QUADRATIC PROBLEMS AND NUMERICAL LINEAR ALGEBRA

Tom Goldstein

UNIVERSITY OF MARYLAND

HESSIAN

$$f(x) = f(x_1, x_2, x_3)$$

Ludwig Otto Hesse

Last lecture…

$$\nabla f(x) = \begin{pmatrix} \partial_1 f(x) \\ \partial_2 f(x) \\ \partial_3 f(x) \end{pmatrix}$$

Today….

$$\nabla^2 f(x) = \begin{pmatrix} \partial_1^2 f(x) & \partial_1 \partial_2 f(x) & \partial_1 \partial_3 f(x) \\ \partial_2 \partial_1 f(x) & \partial_2^2 f(x) & \partial_2 \partial_3 f(x) \\ \partial_3 \partial_1 f(x) & \partial_3 \partial_2 f(x) & \partial_3^2 f(x) \end{pmatrix}$$

Is it symmetric?

TAYLOR'S THEOREM

$$f(x) = f(0) + \underbrace{x f'(0) + \frac{1}{2} x^2 f''(0)}_{\text{What's this?}} + O(x^3)$$

In higher dimensions…

$$f(x) = f(0) + \underbrace{x^T \nabla f(0) + \frac{1}{2} x^T \nabla^2 f(0) x}_{\text{What's this?}} + O(\|x - x_0\|^3)$$

QUADRATIC FORM

$$f(z) = f(x, y) = 4x^2 + 2xy - 3y^2$$

$$f(z) = \frac{1}{2}(x\ \ y)\begin{pmatrix} 8 & 2 \\ 2 & -6 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{2}z^T\begin{pmatrix} 8 & 2 \\ 2 & -6 \end{pmatrix}z$$

In general

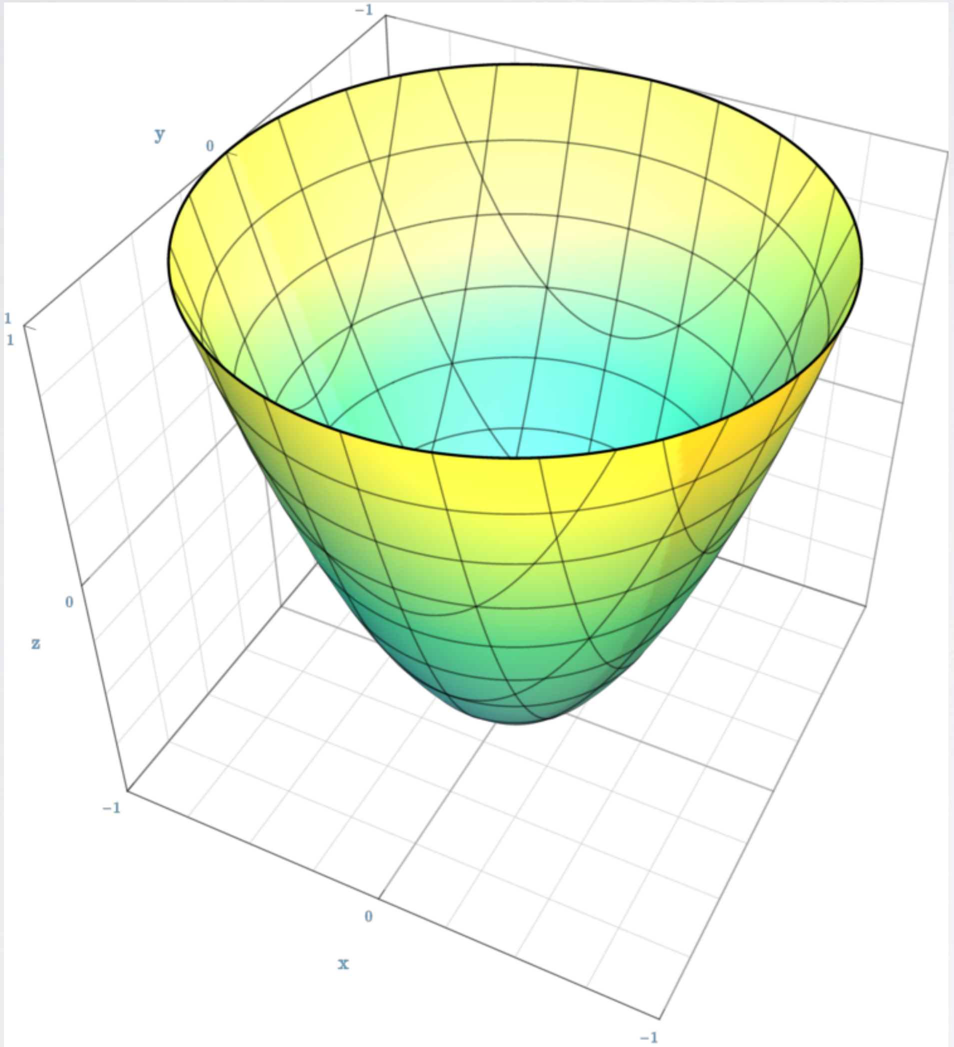
$$f(x) = c + g^T x + \frac{1}{2}x^T H x$$

gradient at 0

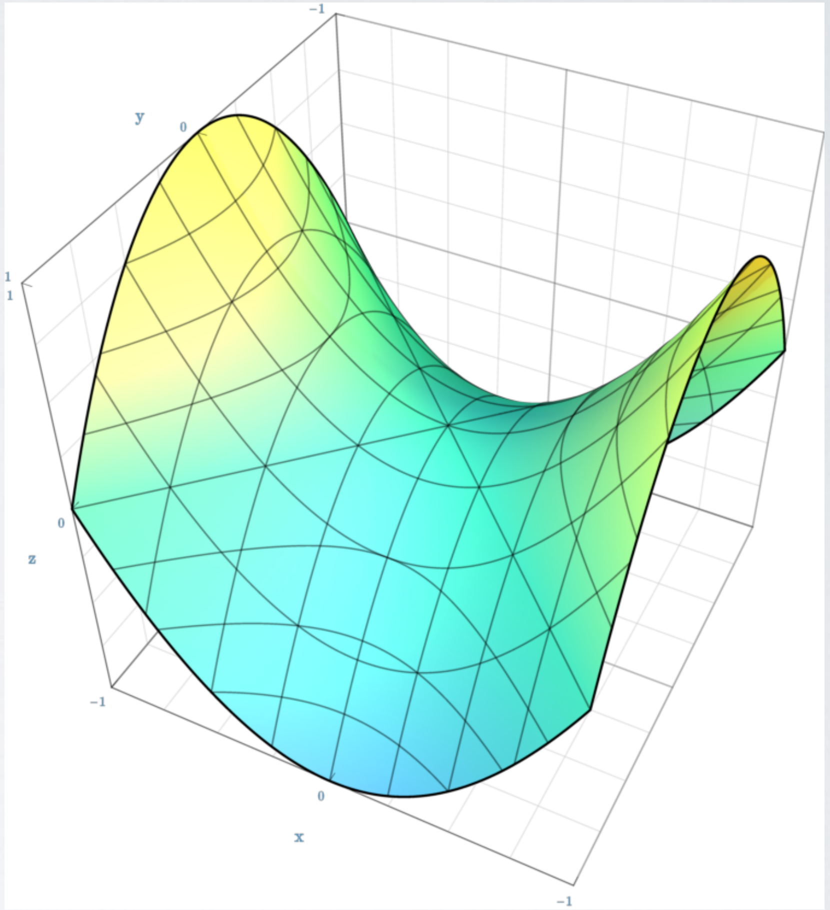
Hessian at 0

EXAMPLES

$$f(x) = c + g^T x + \frac{1}{2} x^T H x$$



Paraboloid



hyperboloid

What's the difference? **Eigenvalues** of the Hessian

FACTORIZATION OF HESSIAN

$$f(x) = c + g^T x + \frac{1}{2} x^T H x$$

Spectral Theorem: $$H = UDU^T$$

diagonal **orthogonal**

$$f(x) = c + g^T x + \frac{1}{2} x^T U D \underline{U^T x}$$

$$y = U^T x$$

$$f(y) = c + (U^T g)^T y + \frac{1}{2} y^T D y$$

$$f(y) = c + \sum_i (U^T g)_i y_i + \frac{1}{2} D_{ii} y_i^2$$

FACTORIZATION OF HESSIAN

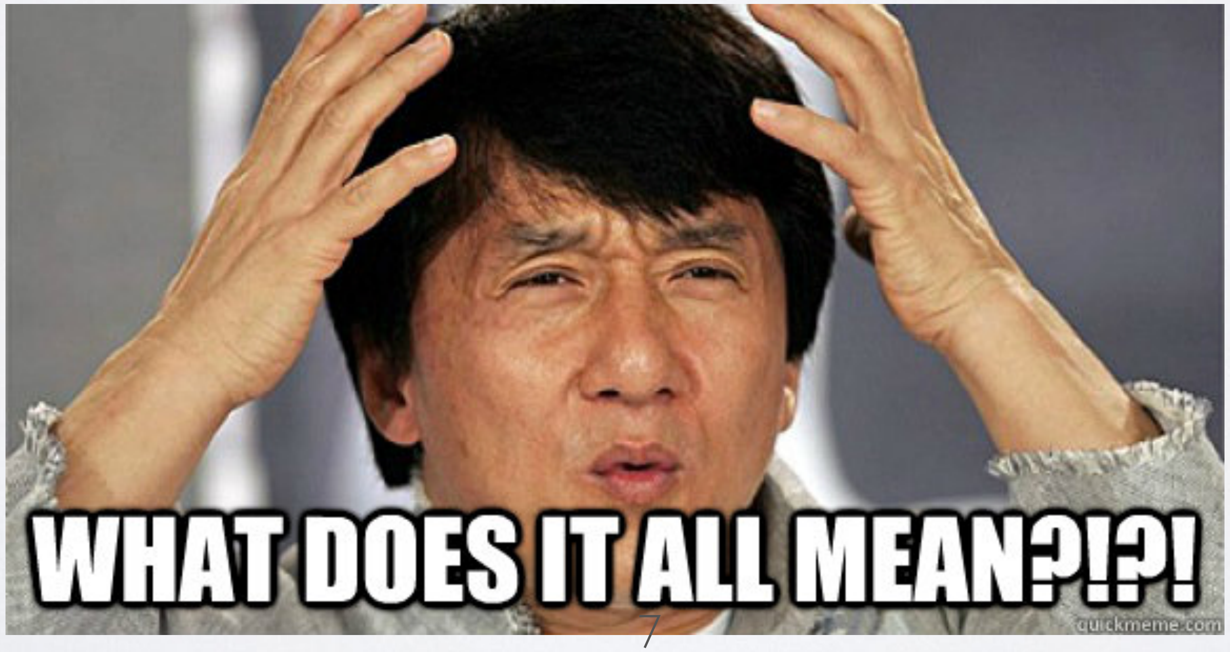
$$f(x) = c + g^T x + \frac{1}{2} x^T H x$$

$$y = U^T x$$

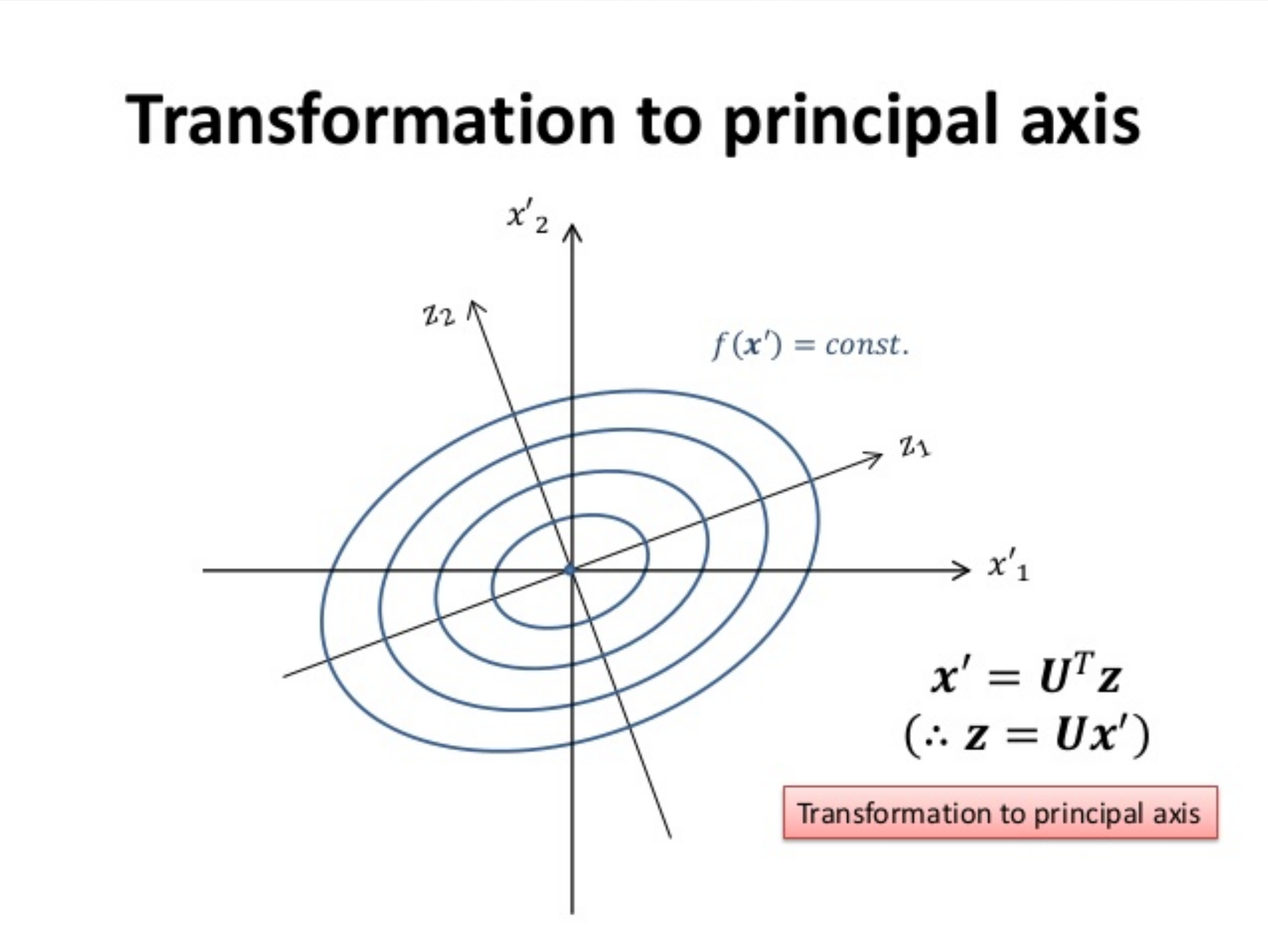
Change of variables

$$f(y) = c + \sum_i (U^T g)_i y_i + \frac{1}{2} D_{ii} y_i^2$$

Curvature along each coordinate



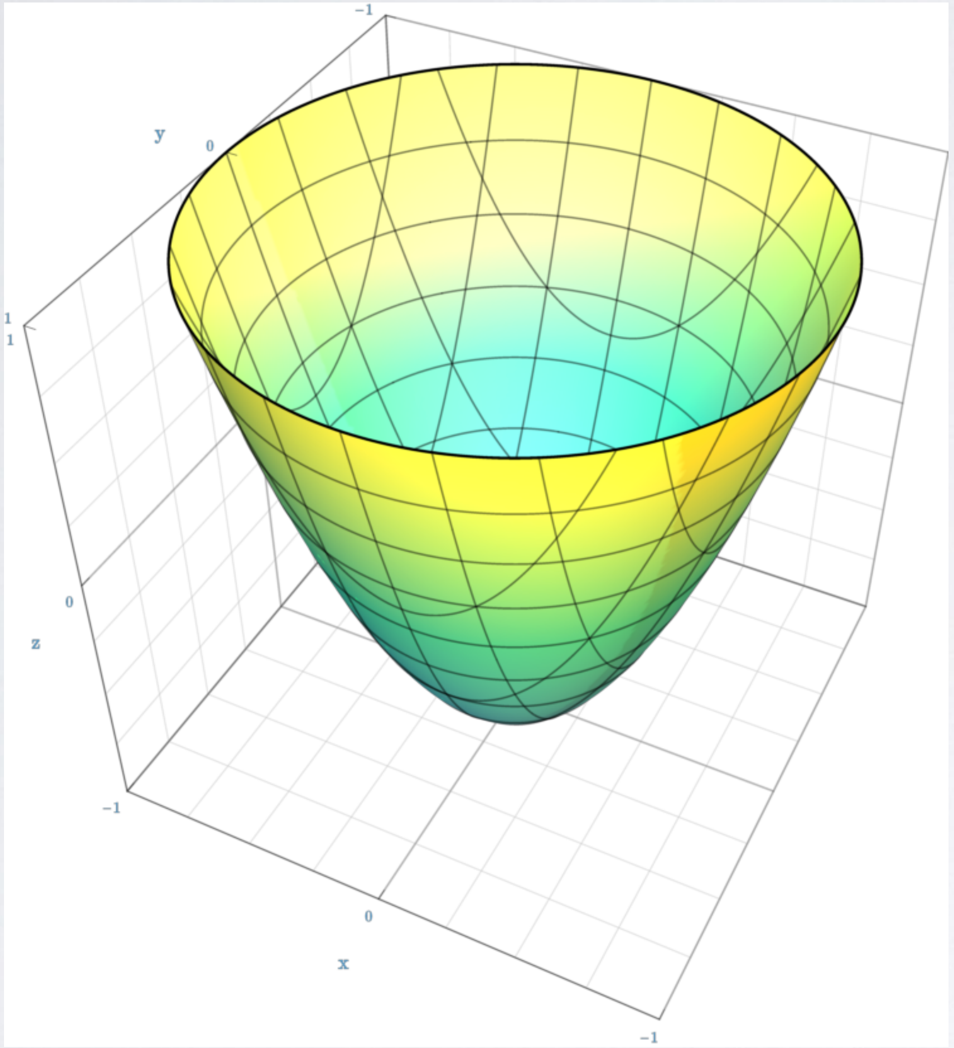
TRANFORMATION



Behavior of a quadratic form determined only by eigenvalues!

EXAMPLES

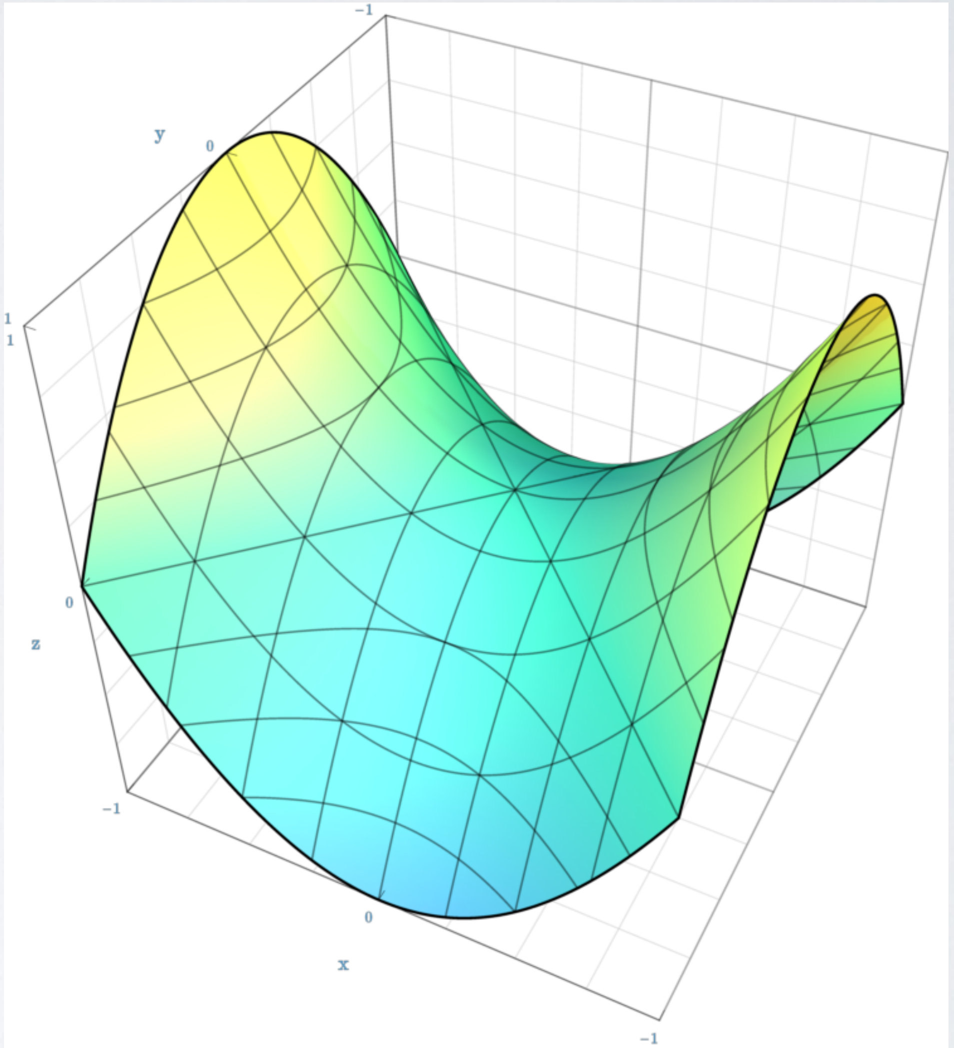
$$f(x) = c + g^T x + \frac{1}{2} x^T H x$$



Positive definite Hessian

Convex

Unique minimizers



Indefinite Hessian

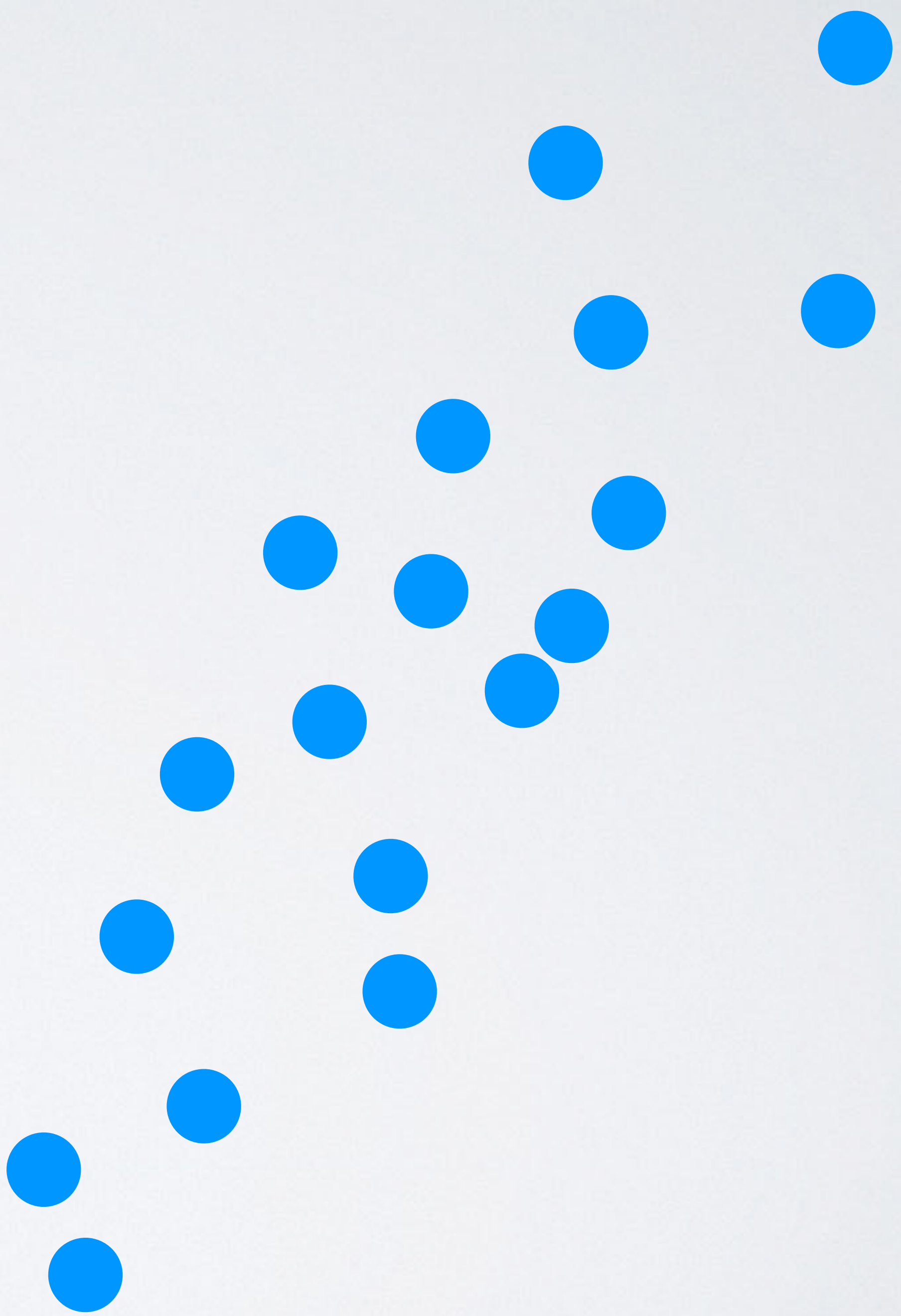
non-convex

No minimizers at all!

Application

PRINCIPLE COMPONENT ANALYSIS

GAUSSIAN DATA



GAUSSIAN DATA

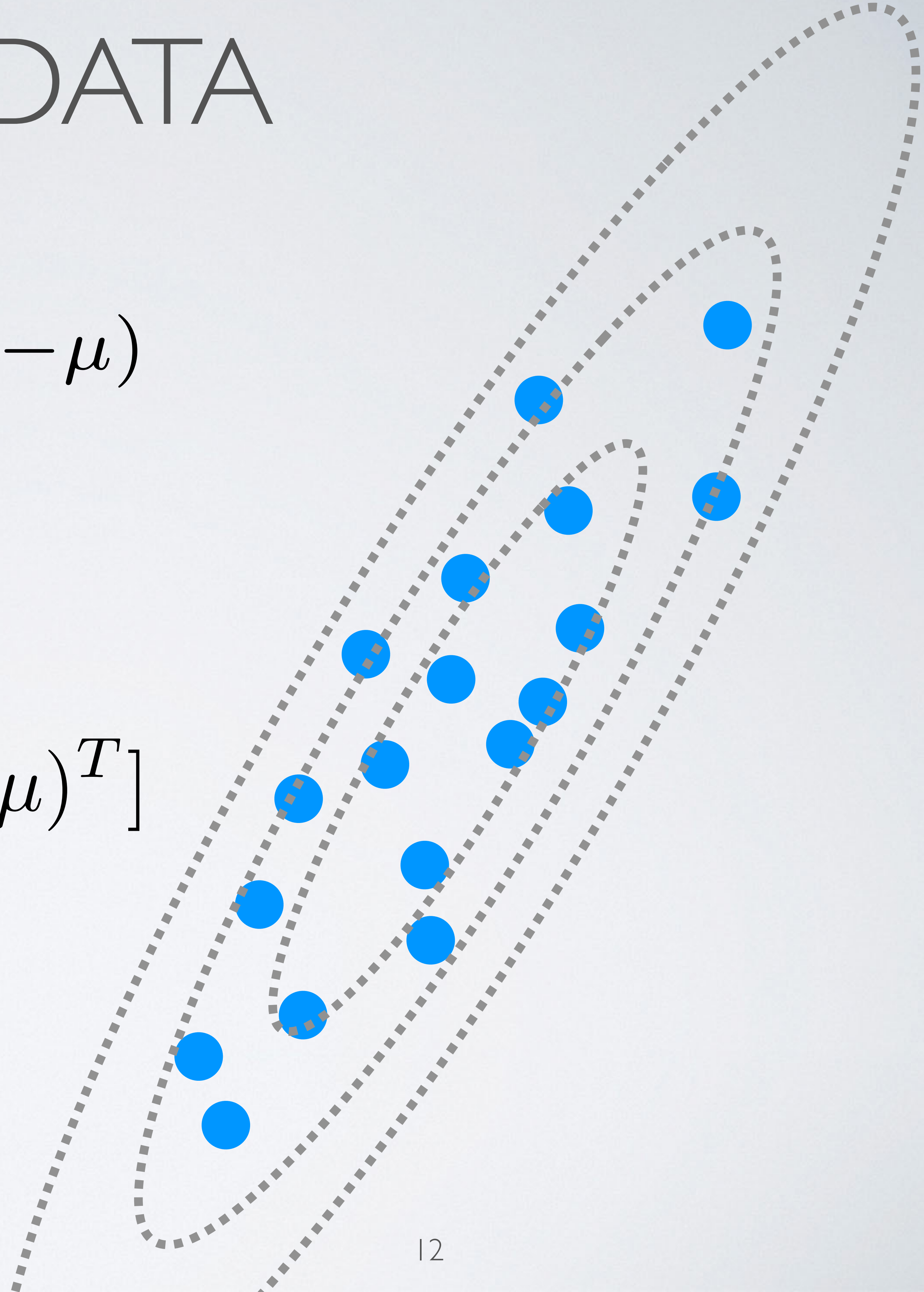
model

$$e^{-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)}$$

calculate

$$\mu = \mathbb{E}_i[x_i]$$

$$\Sigma = \mathbb{E}_i[(x_i - \mu)(x_i - \mu)^T]$$



GAUSSIAN DATA

model

$$e^{-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)}$$

log-likelihood is **quadratic**

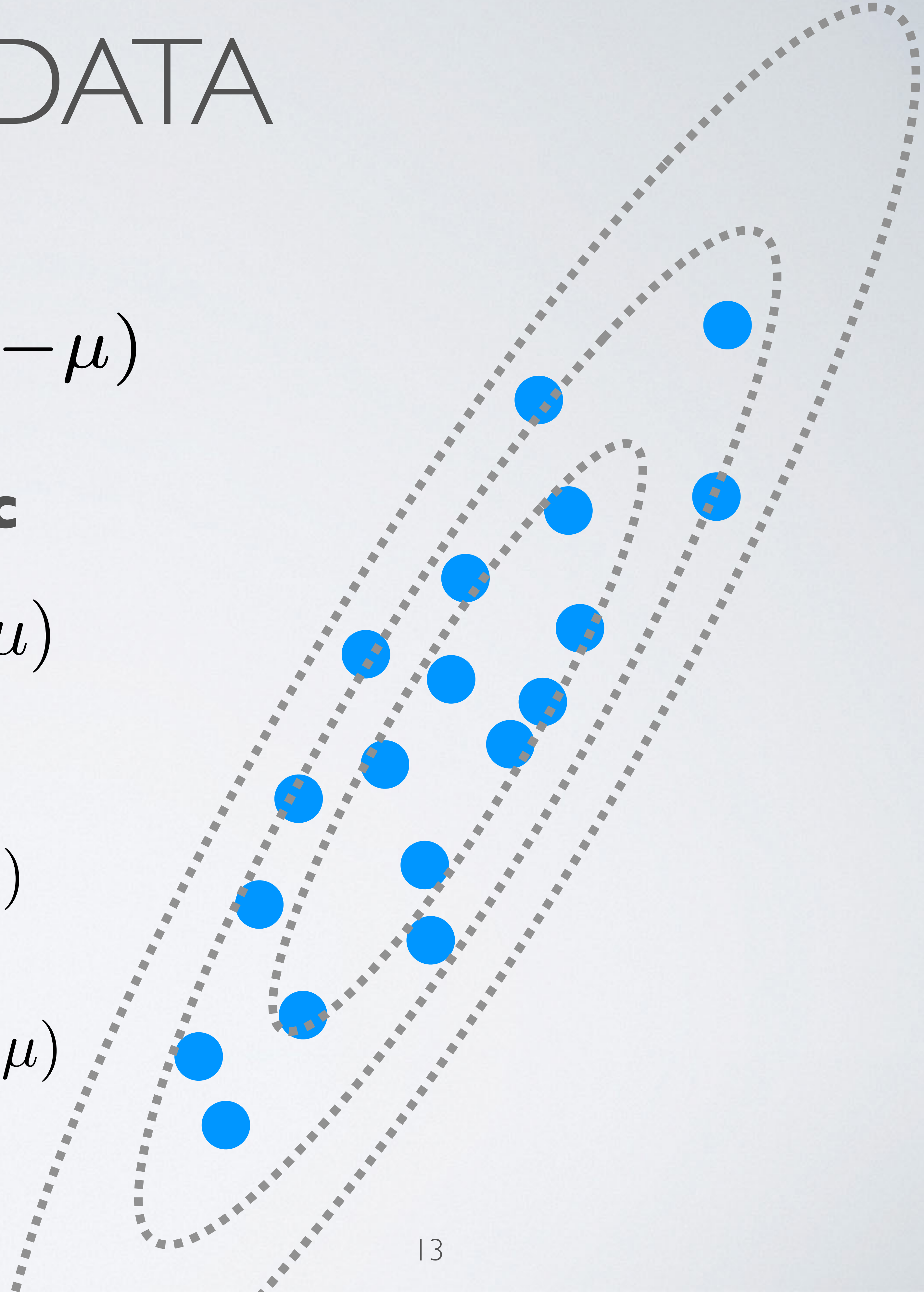
$$-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)$$

factor the covariance

$$-\frac{1}{2}(x-\mu)^t U D^{-1} U^t (x-\mu)$$

change variables: $z \leftarrow U^t(x-\mu)$

$$-\frac{1}{2}z^t D^{-1} z = \sum_j -\frac{1}{2\sigma_j^2} z_j^2$$



GAUSSIAN DATA

model

$$e^{-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)}$$

log-likelihood is **quadratic**

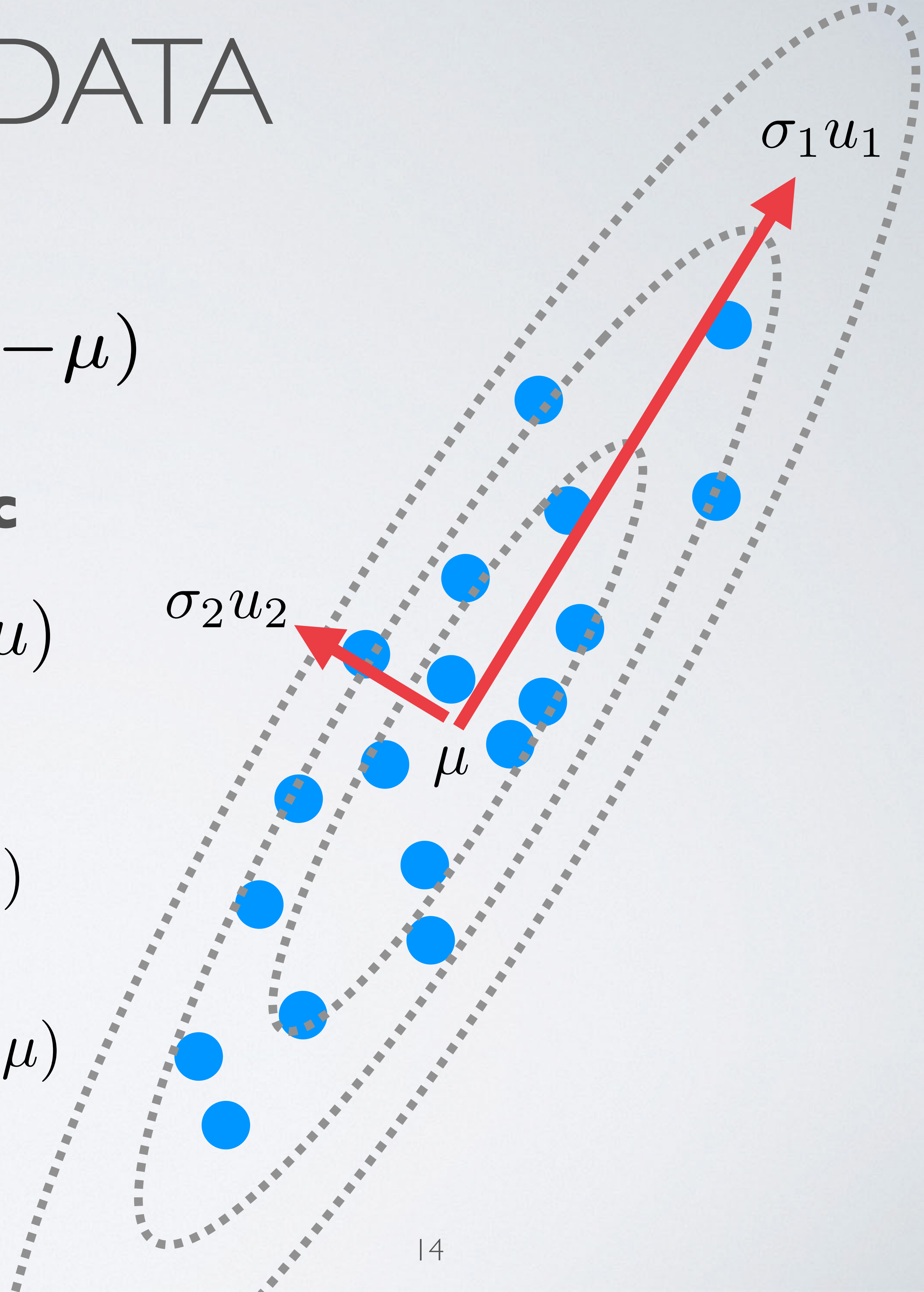
$$-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)$$

factor the covariance

$$-\frac{1}{2}(x-\mu)^t U D^{-1} U^t (x-\mu)$$

change variables: $z \leftarrow U^t(x-\mu)$

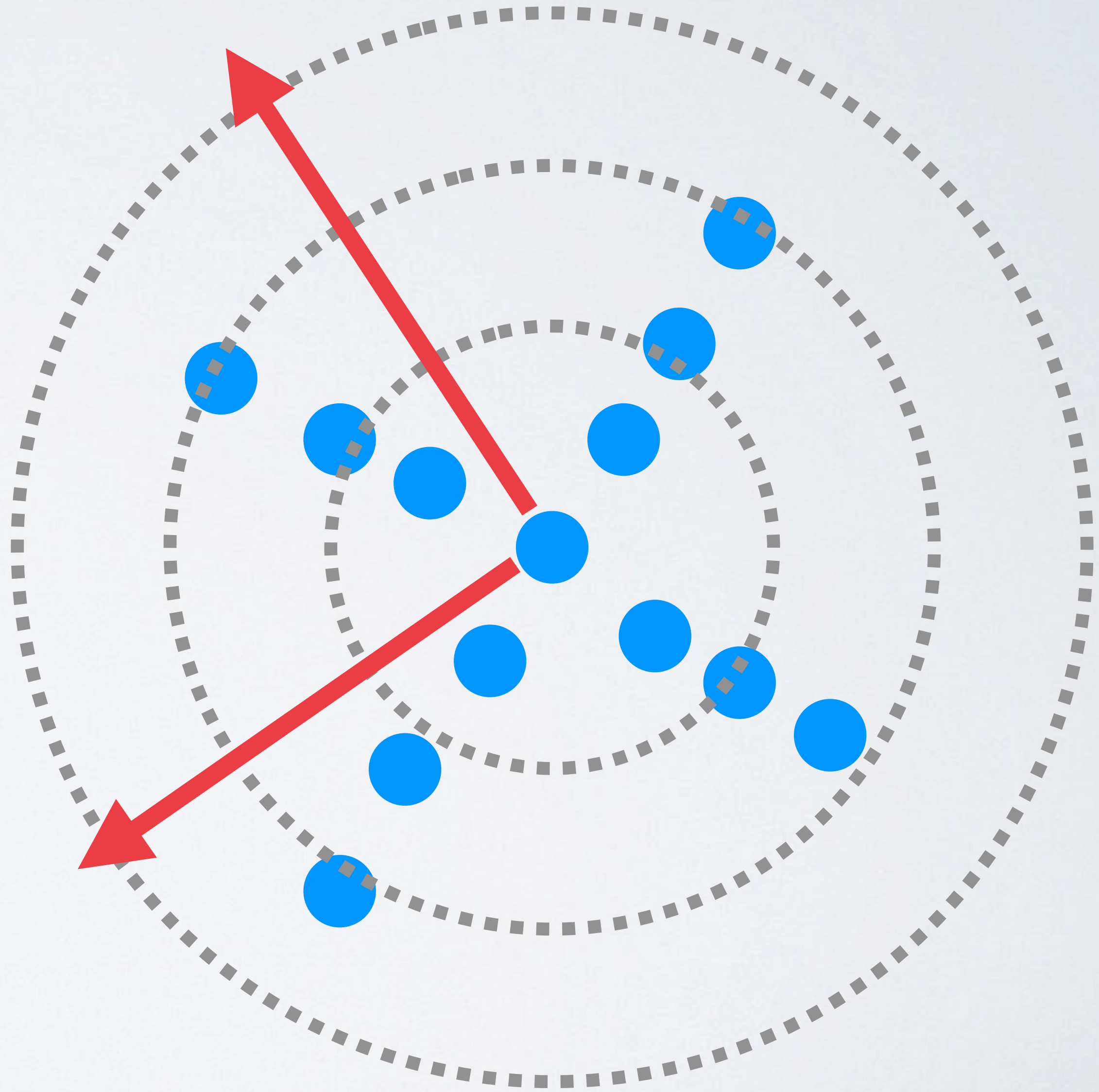
$$-\frac{1}{2}z^t D^{-1} z = \sum_j -\frac{1}{2\sigma_j^2} z_j^2$$



WHAT ABOUT NON-GAUSSIAN DATA?

model

$$e^{-\frac{1}{2}(x-\mu)^t\Sigma^{-1}(x-\mu)}$$



NUMERICAL LINEAR ALGEBRA

MINIMIZING QUADRATIC

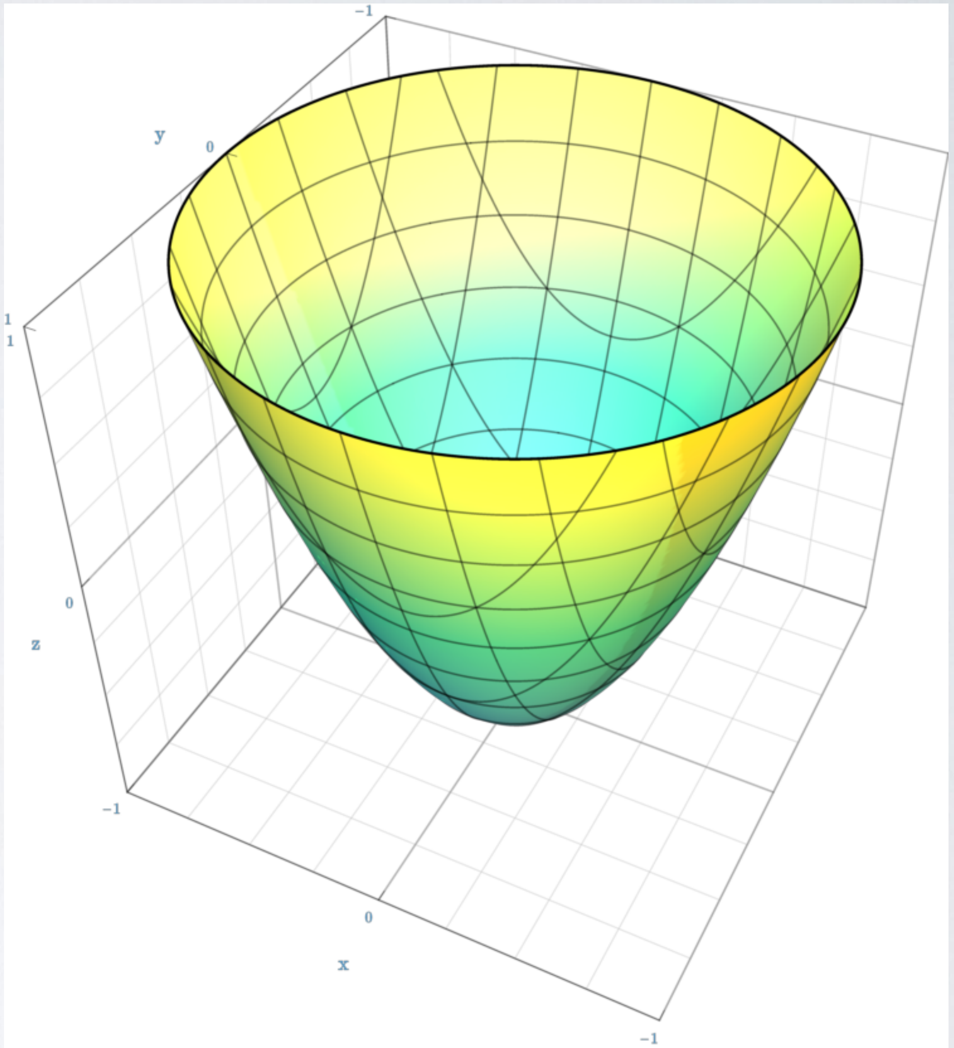
$$f(x) = \frac{1}{2}x^T H x + g^T x + c$$

$$\nabla f(x) = Hx + g = 0$$

$$Hx = -g$$

$$x = -H^{-1} g$$

How do we compute this??



GAUSSIAN ELIMINATION

$$\left[\begin{array}{cccccc|c} a_{11} & a_{12} & a_{13} & a_{14} & \dots & a_{1n} & y_1 \\ 0 & a_{22}' & a_{23}' & a_{24}' & \dots & a_{2n}' & y_2' \\ 0 & a_{32}' & a_{33}' & a_{34}' & \dots & a_{3n}' & y_3' \\ 0 & a_{42}' & a_{43}' & a_{44}' & \dots & a_{4n}' & y_4' \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & a_{m2}' & a_{m3}' & a_{m4}' & \dots & a_{mn}' & y_m' \end{array}\right]$$

$$\downarrow$$

$$\left[\begin{array}{cccccc|c} a_{11} & a_{12} & a_{13} & a_{14} & \dots & a_{1n} & y_1 \\ 0 & a_{22}' & a_{23}' & a_{24}' & \dots & a_{2n}' & y_2' \\ 0 & 0 & a_{33}'' & a_{34}'' & \dots & a_{3n}'' & y_3'' \\ 0 & 0 & a_{43}'' & a_{44}'' & \dots & a_{4n}'' & y_4'' \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & a_{m3}'' & a_{m4}'' & \dots & a_{mn}'' & y_m'' \end{array}\right]$$

WHAT'S WRONG WITH THIS?

Use row 2 to eliminate row 3:

$$A_{k+1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & a_{32}/a_{22} & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}^{-1} A_k$$

$$\kappa = O(a_{32}^2/a_{22}^2)$$

Is this bad? How bad?

WHAT'S WRONG WITH THIS?

Use row 2 to eliminate row 3:

$$A_{k+1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & a_{32}/a_{22} & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}^{-1} A_k$$

Poor conditioning = EXTREME BADNESS

Error is O(κ)

Example: hilb(9) operator in Matlab

BETTER SOLUTION: LU / CHOLESKY

$$Ax = b$$

Factorize! $A = LU$

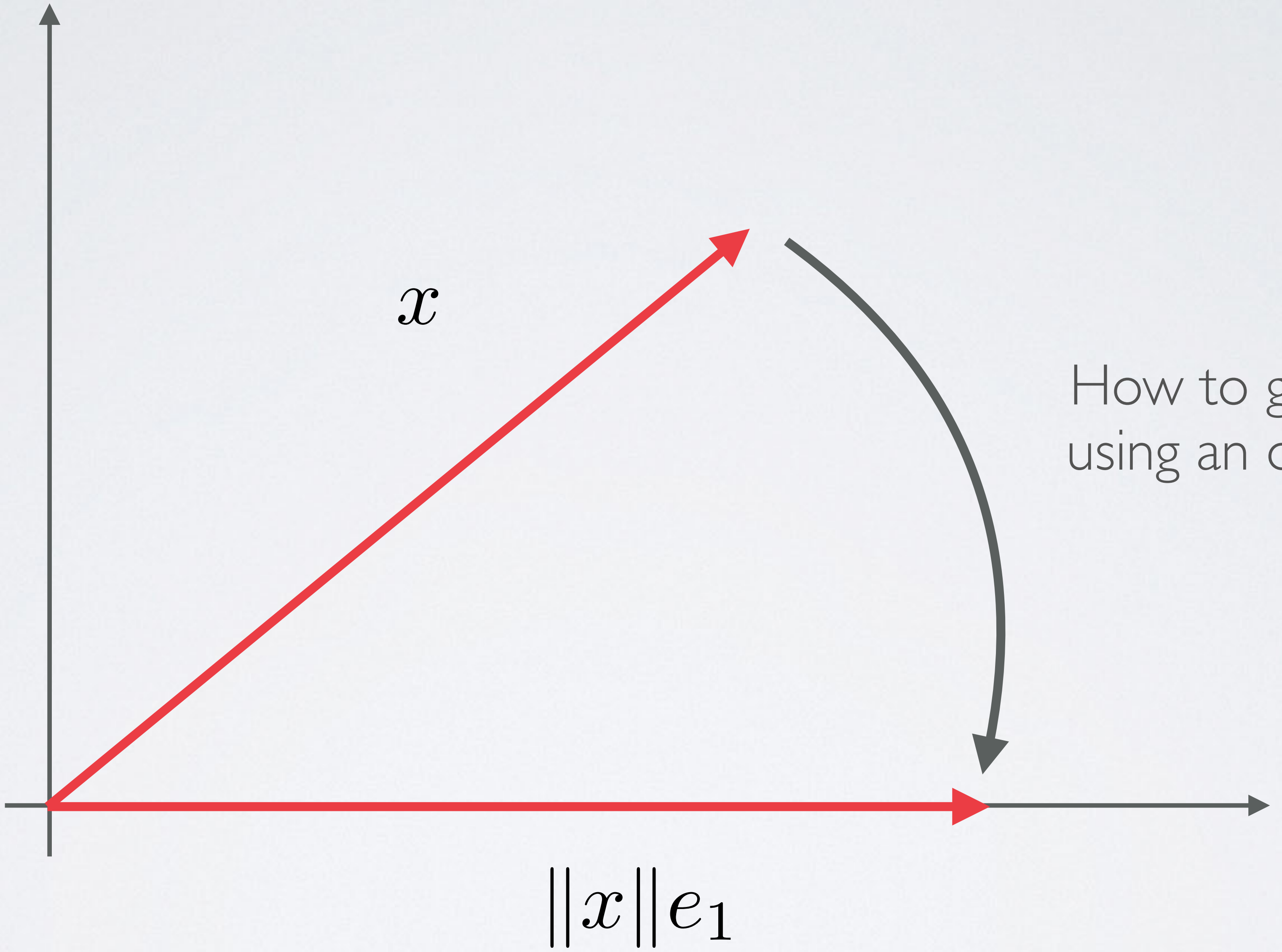
$$\mathbf{L} = \begin{bmatrix} l_{11} & 0 & 0 & \cdots & 0 \\ l_{21} & l_{22} & 0 & & 0 \\ \vdots & & & \ddots & \vdots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ 0 & u_{22} & u_{23} & & u_{2n} \\ 0 & 0 & u_{33} & & u_{3n} \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & u_{nn} \end{bmatrix}$$

Error is O($\sqrt{\kappa}\epsilon$)

Complexity?

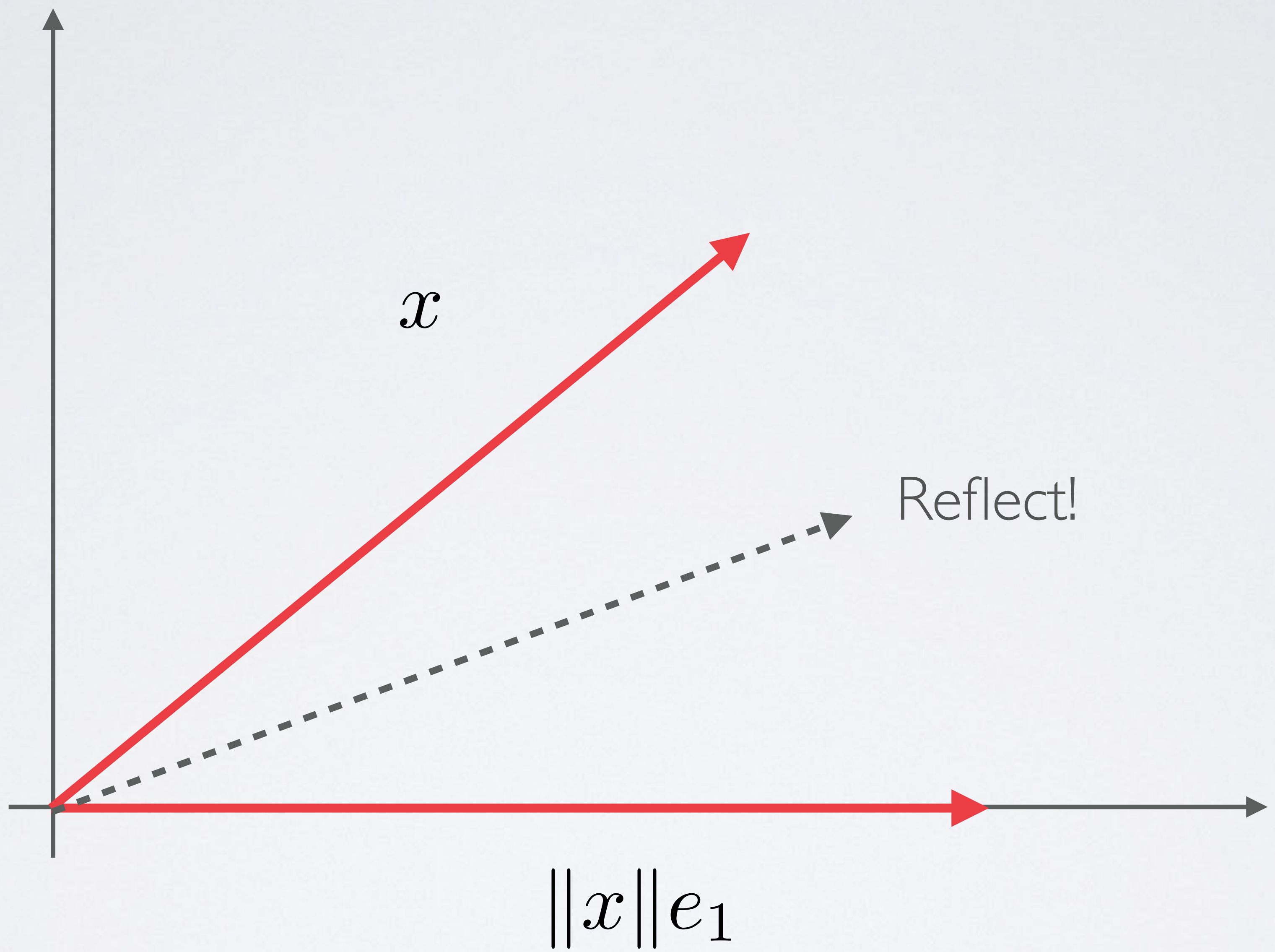
How long does it take to find $A^{-1}b$?

HOUSEHOLDER REFLECTIONS

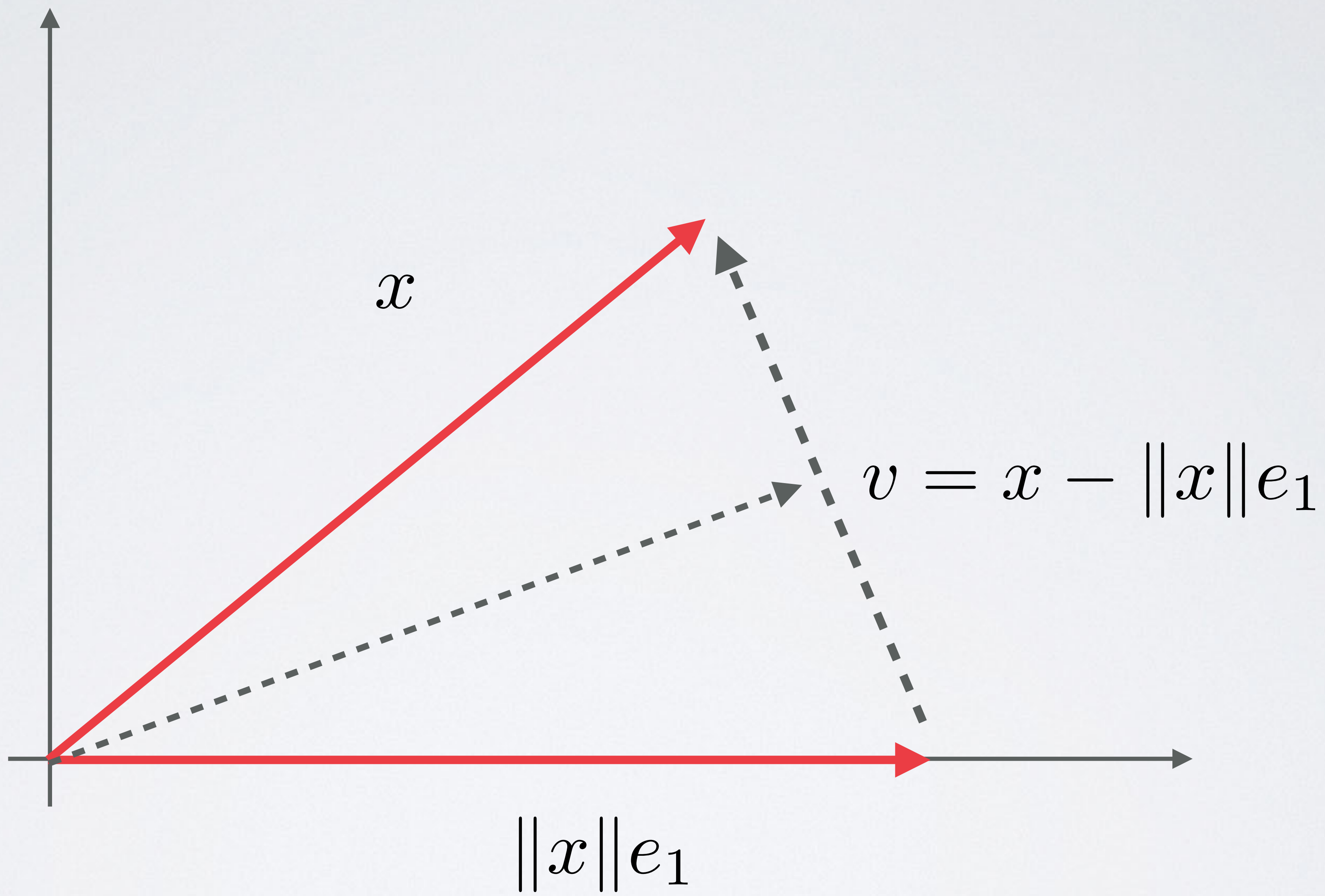


How to get x onto the axis using an orthogonal matrix?

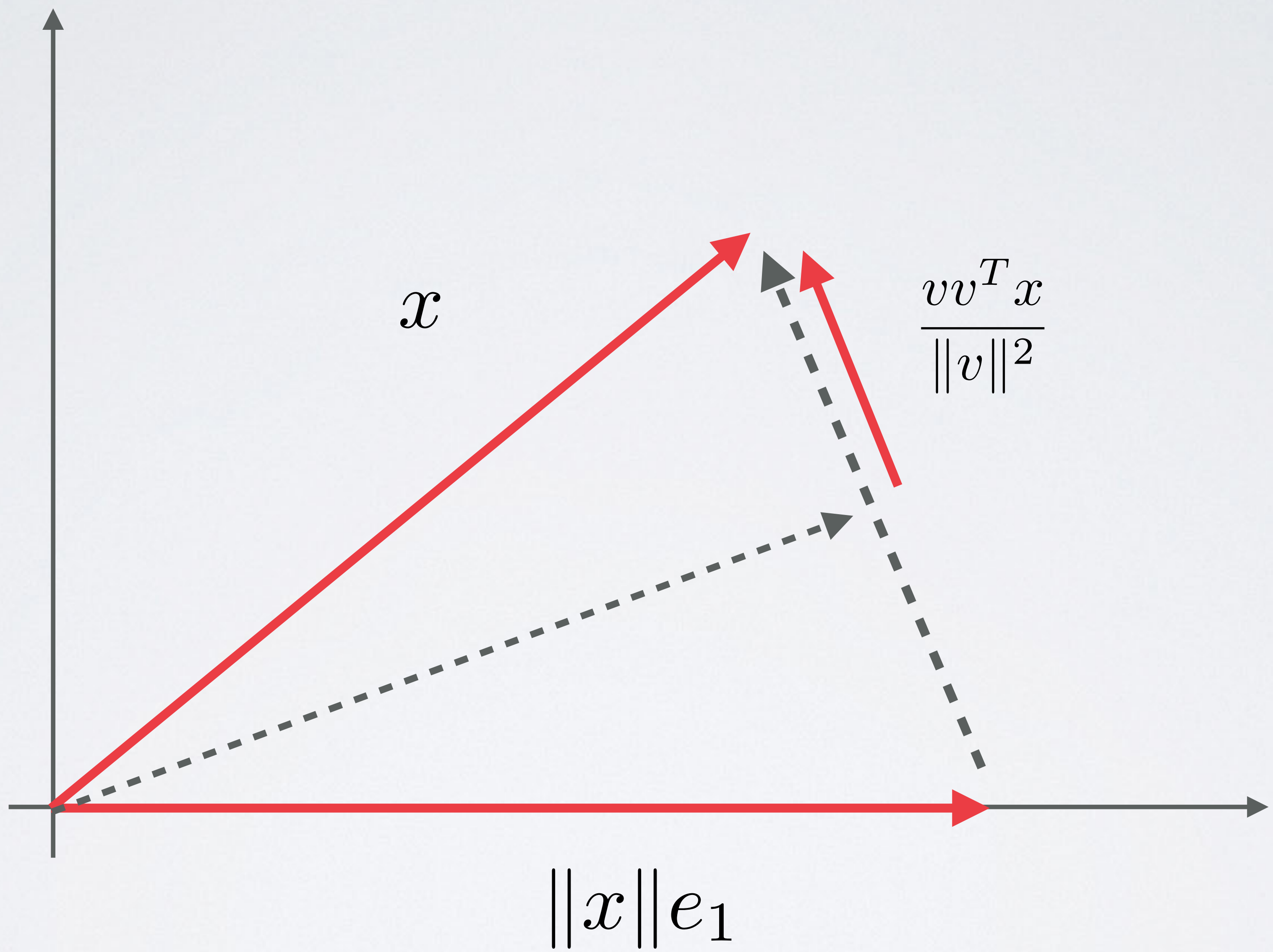
HOUSEHOLDER REFLECTIONS



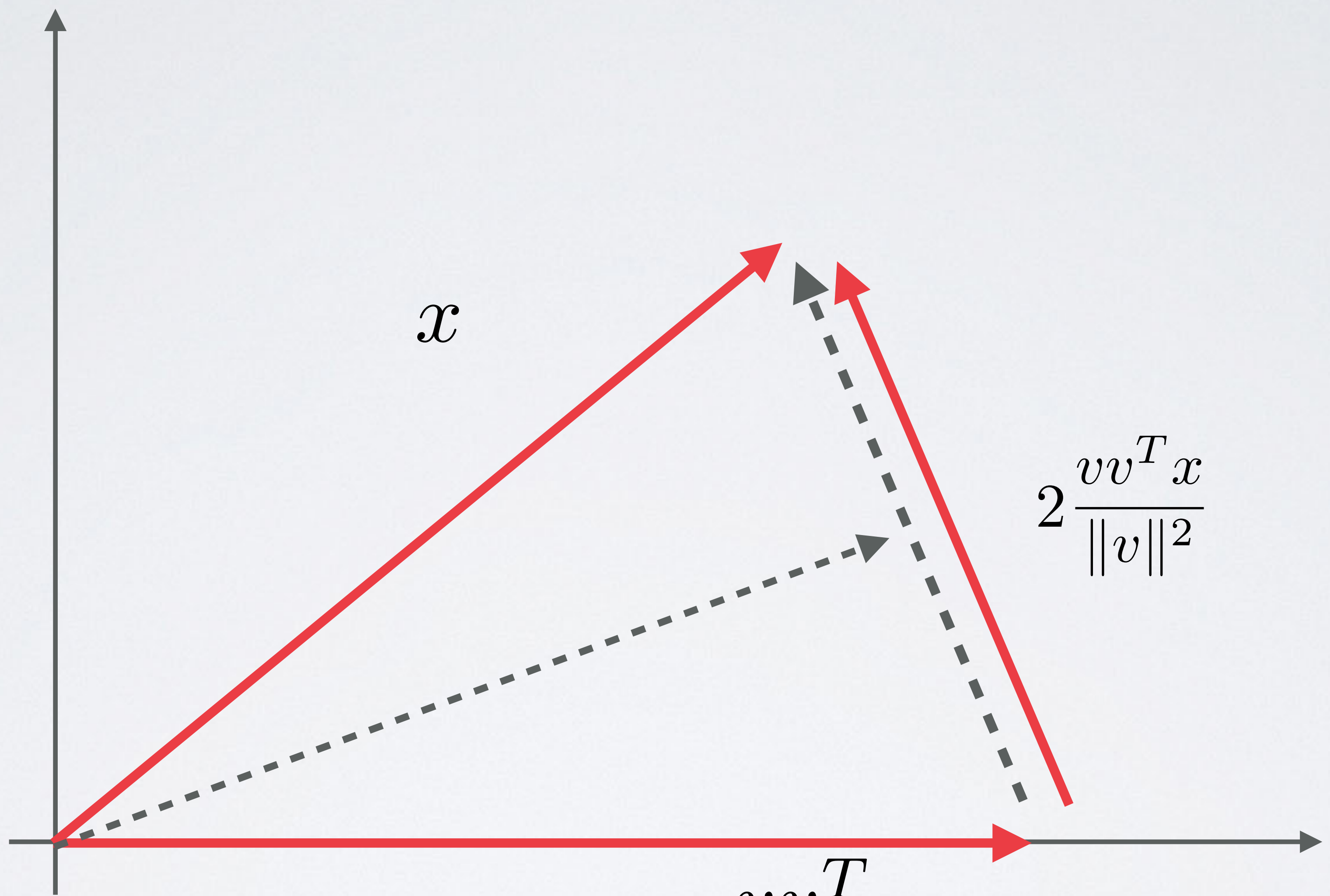
HOUSEHOLDER REFLECTIONS



HOUSEHOLDER REFLECTIONS



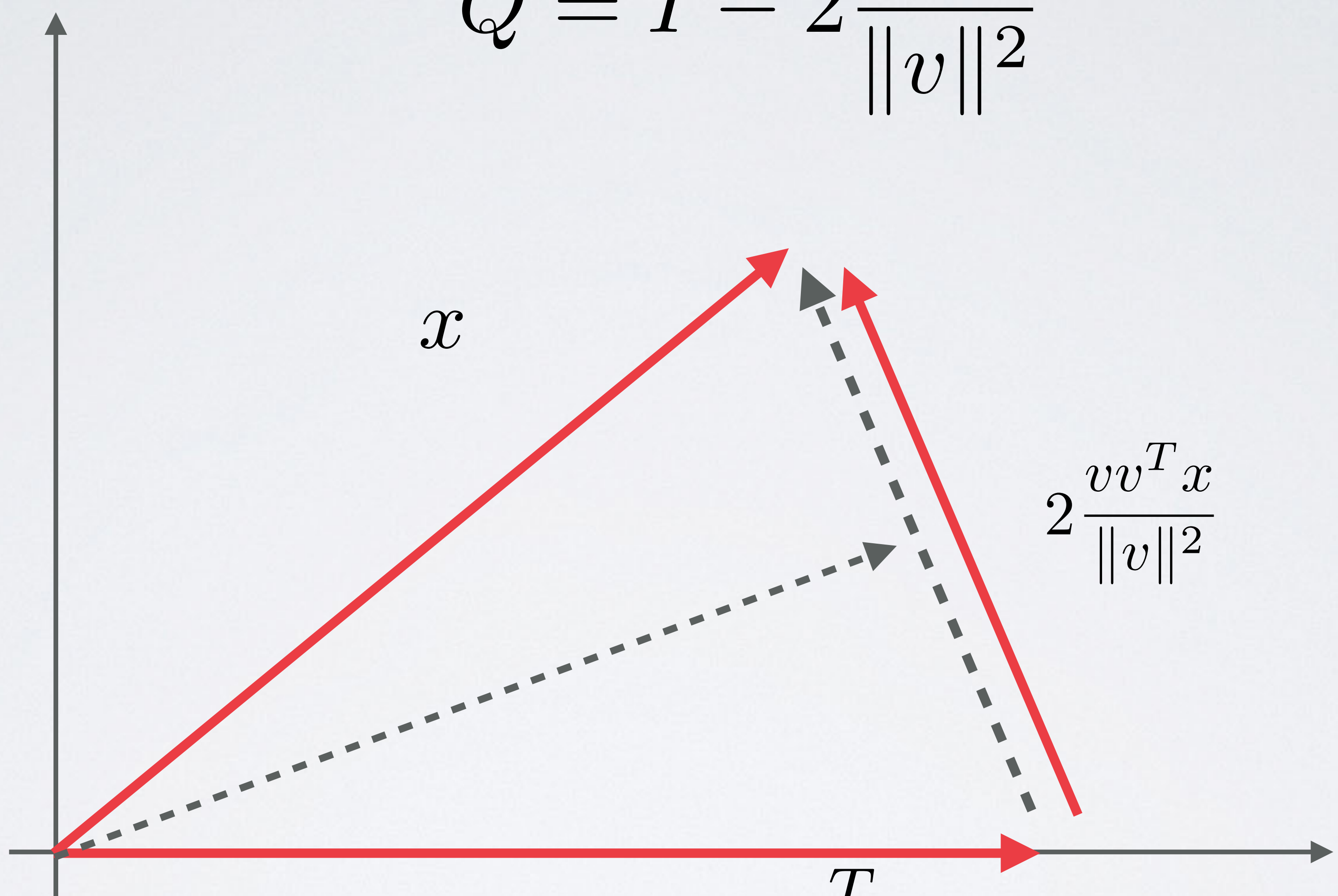
HOUSEHOLDER REFLECTIONS



$$\|x\|e_1 = x - 2\frac{vv^T}{\|v\|^2}x$$

HOUSEHOLDER REFLECTIONS

$$Q = I - 2\frac{vv^T}{\|v\|^2}$$

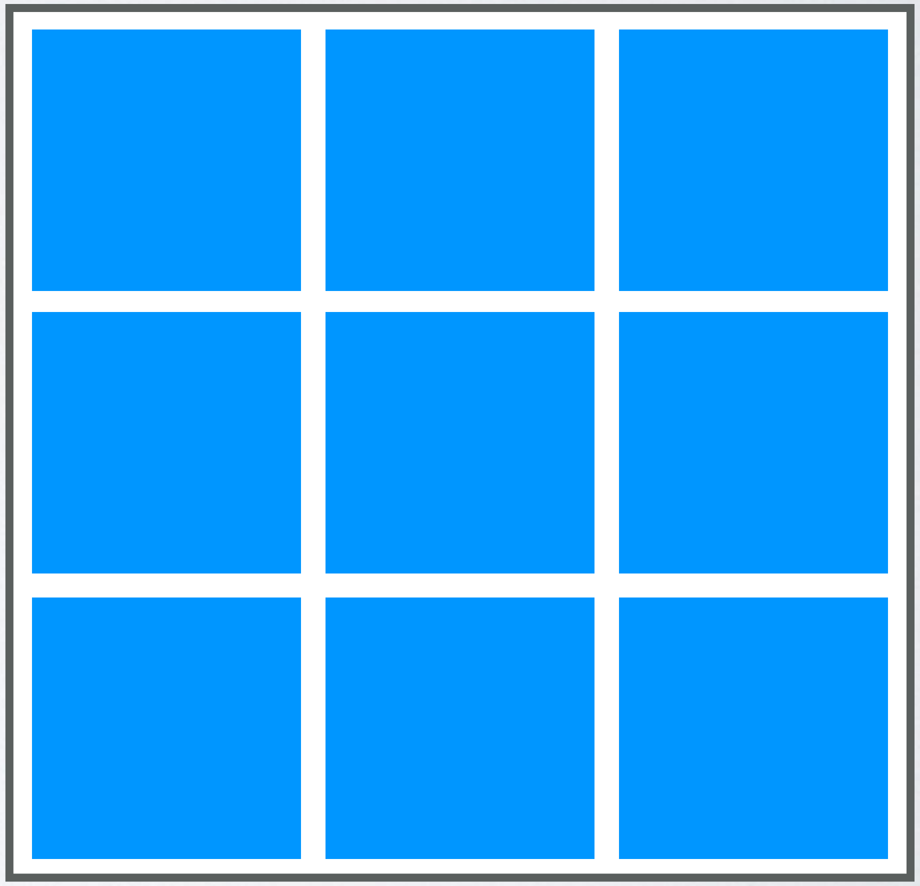


$$\|x\|e_1 = x - 2\frac{vv^T}{\|v\|^2}x$$

QR USING HOUSEHOLDER

Map to e_1

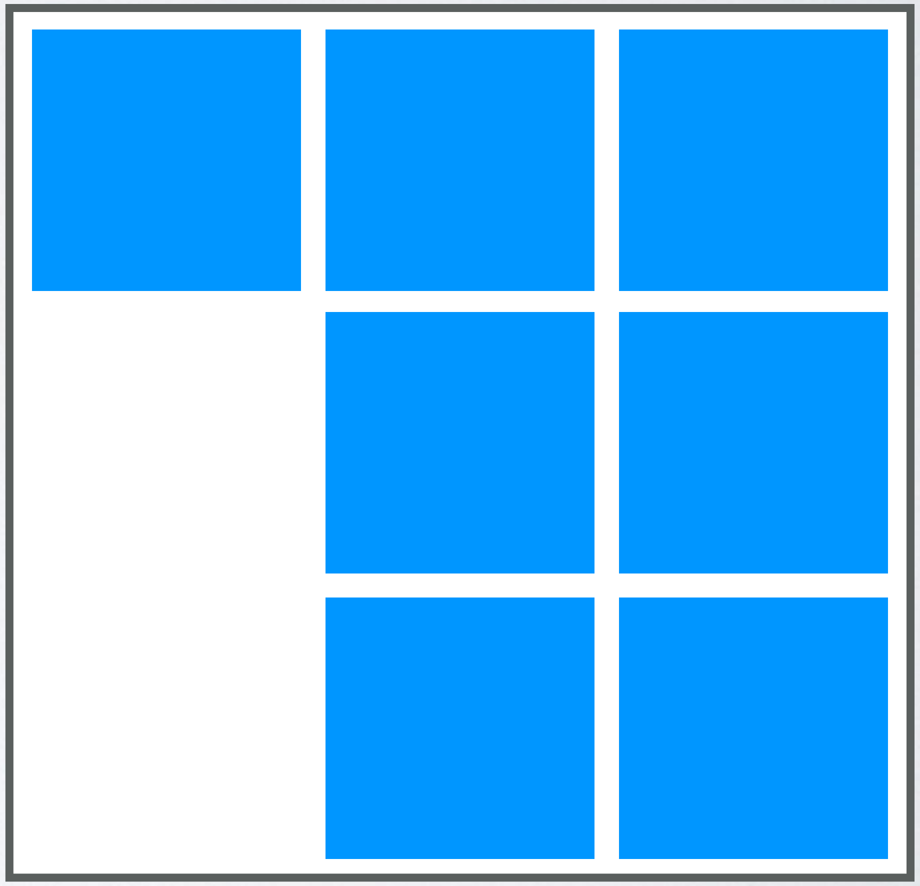
$$Q_1^T Q_1$$



QR USING HOUSEHOLDER

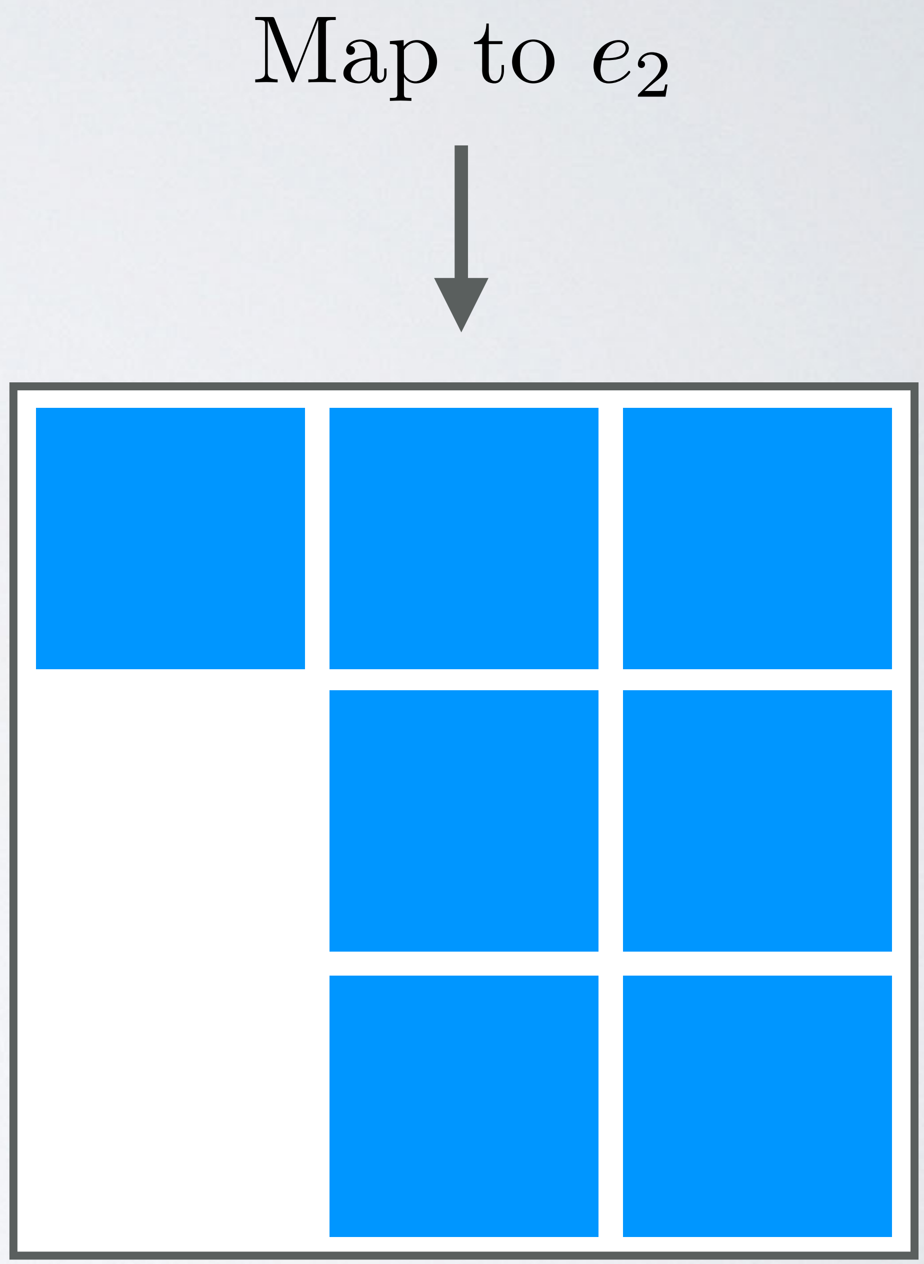
$$Q_1^T$$

Map to e_2



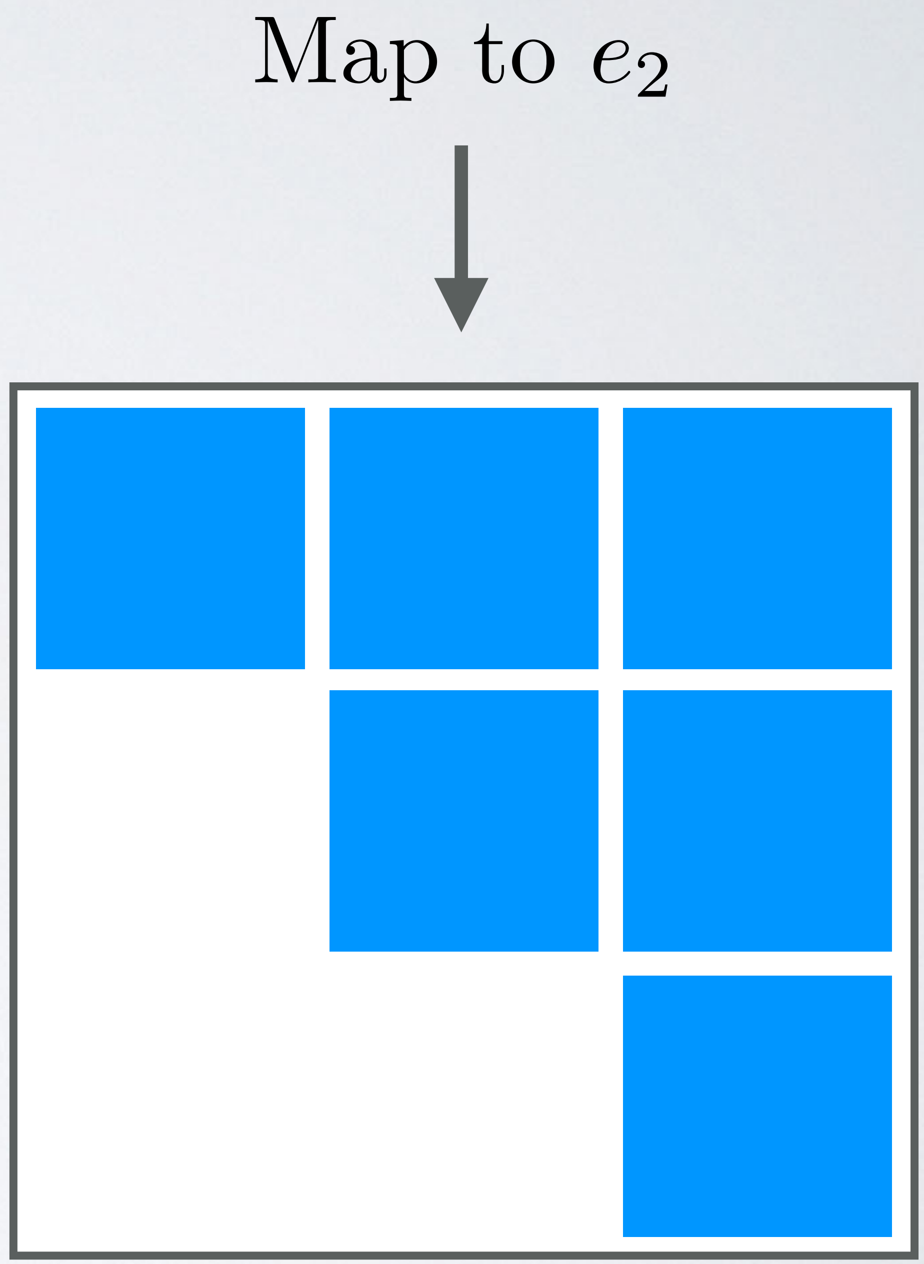
QR USING HOUSEHOLDER

$$Q_1^T Q_2^T Q_2$$



QR USING HOUSEHOLDER

$$Q_1^T Q_2^T$$



BETTER STILL: QR

$$Ax = b$$

$$QRx = b$$

$$\mathbf{Q} = \begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix} \qquad R = \begin{pmatrix} * & * & * \\ 0 & * & * \\ 0 & 0 & * \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Error is O($\sqrt{N}\epsilon$)

$$A^{-1}b = R^{-1}Q^T b$$

Complexity?

THE BEST: SVD

$$Ax = b$$

$$A = USV^T$$

$$A^{-1}b = VS^{-1}U^Tb$$

Advantages:

- SUPER numerically stable
- Can use pseudoinverse
- Non-square problems
- Sometimes the SVD is **FREE**

SLOW: 2-20X worse than QR. Complexity?

CHEATING FATE

Can we get better than cubic complexity?

CHEATING FATE: WOODBURY IDENTITY

$$Ax = b \xrightarrow{O(n^3)} A^{-1}$$

$$(A+\delta)x = b \xrightarrow{O(n^2)} (A+\delta)^{-1}$$

$$(A + \underbrace{U}_{\text{tall}}\underbrace{V}_{\text{fat}})^{-1} = A^{-1} - A^{-1}U\underbrace{(I + VA^{-1}U)}_{\textbf{Small!}}{}^{-1}VA^{-1}$$

EXAMPLE: CHANGE BOUNDARY

$$A = \begin{pmatrix} a_1 & b_1 & 0 & 0 & 0 \\ b_1 & a_2 & b_2 & 0 & 0 \\ 0 & b_2 & a_3 & b_3 & 0 \\ 0 & 0 & b_3 & a_4 & b_4 \\ 0 & 0 & 0 & b_4 & a_5 \end{pmatrix} \qquad B = \begin{pmatrix} a_1 & b_1 & 0 & 0 & c_1 \\ b_1 & a_2 & b_2 & 0 & 0 \\ 0 & b_2 & a_3 & b_3 & 0 \\ 0 & 0 & b_3 & a_4 & b_4 \\ c_1 & 0 & 0 & b_4 & a_5 \end{pmatrix}$$

$$B = A + \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & c_1 \\ c_1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$(A + UV)^{-1} = A^{-1} - A^{-1}U\underline{(I + VA^{-1}U)}^{-1}VA^{-1}$$

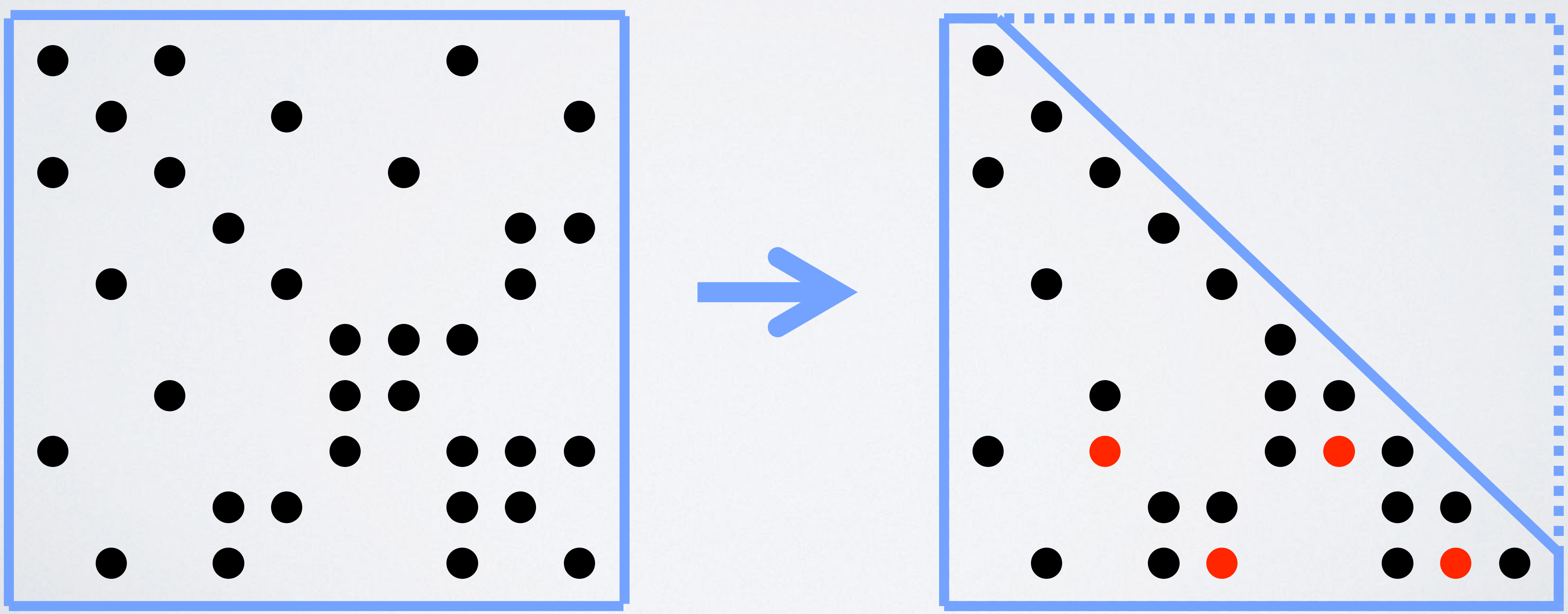
2X2 = CHEAP!

SPARSE SYSTEMS

Sparse $A \longrightarrow A^{-1}$ Dense

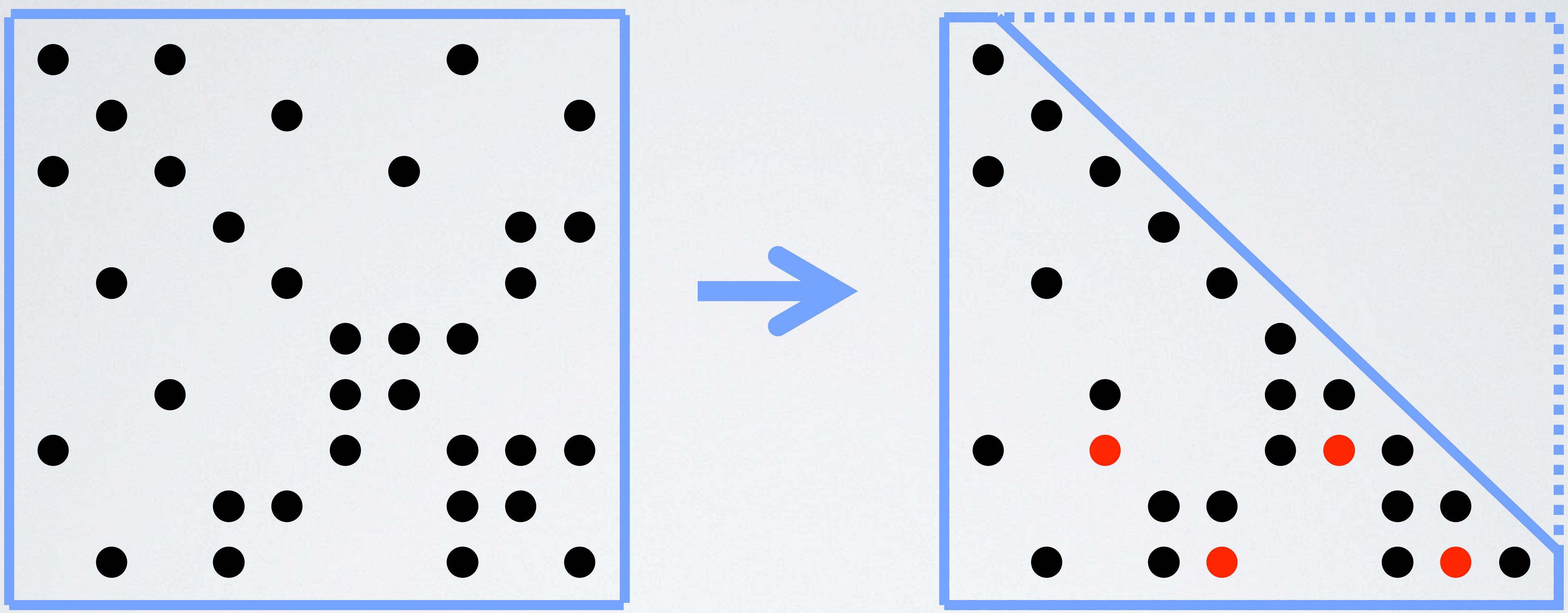
Sparse Matrix

Sparse Cholesky



SPARSE SYSTEMS

Sparse Matrix — Sparse Cholesky



Advantages over inverse:

- Beats complexity bound!
- Lower memory requirements

Disadvantage: Some matrices have bad "**fill-in**"

ITERATIVE METHODS

$$Ax = b$$

Only observe ACTION of A on vectors

Simplest: Richardson iteration

$$x_0 = 0$$
$$x_{k+1} = x_k + \tau(b - Ax_k)$$

Error analysis:
$$\begin{aligned} r_{k+1} &= b - Ax_{k+1} \\ &= b - Ax_k - \tau A(b - Ax_k)) \\ &= (I - \tau A)(b - Ax_k) \end{aligned}$$
$$r_{k+1} = (I - \tau A)r_k$$

CONVERGENCE: RICHARDSON

$$Ax = b$$

$$r_{k+1} = (I - \tau A)r_k$$

CONVERGENCE: RICHARDSON

$$Ax = b$$

$$r_{k+1} = \underline{(I - \tau A)} r_k$$

Only works for PD matrix!

$$1 - \tau \lambda_{max} > -1$$

$$\tau < \frac{2}{\lambda_{max}}$$

$$\tau^\star = \frac{2}{\lambda_{min} + \lambda_{max}} = \arg\min |1 - \tau \lambda_i|$$

$$\|1 - \tau^\star A\|_2 = \frac{\kappa}{\kappa + 1}$$

Condition number

KRYLOV METHODS

$$Ax = b$$

Choose **A-conjugate** basis: $\{p_k\}$

$$\langle p_i, Ap_j \rangle = \underline{\langle p_i, p_j \rangle_A} = 0$$

Orthogonal in A inner-product

$$x = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3 + \alpha_4 p_4$$

$$b = Ax = \alpha_1 Ap_1 + \alpha_2 Ap_2 + \alpha_3 Ap_3 + \alpha_4 Ap_4$$

Try to figure these out

KRYLOV METHODS

$$b = Ax = \alpha_1 Ap_1 + \alpha_2 Ap_2 + \alpha_3 Ap_3 + \alpha_4 Ap_4$$

$$\langle p_1, b\rangle = \alpha_1 \langle p_1, Ap_1\rangle + \alpha_2 \underbrace{\langle p_1, Ap_2\rangle}_{\mathbf{0}} + \alpha_3 \underbrace{\langle p_1, Ap_3\rangle}_{\mathbf{0}} + \alpha_4 \underbrace{\langle p_1, Ap_4\rangle}_{\mathbf{0}}$$

$$\langle p_1, b\rangle = \alpha_1 \langle p_1, Ap_1\rangle$$

$$\alpha_1 = \frac{\langle p_1, b\rangle}{\langle p_1, Ap_1\rangle}$$

Approximation $\quad x_1 = \alpha_1 p_1$

Residual $\quad r_1 = b - Ax_1 = \alpha_2 Ap_2 + \alpha_3 Ap_3 + \alpha_4 Ap_4$

KRYLOV METHODS

Approximation $x_1 = \alpha_1 p_1$

Residual $r_1 = b - Ax_1 = \alpha_2 Ap_2 + \alpha_3 Ap_3 + \alpha_4 Ap_4$

$$\langle p_2, r_1 \rangle = \alpha_2 \langle p_2, Ap_2 \rangle + \alpha_3 \langle p_2, Ap_3 \rangle + \alpha_4 \langle p_2, Ap_4 \rangle$$

$$\alpha_2 = \frac{\langle p_2, r_1 \rangle}{\langle p_2, Ap_2 \rangle}$$

$$x_2 = x_1 + \alpha_2 p_2 = \alpha_1 p_1 + \alpha_2 p_2$$

$$r_2 = r_1 - \alpha_2 Ap_2 = \alpha_3 Ap_3 + \alpha_4 Ap_4$$

GMRES

How to choose $\{p_k\}$?

$$x_1 = 0$$

$$p_1 = b$$

$$\alpha_1 = \frac{\langle p_1, b\rangle}{\langle p_1, Ap_1\rangle}$$

$$x_2 = x_1 + \alpha p_1$$

$$r_2 = b - \alpha A p_1$$

(generalized)
Gram–Schmidt process $\{p_1, r_2\} \rightarrow p_2$

$$\alpha_2 = \frac{\langle p_2, r_2\rangle}{\langle p_2, Ap_2\rangle}$$

$$x_3 = x_2 + \alpha_2 p_2$$

$$r_3 = r_2 - \alpha_2 A p_2$$

Gram–Schmidt process $\{p_1, p_2, r_3\} \rightarrow p_3$

GMRES

Gram–Schmidt process $\{p_1, p_2, r_3\} \longrightarrow p_3$

$$\vdots$$

$$x_k = x_{k-1} + \alpha_{k-1} p_{k-1}$$
$$r_k = r_{k-1} - \alpha_{k-1} A p_{k-1}$$

Gram–Schmidt process $\{p_1, p_2, p_3, \cdots, p_{k-1}, r_k\} \longrightarrow p_k$

GMRES(K)

$$\{p_{k-4}, p_{k-3}, p_{k-2}, p_{k-1}, r_k\} \longrightarrow p_k$$

forget this **remember this**

Pros

- Low memory requirements
- Constant iteration complexity

Cons

- No convergence theory (but works in practice)

CONJUGATE GRADIENTS

ONLY works when matrix is SPD

GMRES: $\{p_{k-4}, p_{k-3}, p_{k-2}, p_{k-1}, r_k\} \longrightarrow p_k$

Observation: $\langle r_k, Ap_j \rangle = 0, \quad \forall j < k-1$

Upshot: only need to remove p_{k-1} from r_k

$$p_k = r_k - p_{k-1} \frac{\langle p_{k-1}, Ar_k \rangle}{\langle p_{k-1}, Ap_{k-1} \rangle}$$

Only need to store x_k, r_k !

Linear iteration complexity!

CONVERGENCE

EXACT solution when k = number of eigenvalues

In terms of condition number

$$\frac{\|x_k - x^\star\|_A}{\|x^\star\|_A} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k$$

Gets BAD when κ is big

Preconditioners

$$Ax = b \xrightarrow{M \approx A^{-1}} MAx = Mb$$

$\kappa(MA) \ll \kappa(A)$

OTHER METHODS

- **MINRES**: solve ANY SYMMETRIC system
  - 2X more expensive than CG
- **QMR**: Works for ANY system
  - Similar to MINRES, but with no guarantees

(REALLY REALLY) LARGE MATRICES

Inversion

$O(N^3)$

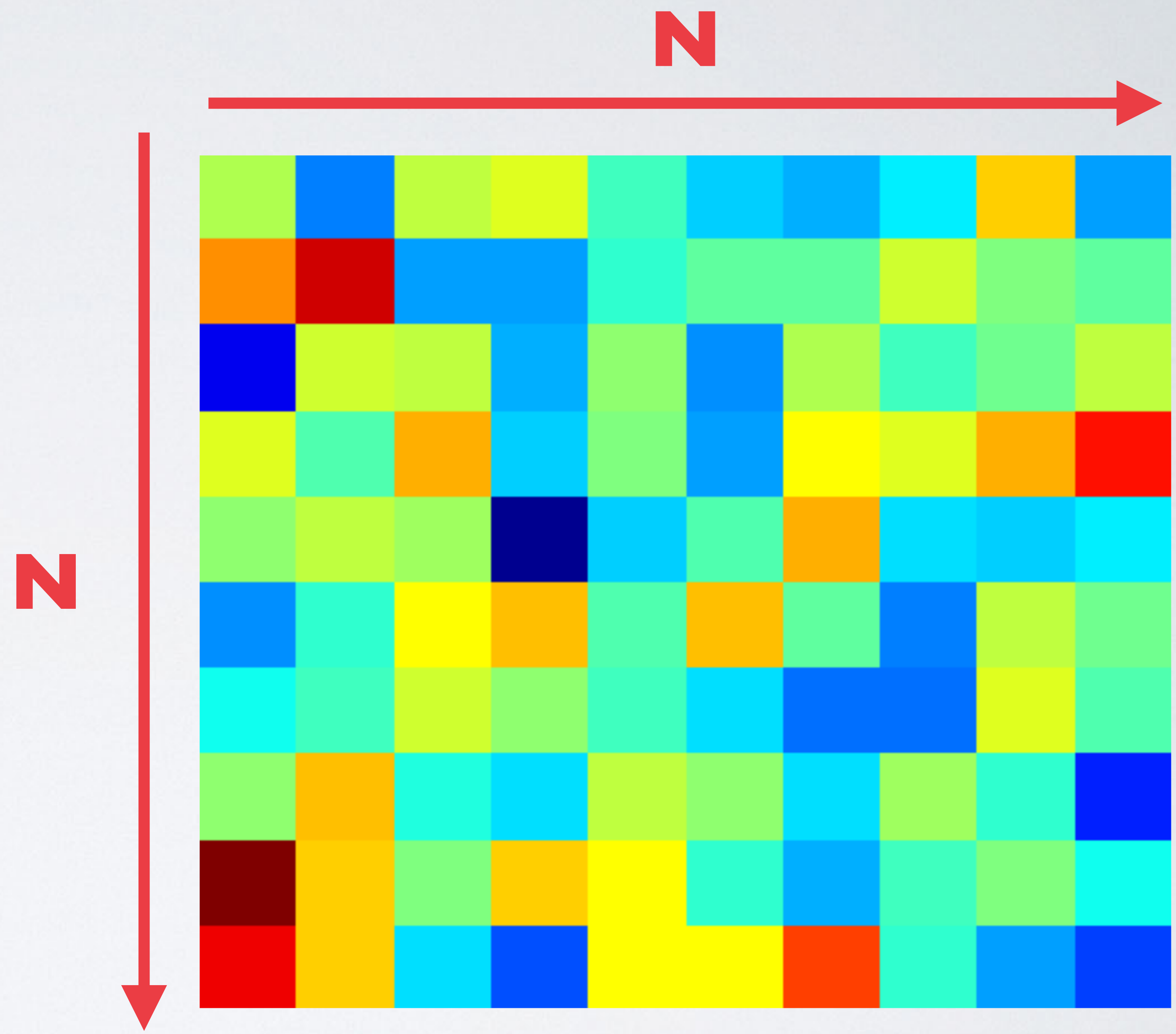
Eigenvectors

$O(N^3)$

Storage

$O(N^2)$

100K x 100k = 80 Gb RAM



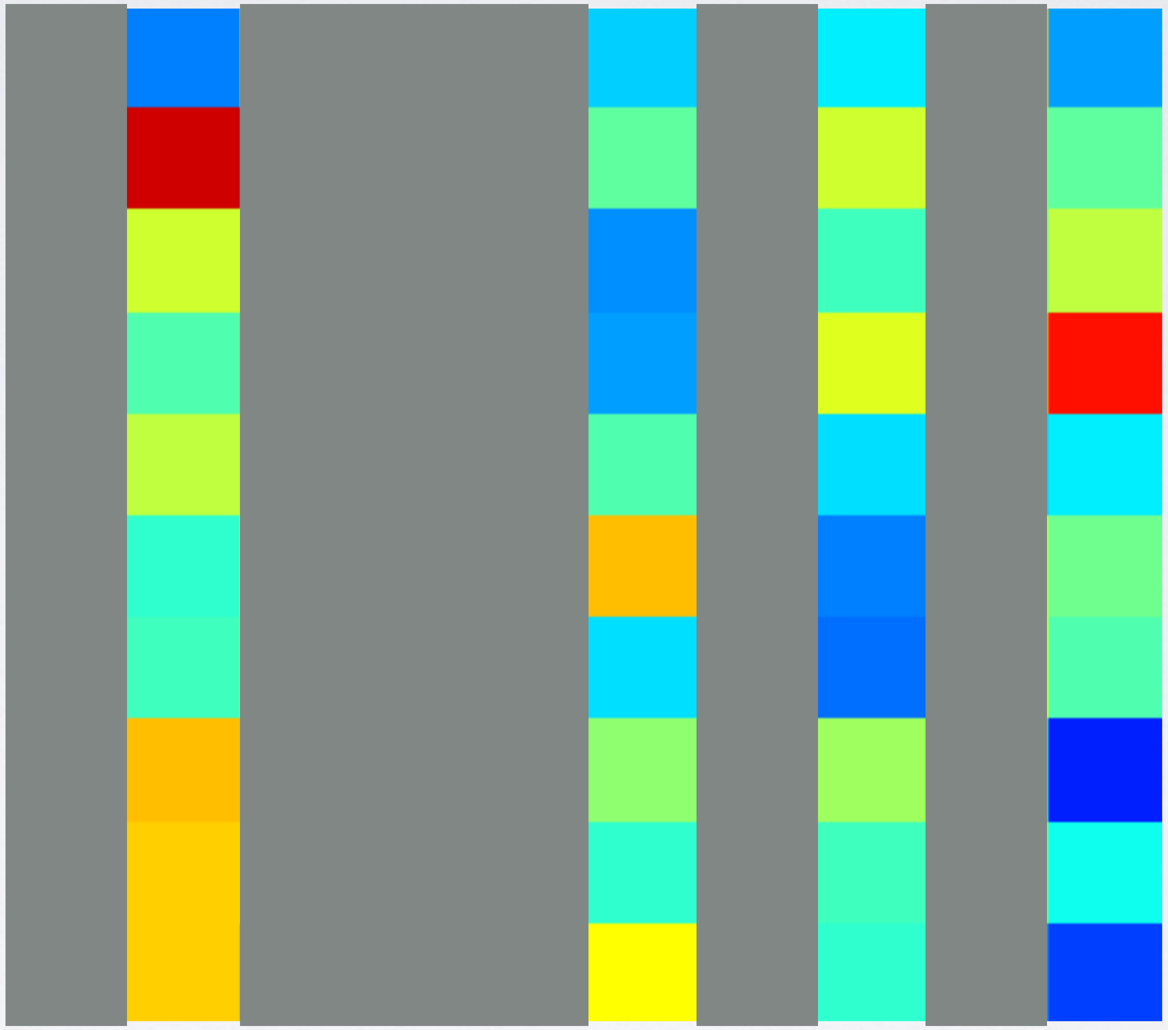
Randomized methods: handle big matrices with low complexity

NYSTROM APPROXIMATION

What if you have to factor a matrix that you don't even have?

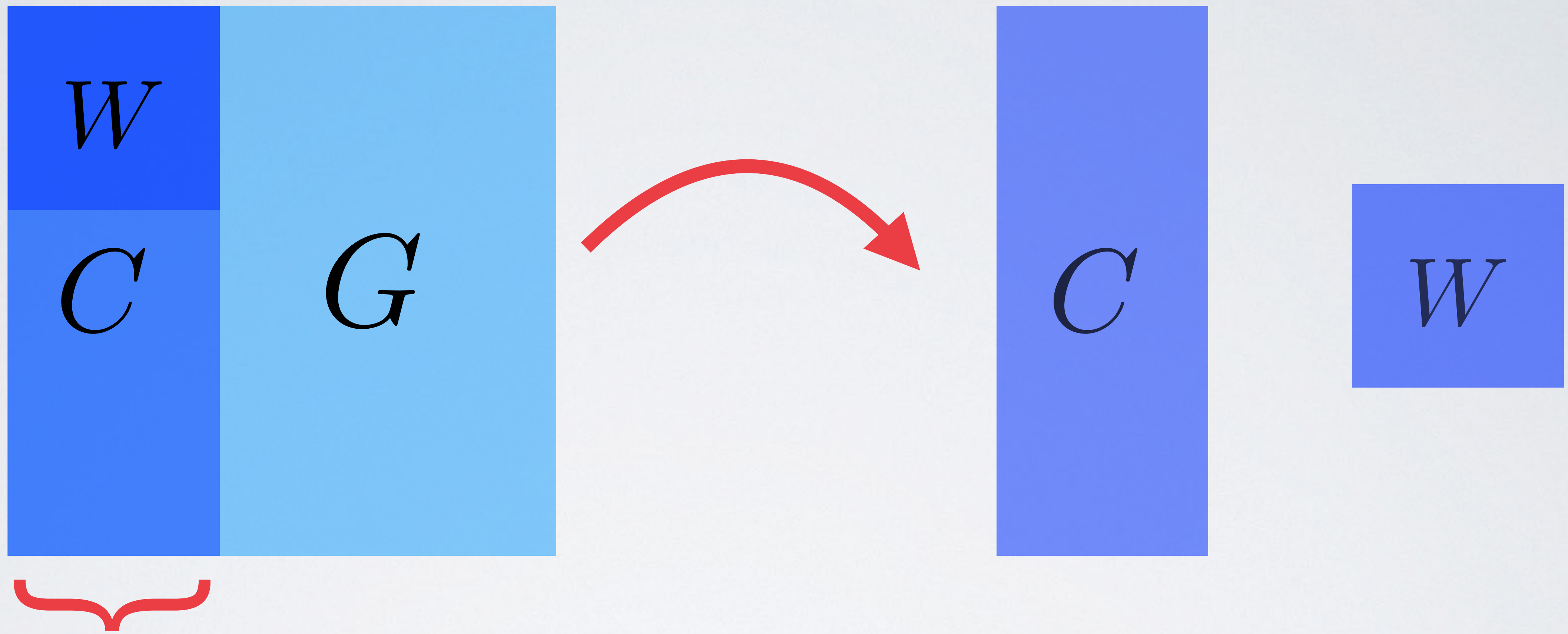
See Williams & Seeger '04

NYSTROM APPROXIMATION



If number of columns is roughly the (effective) rank, then you're ok!

NYSTROM APPROXIMATION



sampled columns

$$G \approx C W^{-1} C^T$$

See Williams & Seeger '04

WHY DOES IT WORK

For any semi-definite matrix…

$$G = X^T X \qquad\qquad X = [X_1 \;\; X_2]$$

sampled

non-sampled

$$G = \begin{bmatrix} X_1^T \\ X_2^T \end{bmatrix} [X_1 \quad X_2] = \begin{bmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_2 \end{bmatrix}$$

$$C$$

NYSTROM APPROXIMATION

$$G = \begin{bmatrix} X_1^T \\ X_2^T \end{bmatrix} \begin{bmatrix} X_1 & X_2 \end{bmatrix} = \begin{bmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_2 \end{bmatrix}$$

$$C$$

$$\widetilde{G}_k = C_k W_k^{-1} C_k^T = \begin{bmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_1 W_k^{-1} X_1^T X_2 \end{bmatrix}$$

when #(sampled columns) = rank

$$\widetilde{G}_k = C_k W_k^{-1} C_k^T = \begin{bmatrix} X_1^T X_1 & X_1^T X_2 \\ X_2^T X_1 & X_2^T X_2 \end{bmatrix}$$

STUFF YOU CAN DO WITH NYSTROM

matrix multiply

$$\mathrm{G}\,\mathbf{z} \approx C\,W^{-1}\,C^{T}\,\mathbf{z}$$

STUFF YOU CAN DO WITH NYSTROM

(Kernel) least squares with ridge penalty

$$\min_x x^T G x - x^T b + \frac{1}{2}\|x\|^2$$

$$(G + I)x = b$$

$$x = (G + I)^{-1} b$$

using matrix inversion lemma

$$\underbrace{(I+G)^{-1} \approx (I + CW^{-1}C^T)^{-1}}_{\text{Big}} = I - C\underbrace{(W + C^T C)}_{\text{Small}}{}^{-1} C^T$$

STUFF YOU CAN DO WITH NYSTROM

SVD

$$G = CW^{-1}C^T$$

$$\xrightarrow{QR}$$

$$= \underbrace{QRW^{-1}R^TQ^T}$$

$$= Q\hat{W}Q^T$$

$$\xrightarrow{\text{SVD}}$$

$$= \underbrace{Q\hat{U}}\Sigma\underbrace{\hat{V}^TQ^T}$$

$$= U\Sigma V^T$$

FOR GENERAL MATRICES?

Randomized SVD

Halko, Martins, Tropp, 2011

Sketch the matrix A

$$A \quad \Omega = B \longrightarrow \text{“Sketch”}$$

Orthogonalize

$$Q = \mathrm{orth}(B)$$

Approximate A

$$A \approx Q \quad Q^T A$$

Factorize small matrix

$$A \approx Q(\hat{U}\Sigma V^T)$$

$$A \approx U \Sigma V^T$$