# Numerical Methods for CSE

AUTUMN 2023

Q+A 22.09.2023

VIDEOS
LECTURE NOTES
↓ ↘
HAND SCRIPT
↓
every week.

interactive
Q+A
↓
discuss

HOMEWORK*
TUTORIALS.
very important
very important

NO BONUS
A MOCK EXAM "BYOD" + "DIY" NOVEMBER

"CODE EXPERT"

**Def** Kronecker produkt of two matrices

$\underline{\underline{A}} \in \mathbb{R}^{m \times n}$ $\quad \underline{\underline{B}} \in \mathbb{R}^{l \times k}$

$m, n, l, k \in \mathbb{N}$

$\underline{\underline{A}} \otimes \underline{\underline{B}} \in \mathbb{R}^{(ml) \times (nk)}$

Block of size $l \times k$

$$\begin{bmatrix} A_{11}\underline{\underline{B}} & A_{12}\underline{\underline{B}} & \dots & A_{1n}\underline{\underline{B}} \\ \vdots & & & \\ A_{m1}\underline{\underline{B}} & A_{m2}\underline{\underline{B}} & \dots & A_{mn}\underline{\underline{B}} \end{bmatrix}$$

$\hookrightarrow (ml) \times (nk)$

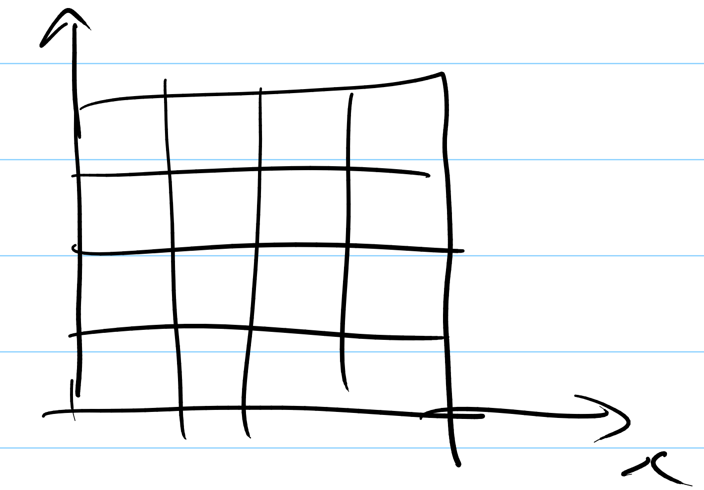
**Example** $I_m \otimes \underline{\underline{A}} = \begin{bmatrix} 1\underline{\underline{A}} & & 0 \\ & 1\underline{\underline{A}} & \\ 0 & & \ddots \\ & & & 1\underline{\underline{A}} \end{bmatrix}$

$\underline{\underline{A}} \times I_m = \begin{bmatrix} a_{11}\underline{I} & a_{12}\underline{I} & \dots & a_{1k} \\ & a_{22}\underline{I} & & \\ \vdots & & & \\ a_{n1} & a_{n2} & \dots & a_{nk}\underline{I} \end{bmatrix}$

$\underline{\underline{A}}$ $n \times k$

$$\begin{bmatrix} \begin{matrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{11} & & \vdots \\ \vdots & & & 0 \\ 0 & \dots & & a_{11} \end{matrix} & \begin{matrix} a_{12} & 0 & \dots & 0 \\ 0 & a_{12} & & 0 \\ \vdots & & \ddots & 0 \\ 0 & \dots & & a_{12} \end{matrix} \\ & \end{bmatrix}$$

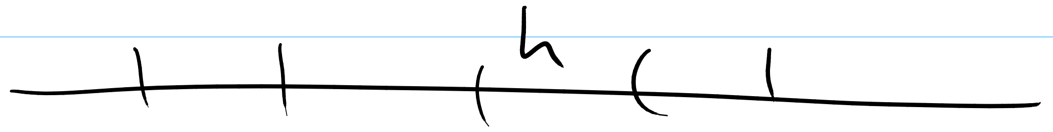
# Partial Differential Equations




$$\boxed{\frac{\partial}{\partial x}} \quad \frac{\partial}{\partial y}$$

$\downarrow$

Matrix




$x_{i-1}$ $x_i$ $x_{i+1}$

$$\underline{u} = \begin{bmatrix} u(x_0) \\ \vdots \\ u(x_{N-1}) \end{bmatrix} \qquad \underline{\underline{A}} \qquad \underline{\underline{D}}_x \quad \underline{\underline{D}}_y$$

$$\frac{\partial}{\partial x} u(x_i) \simeq \frac{u_{i+1} - u_{i-1}}{2h}$$

$$\frac{\partial}{\partial x} \quad \frac{\partial}{\partial y} \qquad \underline{\underline{D}}_x \otimes \underline{\underline{D}}_y$$

29.09.2023

# Q3.0.1.11.C.

**(Q3.0.1.11.C)** [A $\int_0^1 e^x\,\mathrm{d}e$-type problem] We know the solution $\mathbf{x} \in \mathbb{R}^n$ and the right-hand-side vector $\mathbf{b} \in \mathbb{R}^n$ of the $n \times n$ (Toeplitz) tridiagonal linear system of equations

$$\begin{bmatrix} \alpha & \beta & 0 & \dots & & & \dots & 0 \\ \beta & \alpha & \ddots & & & & & \vdots \\ 0 & \beta & \ddots & \ddots & & & & \\ \vdots & & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \ddots & \vdots \\ & & & & & \ddots & \beta & 0 \\ \vdots & & & & & \beta & \alpha & \beta \\ 0 & \dots & & & \dots & 0 & \beta & \alpha \end{bmatrix} \mathbf{x} = \mathbf{b}\,.$$

Which overdetermined linear system of equations of maximal size has the vector $[\alpha, \beta]^\top \in \mathbb{R}^2$ as its solution?

$$\begin{cases} \alpha x_1 + \beta \boxed{x_2} = b_1 \\ \beta \boxed{x_1} + \alpha x_2 + \beta \boxed{x_3} = b_2 \\ \beta \boxed{x_2} + \alpha x_3 + \beta \boxed{x_4} = b_3 \\ \beta x_3 + \alpha x_4 + \beta x_5 = b_4 \\ \beta x_4 + \alpha x_5 + \beta x_6 = b_5 \\ \vdots \\ \beta \boxed{x_{n-2}} + \alpha x_{n-1} + \beta \boxed{x_n} = b_{n-1} \\ \beta \boxed{x_{n-1}} + \alpha x_n = b_n \end{cases}$$

$$\begin{bmatrix} x_1 & x_2 \\ x_2 & x_1 + x_3 \\ x_3 & x_2 + x_4 \\ x_4 & x_3 + x_5 \\ \vdots & \vdots \\ x_{n-1} & x_{n-2} + x_n \\ x_n & x_{n-1} \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ \\ b_n \end{bmatrix}$$

**(Q3.1.1.14.C)** Given a matrix $\mathbf{B} \in \mathbb{R}^{m,n}$, a vector $\mathbf{c} \in \mathbb{R}^m$, and $\lambda > 0$, define

$$\{\mathbf{x}^*\} := \underset{\mathbf{x}\in\mathbb{R}^n}{\operatorname{argmin}} \|\mathbf{B}\mathbf{x} - \mathbf{b}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 \subset \mathbb{R}^n .$$

looks like a penalisation :

$x \neq 0 \Rightarrow \lambda \|x\|_2^2$ clear increase !

but actually no deeper meaning than for trick

State an overdetermined linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$, of which $\mathbf{x}^*$ is a least-squares solution.

↳ a method to address linear least squares problem for $\underline{\underline{B}}$ with $\underline{\underline{B}}$ not full rank

(i.e. (~~some of~~) the columns of $\underline{\underline{B}}$ are linear dependent)

Pick a small $\lambda > 0$

$$\underline{\underline{A}} = \begin{bmatrix} \underline{\underline{B}} \\ \sqrt{\lambda}\, \underline{\underline{I}}_n \end{bmatrix} \qquad \underline{b} = \begin{bmatrix} \underline{c} \\ \underline{0} \end{bmatrix} \in \mathbb{R}^{m+n}$$

↳ linear indep. columns of $\underline{\underline{A}}$

$$\lambda \|\underline{x}\|_2^2 = \langle \sqrt{\lambda}\, \underline{\underline{I}}\, \underline{x}, \sqrt{\lambda}\, \underline{\underline{I}}\, \underline{x} \rangle$$

$$\underset{\underline{x}\in\mathbb{R}^n}{\operatorname{argmin}} \|\underline{\underline{A}}\,\underline{x} - \underline{b}\|_2^2$$

$(m+n) \times n$ has Rank $n$.

$\Rightarrow$ QR-decomposition will work. !

Possible advantage: if $\underline{\underline{B}}$ is sparse, so is $\underline{\underline{A}}$, so QR-dec. via Givens-Rotations might be much less expensive than SVD($\underline{\underline{B}}$).

Question: difference in advantages CSC / CRS

CSS: good for slicing colomns
CRS: good for slicing rows
both are good for internal +, * (pointwise)
Matrix * vector
(might be faster)

Remark other formats are better for a fast construction.

**(Q2.7.1.5.E)** For a given matrix $\mathbf{A} \in \mathbb{R}^{m,n}$, $m, n \in \mathbb{N}$, we define the square matrix

$$\mathbf{W}_{\mathbf{A}} := \begin{bmatrix} \mathbf{O}_{m,m} & \mathbf{A} \\ \mathbf{A}^\top & \mathbf{O}_{n,n} \end{bmatrix} \in \mathbb{R}^{m+n,m+n}.$$

Outline the implementation of an efficient C++ function

```
void crsAtoW(std::vector<double> &val,
std::vector<unsigned int> &col_ind,
std::vector<unsigned int> &row_ptr);
```

whose arguments supply the three vectors defining the matrix $\mathbf{A}$ in CRS format and which overwrites them with the corresponding vectors of the CRS-format description of $\mathbf{W}_{\mathbf{A}}$.

Remember that the CRS format of a matrix $\mathbf{A} \in \mathbb{R}^{m,n}$ is defined by

$$\texttt{val}[k] = (\mathbf{A})_{i,j} \quad \Leftrightarrow \quad \begin{cases} \texttt{col\_ind}[k] = j, \\ \texttt{row\_ptr}[i] \leq k < \texttt{row\_ptr}[i+1], \end{cases} \quad 1 \leq k \leq \operatorname{nnz}(\mathbf{A}).$$

It may be convenient to use `std::vector::resize(n)` that resizes a vector so that it contains n elements. If n is smaller than the current container size, the content is reduced to its first n elements, removing those beyond (and destroying them). If n is greater than the current container size, the content is expanded by inserting at the end as many elements as needed to reach a size of n using their default value.

Most important: how to implement $\underline{A}^T$ ?

This function does it:

```
CRSMatrix sparse_transpose(const CRSMatrix& input) {
    CRSMatrix res{
        input.m,
        input.n,
        input.nz,
        std::vector<double>(input.nz, 0.0),
        std::vector<int>(input.nz, 0),
        std::vector<int>(input.m + 2, 0) // one extra
    };

    // count per column
    for (int i = 0; i < input.nz; ++i) {
        ++res.rowPtr[input.colIndex[i] + 2];
    }

    // from count per column generate new rowPtr (but shifted)
    for (int i = 2; i < res.rowPtr.size(); ++i) {
        // create incremental sum
        res.rowPtr[i] += res.rowPtr[i - 1];
    }

    // perform the main part
    for (int i = 0; i < input.n; ++i) {
        for (int j = input.rowPtr[i]; j < input.rowPtr[i + 1]; ++j) {
            // calculate index to transposed matrix at which we should p
            const int new_index = res.rowPtr[input.colIndex[j] + 1]++;
            res.val[new_index] = input.val[j];
            res.colIndex[new_index] = i;
        }
    }
    res.rowPtr.pop_back(); // pop that one extra

    return res;
}
```

**(Q2.6.0.25.F)** [Loss of stability] By direct block-wise Gaussian elimination we found the following solution formulas for a block-partitioned linear system of equations with $\mathbf{D} \in \mathbb{R}^{n,n}$, $\mathbf{c}, \mathbf{b} \in \mathbb{R}^n$, $\alpha \in \mathbb{R}$, $\mathbf{y} \in \mathbb{R}^{n+1}$:

$$\mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{D} & \mathbf{c} \\ \mathbf{b}^\top & \alpha \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \xi \end{bmatrix} = \mathbf{y} := \begin{bmatrix} \mathbf{y}_1 \\ \eta \end{bmatrix}, \tag{2.6.0.7}$$

$$\xi = \frac{\eta - \mathbf{b}^T \mathbf{D}^{-1} \mathbf{y}_1}{\alpha - \mathbf{b}^\top \mathbf{D}^{-1} \mathbf{c}}, \tag{2.6.0.8}$$

$$\mathbf{x}_1 = \mathbf{D}^{-1}(\mathbf{y}_1 - \xi \mathbf{c}).$$

Use these formulas to compute the solution of the $2 \times 2$ linear system of equations

$$\begin{bmatrix} \delta & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ \xi \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix},$$

assuming $|\delta| < \frac{1}{2}\mathrm{EPS}$ and using floating point arithmetic.

**Hint.** Remember that, if $|\delta| < \frac{1}{2}\mathrm{EPS}$, in floating point arithmetic

$$1 \tilde{+} \delta = 1 \quad \text{and} \quad 2 \tilde{+} \delta^{-1} = \delta^{-1}.$$

This is compatible with the "Axiom" of roundoff ana[…]s Ass. 1.5.3.11

$$\xi = \frac{2 - \frac{1}{\delta}}{1 - \frac{1}{\delta}} = \frac{1}{1-\delta^{-1}} + 1$$

$$1 + \delta = 1$$

$$-2 - \frac{1}{\delta} = -\frac{1}{\delta^{-1}} \Rightarrow \quad 1 - 2 - \frac{1}{\delta} = 1 - \frac{1}{\delta}$$

$$\underline{\underline{2 - \delta^{-1}}} = 2 - \underline{\underline{(2 + \delta^{-1})}} = -\delta^{-1}$$

$$1 - \delta^{-1} = 1 - (2 + \delta^{-1}) = -1 - \delta^{-1}$$

$$\xi = \frac{-\delta^{-1}}{-\delta^{-1} - 1} = \frac{+1}{+1+\delta} = \frac{1}{1} = 1$$

$$x_1 = \delta^{-1}(1 - 1) = 0 \Rightarrow \begin{bmatrix} -1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} \delta & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

:(

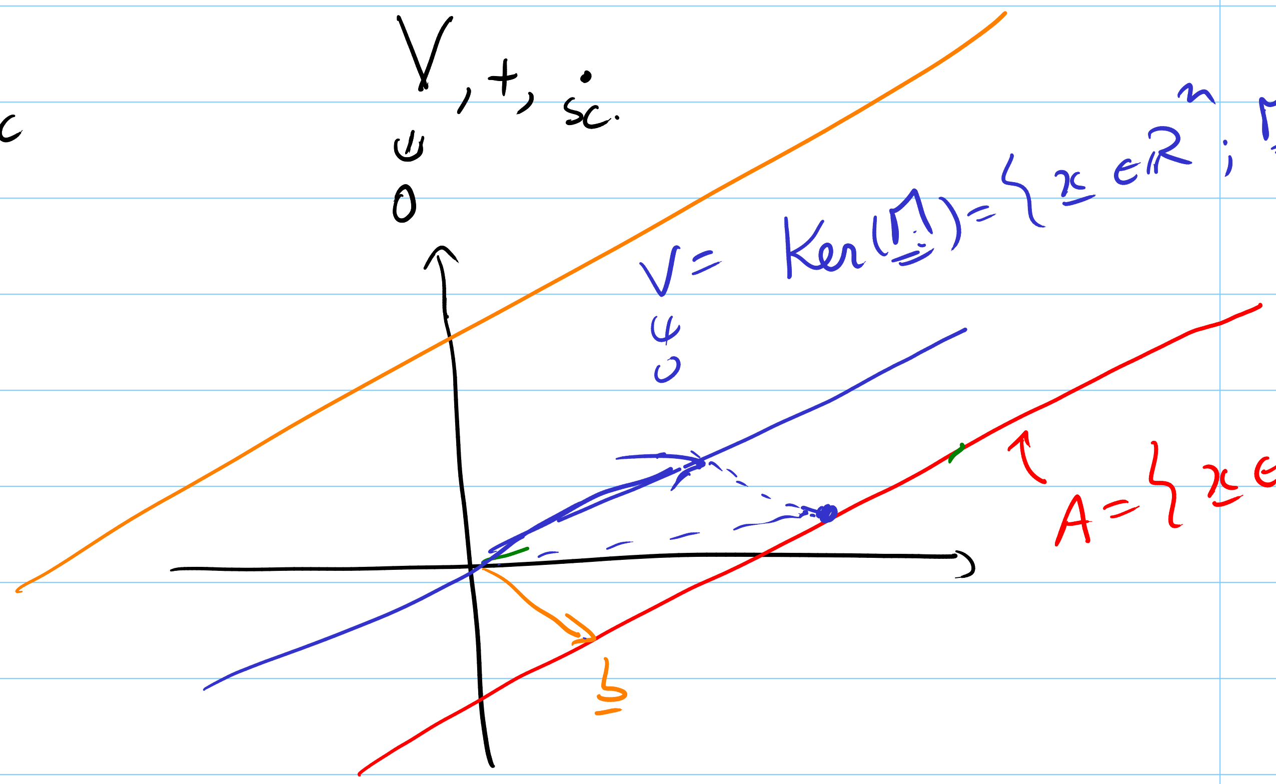
If we compute "correctly", we get the wrong answer! :)

06.10.2023

affine space $\neq$ linear space
vektor

$A, +, \cdot_{sc}$ $\qquad$ $V, +, \cdot_{sc}.$

$V \ni 0$



$V = \mathrm{Ker}(\underline{\underline{M}}) = \{ \underline{x} \in \mathbb{R}^n ; \underline{\underline{M}}\,\underline{x} = \underline{0} \}$

$V \ni 0$

$A = \{ \underline{x} \in \mathbb{R}^n ; \underline{\underline{M}}\,\underline{x} = \underline{b} \}$

$\underline{b}$

$\underline{x} \in A \iff \underline{x} = \underline{b} + \underline{v}$ with $\underline{v} \in V$

Could you please explain what the algorithm 2.10.1 from the exercise sheet does?
I don't quite understand why we can use the Sherman-Morris-Woodbury Formula for sub-problem c)

Use SMW-Formula in order to avoid solving

$\underline{\underline{M}}\,\underline{x} = \underline{ev}$ with a full matrix $\underline{\underline{M}}$

Use the fact that $\underline{\underline{M}} = \mathrm{diag}(\underline{d}) + \underline{u}\,\underline{v}^T$

Sparse $\qquad$ Rang 1-modification

$\mathrm{dig}(\underline{d})\,\underline{x} = \underline{b}$ is solvable
dirctly in $O(n)$
Operations

Hence $O(n)$ instead of $O(n^3)$.

13.10.2023

**(Q3.4.4.13.A)** Let $\mathbf{M} \in \mathbb{R}^{n,n}$ be symmetric and positive definite (*s.p.d.*) and $\mathbf{A} \in \mathbb{R}^{m,n}$. Devise an algorithm for computing

$$\underset{\mathbf{x}\in B}{\operatorname{argmax}}\|\mathbf{A}\mathbf{x}\| \quad , \quad B := \{\mathbf{x} \in \mathbb{R}^n,\ \mathbf{x}^\top \mathbf{M}\mathbf{x} = 1\} \ ,$$

also based on the SVD of $\mathbf{M}$.

$$\underline{M} = \underline{U}\,\underline{\Sigma}\,\underline{U}^T \qquad \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n > 0$$

$$\underline{D} = \sqrt{\underline{\Sigma}}$$

Write the condition: $\underline{x}^T \underline{M}\,\underline{x} = 1$

$$\underline{x}^T \underline{U}\,\underline{\Sigma}\,\underline{U}^T \underline{x} = 1$$

$$\underbrace{\underline{x}^T \underline{U}\,\underline{D}^T}_{\underline{y}^T}\underbrace{\underline{D}\,\underline{U}^T\underline{x}}_{\underline{y}} = 1$$

orthonormal.

$$\underline{y}^T\underline{y} = 1 \quad \text{with} \quad \underline{y} = \underline{D}\,\underline{U}^T\underline{x} \iff \underline{U}\,\underline{D}^{-1}\underline{y} = \underline{x}$$

$\underline{D}^{-1}|$

$\underline{U}|$

$$\underset{\substack{x\in\mathbb{R}^n \\ \underline{x}^T\underline{M}\,\underline{x}=1}}{\operatorname{argmax}} \|\underline{A}\,\underline{x}\| =$$

$$\underset{\substack{\underline{y}=\underline{D}\underline{U}^T\underline{x} \\ \underline{y}^T\underline{y}=1}}{\operatorname{argmax}} \|\underbrace{\underline{A}\,\underline{U}\,\underline{D}^{-1}}_{\underline{B}}\underline{y}\| = \underline{U}\,\underline{D}^{-1}\underset{\substack{\underline{y}\in\mathbb{R}^n \\ \underline{y}^T\underline{y}=1}}{\operatorname{argmax}}\|\underline{B}\,\underline{y}\|$$

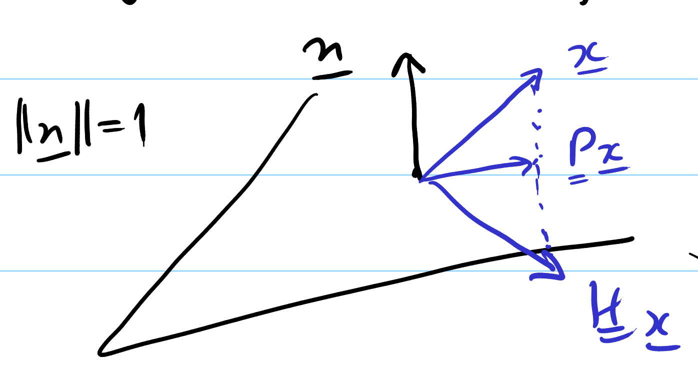
which $\underline{x}$ makes $\|\underline{A}\,\underline{x}\| = \min$

See (3.4.4.3) for the solution of this problem.

Sol. is the first right singular vector of $\underline{B}$ → $\underline{y}$

Sol $\underline{x} = \underline{U}\,\underline{D}^{-1}\underline{y}$

Q Why $\det H = -1$ for any Householder Matrix?




$\underline{n} \in \mathbb{R}^d$

$\|\underline{n}\| = 1$

$\to$ plane $\perp \underline{n}$

$\dim V = d-1$

Take basis (orthonormal) $\underline{v}_1, \dots, \underline{v}_{d-1} \in V$

$\underline{\underline{H}}\,\underline{v}_1 = \underline{v}_1, \dots, \underline{\underline{H}}\,\underline{v}_{d-1} = \underline{v}_{d-1}$

$\underline{\underline{H}}\,\underline{n} = -n$

$\to$ 1 is EW of $\underline{\underline{H}}$ of Multiplicity $d-1$

$-1$ is EW of $\underline{\underline{H}}$ of multiplicity 1

$\det \underline{\underline{H}} = \lambda_1 \lambda_2 \lambda_2 \dots \lambda_n = (1)^{d-1}(-1) = -1$

Q How do we solve Linear Systems of Eq.?

1) Want good precision? + Moderate precision.

2) how expensive?

good precision, not too big Matrix $\Rightarrow$ LU-decomp.

A symmetric pos.def $\Rightarrow$ Cholesky-decomp.

big Matrix, sparse (banded)

$\to$ there might be some direct methods that keep sparsity, so might be feasable QR; LU

in general LU would be not feasable.

less precision or Large sparse Matrix $\underline{A}$

$\Rightarrow$ iterative methods

Krylov-type Method

$\nearrow$ symmetric CG

$\searrow$ non-symm. others

Krylov-type methods use only

$\underline{\underline{A}}\, \underline{x}$

$\Rightarrow$ if $\underline{\underline{A}}$ is sparse $\Rightarrow$ $O(n, n^2)$

Note: CG slow if $\mathrm{cond}(\underline{\underline{A}})$ is big.

$\rightarrow$ use pre-conditioning

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \Rightarrow \underline{\underline{B}}^{-1}\underline{\underline{A}}\,\underline{x} = \underline{\underline{B}}^{-1}\underline{b}$$

$\underline{\underline{B}}^{-1} \,|$

if $\underline{\underline{B}}^{-1} = \underline{\underline{A}}^{-1} \Rightarrow \underline{x} = \underline{\underline{B}}^{-1}\underline{b}$ 😮

use $\underline{\underline{B}}^{-1} \approx \underline{\underline{A}}^{-1}$

ex $\underline{\underline{B}} = \mathrm{diag}(\underline{\underline{A}}) \rightarrow \underline{\underline{B}}^{-1}$ cheap

$\Rightarrow$ solve (CG) for $\underline{\underline{B}}^{-1}\underline{\underline{A}}\,\underline{x} = \underline{\underline{B}}^{-1}\underline{b}$

(Q) Polar decomposition

20.10.2023

$$\underline{\underline{A}} = \underline{\underline{U}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^T = \underbrace{\underline{\underline{U}}\,\underline{\underline{V}}^T}_{\underline{\underline{Q}}}\,\underbrace{\underline{\underline{V}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^T}_{\underline{\underline{H}}\text{ spd.}}$$

$I = V^T V$

$$= \underbrace{U \Sigma U^T}_{\underline{\underline{\tilde{H}}}\text{ spd.}}\, \underbrace{U V^T}_{\underline{\underline{\tilde{Q}}}} =$$

$m \geq n$ economical SVD

$\underline{\underline{U}} \in \mathbb{R}^{m \times n}$, $\underline{\underline{\Sigma}} = \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_n \end{bmatrix}$

$\underline{\underline{V}} \in \mathbb{R}^{n \times n}$

$\underline{\underline{V}}\,\underline{\underline{\Sigma}}\,\underline{\underline{V}}^T \sim n^2$ Operations (Storage)

$\underline{\underline{U}} \sim m \cdot n$ Operations (Storage)

$\Rightarrow$ not cheaper $O(mn + n^2)$

$+n$ $\rightarrow$ dominates!

3.12.d . Main messages:

① avoid using economical QR-decomposition

② if using it, then be aware of dimensions!

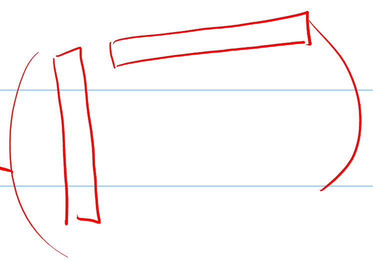
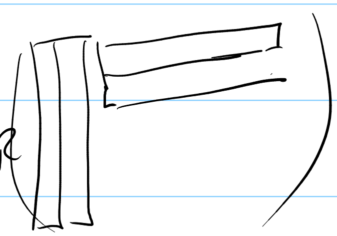
3.π. $\underline{X}_1 = \underline{A}_1 \underline{B}_1^T \in \mathbb{R}^{m\times n}$ $\quad \underline{X}_2 = \underline{A}_2 \underline{B}_2^T$ $\quad \underline{A}_1, \underline{A}_2 \in \mathbb{R}^{m\times k}$

$\underline{X} = [X_1 \ X_2] \in \mathbb{R}^{m,2n}$ $\quad \underline{B}_1, \underline{B}_2 \in \mathbb{R}^{n\times k}$

$$\mathcal{R}(\underline{X}) = \mathcal{R}(\underline{X}_1) + \mathcal{R}(\underline{X}_2)$$

$$\Rightarrow \dim \mathcal{R}(X) \le \underbrace{\dim \mathcal{R}(X_1)}_{\le k} + \underbrace{\dim \mathcal{R}(X_2)}_{\le k} = 2k \Rightarrow$$

$\dim \mathcal{R}(X) \le m$

dim R  = 1 $\qquad$ dim R  = k = Rank(A)

$\dim \mathcal{R}(X) \le \min\{m, 2k\}$.

SVD of $\underline{X}$ without assembling $X$?

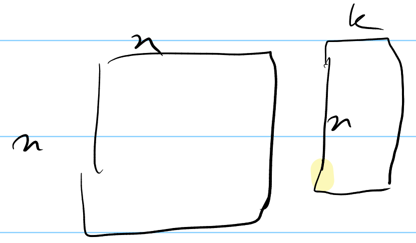
i.e. by using only the factors

$\underline{A}_1, \underline{B}_1, \underline{A}_2, \underline{B}_2$.

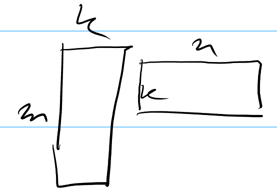
economical!

$\underline{B}_1 = \underline{Q}_1 \underline{R}_1$

$\underline{B}_2 = \underline{Q}_2 \underline{R}_2$

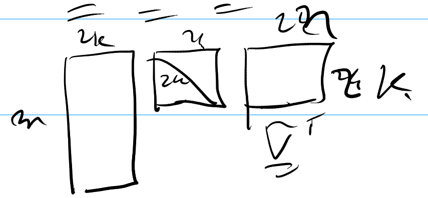


$\underline{X} = [\underline{A}_1 \underline{R}_1^T \underline{Q}_1^T \quad \underline{A}_2 \underline{R}_2^T \underline{Q}_2^T] =$



$$= \underbrace{[\overset{m\times n}{\underline{A}_1 R_1^T} \quad \overset{m\times n}{\underline{A}_2 \underline{R}_2^T}]}_{\underline{Z}^{m\times 2n}} \underbrace{\begin{bmatrix} Q_1^T & 0 \\ 0 & Q_2^T \end{bmatrix}}_{\underline{Q} \in \mathbb{R}^{2k\times 2n}}$$

$\underline{Z} = \underline{\tilde{U}} \underline{\tilde{\Sigma}} \underline{\tilde{V}}^T \qquad \underline{\tilde{U}} \in \mathbb{R}^{m,2k}, \quad \underline{\tilde{V}} \in \mathbb{R}^{2k\times 2n}$



$$X = \underline{\tilde{U}}\,\underline{\check{\Sigma}}\,\underbrace{\underline{\tilde{V}}^T \underline{Q}^T}_{}$$

$$\tilde{U} = U, \quad \check{\Sigma} = \Sigma, \quad \tilde{V}^T Q^T = Q$$

Implementation:

15.6.
$$I_{nk} = \begin{array}{c} k \\ \begin{bmatrix} 1 & & \\ & 1 & \\ & & \ddots\ 1 \\ & & \\ \end{bmatrix} \end{array} n$$

one call of full QR-decomposition (Householder) gives whole $n \times n$ matrix $\underline{Q}_1$

$$\underline{Q}_1 \underline{I}_{nk} = \begin{bmatrix} \uparrow & & \uparrow \\ \underline{q}_1 & \dots & \underline{q}_k \end{bmatrix} = \text{first } k \text{ columns of } \underline{Q}.$$

(Q) Meaning of interpolation operator?

$t_1 \quad \underline{y}_1 \in \mathbb{R}^n$

$\wedge$

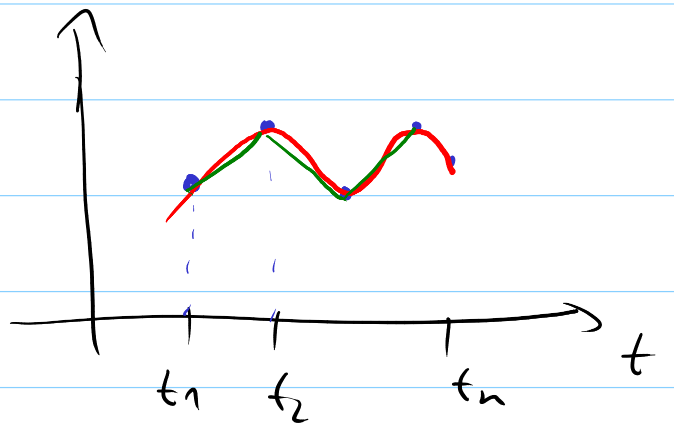
$t_2 \quad \underline{y}_2 \in \mathbb{R}^n$

$\wedge$

$\vdots \quad \vdots$

$\wedge$

$t_m \quad \underline{y}_m$




Want a function $f : [t_1, t_m] \to \mathbb{R}^n$ such that

$$f(t_j) = \underline{y}_j \quad \text{for all } j = 1, \dots, m.$$

$f$ should have some desired properties

applications    mathematical feasable

$$V = \operatorname{span}\{b_1, b_2, \dots, b_n\} \ni f$$

$$f(t) = \sum_{k=1}^{n} c_k b_k(t)$$

$n$ conditions $\to$ Linear System $n \times n$ for $c_1, \dots, c_n$.

(C) V. Gradinaru

Which $b_1, \dots, b_n$ to choose?

$V = P_n \Rightarrow b_k(t) = t^{k-1}$ :(

$\Downarrow$

linear system bad conditioned!

+ better basis!

+ suitable space.

$b_k(t) = e^{ikt} = \cos(kt) + i\sin(kt)$

$\Rightarrow$ trigonometric polynomials

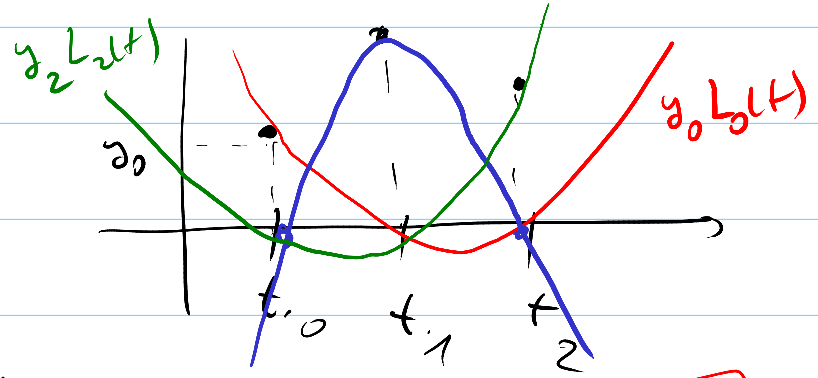
$\Rightarrow$ very fast and accurate algorithms via FFT.

$\rightarrow$ complete basis in $L^2$

(Q) Lagrange interpolation.

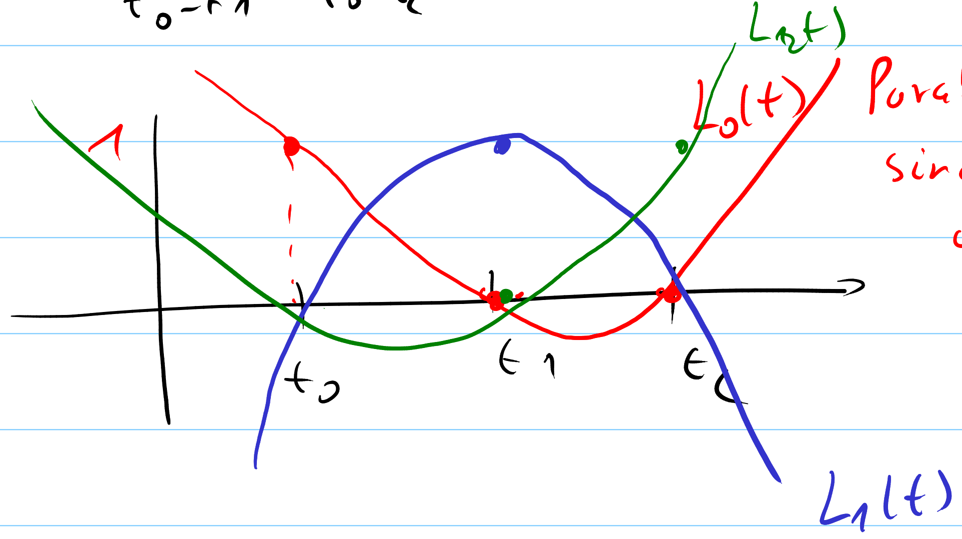
Let us take 3 measurements.

| $t_0$ | $t_1$ | $t_2$ |
|---|---|---|
| $y_0$ | $y_1$ | $y_2$ |




$$L_0(t) = \frac{t-t_1}{t_0-t_1}\,\frac{t-t_2}{t_0-t_2} \Rightarrow L_0(t_0) = \underbrace{\boxed{\frac{t_0-t_1}{t_0-t_1}}}_{1}\,\underbrace{\boxed{\frac{t_0-t_2}{t_0-t_2}}}_{1} = 1$$

$$L_0(t_1) = \frac{0}{t_0-t_1}\,\frac{t_1-t_2}{t_0-t_2} \Rightarrow L_0(t_1) = L_0(t_2) = 0$$




Parabola since polynomial of degree.

$$f(t) = y_0 L_0(t) + y_1 L_1(t) + y_2 L_2(t)$$

$$f(t_0) = y_0 \cdot 1 + y_1 \cdot 0 + y_2 \cdot 0 = y_0$$

$$f(t_1) = y_0 \cdot 0 + y_1 \cdot 1 + y_2 \cdot 0 = y_1$$

$$f(t_2) = y_0 \cdot 0 + y_1 \cdot 0 + y_2 \, 1 = y_2$$

$\uparrow$

"$\delta$"-Property

$$L_j(t_k) = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k. \end{cases}$$