

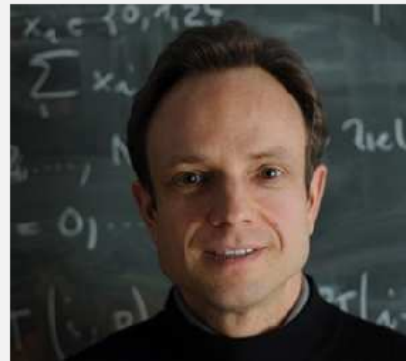
Course Video

Section 3.2: Normal Equation Methods

Prof. R. Hiptmair, SAM, ETH Zurich

Date: September 23, 2020

(C) Seminar für Angewandte Mathematik, ETH Zürich



Dependencies. Section 3.1.2, Section 2.5, [Section 2.8]

Duration: 5 minutes



Note: Possible minor *mismatch of video and tablet notes!*

[Corrections and updates can be incorporated into tablet notes only]

III. Direct Methods for Linear Least Squares Problems

3.2. Normal Equation Methods

Setting: $OD\text{-}LSE$ $Ax = b$, $A \in \mathbb{R}^{m,n}$, $m \geq n$, $b \in \mathbb{R}^m$

NEQ: $A^T A x = A^T b \Leftrightarrow \{x\} = \text{lsq}(A, b)$

Assume: $FRC \Leftrightarrow \text{rank}(A) = n \Leftrightarrow A^T A$ regular

Algorithm: Normal equation method to solve full-rank least squares problem $Ax = b$

- 1 Compute **regular** matrix $C := A^T A \in \mathbb{R}^{n,n}$.
- 2 Compute right hand side vector $c := A^T b$.
- 3 Solve **s.p.d.** ($\rightarrow$ Def. 1.1.2.6) linear system of equations: $Cx = c \rightarrow \S 2.8.0.13$

Definition 1.1.2.6. Symmetric positive definite (s.p.d.) matrices $\rightarrow$,

$M \in \mathbb{K}^{n,n}$, $n \in \mathbb{N}$, is **symmetric (Hermitian) positive definite** (s.p.d.), if

$$M = M^H \text{ and } \forall x \in \mathbb{K}^n: x^H M x > 0 \Leftrightarrow x \neq 0.$$

If $x^H M x \geq 0$ for all $x \in \mathbb{K}^n \triangleright M$ **positive semi-definite**.



$$x^T C x = x^T A^T A x = (A x)^T (A x) = \|A x\|_2^2 > 0 \iff x \neq 0$$

FRC : $\mathcal{N}(A) = \{0\}$

C++ code 3.2.0.1: Solving a linear least squares problem via normal equations

```

2  /// Solving the overdetermined linear system of equations
3  /// Ax = b by solving normal equations (3.1.2.2)
4  /// The least squares solution is returned by value
5  VectorXd normeqsolve(const MatrixXd &A, const VectorXd &b) {
6      if (b.size() != A.rows()) throw runtime_error("Dimension mismatch");
7      // Use Cholesky factorization for s.p.d. system matrix, § 2.8.0.13
8      VectorXd x = (A.transpose()*A).llt().solve(A.transpose()*b);
9      return x;
10 }

```

① $O(mn^2)$ ③ ② $O(mn)$

Special LU-type factorization for s.p.d. Matrices : $O(n^3)$

$$\begin{bmatrix} A^T \end{bmatrix} \begin{bmatrix} A \end{bmatrix} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} A^T \end{bmatrix} \begin{bmatrix} b \end{bmatrix}$$

① ②

$$\Leftrightarrow \begin{bmatrix} A^T A \end{bmatrix} \begin{bmatrix} x \end{bmatrix} = \begin{bmatrix} A^T \end{bmatrix} \begin{bmatrix} b \end{bmatrix}$$

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

step ①: cost $O(mn^2)$
 step ②: cost $O(nm)$
 step ③: cost $O(n^3)$

} $\blacktriangleright$ cost $O(n^2m + n^3)$ for $m, n \rightarrow \infty$.

Linear in m , if n fixed/small

Ex 3.2.0.4: NEQ & roundoff errors



$$A = \begin{bmatrix} 1 & 1 \\ \delta & 0 \\ 0 & \delta \end{bmatrix} \Rightarrow A^T A = \begin{bmatrix} 1+\delta^2 & 1 \\ 1 & 1+\delta^2 \end{bmatrix}$$

If $\delta \approx \sqrt{\text{EPS}}$, then $1 + \delta^2 = 1$ in $\mathbb{M}$.

Computed:

$$A^T A = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$1 = (1+\delta^2)(1+\epsilon) \Rightarrow |\epsilon| = \frac{\delta^2}{1+\delta^2} < \text{EPS}$$

$\rightarrow$ complies w/ "axiom of floating point arithmetic"

C++ code 3.2.0.5:

```

2  int main() {
3      MatrixXd A(3,2);
4      // Inquire about machine precision → Ex. 1.5.3.12
5      double eps = std::numeric_limits<double>::epsilon();
6      // « initialization of matrix → § 1.2.1.3
7      A << 1, 1, sqrt(eps), 0, 0, sqrt(eps);
8      // Output rank of A^T A
9      std::cout << "Rank of A: " << A.fullPivLu().rank() << std::endl
10         << "Rank of A^T A: "
11         << (A.transpose() * A).fullPivLu().rank() << std::endl;
12      return 0;
13 }

```

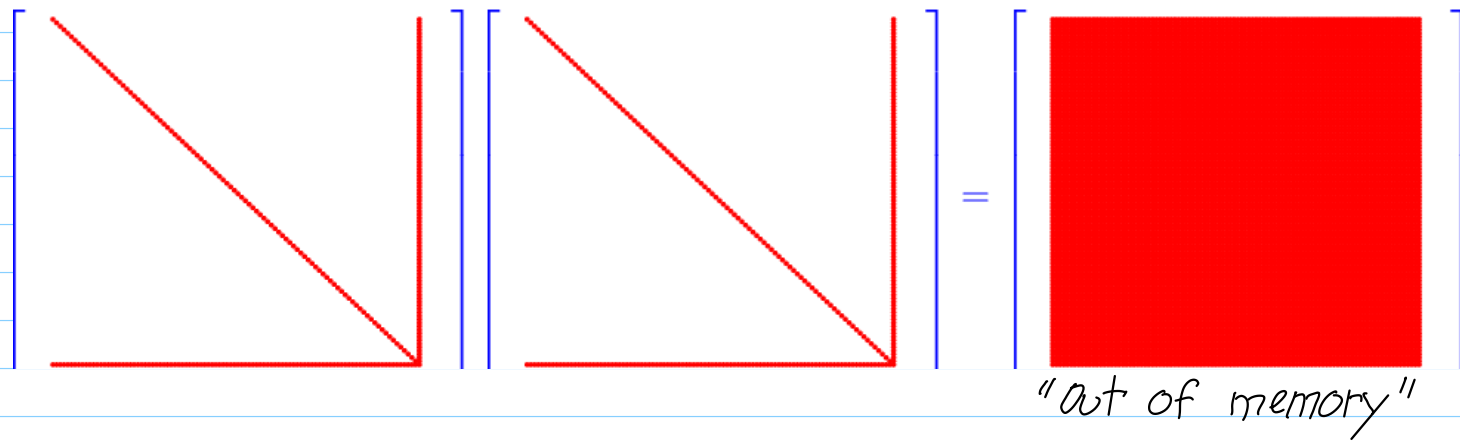
Output:

1 Rank of A: 2
 2 Rank of A^T A: 1

③

Rem 3.2.0.6: NEQ: Loss of sparsity

[A concern only for large m, n]
 A sparse $\Rightarrow A^T A$ sparse



A trick to avoid computing $A^T A$:

§ 3.2.0.7 Extended normal equations

NEQ $A^T A x = A^T b \Leftrightarrow A^T (\underbrace{b - Ax}_{\text{residual } r}) = 0$

Idea: r as new unknown & split equations

$$\left. \begin{array}{l} r = b - Ax \\ A^T r = 0 \end{array} \right\} \Rightarrow \underbrace{\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix}}_{\in \mathbb{R}^{m+n, m+n}, \text{ but sparse, if } A \text{ sparse}} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}$$

Review questions:

Answer review questions 3.2.0.11 without aids

4

5

⑦