) = \frac{1}{\sqrt{2^n\,n!}} \cdot \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} \cdot e^{-\frac{m\omega x^2}{2\hbar}} \cdot H_n\left(\sqrt{\frac{m\omega}{\hbar}}x\right), \qquad n = 0, 1, 2, \ldots.$$

$$|n\rangle = \left(2^n n! \sqrt{\pi}\right)^{-1/2} \mathrm{e}^{-q^2/2} H_n(q)$$

Kompakten zapis lastnih stanj $\psi_n(x)$

- Ustrezni energijski nivoji:

$$E_n = \hbar\omega\left(n + \frac{1}{2}\right) = (2n+1)\frac{\hbar}{2}\omega$$

https://en.wikipedia.org/wiki/Quantum_harmonic_oscillator

Harmonski oscilator

- Vsako stanje lahko zapisemo kot superpozicijo lastnih stanj (srečali boste npr koherentno stanje ipd…):

$$|\alpha\rangle = \sum_{n=0}^{\infty} |n\rangle\langle n|\alpha\rangle = \sum_{n=0}^{\infty} |n\rangle\, c_n = \sum_{n=0}^{\infty} \psi_n(x)\, c_n$$

- Ekvivalentno lahko rečemo, da smo stanje zapisali v **ortogonalni bazi Hilbertovega prostora** (lastne. f. = neskončno dimenzij). Tu imamo seveda definiran skalarni produkt in 'matrične' vrednosti operatorjev, na primer:

$$\langle n|n\rangle = \int_{-\infty}^{\infty} dx\; |\psi_n(x)|^2 = 1, \quad \langle n|m\rangle = \int_{-\infty}^{\infty} dx\, \psi_n^*(x)\psi_m(x) = \delta_{nm}$$

$$q_{ij} = \langle i|q|j\rangle = \frac{1}{2}\sqrt{i+j+1}\,\delta_{|i-j|,1}$$

$$\langle i\,|q^2|\,j\rangle = \frac{1}{2}\left[\sqrt{j(j-1)}\delta_{i,j-2} + (2j+1)\delta_{i,j} + \sqrt{(j+1)(j+2)}\delta_{i,j+2}\right]$$

https://en.wikipedia.org/wiki/Quantum_harmonic_oscillator

AnHarmonski oscilator

- **Anharmonski** oscilator je definiran kot harmonski primer z dodatkom (majhnega) dodatnega člena s potenco četrtega reda:

$$H = H_0 + \lambda q^4 \qquad \hat{H}\,|\psi\rangle = E\,|\psi\rangle$$

- Ta problem se najlepše rešuje v **bazi harmonskega oscilatorja:**
  - **Analitični** pristop je perturbacijski (nekaj dokumentacije na učilnici na spletu).
  - Mi tu poskusamo račun resiti **numerično (~ točno).**
  - Nove lastne funkcije, zapisane v tej bazi, bodo seveda malo drugačne.
    - Drugačne bodo tudi energije.
  - Torej, poskusamo zapisati Hamiltonovo enačbo v tej bazi in dobimo nediagonalno matriko (ki pa ni zelo daleč od diagonalne - več-diagonalna (pasovna)…)

$$\langle i\,|H|\,j\rangle = \langle i\,\left|H_0 + \lambda q^4\right|\,j\rangle = H_{ij} \neq E_i\,\delta_{i,j}$$

- **Tako pridemo do problema diagonalizacije: nove lastne funkcije bodo določene v bazi, kjer bo zgornja matrika diagonalna!**

AnHarmonski oscilator

- Bolj natančno: rešujemo torej spet enačbo:

$$\hat{H}\,|\psi\rangle = E\,|\psi\rangle$$

- Vendar pa nastavek za lastno funkcijo zapišemo v **bazi harmonskega oscilatorja:**

$$|\psi\rangle = \sum_{n=0}^{\infty} |n\rangle\langle n|\psi\rangle = \sum_{n=0}^{\infty} |n\rangle\, c_n = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ \vdots \end{bmatrix}$$

Vektor v (neskončno-dim) Hilbertovem prostoru

- Nove vektorje znova zapisemo v originalni bazi in primerjamo komponente (ortogonalnost!):

$$\hat{H}|\psi\rangle = \sum_{n=0}^{\infty} \hat{H}|n\rangle c_n,$$

$$E|\psi\rangle = E\sum_{m=0}^{\infty} |m\rangle\langle m \mid \psi\rangle = E\sum_{m=0}^{\infty} |m\rangle c_m,$$

$$\sum_{n=0}^{\infty} \hat{H}|n\rangle c_n = E\sum_{m=0}^{\infty} |m\rangle\langle m \mid \psi\rangle = E\sum_{m=0}^{\infty} |m\rangle c_m, \qquad \langle m|\cdot$$

$$E\, c_m = \sum_{n=0}^{\infty} \langle m|\hat{H}|n\rangle c_n$$

Matrični zapis v Hilbertovem prostoru

$$\Longrightarrow \quad E \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \\ \vdots \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & \dots & H_{1n} & \dots \\ H_{21} & H_{22} & & & \\ \vdots & \vdots & \ddots & \vdots & \\ H_{m1} & H_{m2} & \dots & H_{mn} & \dots \\ \vdots & \vdots & & \vdots & \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \\ \vdots \end{bmatrix}$$

$$E\,\mathbf{v} = \underline{H}\,\mathbf{v}$$

- **Očitno je to res le problem lastnih vrednosti v matričnem zapisu…**
- **Numerično moramo izbrati končno dimenzij (N), kar je del našega približka!**

Dodatne misli za domačo nalogo

- **Potenčne metode (najpreprostejše iteracije) z Hotelingovo redukcijo, ki ste jih (lahko) srečali pri Num. metodah prosim uporabite v tej nalogi le kot dodatno opcijo!**
  - Osebno se mi zdijo prevec ne-elegantne…
- Za DN **implementirajte vsaj eno metodo 'na roke')** (Jacobi, Givens, Householder).
  - QR ste tudi (lahko) spoznali že pri Num. metodah, lahko **dodate** tu.
  - Čim več, tem bolje …
- Primerjajte svoje metode tudi z 'vgrajenimi' metodami (hitrost, natančnost…).
- Investirajte čas v lepe grafe:
  - Kako so energije stanj odvisne od anharmoničnega parametra λ?
  - Kako se energije stanj spreminjajo z dimenzijo matrike H (N-dim)?
  - Znate lepo prikazati tudi lastne vektorje?
- Ugotovite, kateri od načinov izračuna perturbacijske matrike je najboljši (natačnost, hitrost..) in poskusi povedati tudi, zakaj:
  - $[q^4]$, $[q^2]^2$,$[q]^4$