Přednáška 18 – QR faktorizace I.

<u>Opáčko</u>: LU faktorizace / Gaussova eliminace může být výrazně nestabilní ⇝
⇝ hledáme jiné faktorizace / algoritmy pro převod $A\vec{x} = \vec{b}$ na horní-△ systém. Intuitivně jsme si odvodili, že možná ne-stabilita LU/GE by se vyřešila používáním <u>unitárních transf.</u>

<u>Lingebra 1</u>: $\forall A \in \mathbb{R}^{n\times n}\ \exists Q$ unitární, $\exists R$ horní-△: $A = QR$

Navíc Q, R lze spočítat Gram-Schmidt ortogonalizací $\{A_{:,1}, A_{:,2}, \dots, A_{:,n}\}$ (tj. sloupce matice A)

Pak

$$A\vec{x} = \vec{b} \iff QR\vec{x} = \vec{b} \iff R\vec{x} = Q^T\vec{b} \quad \& \quad Q \text{ unitární}$$

tj. umíme přesně to, co jsme chtěli. Zbývá vyjasnit:

1. Umíme spočítat/aplikovat Q, R stabilně?
2. Lze QR „rovnou aplikovat"? Ve stejném smyslu jako „GE je přímá aplikace LU".

Výpočet $A = QR$: Gramm-Schmidt. ortog.

<u>Krok 1</u>: $\vec{q}_1 = \dfrac{\vec{A}_{:,1}}{\|\vec{A}_{:,1}\|_2}$ & $r_{11} := \|\vec{A}_{:,1}\|_2$

<u>Krok 2</u>: $\vec{v}_2 := \vec{A}_{:,2} - \langle q_1, \vec{A}_{:,2}\rangle \cdot q_1 = \left(I - q_1 q_1^T\right)\vec{A}_{:,2}$

$\vec{q}_2 := \vec{v}_2 / \|\vec{v}_2\|_2 \qquad r_{1,2} := \langle q_1, \vec{A}_{:,2}\rangle \quad \& \quad r_{22} = \|\vec{v}_2\|$

⇝ tudíž platí $\left[\vec{A}_{:,1}, \vec{A}_{:,2}\right] = \left[\vec{q}_1, \vec{q}_2\right] \cdot \begin{bmatrix} r_{11} & r_{12} \\ 0 & r_{22} \end{bmatrix}$

<u>Krok 3</u>: $\vec{v}_3 := \vec{A}_{:,3} - \langle \vec{q}_1, \vec{A}_{:,3}\rangle \vec{q}_1 - \langle \vec{q}_2, \vec{A}_{:,3}\rangle \vec{q}_2 =$

$$= \left(I - q_1 q_1^T - q_2 q_2^T\right)\vec{A}_{:,3} \overset{(*)}{=} \left(I - \vec{q}_2\vec{q}_2^{\,T}\right)\left(I - \vec{q}_1\vec{q}_1^{\,T}\right)\vec{A}_{:,3}$$

(*) protože $q_1 \perp q_2$ ... neboli $\vec{q}_1^{\,T}\vec{q}_2 = 0$

$\vec{q}_3 := \vec{v}_3 / \|\vec{v}_3\|_2$ & $r_{13} = \langle \vec{q}_1, \vec{A}_{:,3}\rangle$, $r_{23} = \langle \vec{q}_2, \vec{A}_{:,3}\rangle$, $r_{33} = \|\vec{v}_3\|$

⇝ a opět máme $\left[\vec{A}_{:,1}, \vec{A}_{:,2}, \vec{A}_{:,3}\right] = \left[\vec{q}_1, \vec{q}_2, \vec{q}_3\right] \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ 0 & r_{22} & r_{23} \\ 0 & 0 & r_{33} \end{bmatrix}$

Pozorování: $r_{23} = \langle \vec{q}_2, \vec{A}_{:,3} \rangle = \langle \vec{q}_2, (I - \vec{q}_1\vec{q}_1^T)\vec{A}_{:,3} \rangle$ ..... stejně jako v (*)

⇝ obdobně i pro $r_{24}, r_{34}, r_{25}, \dots, r_{45}, \dots$ ⇝ který vzoreček vybrat?

Krok n: $\vec{V}_n := \left(I - \sum_{i=1}^{n-1} \vec{q}_i \vec{q}_i^T\right)\vec{A}_{:,n} = \left(\prod_{i=1}^{n-1} (I - \vec{q}_i\vec{q}_i^T)\right)\vec{A}_{:,n}$

$\vec{q}_n := \vec{V}_n / \|\vec{V}_n\|_2$ & $r_{in} = \langle \vec{q}_i, \vec{A}_{:,i} \rangle$ & $r_{nn} = \|\vec{V}_n\|_2$

⇝ a získáváme $\underbrace{[\vec{A}_{:,1}, \dots, \vec{A}_{:,n}]}_{A} = \underbrace{[\vec{q}_1, \dots, \vec{q}_n]}_{Q} \underbrace{\begin{bmatrix} r_{11} & \cdots & r_{1n} \\ 0 & \ddots & \vdots \\ 0 & \cdots\ 0 & r_{nn} \end{bmatrix}}_{R}$

Přirozeně vidíme 2 způsoby výpočtu:

tzv. klasický → $r_{ij} = \langle \vec{A}_{:,j}, \vec{q}_i \rangle$

tzv. modifikovaný → $r_{ij} = \langle \vec{q}_i, (I - \vec{q}_{i-1}\vec{q}_{i-1}^T) \cdot \ldots \cdot (I - \vec{q}_1\vec{q}_1^T) \cdot \vec{A}_{:,j} \rangle$

na papíře jsou ekvivalentní ⇝ naprogramované nikoliv

⇝ který byste tipovali na vítěze? (python demo)

Jak je na tom stabilita výpočtu?

A ... vstupní data, $\hat{Q}, \hat{R}$ produkt algoritmu výše:

$\|A - \hat{Q}\hat{R}\| \leq C \cdot \|A\| \cdot \varepsilon_{mach}$ ..... pro oba způsoby

$$\|I - \hat{Q}^T\hat{Q}\| \leq \begin{cases} C \cdot \varkappa(A)^2 \cdot \varepsilon_{mach} & \text{☹ ..... klasický} \\ C \cdot \varkappa(A) \cdot \varepsilon_{mach} & \text{☹ ..... modifikovaný} \end{cases}$$

Kdybychom počítali přesně pak $= 0$ ⇝ $\hat{Q}$ by měla mít ON sloupce.

To byla ta motivace proč dělat QR rozklad/řešič! Platí to i na PC?

⇝ ne nutně ☹

Jinými slovy: • obě verze Gram-Schmidta spočítají rozklad „stabilně" ve smyslu relativního rezidua, tj. $\|A - \hat{Q}\hat{R}\| / \|A\| \leq C \cdot \varepsilon_{mach}$.

• obě verze trpí <u>numerickou ztrátou ortogonality</u> (pro špatně podmíněné matice)

Kvůli tomu se obecně tyto algoritmy považují za nestabilní.

<u>Navíc:</u> matice, které aplikujeme na sloupce A, tj. $I - \sum_{i=1}^{k} \vec{q}_i \vec{q}_i^T = (I - \vec{q}_1 \vec{q}_1^T) \cdot \ldots \cdot (I - \vec{q}_k \vec{q}_k^T)$ <u>nejsou unitární</u>

$\Rightarrow$ potenciálně stejný problém jako u G.E./LU!

$\rightsquigarrow$ <u>Lze lépe:</u>
- Givensovy rotace
- Householderovy reflexe

GRAM-SCHMIDT graficky

$[\vec{a}_1, \vec{a}_2, \vec{a}_3] = A$