

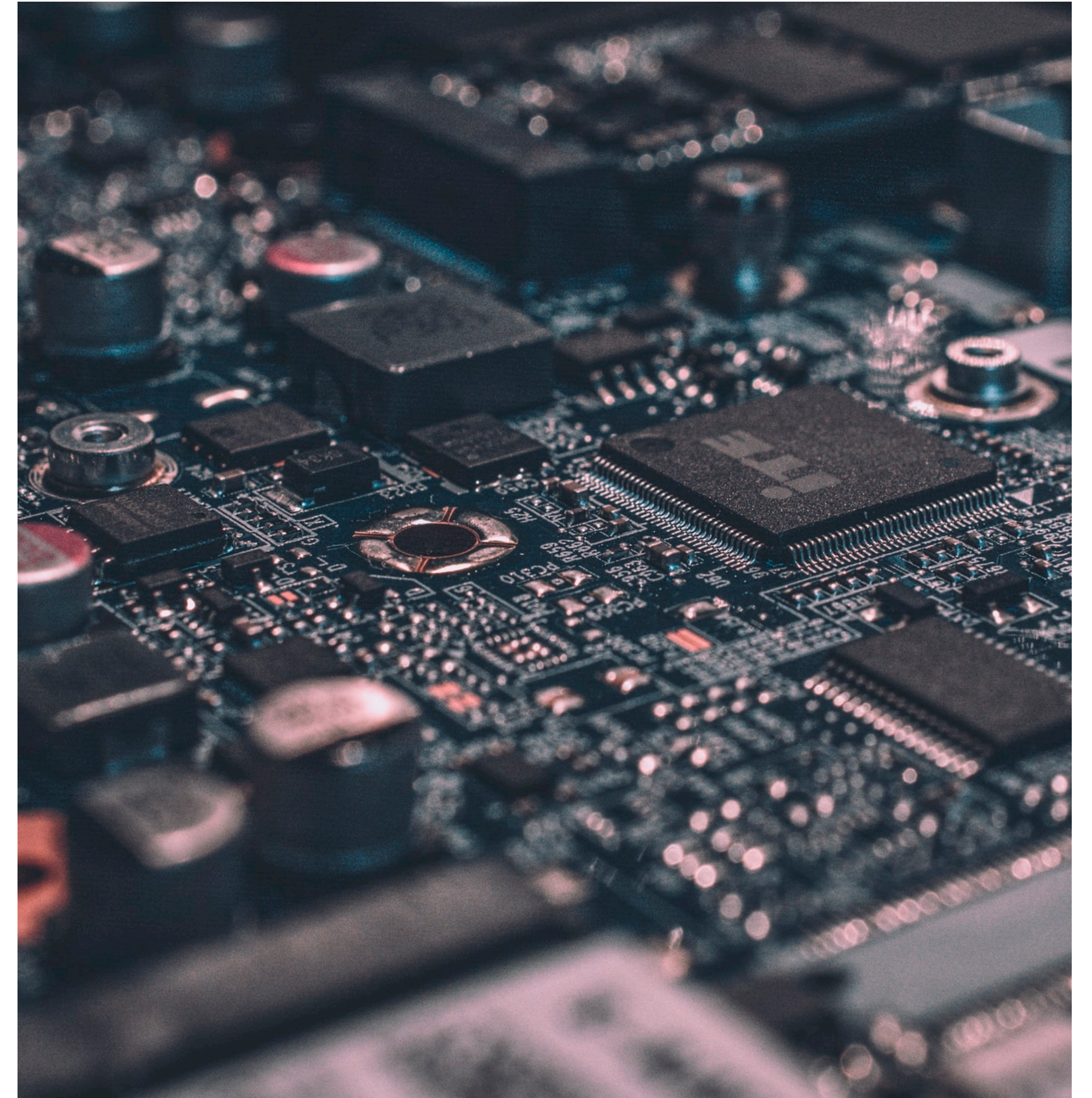
ORF307 – Optimization

3. Least squares

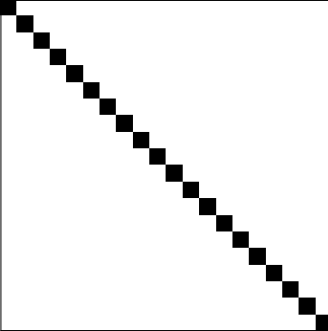
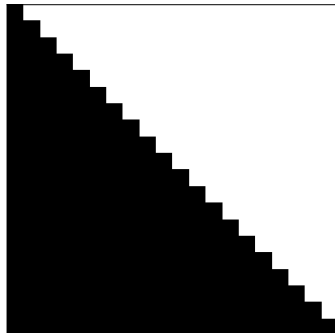
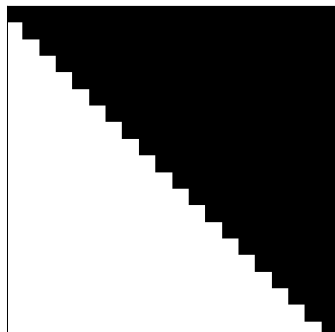
Recap

Flop counts

- Computers store real numbers in **floating-point format**
- Basic arithmetic operations (addition, multiplication, etc...) are called **floating point operations (flops)**
- **Algorithm complexity:** total number of flops needed as function of dimensions
- **Execution time** \approx (flops)/(computer speed)
[Very grossly approximated]
- Modern computers can go at 1 Gflop/sec (10^9 flops/sec)



Summary of easy linear systems

		method	flops
	diagonal $A = \text{diag}(a_1, \dots, a_n)$	$x_i = b_i / a_i$	n
	lower triangular $A_{ij} = 0 \text{ for } i < j$	forward substitution	n^2
	upper triangular $A_{ij} = 0 \text{ for } i > j$	backward substitution	n^2
	permutation $P_{ij} = 1 \text{ if } j = \pi_i \text{ else } 0$	inverse permutation	0

The factor-solve method for solving $Ax = b$

1. **Factor** A as a product of simple matrices:

$$A = A_1 A_2 \cdots A_k, \quad \longrightarrow \quad A_1 A_2 \cdots A_k x = b$$

(A_i diagonal, upper/lower triangular, permutation, etc)

2. **Compute** $x = A^{-1}b = A_k^{-1} \cdots A_1^{-1}b$
by solving k “easy” systems



$$A_1 x_1 = b$$

$$A_2 x_2 = x_1$$

$$\vdots$$

$$A_k x = x_{k-1}$$

Note: step 2 is much cheaper than step 1

Multiple right-hand sides

You now have factored A and you want to solve d linear systems with different right-hand side m -vectors b_i

$$Ax = b_1 \quad Ax = b_2 \quad \dots \quad Ax = b_d$$

Factorization-caching procedure

1. Factor $A = A_1, \dots, A_k$ **only once** (expensive)
2. Solve all linear systems using **the same factorization** (cheap)

Solve many “at the price of one”

LU Factorization

Every invertible matrix A can be factored as

$$A = PLU \quad \longrightarrow \quad P^T A = LU$$

P permutation, L lower triangular, U upper triangular

Procedure

- Similar to Gaussian elimination (rewrite $Ax = b$ as $Ux = d$)
- Permutation P avoids divisions by 0
- One of infinite possible combinations of P, L, U

Complexity

- $(2/3)n^3$ flops
- Less if A has special structure (sparse, diagonal, etc)

LU Solution

$$Ax = b, \quad \Rightarrow \quad PLUx = b$$

Iterations

1. *Permutation*: Solve $Px_1 = b$ (0 flops)
2. *Forward substitution*: Solve $Lx_2 = x_1$ (n^2 flops)
3. *Backward substitution*: Solve $Ux = x_2$ (n^2 flops)

Complexity

- Factor + solve: $(2/3)n^3 + 2n^2 \approx (2/3)n^3$ (for large n)
- Just solve (prefactored): $2n^2$

LL^T (Cholesky) Factorization

Every positive definite matrix A can be factored as

$$A = LL^T$$

L lower triangular

Procedure

- Works only on **symmetric with positive definite** matrices
- No need to permute as in LU
- One of infinite possible choices of L

Complexity

- $(1/3)n^3$ flops (half of LU decomposition)
- Less if A has special structure (sparse, diagonal, etc)

LL^T (Cholesky) Solution

$$Ax = b, \quad \Rightarrow \quad LL^T x = b$$

Iterations

1. *Forward substitution*: Solve $Lx_1 = b$ (n^2 flops)
2. *Backward substitution*: Solve $L^T x = x_1$ (n^2 flops)

Complexity

- Factor + solve: $(1/3)n^3 + 2n^2 \approx (1/3)n^3$ (for large n)
- Just solve (prefactored): $2n^2$

Today's lecture

Least squares

- Least squares optimization
- Gram matrix
- Solving least squares
- Example

Least squares optimization

Solving overdetermined linear systems

You have an *overdetermined* $m \times n$ linear system ($m > n$)

$$Ax = b$$

(with tall A)

example

Typically no solution

$$\begin{bmatrix} 2 & 0 \\ -1 & 1 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

Least squares problem

residual vector

$$r = Ax - b$$



Goal: make it as small as possible

minimize $\|r\|$

Least squares problem

minimize $\|Ax - b\|_2^2$

- x is the *decision variable*
- $\|Ax - b\|_2^2$ is the *objective function*

Least squares solution

$$\text{minimize } \|Ax - b\|_2^2$$

optimality condition x^* is a *solution* of least squares problem if
 $\|Ax^* - b\|^2 \leq \|Ax - b\|^2$, for any n -vector x

x^* need not (and usually does not) satisfy $Ax^* = b$

What happens if x^* does satisfy $Ax^* = b$?

Column interpretation

$$A = \begin{bmatrix} a_1, \dots, a_n \end{bmatrix}, \quad a_1, \dots, a_n \text{ are columns of } A$$

Goal: find a linear combination of the columns of A that is closest to b

$$\|Ax - b\|^2 = \|(x_1 a_1 + \dots + x_n a_n) - b\|^2$$

If x^* is a solution of the least squares problem, the m -vector

$$Ax^* = x_1^* a_1 + \dots + x_n^* a_n$$

is the closest to b among all linear combinations of the columns of A

Row interpretation

$$A = \begin{bmatrix} \tilde{a}_1^T \\ \vdots \\ \tilde{a}_m^T \end{bmatrix}, \quad \tilde{a}_1^T, \dots, \tilde{a}_m^T \text{ are rows of } A$$

The residual components are $r_i = \tilde{a}_i^T x - b_i$

Goal minimize sum of squares of the residuals

$$\|Ax - b\|^2 = (\tilde{a}_1^T x - b_1)^2 + \dots + (\tilde{a}_m^T x - b_m)^2$$

Comparison

- Solving $Ax = b$ forces all residuals to be zero
- Least squares attempts to make them small

Example

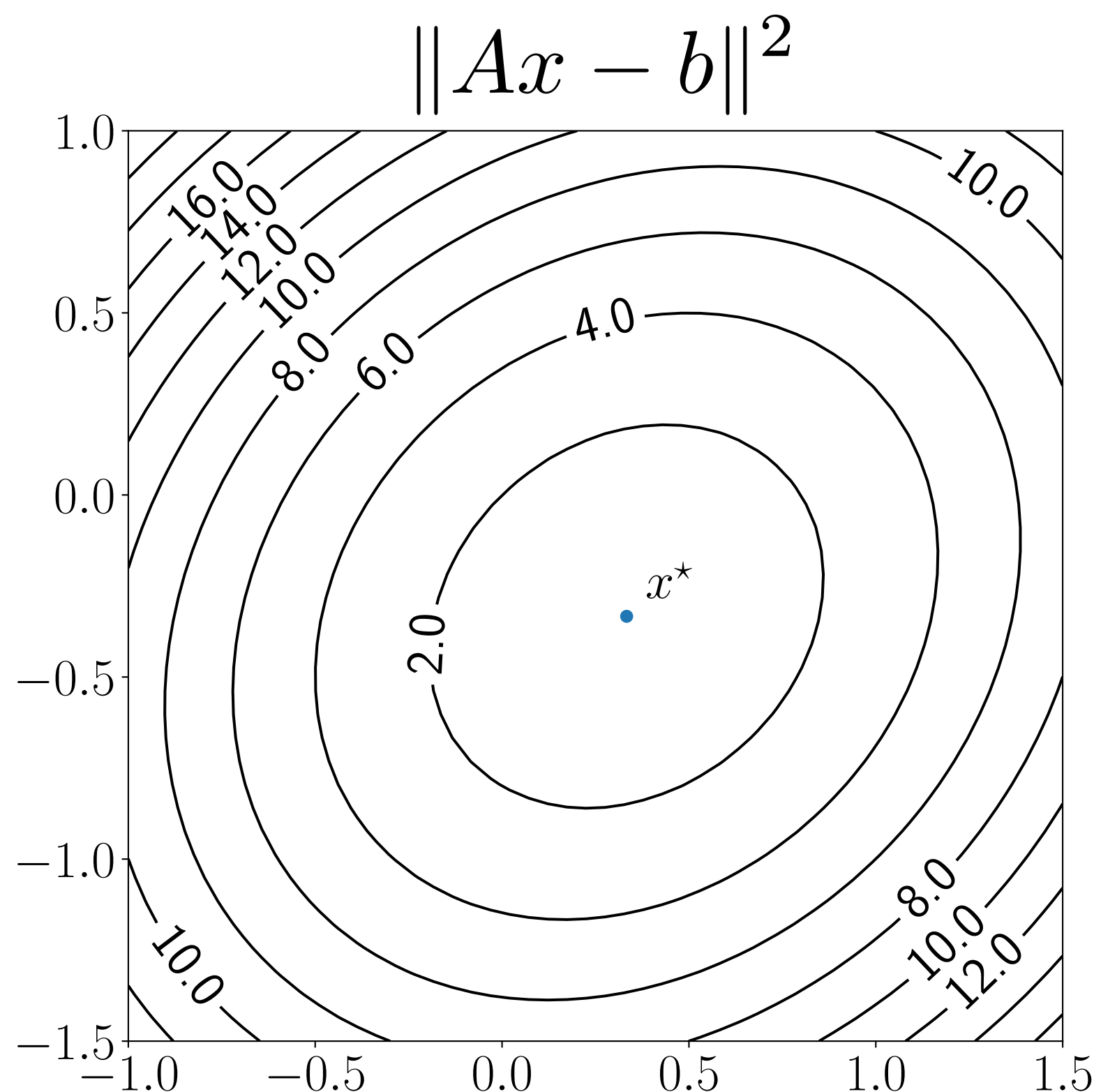
$$\begin{matrix} A \\ \begin{bmatrix} 2 & 0 \\ -1 & 1 \\ 0 & 2 \end{bmatrix} \end{matrix} \begin{matrix} \\ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{matrix} = \begin{matrix} b \\ \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \end{matrix}$$

Least squares problem

Compute x to minimize

$$\|Ax - b\|^2 = (2x_1 - 1)^2 + (-x_1 + x_2)^2 + (2x_2 + 1)^2$$

Solution $x^* = (1/3, -1/3)$ (via calculus)



Interpretations

- $\|Ax^* - b\|^2 = 2/3$ smallest possible value of $\|Ax - b\|^2$
- $Ax^* = (2/3, -2/3, -2/3)$ is the linear combination of columns of A closest to b

Gram matrix

Gram matrix

Given an $m \times n$ matrix A with columns a_1, \dots, a_n

the **Gram matrix** of A is

$$A^T A = \begin{bmatrix} a_1^T a_1 & a_1^T a_2 & \dots & a_1^T a_n \\ a_2^T a_1 & a_2^T a_2 & \dots & a_2^T a_n \\ \vdots & \vdots & \ddots & \vdots \\ a_n^T a_1 & a_n^T a_2 & \dots & a_n^T a_n \end{bmatrix}$$

Very useful in least squares problems

Gram matrix

Invertibility

A has linearly independent columns if and only if $A^T A$ is invertible

Proof

We show that $Ax = 0 \iff A^T Ax = 0$

\Rightarrow if $Ax = 0$ then we can write

$$A^T Ax = A^T (Ax) = A^T 0 = 0$$

\Leftarrow if $A^T Ax = 0$ then we can write

$$0 = x^T 0 = x^T (A^T Ax) = x^T A^T Ax = \|Ax\|^2$$

which implies that $Ax = 0$ (definition of norm)



Positive (semi)definiteness of Gram matrix

Positive semidefinite (always)

$$x^T A^T A x = (Ax)^T (Ax) = \|Ax\|^2 \geq 0, \quad \text{for any } n\text{-vector } x$$

Positive definite

$A^T A$ is positive definite if and only if A has linearly independent columns

Proof

If the columns of A are linearly independent, then

$$Ax \neq 0 \text{ for any } x \neq 0$$

Therefore, $x^T A^T A x = \|Ax\|^2 > 0$ (definition of norm) 

Solving least squares problems

Main assumption

Least squares problem

$$\text{minimize } \|Ax - b\|_2^2$$

A has **linearly independent columns**

True in most practical examples such as data fitting (next lecture)

Calculus derivation

$$f(x) = \|Ax - b\|^2 = \sum_{i=1}^m \left(\sum_{j=1}^n A_{ij}x_j - b_i \right)^2$$

The solution x^* satisfies

$$\nabla f(x^*)_k = \frac{\partial f}{\partial x_k}(x^*) = 0,$$

for $k = 1, \dots, n$



$$\begin{aligned} \frac{\partial f}{\partial x_k}(x) &= 2 \sum_{i=1}^m \left(\sum_{j=1}^n A_{ij}x_j - b_i \right) (A_{ik}) \\ &= 2 \sum_{i=1}^m (A^T)_{ki} (Ax - b)_i \\ &= 2(A^T(Ax - b))_k \end{aligned}$$

Calculus derivation in vector form

$$f(x) = \|Ax - b\|^2 = (Ax - b)^T (Ax - b) = x^T A^T A x - 2(A^T b)^T x + b^T b$$

$$\nabla f(x^*) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(x^*) \\ \vdots \\ \frac{\partial f}{\partial x_n}(x^*) \end{bmatrix} = 2A^T A x^* - 2A^T b = 2A^T (Ax^* - b) = 0$$

$n \times n$
square
linear system

normal equations

$(A^T A)x^* = A^T b$

Optimality

For x^* such that $A^T A x^* = A^T b$, we have

$$\begin{aligned}\|Ax - b\|^2 &= \|(Ax - Ax^*) + (Ax^* - b)\|^2 \\&= \|A(x - x^*)\|^2 + \|Ax^* - b\|^2 + 2(A(x - x^*))^T (Ax^* - b) \\&= \|A(x - x^*)\|^2 + \|Ax^* - b\|^2 + 2(x - x^*)^T \underbrace{A^T (Ax^* - b)}_{(A^T (Ax^* - b) = 0)} \\&= \|A(x - x^*)\|^2 + \|Ax^* - b\|^2\end{aligned}$$

Therefore, for any x , we have

$$\|Ax - b\|^2 \geq \|Ax^* - b\|^2$$

If equality holds, $A(x - x^*) = 0 \Rightarrow x = x^*$
since columns of A are linearly independent

Solving normal equations

$$(A^T A)x^* = A^T b$$

Inversion

$$x^* = (A^T A)^{-1} A^T b$$

Pseudo-inverse

$$A^\dagger = (A^T A)^{-1} A^T$$

Factor-solve method

A has linearly independent columns



$A^T A$ is **symmetric positive-definite**

Cholesky factorization

$$A^T A = LL^T$$

Which method is faster?

Solving normal equations with Cholesky

1. Form linear system $A^T A x = A^T b$
 - Form $M = A^T A$ ($2mn^2$ flops)
 - Form $q = A^T b$: ($2mn$ flops)
2. Factor $M = LL^T$ ($(1/3)n^3$ flops)
3. Solve $LL^T x = q$ ($2n^2$ flops)
(with forward/backward substitution)

Complexity

- Factor + solve: $2mn^2 + 2mn + (1/3)n^3 + 2n^2 \approx 2mn^2$
- Solve given a new b (prefactored): $2mn + 2n^2 \approx 2mn$

Example

Optimal advertising

m demographic groups
we want to advertise to



v^{des} is the m -vector
of desired views/impressions

n advertising channels
(web publishers, radio, print, etc.)



s is the n -vector
of purchases

$m \times n$ matrix A gives
demographic reach of channels



A_{ij} is the number of views
for group i and dollar spent
on channel j (1000/\$)

Views across demographic groups

$$v = As$$

Goal

minimize $\|As - v^{\text{des}}\|^2$

Optimal advertising Results

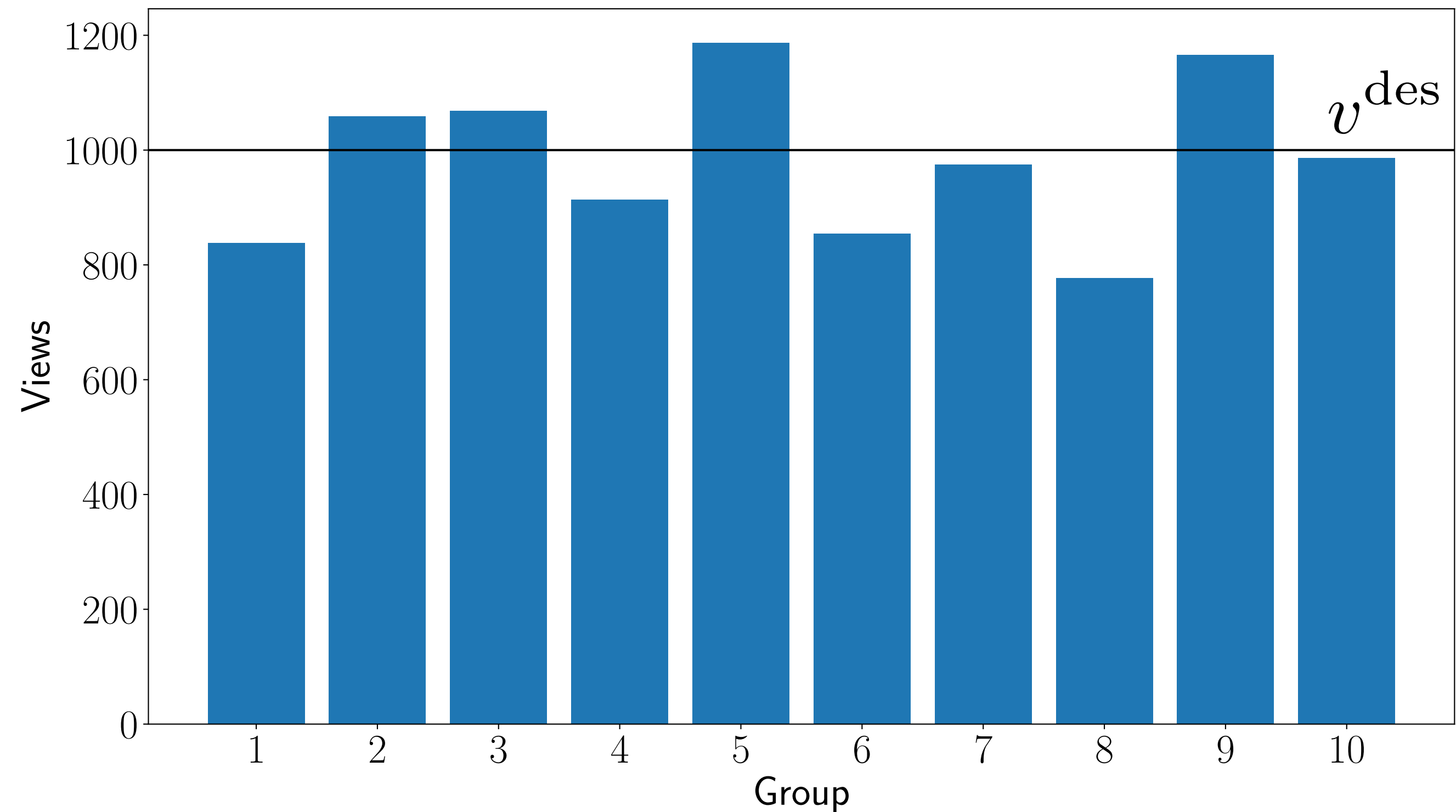
$m = 10$ groups, $n = 3$ channels

desired views vector $v^{\text{des}} = (10^3)\mathbf{1}$

minimize $\|As - v^{\text{des}}\|^2$



optimal spending $s^* = (62, 100, 1443)$



Optimal advertising

Reusing factorization on large example

$m = 100,000$ groups, $n = 5,000$ channels

$$\text{minimize } \|As - v^{\text{des}}\|^2$$

First solve

desired views $v^{\text{des},1} = (10^3)\mathbf{1}$

1. Form linear system $Mx = q$
where $M = A^T A$, $q = A^T b$
2. Factor $M = LL^T$
3. Solve $LL^T x = q$

Complexity

$$2mn^2$$

Time: 9 sec

Second solve

desired views $v^{\text{des},2} = 500\mathbf{1}$

1. Form $q = A^T b$
2. Solve $LL^T x = q$

Complexity

$$2mn$$

Time: 0.37 sec

Pseudoinverse

Time: 263 sec

Least squares

Today, we learned to:

- **Define and recognize** least squares problems
- **Solve** least squares problems using Cholesky factorization
- **Understand** the benefits of reusing factorizations

References

- S. Boyd, L. Vandenberghe: Introduction to Applied Linear Algebra – Vectors, Matrices, and Least Squares
 - Chapter 12: least squares

Next lecture

- Least squares and data fitting