Numerical Methods for Computational Science and Engineering

Fall Semester 2017 (HS17)

Prof. Rima Alaifari, SAM, ETH Zurich

Example (2.6.13): Low-rank modification of an LSE

Task: Solve $Ax = b$ $\quad A \in \mathbb{R}^{n,n}_{reg}$

Then solve $\tilde{A}x = \tilde{b}$ $\quad (**)$ $\quad A - \tilde{A}$ is low-rank

E.g. $A, \tilde{A}$ differ by only one entry:

$$\mathbf{A}, \tilde{\mathbf{A}} \in \mathbb{K}^{n,n}: \quad \tilde{a}_{ij} = \begin{cases} a_{ij} & \text{, if } (i,j) \neq (i^*, j^*) \,, \\ z + a_{ij} & \text{, if } (i,j) = (i^*, j^*) \,, \end{cases} \quad , \quad i^*, j^* \in \{1, \dots, n\} \,.$$

$$\blacktriangleright \quad \boxed{\tilde{\mathbf{A}} = \mathbf{A} + z \cdot \mathbf{e}_{i^*} \mathbf{e}_{j^*}^T} \,.$$

$$z \cdot e_{i^*} e_{j^*}^T = \begin{array}{cc} \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & z & & \vdots \\ \vdots & & & & \vdots \\ 0 & \cdots & \cdots & & 0 \end{bmatrix} & \leftarrow i^* \\ \uparrow j^* & \end{array}$$

General rank-1 modification:

$$\tilde{A} = A + uv^T \qquad u, v \in \mathbb{R}^n \setminus \{0\}$$

Trick: Use block elimination on the block partitioned system:

$$\underbrace{\begin{bmatrix} A & u \\ v^T & -1 \end{bmatrix}}_{(n+1) \times (n+1)\text{-matrix}} \begin{bmatrix} \tilde{x} \\ \xi \end{bmatrix} = \begin{bmatrix} \tilde{b} \\ 0 \end{bmatrix} \qquad (*)$$

Verify: (I) $A\tilde{x} + u\xi = \tilde{b}$

(II) $v^T\tilde{x} - \xi = 0 \iff \xi = v^T\tilde{x}$

$(I)+(II) \iff A\tilde{x} + uv^T\tilde{x} = \tilde{b}$

$\iff \underbrace{(A+uv^T)}_{\tilde{A}}\tilde{x} = \tilde{b}$

Solving for $\tilde{x}$ in (*) is equiv. to solving for $\tilde{x}$ in (**)

Block eliminating $\tilde{x}$:

$$\tilde{x} = A^{-1}(\tilde{b} - u\xi) \leftarrow$$

$$\Rightarrow v^T A^{-1}(\tilde{b} - u\xi) - \xi = 0$$

$$(v^T A^{-1} u + 1)\xi = v^T A^{-1}\tilde{b}$$

$$\xi = \frac{v^T A^{-1}\tilde{b}}{v^T A^{-1} u + 1}$$

$$\tilde{x} = (A^{-1}\tilde{b}) - (A^{-1}u)\frac{v^T(A^{-1}\tilde{b})}{v^T(A^{-1}u)+1}$$

assuming LU decomp. of A already available: cost $O(n^2)$

[itself $O(n^3)$]

solving for rank-1 perturbation costs about the same as solving for A & new RHS

More generally: rank-k perturbation

Lemma 2.6.22. Sherman-Morrison-Woodbury formula

For regular $\mathbf{A} \in \mathbb{K}^{n,n}$, *and* $\mathbf{U}, \mathbf{V} \in \mathbb{K}^{n,k}$, $n, k \in \mathbb{N}$, $k \leq n$, *holds*

$$(\mathbf{A} + \mathbf{U}\mathbf{V}^H)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{I}_k + \mathbf{V}^H\mathbf{A}^{-1}\mathbf{U})^{-1}\mathbf{V}^H\mathbf{A}^{-1},$$

if $\mathbf{I}_k + \mathbf{V}^H\mathbf{A}^{-1}\mathbf{U}$ *is regular.*

$\tilde{A} = A + UV^H \qquad U, V \in \mathbb{K}^{n,k}$

$I + V^H A^{-1} U$ $k \times k$ matrix

$\rightsquigarrow$ solving for a small system only [k is small]

Under some conditions: $c(I + V^H A^{-1} U) \le c(A)\, c(A + UV^H)$

if problem is well-conditioned for $A, \tilde{A}$, then solving for $I + V^H A^{-1} U$ also well-conditioned

If $\tilde{A} - A$ has only a few nonzero columns (or rows)

$\rightsquigarrow$ computing $UV^H \approx \Theta(nk^2)$

2.7. Sparse Linear Systems

Sparse: most entries are zero

Notion 2.7.1. Sparse matrix

$\mathbf{A} \in \mathbb{K}^{m,n}$, $m, n \in \mathbb{N}$, is sparse, if

$$\mathrm{nnz}(\mathbf{A}) := \#\{(i,j) \in \{1,\dots,m\} \times \{1,\dots,n\}: a_{ij} \neq 0\} \ll mn\,.$$

More mathematical:

Definition 2.7.3. Sparse matrices

Given a strictly increasing sequences $m : \mathbb{N} \mapsto \mathbb{N}$, $n : \mathbb{N} \mapsto \mathbb{N}$, a family $(\mathbf{A}^{(l)})_{l\in\mathbb{N}}$ of matrices with $\mathbf{A}^{(l)} \in \mathbb{K}^{m_l,n_l}$ is sparse (opposite: dense), if

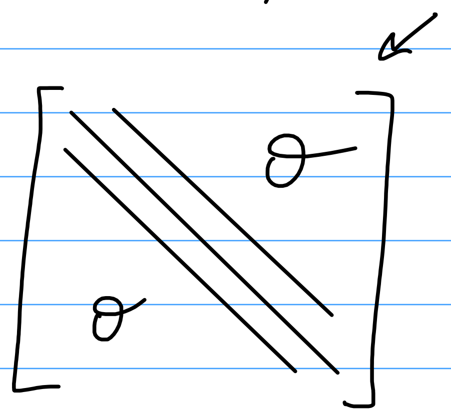
$$\lim_{l\to\infty} \frac{\mathrm{nnz}(\mathbf{A}^{(l)})}{n_l m_l} = 0\,.$$

Example: Arrow matrix $A_n \in \mathbb{R}^{n,n}$

$\mathrm{nnz}(A_n) = n + 2(n-1) = 3n-2$

$\lim_{n\to\infty} \frac{3n-2}{n^2} = 0$

Further examples: diagonal matrix, band matrix



2.7.1 Sparse Matrix Storage Formats

Goal: required memory $\sim nnz(A)$

cost of matrix x vector $\sim nnz(A)$

Example (2.7.6): COO / triplet format

$\rightsquigarrow$ list of triplets $(i, j, (A)_{i,j})$

```
struct TripletMatrix {
  size_t m,n;             // Number of rows and columns
  vector<size_t> I;       // row indices
  vector<size_t> J;       // column indices
  vector<scalar_t> a;     // values associated with index pairs
};
```

$\hookrightarrow$ have the same size $\geq nnz(A)$

This format allows repetition of index pairs

$\hookrightarrow$ this needs convention

triplets with same index pair (i, j):

values of these triplets are added to form $(A)_{i,j}$

suppose $I = (0, 0, 1, 3, 1, 0)$

$J = (2, 1, 0, 3, 0, 2)$

$a = (\underline{1}, 1, 1, 1, 2, \underline{3})$

$$A = \begin{pmatrix} 0 & 1 & 4 & 0 \\ 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

C++-code 2.7.7: Matrix×vector product $\mathbf{y} = \mathbf{A}\mathbf{x}$ in triplet format

```
void multTriplMatvec(const TripletMatrix &A,
                     const vector<scalar_t> &x,
                     vector<scalar_t> &y)
 for (size_t k=0; k<A.a.size(); k++) {
   y[A.I[k]] += A.a[k]*x[A.J[k]];
 }
```

Computational effort: $O(\text{A.a.size()})$

length of vector a (potentially $> nnz(A)$)

Example (2.7.9): CRS format (CCS format)

compressed row storage — column

sparse matrix $A = (a_{ij}) \in \mathbb{K}^{n,n}$

stored as 3 contiguous arrays:

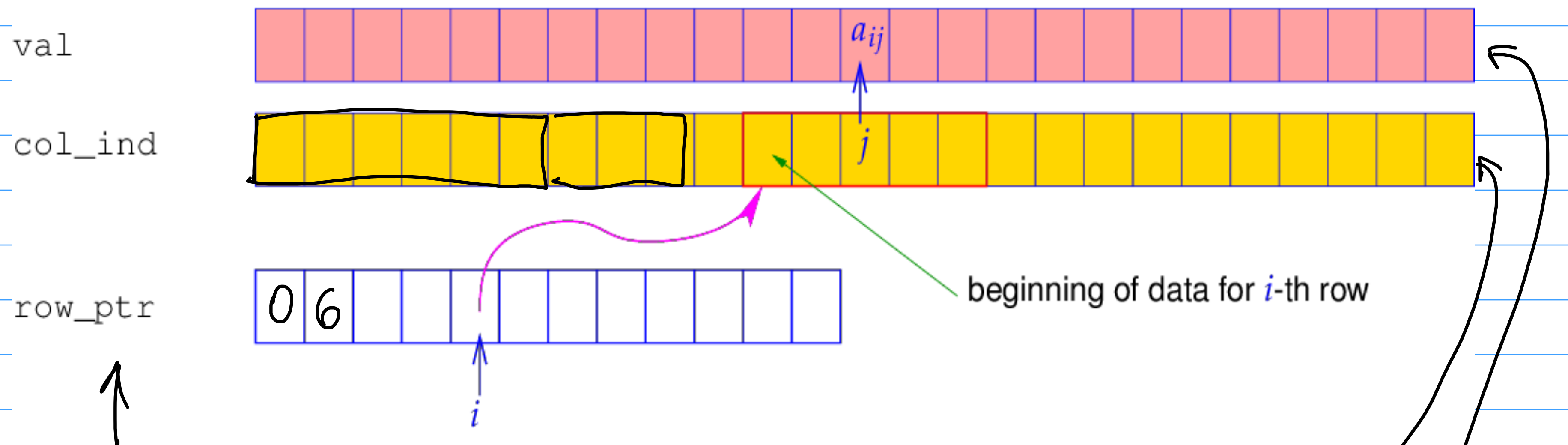
```
vector<scalar_t> val
  vector<size_t> col_ind
  vector<size_t> row_ptr
```

size $\operatorname{nnz}(\mathbf{A}) := \#\{(i,j) \in \{1,\dots,n\}^2,\ a_{ij} \neq 0\}$

size $\operatorname{nnz}(\mathbf{A})$

size $n+1$ & $\texttt{row\_ptr}[n+1] = \operatorname{nnz}(\mathbf{A}) + 1$ (sentinel value)

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



indicates index in above vectors at which new row is started

vector of column indices

vector of nonzero coeffs

Example:

$$
\mathbf{A} = \begin{bmatrix} 10 & 0 & 0 & 0 & -2 & 0 \\ 3 & 9 & 0 & 0 & 0 & 3 \\ 0 & 7 & 8 & 7 & 0 & 0 \\ 3 & 0 & 8 & 7 & 5 & 0 \\ 0 & 8 & 0 & 9 & 9 & 13 \\ 0 & 4 & 0 & 0 & 2 & -1 \end{bmatrix}
$$

`val`-vector:

10	-2	3	9	3	7	8	7	3 ... 9	13	4	2	-1

`col_ind`-array:

1	5	1	2	6	2	3	4	1 ... 5	6	2	5	6

`row_ptr`-array:

1	3	6	9	13	17	20

[indexing starting at 1]

val(4) col_ind(4) = 2 $\Rightarrow$ $\underline{j=2}$

row_ptr(i) $\leq$ 4 < row_ptr(i+1)

$\underline{i=2}$

Q: Find col_ind, row_ptr

$$
A_1 = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \quad A_2 = \begin{bmatrix} a & b & c \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad A_3 = \begin{bmatrix} 0 & 0 & 0 \\ a & b & c \\ 0 & 0 & 0 \end{bmatrix}
$$

[indexing from 0]

A_1, A_2, A_3: col_ind = [0 1 2]

row_ptr(i+1) = row_ptr(i) + nnz($(A)_{i,j}$)

row_ptr(n+1) = nnz($A_{1,2,3}$)

A_1: row_ptr = [0 1 2 3]

A_2: row_ptr = [0 3 3 3]

A_3: row_ptr = [0 0 3 3]

CCS: compressed Column-storage format

[column-major]

Sparse matrices in EIGEN:

```
SparseMatrix<double> Asp(1000,1000);
SparseMatrix<double,RowMajor> Bsp;
```

← default is CCS

← CRS

Initialization of sparse matrix:

```
Asp.insert(i,j) = v_ij;
```

↑ inserts a new element at index i, j

insert( ) assumes that this element does not yet exist

```
Asp.coeffRef(i,j) = v_ij;
Asp.coeffRef(i,j) += w_ij;
```

↑ updates entry at index i, j

if insert( ) or coeffRef( ) are used: need

```
Asp.makeCompressed();
```

← at the end to get CCS/CRS format

Note: when inserting new elements

→ possibly multiple reallocations during insertion procedure

cost of inserting new element ~ $\Theta(nnz)$

↑ current # of nonzeros

Two ways to avoid this:

1. Use triplet format for initialization & then change to CRS / CCS format

```
std::vector <Eigen::Triplet <double > > triplets;
// ..  fill the std::vector triplets ..
Eigen::SparseMatrix<double, Eigen::RowMajor> spMat(rows, cols);
spMat.setFromTriplets(triplets.begin(), triplets.end());
```

↑ takes a vector of triplets (i, j, value(i,j)) & outputs a sparse matrix in CRS / CCS store triplets explicitly in an std::vector & don't use complex data structure.

Triplet in EIGEN:

```
unsigned int row_idx = 2;
unsigned int col_idx = 4;
double value = 2.5;
Eigen::Triplet<double> triplet(row_idx,col_idx,value);
std::cout << '(' << triplet.row() << ',' << triplet.col()
          << ',' << triplet.value() << ')' << std::endl;
```

2. "Reserve" enough space in each row (if in RowMajor) for nonzero entries ⇝ helpful if we have good estimate of nnz for each row

C++11-code 2.7.21: Accessing entries of a sparse matrix: potentially inefficient!

```
unsigned int rows, cols, ~~max_no_nnz_per_row~~ nr;
.....
SparseMatrix<double, RowMajor> mat(rows, cols);
mat.reserve(RowVectorXi::Constant(cols, ~~max_no_nnz_per_row~~ nr));
// do many (incremental) initializations
for ( ) {
  mat.insert(i,j) = value_ij;
  mat.coeffRef(i,j) += increment_ij;
}
mat.makeCompressed();
```

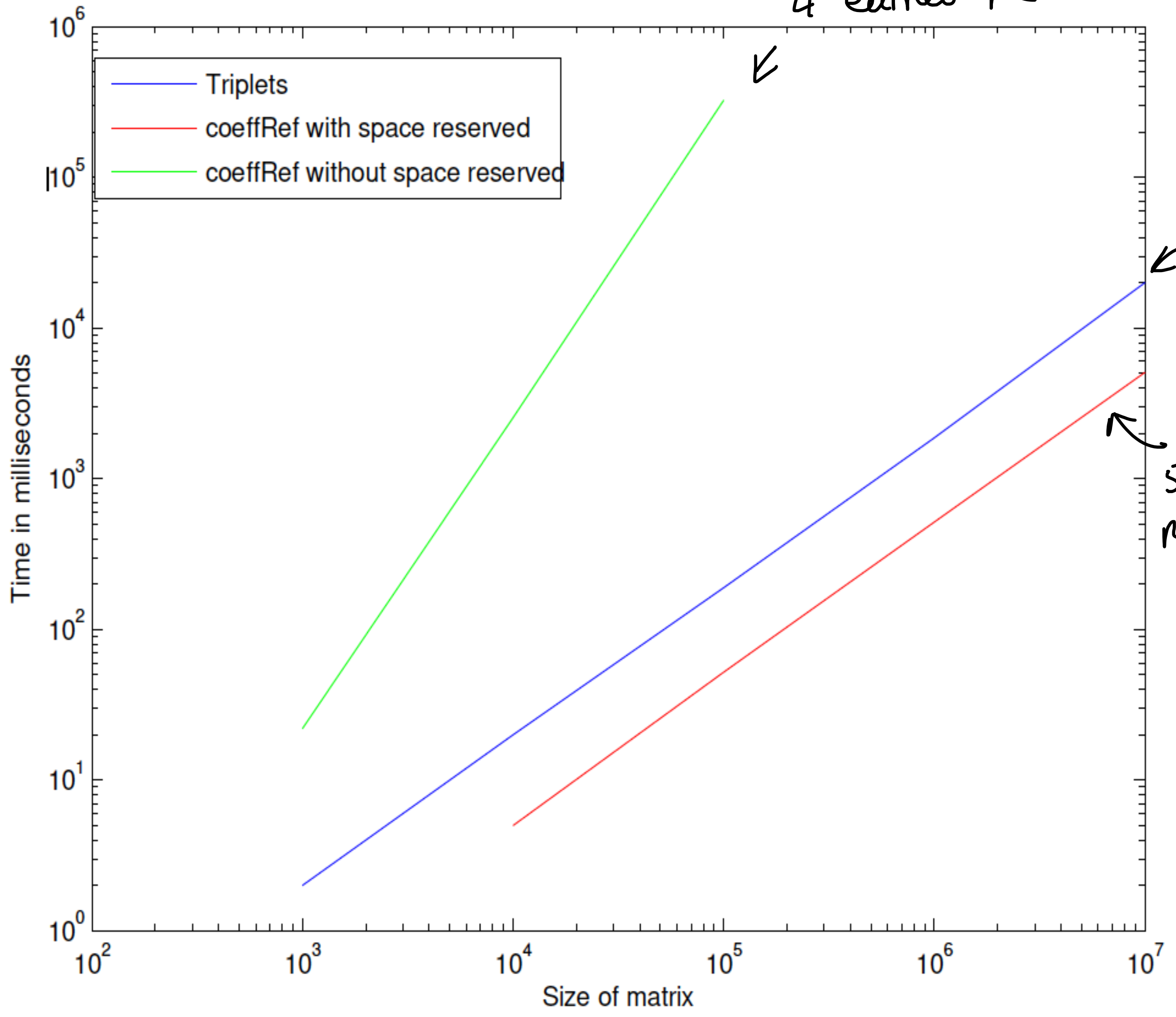
$\underbrace{(nr, nr, \dots, nr)}_{\text{of length cols}}$

reserve space for nr nonzero entries per row

<u>Example</u>: Runtimes of initialization of band matrix with bandwidth 2 (i.e. 5 nonzero diagonals)

⇒ max of 5 nonzeros per row

(2) 4 entries per row are reserved



triplets

(2) 5 entries per row reserved

Efficient sparse initialization : $\Theta(n)$

$nnz = \Theta(n)$

Note: it's advisable to also use reserve with triplet format!

```
std::vector <Eigen::Triplet <double >    > triplets;
 triplets.reserve(nnz);
// ..   fill the std::vector triplets ..
Eigen::SparseMatrix<double, Eigen::RowMajor> spMat(rows, cols);
spMat.setFromTriplets(triplets.begin(), triplets.end());
```

2.7.4 Direct solution of sparse LSE

Sparse matrix is stored as dense $\rightarrow$ solver won't exploit sparsity

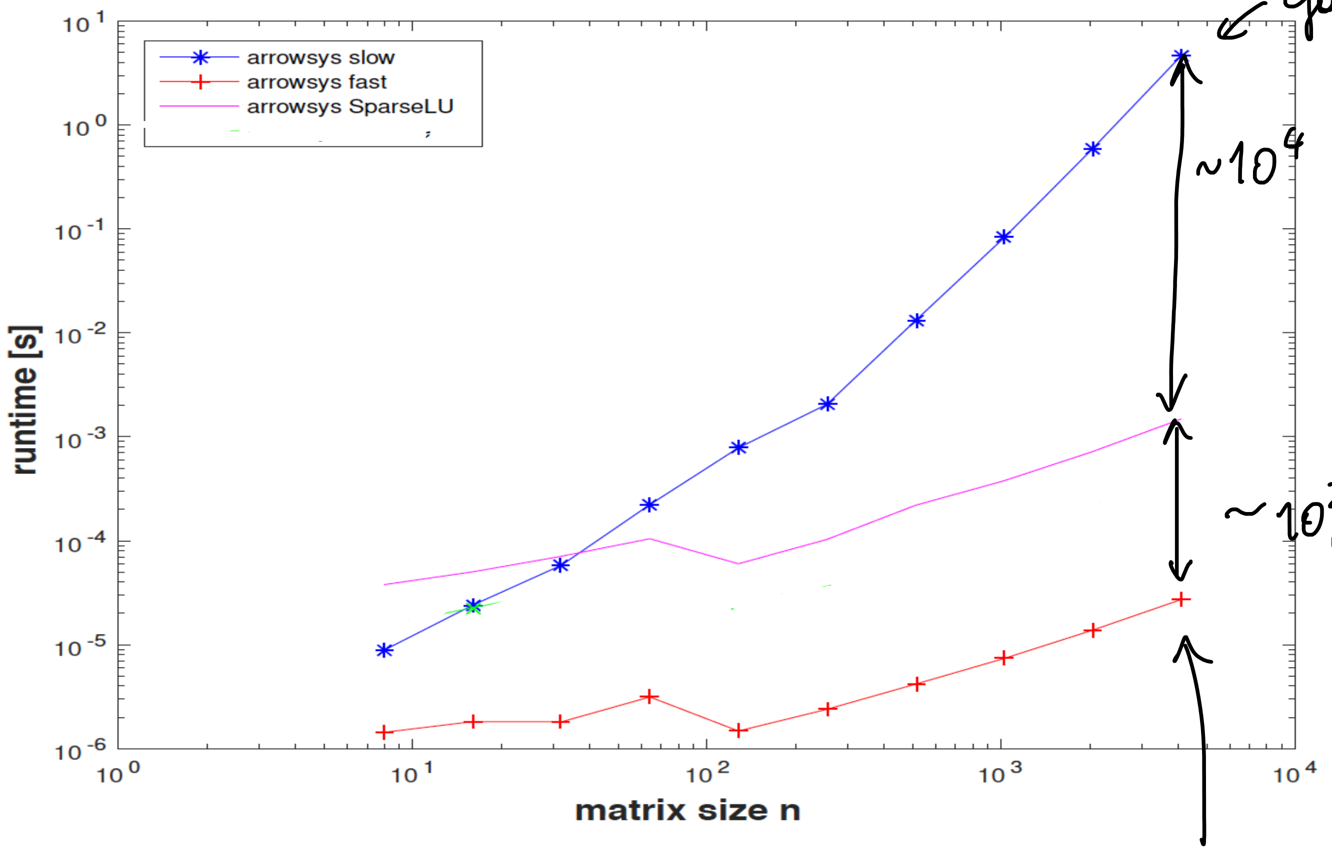
Sparse matrix format: there are solvers that exploit sparsity & avoid unnecessary calculations (working with zero entries)

SparseLU: "clever" LU decomposition of sparse matrix

C++-code 2.7.36: Function for solving a sparse LSE with EIGEN ➜ GITLAB

```cpp
using SparseMatrix = Eigen::SparseMatrix<double>;
// Perform sparse elimination
void sparse_solve(const SparseMatrix& A, const VectorXd& b, VectorXd&
   x) {
  Eigen::SparseLU<SparseMatrix> solver(A);
  x = solver.solve(b);
}
```

Example: LSE with arrow matrix



gaussian elimination

$\sim 10^4$

$\sim 10^2$ sparseLU: general (pivoting, ...)

clever block elimination

Cost of sparse solvers:

between $\Theta(nnz^{3/2})$ and $\Theta(nnz^{5/2})$

should only depend on nnz

Sparse matrix solvers: very sophisticated

→ use them & don't implement yourself

3. Direct methods for Linear Least Squares Problems

Motivation: Problem of parameter estimation

Model: $f(x) = a_1 x_1 + \dots + a_n x_n \quad f: \mathbb{R}^n \to \mathbb{R}$

Suppose we have series of measurements

$$\left(x^{(k)}, y^{(k)}\right)_{k=1}^{n} \qquad x^{(k)} \in \mathbb{R}^n, \; y^{(k)} \in \mathbb{R}$$

where $x^{(k)} \mapsto y^{(k)} = f(x^{(k)})$

Goal: with this series of experiments estimate parameters $a_1, \dots, a_n$

Example: modelling causal relationships between parameters in biological systems

We can write this in matrix form:

$$\begin{pmatrix} x_1^{(1)} & x_2^{(1)} & \dots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \dots & x_n^{(2)} \\ \vdots & & & \\ x_1^{(n)} & x_2^{(n)} & \dots & x_n^{(n)} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

i.e. do n trials stored columnwise in a matrix $X \in \mathbb{R}^{n,n}$; outcomes stored in $y \in \mathbb{R}^n$

Estimate parameters by solving

$$X^T a = y \qquad a \in \mathbb{R}^n$$

"linear regression"

More generally:

$$f(x) = a_1 f_1(x) a_2 f_2(x) + \dots + a_n f_n(x)$$

Record $(x^{(k)}, y^{(k)})_{k=1}^n$ $\qquad y^{(k)} = f(x^{(k)})$

Estimate $a_1, \dots, a_n$ by solving

$$\begin{pmatrix} f_1(x^{(1)}) & f_2(x^{(1)}) & \dots & f_n(x^{(1)}) \\ \vdots & & & \\ f_1(x^{(n)}) & f_2(x^{(n)}) & \dots & f_n(x^{(n)}) \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

Example: Polynomial regression:

$$f(x) = a_1 + a_2 x + a_3 x^2 + \dots + a_n x^{n-1}$$

$$\begin{pmatrix} 1 & x^{(1)} & (x^{(1)})^2 & \dots & (x^{(1)})^{n-1} \\ \vdots & \vdots & & & \\ 1 & x^{(n)} & (x^{(n)})^2 & \dots & (x^{(n)})^{n-1} \end{pmatrix} \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

Regression with n parameters and n $(x^{(k)}, y^{(k)})$ pairs is problematic:

- In practice: would want to incorporate many more measurements to avoid overfitting.

$\rightsquigarrow$ this than yields an overdetermined system

- In general: won't have a solution (because neither model will be perfect nor will measurements be exact)

Why? overdetermined system:

$$(*) \quad \begin{bmatrix} \\ A \\ \\ \end{bmatrix} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} \\ b \\ \\ \end{bmatrix} \qquad A \in \mathbb{R}^{m,n} \quad m \gg n$$

range $\mathcal{R}(A) = \{ y \in \mathbb{R}^m : \exists\, x \in \mathbb{R}^n \text{ s.t. } Ax = y \}$

writing $(*)$ as $(A)_{:,1} x_1 + (A)_{:,2} x_2 + \ldots + (A)_{:,n} x_n = b$

$\dim \mathcal{R}(A) = \operatorname{rank}(A) \leq n$

$\Rightarrow \mathcal{R}(A)$ is at most an n-dim. subspace of $\mathbb{R}^m$

perturbing $\underset{\in \mathcal{R}(A)}{\underset{\uparrow}{b}}$ to b^δ: very likely that

$b^\delta \notin \mathcal{R}(A) \Rightarrow Ax = b^\delta$ is not solvable

Instead of solving exactly: only search for good approximation

$Ax \approx b$

More precisely: minimize norm of residual $Ax-b$

$$\| Ax-b \|_2$$

→ concept of least-squares solution!

3.1. Least squares solutions

Definition 3.1.3. Least squares solution

For given $\mathbf{A} \in \mathbb{K}^{m,n}$, $\mathbf{b} \in \mathbb{K}^m$ the vector $\mathbf{x} \in \mathbb{R}^n$ is a least squares solution of the linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$, if

$$\mathbf{x} \in \underset{\mathbf{y} \in \mathbb{K}^n}{\operatorname{argmin}} \|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2 ,$$

$$\Updownarrow$$

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 = \inf_{\mathbf{y} \in \mathbb{K}^n} \|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2 .$$

Example of parameter estimation

$$X^T a = y$$

$$a = \underset{p \in \mathbb{R}^n}{\operatorname{argmin}} \sum_{k=1}^{m} \left| (x^{(k)})^T \cdot p - y^{(k)} \right|^2$$

$$\operatorname{lsq}(\mathbf{A}, \mathbf{b}) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x} \text{ is a least squares solution of } \mathbf{A}\mathbf{x} = \mathbf{b}\} \subset \mathbb{R}^n . \tag{3.1.4}$$

$x \in \operatorname{lsq}(A,b)$: Ax is closest element in $R(A)$ to b

i.e. projection of b on $R(A)$

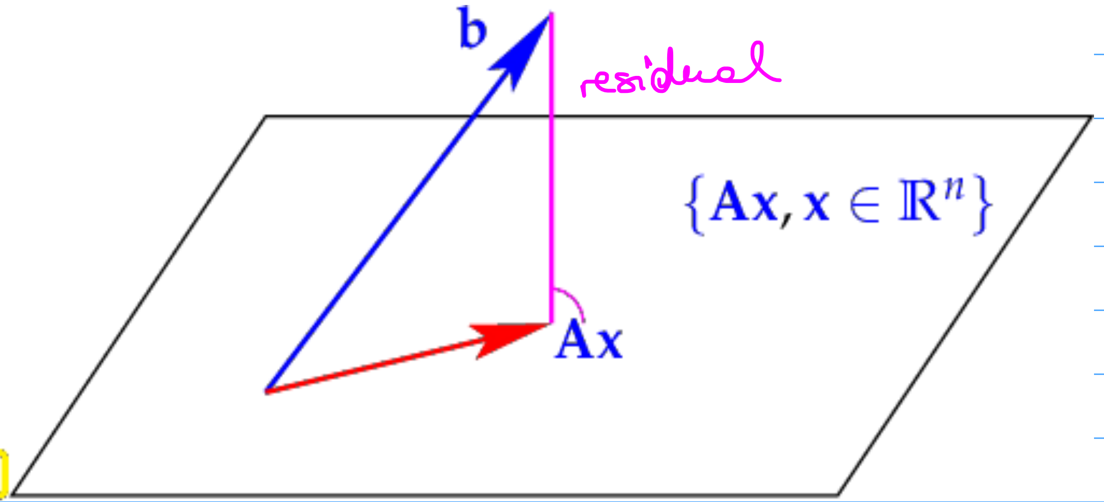


Fig. 96

Theorem 3.1.9. Existence of least squares solutions

For any $\mathbf{A} \in \mathbb{R}^{m,n}$, $\mathbf{b} \in \mathbb{R}^m$ a least squares solution of $\mathbf{A}\mathbf{x} = \mathbf{b}$ (→ Def. 3.1.3) exists.

The normal equation:

Recall basic LA result:

Lemma 3.1.21. Kernel and range of (Hermitian) transposed matrices

For any matrix $\mathbf{A} \in \mathbb{K}^{m,n}$ *holds*

$$\mathcal{N}(\mathbf{A}) = \mathcal{R}(\mathbf{A}^H)^\perp \;, \quad \mathcal{N}(\mathbf{A})^\perp = \mathcal{R}(\mathbf{A}^H) \;.$$

$z \in Y^\perp \iff \langle z, y \rangle = 0 \quad \forall y \in Y$

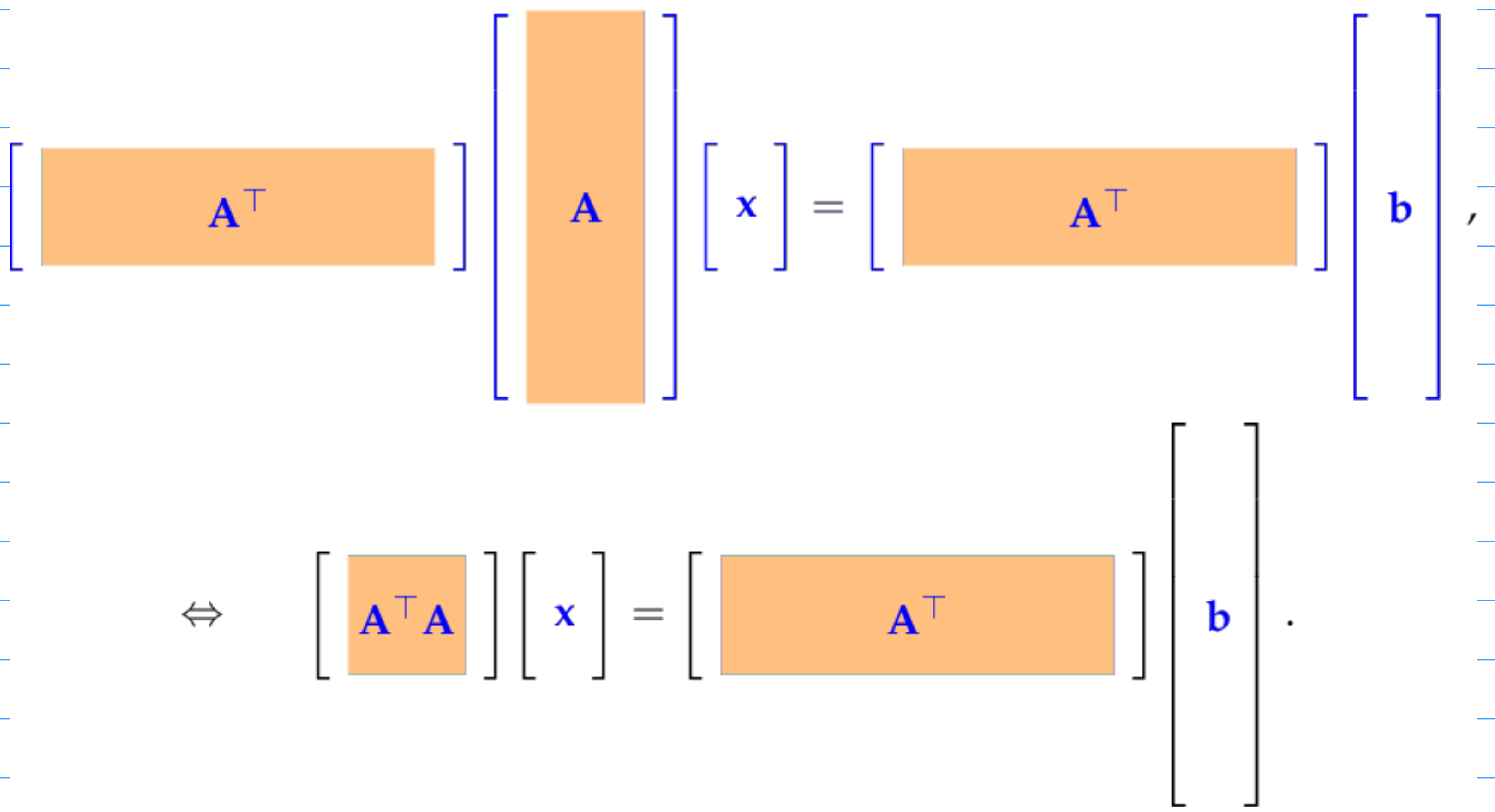
Theorem 3.1.10. Obtaining least squares solutions by solving normal equations

The vector $\mathbf{x} \in \mathbb{R}^n$ *is a least squares solution* (→ *Def. 3.1.3*) *of the linear system of equations* $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{A} \in \mathbb{R}^{m,n}$, $\mathbf{b} \in \mathbb{R}^m$, if and only if *it solves the* normal equations

$$\boxed{\mathbf{A}^\top \mathbf{A}\mathbf{x} = \mathbf{A}^\top \mathbf{b}} \;. \tag{3.1.11}$$

Proof: $x \in lsq(A,b) \iff Ax$ is closest element in $\mathcal{R}(A)$ to $b \iff Ax - b \in \mathcal{R}(A)^\perp = \mathcal{N}(A^T)$

$\iff A^T(Ax-b) = 0$.



$A^TAx = A^Tb$ LSE with $A^TA \in \mathbb{R}^{n,n}$

A^TA s.p. semi-def.

Uniqueness of least-squares solution?

In general: it need not be unique

Uniqueness equivalent to $\mathcal{N}(A^\top A) = \{0\}$

Theorem 3.1.18. Kernel and range of $\mathbf{A}^\top\mathbf{A}$

For $\mathbf{A} \in \mathbb{R}^{m,n}$, $m \geq n$, holds

$$\mathcal{N}(\mathbf{A}^\top\mathbf{A}) = \mathcal{N}(\mathbf{A}), \qquad (3.1.19)$$

$$\mathcal{R}(\mathbf{A}^\top\mathbf{A}) = \mathcal{R}(\mathbf{A}^\top). \qquad (3.1.20)$$

least sq. solution is unique $\Leftrightarrow$ $\mathcal{N}(A^\top A) = \{0\}$

$\Leftrightarrow$ $\mathcal{N}(A) = \{0\}$

$\Leftrightarrow$ $\mathcal{R}(A^\top) = \mathbb{R}^n$ $\Leftrightarrow$ $\operatorname{rank}(A) = n$

full rank condition

Corollary 3.1.22. Uniqueness of least squares solutions

If $m \geq n$ and $\mathcal{N}(\mathbf{A}) = \{\mathbf{0}\}$, then the linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{A} \in \mathbb{R}^{m,n}$, $\mathbf{b} \in \mathbb{R}^m$, has a unique least squares solution ($\rightarrow$ 3.1.3)

$$\mathbf{x} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}, \qquad (3.1.23)$$

that can be obtained by solving the normal equations (3.1.11).

3.1.3. Generalized solutions & Moore-Penrose Pseudo Inverse

How to overcome possible non-uniqueness?

Pick least-squares solution with minimal norm!

Definition 3.1.32. Generalized solution of a lineasr system of equations

The generalized solution $\mathbf{x}^\dagger \in \mathbb{R}^n$ of a linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{A} \in \mathbb{R}^{m,n}$, $\mathbf{b} \in \mathbb{R}^m$, is defined as

$$\mathbf{x}^\dagger := \operatorname{argmin}\{\|\mathbf{x}\|_2 : \mathbf{x} \in \operatorname{lsq}(\mathbf{A},\mathbf{b})\}. \qquad (3.1.33)$$

Theorem: The generalized solution x^+ is unique.

① Suppose $x_0 \in lsq(A,b) \Leftrightarrow A^T A x_0 = A^T b$

Then $lsq(A,b) = x_0 + \mathcal{N}(A^T A)$ Why?

Suppose $x_1 \in lsq(A,b) \Leftrightarrow A^T A x_1 = A^T b$

$$A^T A(x_0 - x_1) = A^T b - A^T b = 0$$

$$\Rightarrow x_0 - x_1 \in \mathcal{N}(A^T A) = \mathcal{N}(A)$$

($lsq(A,b)$ is affine subspace parallel to $\mathcal{N}(A)$)

$$lsq(A,b) = x_0 + \mathcal{N}(A)$$

[cf.: uniqueness of least-sq. sol.: if & only if $\mathcal{N}(A) = \{0\}$]

② generalized solutions are elements of $\mathcal{N}(A)^\perp$:

for any $x \in lsq(A,b)$

$$x = \tilde{x}_0 + \tilde{x}^N \qquad \tilde{x}_0 \in \mathcal{N}(A)^\perp$$

$$\tilde{x}^N \in \mathcal{N}(A)$$

$$\|x\|_2^2 = \|\tilde{x}_0\|_2^2 + \|\tilde{x}^N\|_2^2 \qquad [\mathbb{R}^n = \mathcal{N}(A) \oplus \mathcal{N}(A)^\perp]$$

$\tilde{x}_0 \in lsq(A,b)$:

$$A^T A(\tilde{x}_0) = A^T A(x - \tilde{x}^N) = A^T A x - \underbrace{A^T A \tilde{x}^N} = A^T b - 0 = A^T b$$

$$\Rightarrow \tilde{x}_0 \in lsq(A,b)$$

but: $\|\tilde{x}_0\|_2 \leq \|x\|_2$

$\Rightarrow$ minimal norm solution has to be in $\mathcal{N}(A)^\perp$ $\Rightarrow$ $x^+ \in \mathcal{N}(A)^\perp$

[if x^+ had nullspace component $\to$ we can find another l. sq. solution with $\leq$ norm]

③ Uniqueness:

Suppose $x^+, \tilde{x}^+$ are generalized solutions

$$A^TAx^+ = A^Tb, \quad A^TA\tilde{x}^+ = A^Tb$$

$$x^+, \tilde{x}^+ \in \mathcal{N}(A)^\perp \Rightarrow x^+ - \tilde{x}^+ \in \mathcal{N}(A)^\perp$$

$$A^TA(x^+ - \tilde{x}^+) = A^Tb - A^Tb = 0$$

$$\Rightarrow x^+ - \tilde{x}^+ \in \mathcal{N}(A)$$

$$x^+ - \tilde{x}^+ = 0 \Rightarrow x^+ = \tilde{x}^+$$

Formula for generalized solution:

$$x^+ \in \mathcal{N}(A)^\perp$$

$\Rightarrow$ given a basis $\{v_1, \dots, v_k\} \subset \mathbb{R}^n$ of $\mathcal{N}(A)^\perp$

($\dim \mathcal{N}(A)^\perp = k$)

$$V = [v_1 \dots v_k] \in \mathbb{R}^{n,k}$$

one can always $y \in \mathbb{R}^k$ s.t.

$$Vy = x^+$$

$$\Rightarrow V^TA^TAVy = V^TA^Tb$$

(reduced normal equations)

$(AV)^TAV$

$$\mathbf{V}^\top \mathbf{A}^\top \mathbf{A} \mathbf{V}\, \mathbf{y} = \mathbf{V}^\top \mathbf{A}^\top \mathbf{b} \tag{3.1.36}$$

$$\Updownarrow$$

$$\begin{bmatrix} \mathbf{V}^\top \end{bmatrix} \begin{bmatrix} \mathbf{A}^\top \end{bmatrix} \begin{bmatrix} \mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{y} \end{bmatrix} = \begin{bmatrix} \mathbf{V}^\top \end{bmatrix} \begin{bmatrix} \mathbf{A}^\top \end{bmatrix} \begin{bmatrix} \mathbf{b} \end{bmatrix} .$$

$k \times k$ system of equations

Note: by construction $\mathcal{N}(AV) = \{0\}$

$\Rightarrow \mathcal{N}(V^T A^T A V) = \{0\}$

$V^T A^T A V y = V^T A^T b$

is uniquely solvable

If y is unique solution $\Rightarrow$ $x^\dagger = Vy$

Theorem 3.1.37. Formula for generalized solution

Given $\mathbf{A} \in \mathbb{R}^{m,n}$, $\mathbf{b} \in \mathbb{R}^m$, the generalized solution $\mathbf{x}^\dagger$ of the linear system of equations $\mathbf{A}\mathbf{x} = \mathbf{b}$ is given by

$$\mathbf{x}^\dagger = \mathbf{V}(\mathbf{V}^\top \mathbf{A}^\top \mathbf{A} \mathbf{V})^{-1} (\mathbf{V}^\top \mathbf{A}^\top \mathbf{b}) ,$$

where $\mathbf{V}$ is any matrix whose columns form a basis of $\mathcal{N}(\mathbf{A})^\perp$.

Note: $V(V^T A^T A V)^{-1} V^T A^T$ is called

Moore–Penrose Pseudoinverse A^+ of A

A^+ does not depend on choice of V

3.2. Normal Equation Methods

Suppose we have A with full rank condition

$A \in \mathbb{R}^{m,n}$ $\quad$ $\mathrm{rank}(A) = n$ $\quad$ $m > n$

Algorithm: Normal equation method to solve full-rank least squares problem $\mathbf{A}\mathbf{x} = \mathbf{b}$

❶ Compute regular matrix $\mathbf{C} := \mathbf{A}^\top\mathbf{A} \in \mathbb{R}^{n,n}$.
❷ Compute right hand side vector $\mathbf{c} := \mathbf{A}^\top\mathbf{b}$.
❸ Solve s.p.d. (→ Def. 1.1.8) linear system of equations: $\mathbf{C}\mathbf{x} = \mathbf{c}$ → § 2.8.13

pos. def. $x^T C x = x^T A^T A x = \|Ax\|^2 > 0$

$\uparrow$
$x \neq 0$

step ❶: cost $O(mn^2)$
step ❷: cost $O(nm)$
step ❸: cost $O(n^3)$

▶ cost $O(n^2 m + n^3)$ for $m, n \to \infty$.

linear in m
assuming n small & fixed