COSC6364
Lecture 5: QR factorization

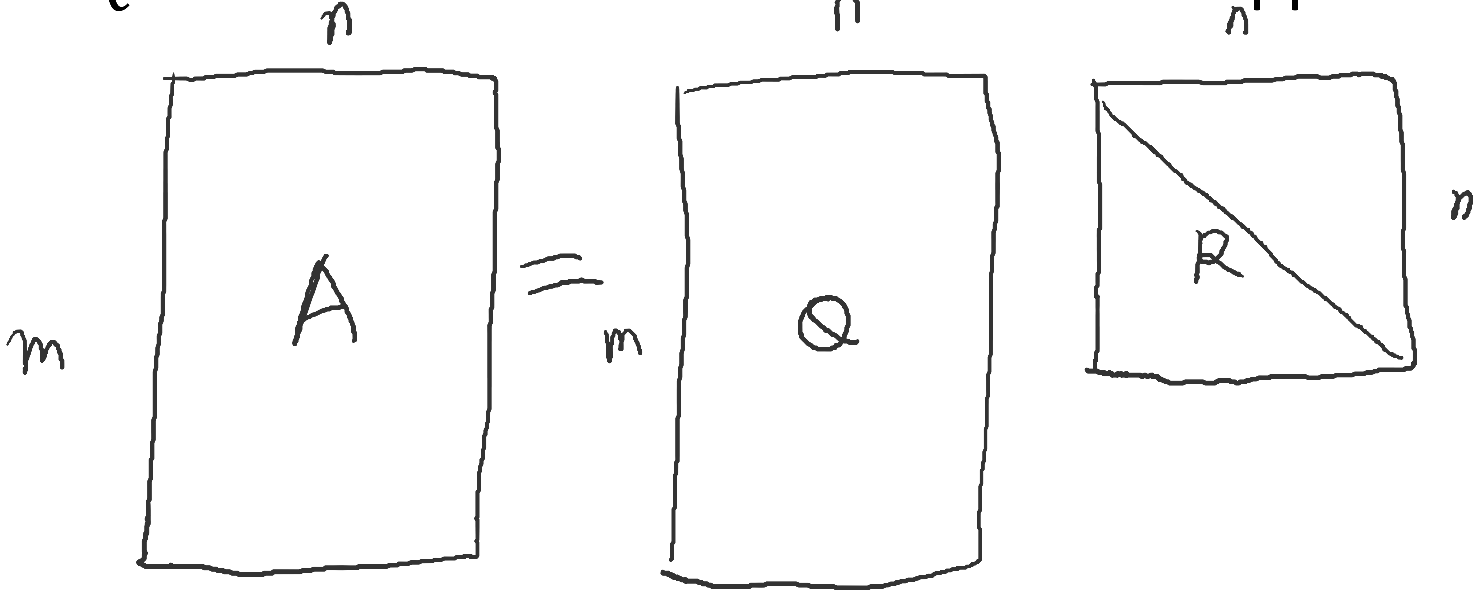
NLA Chapter 7, 8, 10

Probably the most important matrix factorization in NLA

QR factorization: (the reduce QR form)

$$A = \hat{Q}\hat{R}$$

Where $\hat{Q}$ columns are orthonormal and $\hat{R}$ is upper triangular



Reduced QR Factorization

- We are interested in the **column spaces** of a matrix A.
- In fact, we are interested in the successive spaces spanned by the columns $\langle a_1, a_2, \dots \rangle$ of A
$$\langle a_1 \rangle \subset \langle a_1, a_2 \rangle \subset \langle a_1, a_2, a_3 \rangle \subset \cdots$$
- Notation: $\langle x, y, \dots \rangle$ denotes the subspace spanned by vectors inside the brackets.
- The idea of QR: to construct a sequence of orthonormal vectors $q_1, q_2, \dots$ that span these successive spaces:
- $\langle a_1 \rangle = \langle q_1 \rangle$
  $\langle a_1, a_2 \rangle = \langle q_1, q_2 \rangle$
  $\langle a_1, a_2, a_3 \rangle = \langle q_1, q_2, q_3 \rangle$

  ...
- How to express these in terms of matrix?

Reduced QR Factorization cont'd

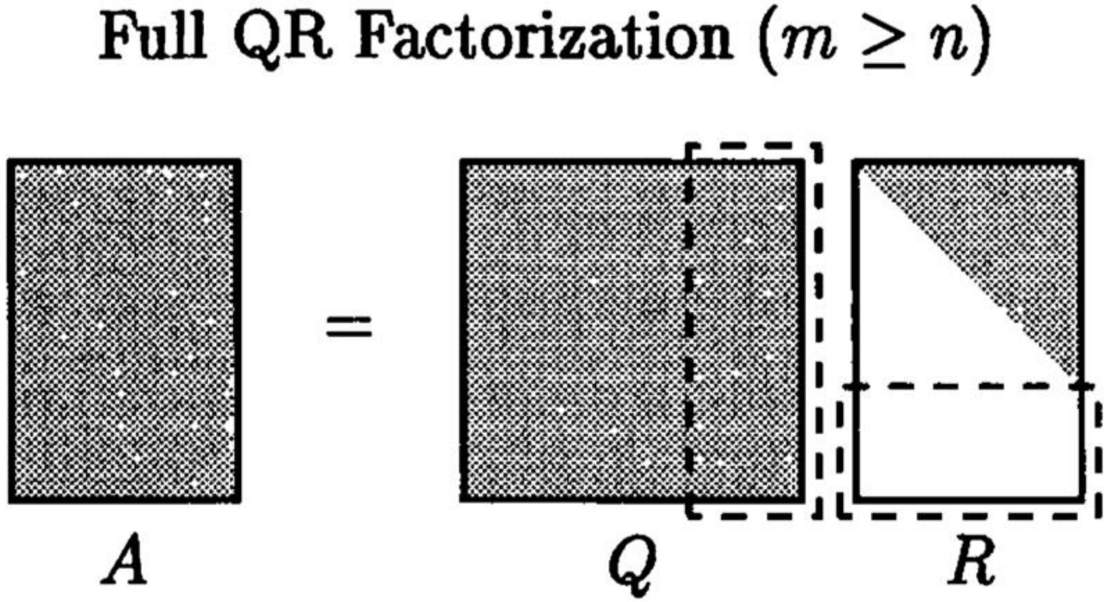
- To be precise, assume $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$ has full rank n. We want the sequence $q_1, q_2, \dots, q_n$ to have the property: $\langle q_1, q_2, \dots, q_j \rangle = \langle a_1, a_2, \dots, a_j \rangle,\ j = 1, \dots, n$
- This amounts to

$$\left[\begin{array}{c|c|c|c} & & & \\ a_1 & a_2 & \cdots & a_n \\ & & & \end{array}\right] = \left[\begin{array}{c|c|c|c} & & & \\ q_1 & q_2 & \cdots & q_n \\ & & & \end{array}\right] \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & & \vdots \\ & & \ddots & \\ & & & r_{nn} \end{bmatrix},$$

- Or, $A = \hat{Q}\hat{R}$, where $\hat{Q}$ is a $m \times n$ orthonormal columns, and $\hat{R}$ is $n \times n$ upper triangular matrix. This is the reduced QR factorization.

Full QR factorization

- The full QR factorization extends $m - n$ columns to $\hat{Q}$ to make it square $m \times m$. (Analogous to Full SVD vs. Reduced SVD)



- The silent columns of Q forms $\mathrm{range}(A)^{\perp}$

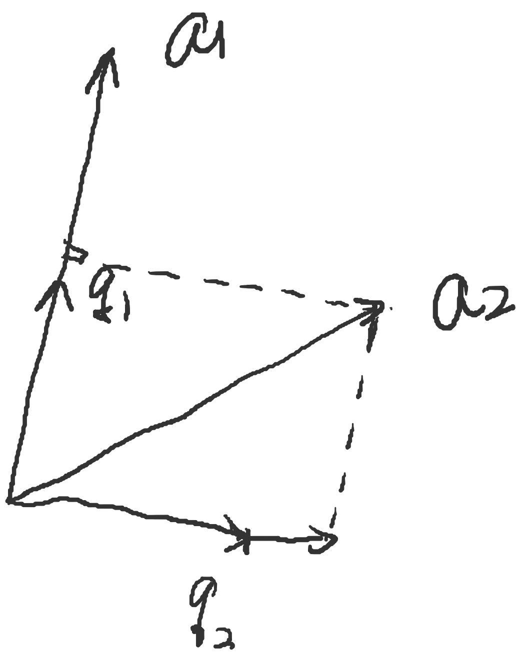
Gram-Schmidt Orthogonalization

- Look at the QR factorization in the form:

$$
\begin{aligned}
a_1 &= r_{11} q_1, \\
a_2 &= r_{12} q_1 + r_{22} q_2, \\
&\vdots \\
a_n &= r_{1n} q_1 + r_{2n} q_2 + \cdots + r_{nn} q_n
\end{aligned}
$$

- Gram-Schmidt orthogonalization computes the r_{ij} from these equations, from top to bottom.
- Can you figure out how to compute $r_{11}, r_{12}, r_{22}, \ldots$? (Hint: note that the $\{q_i\}$ are orthonormal!)

GS-Ortho



$$q_1 = \frac{a_1}{\|a_1\|}, \quad r_{11} = \|a_1\|$$

$$v_2 = a_2 - (q_1^T a_2) q_1, \quad q_2 = \frac{v_2}{\|v_2\|}$$

$$r_{12} = q_1^T a_2, \quad r_{22} = \|v_2\|$$

$$v_3 = a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2$$

$$r_{13} = q_1^T a_3, \quad r_{23} = q_2^T a_3$$

$$r_{33} = \|v_3\|$$

$$\cdots$$

$$v_j = a_j - (q_1^T a_j) q_1 - (q_2^T a_j) q_2 - \cdots - (q_{j-1}^T a_j) q_{j-1}$$

$$r_{1j} = q_1^T a_j, \quad r_{2j} = q_2^T a_j, \quad \cdots, \quad r_{j-1,j} = q_{j-1}^T a_j$$

$$r_{jj} = \|v_j\|$$

Classical Gram-Schmidt (CGS): Careful, may not be stable!

Conceptually simple to understand but numerical unstable

QR factorization by CGS:

Algorithm 7.1. Classical Gram–Schmidt (unstable)

for $j = 1$ **to** n

$\quad v_j = a_j$

$\quad$ **for** $i = 1$ **to** $j - 1$

$\quad\quad r_{ij} = q_i^* a_j$

$\quad\quad v_j = v_j - r_{ij} q_i$

$\quad r_{jj} = \|v_j\|_2$

$\quad q_j = v_j / r_{jj}$

Existence and Uniqueness of (reduced) QR

- Does QR factorization always exist?
  - Yes! You just saw how to create one…
- But is it unique?
  - Pretty much yes, but under the following conditions:
  - $A \in \mathbb{R}^{m \times n}$ $(m \geq n)$ is full rank
  - If you insist $r_{jj} > 0$

Solution of $Ax = b$ by QR Factorization

How to solve x from $Ax = b$?

If $A \in \mathbb{R}^{m \times m}$ is non-singular, then we can rewrite the equation as

$$QRx = b$$
$$Rx = Q^T b$$

And the last equation is easy to solve because R is upper triangular. So here's the algorithm to solve $Ax = b$ by QR:

1. Compute QR factorization of $A = QR$
2. Compute $y = Q^T b$
3. Solve $Rx = y$

This is actually an excellent way to solve a linear equation but not the standard one; the standard one is LU with partial pivoting.

Gram-Schmidt Orthogonalization

- GS is a “triangular orthogonalization”, which makes the columns of a matrix orthonormal via a sequence of right triangular matrix multiplications.
- Soon we’ll learn another approach of doing QR factorization called “orthogonal triangularization” (Householder QR).

GS as a sequence of orthogonal projections

Let $A \in \mathbb{R}^{m \times n}, m \geq n$ be a full rank matrix with columns $\{a_i\}$.

We can express the GS as a sequence of orthogonal projections:

$q_1 = \frac{P_1 a_1}{\|P_1 a_1\|}, q_2 = \frac{P_2 a_2}{\|P_2 a_2\|}, \ldots$

Each P_j denotes an orthogonal projector that projects a_j orthogonally onto the space orthogonal to $\langle q_1, q_2, \ldots, q_{j-1} \rangle$.

What is the projector in terms of q_j?

$$P_j = I - Q_{j-1} Q_{j-1}^T$$

Why? (Hint: what does the projector qq^T do? What does $I - qq^T$ do?)

Modified Gram-Schmidt (MGS)

- CGS is easy to understand but unfortunately not very stable.
- A mathematically equivalent version (MGS) however is stable.
- The CGS computes

$$v_j = P_j a_j$$

- Key idea (why?):

$$v_j = P_{\perp q_{j-1}} \cdots P_{\perp q_2} P_{\perp q_1} a_j$$

- Initiate $v_j = a_j, j = 1, \dots, n$.
- In the i iteration, apply $P_{\perp q_i}$ to all $v_j, j > i$.
- And that’s it! Essentially, different order of updating v_j is the difference between CGS and MGS.

MGS:

Mathematically equivalent to CGS but numerically superior

Algorithm 8.1. Modified Gram–Schmidt

for $i = 1$ **to** n

$\quad v_i = a_i$

for $i = 1$ **to** n

$\quad r_{ii} = \|v_i\|$

$\quad q_i = v_i / r_{ii}$

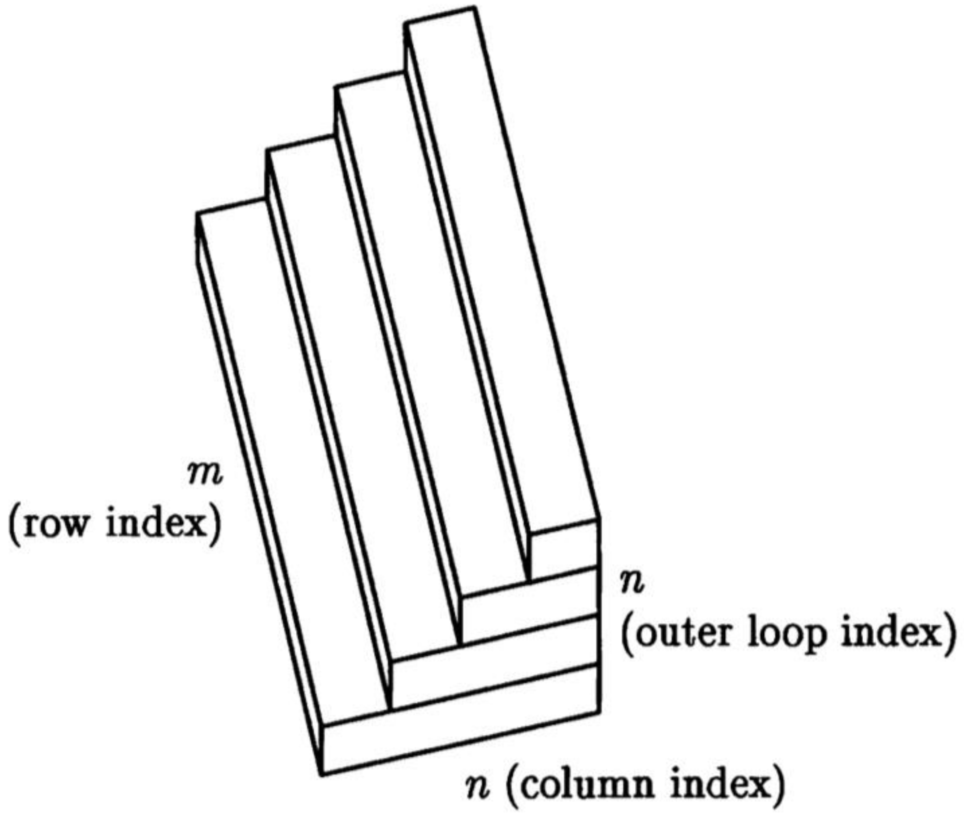
$\quad$ **for** $j = i + 1$ **to** n

$\quad\quad r_{ij} = q_i^* v_j$

$\quad\quad v_j = v_j - r_{ij} q_i$

How much does Gram-Schmidt orthogonalization cost?

- Unit of cost: floating point operations (FLOP)
- Both CGS and MGS cost about the same: $2mn^2$ FLOPs.



Gram-Schmidt as Triangular Orthogonalization

The way to look at MGS in matrix form:

The first outer iteration:

$$\left[\begin{array}{c|c|c|c} & & & \\ v_1 & v_2 & \cdots & v_n \\ & & & \end{array}\right]\begin{bmatrix} \frac{1}{r_{11}} & \frac{-r_{12}}{r_{11}} & \frac{-r_{13}}{r_{11}} & \cdots \\ & 1 & & \\ & & 1 & \\ & & & \ddots \end{bmatrix} = \left[\begin{array}{c|c|c|c} & & & \\ q_1 & v_2^{(2)} & \cdots & v_n^{(2)} \\ & & & \end{array}\right].$$

In general, each iteration is a right upper triangular matrix multiplication: $AR_1R_2\cdots R_n = \hat{Q}$.

$$R_2 = \begin{bmatrix} 1 & & & \\ & \frac{1}{r_{22}} & \frac{-r_{23}}{r_{22}} & \cdots \\ & & 1 & \\ & & & \ddots \end{bmatrix}, \quad R_3 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & \frac{1}{r_{33}} & \cdots \\ & & & \ddots \end{bmatrix}, \ldots .$$

Householder Triangularization

Householder triangularization (HT) is another method to compute QR factorization. Compared to GS orthogonalization,

- HT is more numerically stable.
- Loss of ability as basis for iterative methods
- HT is cheaper than GS (less FLOPs)
- It’s orthogonal triangularization, rather than triangular orthogonalization:

Householder vs. Gram-Schmidt

The GS iteration applies a succession of elementary triangular matrices R_k on the right of A:

$$A \underbrace{R_1 R_2 \dots R_n}_{\hat{R}^{-1}} = \hat{Q}$$

so that the resulting matrix has orthonormal columns.

In contrast, the Householder method applies a succession of elementary unitary matrices Q_k on the left of A:

$$\underbrace{Q_n \dots Q_2 Q_1}_{Q^*} A = R$$

so that the resulting matrix is upper triangular.

Triangularization by Introducing Zeros

At the heart of Householder Triangularization is the Householder reflector (orthogonal matrices) that <u>introduces zeros below the diagonal in the kth column while preserving all the zeros previously introduced</u>.

$$\underset{A}{\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}} \xrightarrow{Q_1} \underset{Q_1A}{\begin{bmatrix} \boldsymbol{\times} & \boldsymbol{\times} & \boldsymbol{\times} \\ \mathbf{0} & \boldsymbol{\times} & \boldsymbol{\times} \\ \mathbf{0} & \boldsymbol{\times} & \boldsymbol{\times} \\ \mathbf{0} & \boldsymbol{\times} & \boldsymbol{\times} \\ \mathbf{0} & \boldsymbol{\times} & \boldsymbol{\times} \end{bmatrix}} \xrightarrow{Q_2} \underset{Q_2Q_1A}{\begin{bmatrix} \times & \times & \times \\ & \boldsymbol{\times} & \boldsymbol{\times} \\ & \mathbf{0} & \boldsymbol{\times} \\ & \mathbf{0} & \boldsymbol{\times} \\ & \mathbf{0} & \boldsymbol{\times} \end{bmatrix}} \xrightarrow{Q_3} \underset{Q_3Q_2Q_1A}{\begin{bmatrix} \times & \times & \times \\ & \times & \times \\ & & \boldsymbol{\times} \\ & & \mathbf{0} \\ & & \mathbf{0} \end{bmatrix}} \qquad (10.1)$$

Householder Reflector (1/3)

How do we construct such orthogonal matrices $Q_1, Q_2, \ldots$?

Here's a standard approach. Each Q_k is taken to be:

$$Q_k = \begin{bmatrix} I & 0 \\ 0 & F \end{bmatrix}$$

where $I \in \mathbb{R}^{(k-1)\times(k-1)}$, and $F \in \mathbb{R}^{(m-k+1)\times(m-k+1)}$ is orthogonal.

Q_k introduces zeros into the kth column. We choose F to be a particular matrix called <u>Householder Reflector</u>

Householder Reflector (2/3)

Suppose, at the beginning of step k, the entries $k, k+1, \ldots, m$ of the kth column are given by vector $x \in \mathbb{R}^{m-k+1}$. The Householder reflector F should effect the following map:

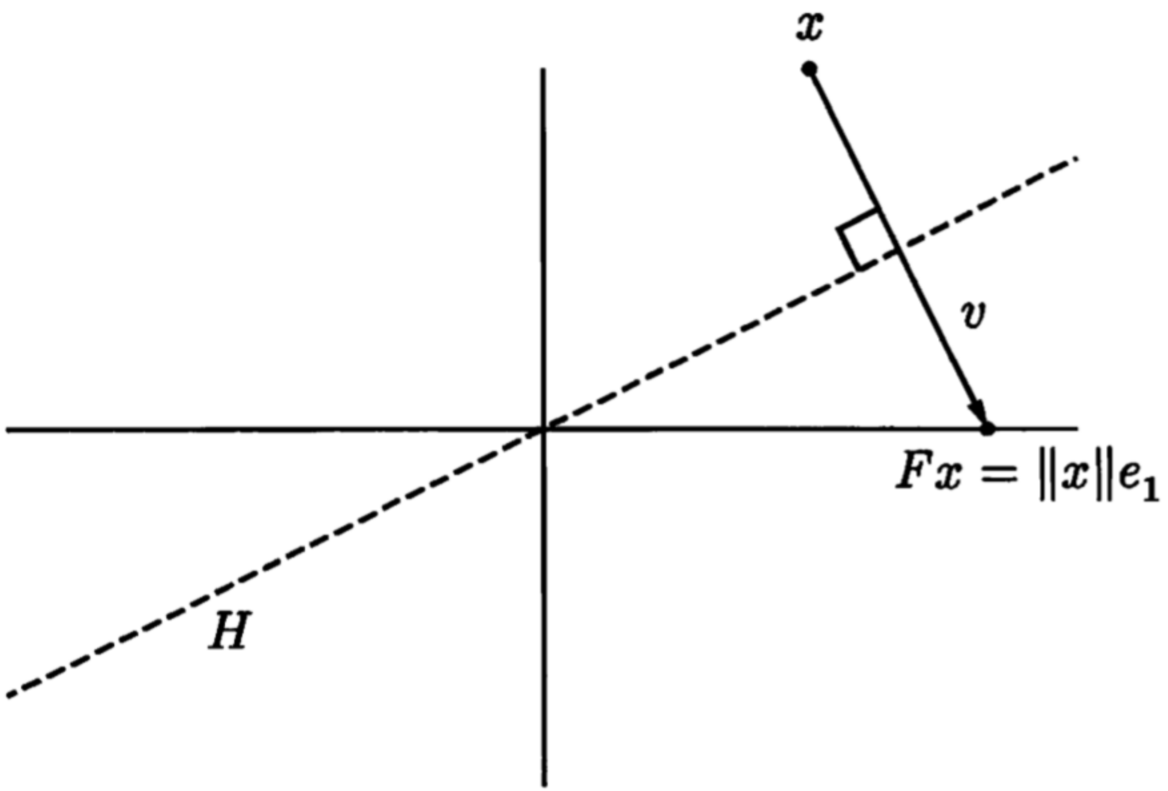
$$x = \begin{bmatrix} \times \\ \times \\ \times \\ \vdots \\ \times \end{bmatrix} \quad \xrightarrow{F} \quad Fx = \begin{bmatrix} \|x\| \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \|x\| e_1. \qquad (10.3)$$

(Why should the first component of Fx be $\|x\|$?)

Householder Reflector (3/3)

The reflector F will reflect across the hyperplane H orthogonal to $v = \|x\|e_1 - x$.

A hyperplane is a generalization of plane in 2D (e.g. 3-dimensional subspace in 4D space). A hyperplane can be characterized as all the vectors orthogonal to a fixed vector (normal vector).



Projector and Reflector

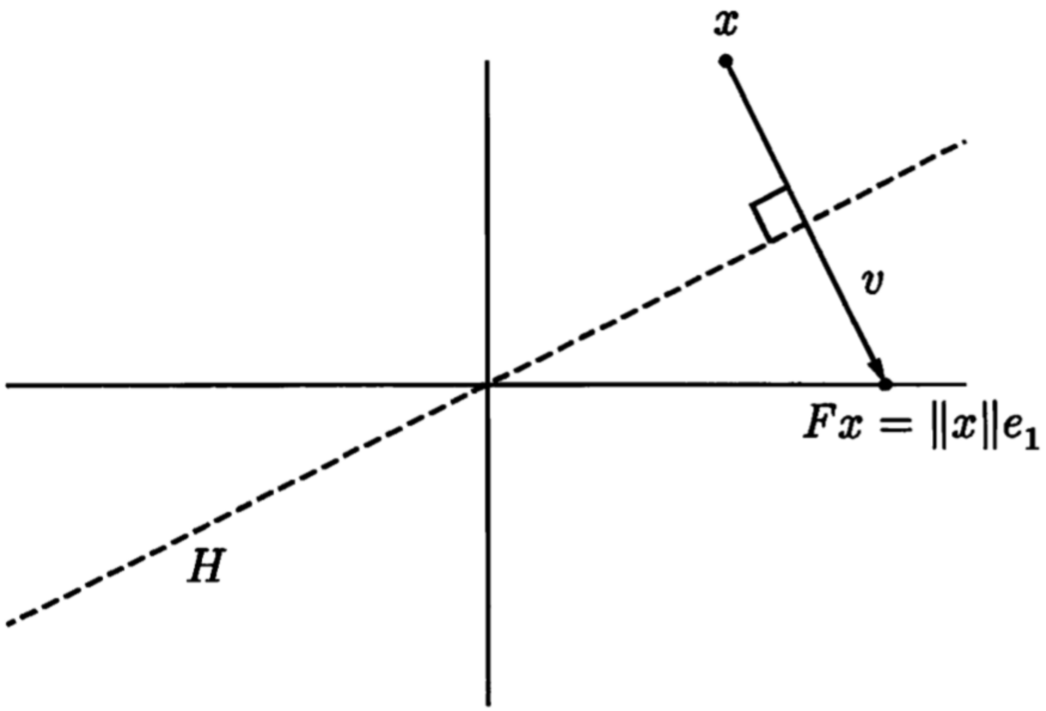
Remember the projectors? What is the projector that maps a vector y to $\langle v \rangle$? What projector maps y to H?

$$P = I - \frac{vv^T}{v^T v}$$

To reflect y across H, we must not stop at projection onto H; we must go exactly twice as far in the same direction:

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

(Which one of P, F is orthogonal?)

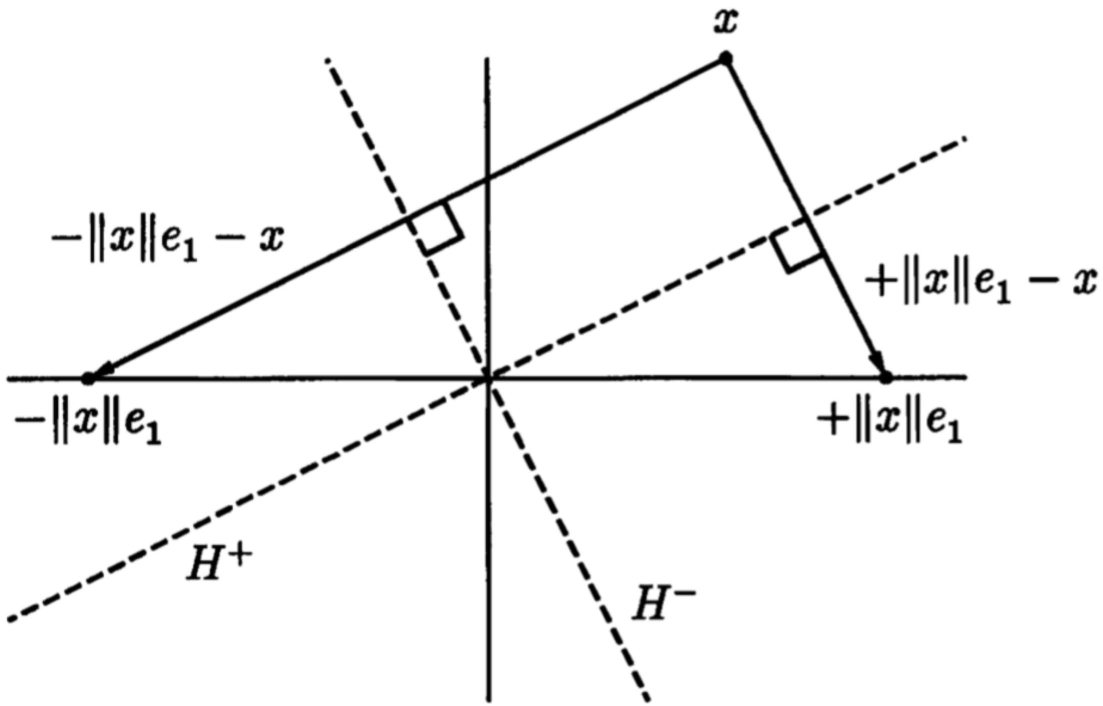


The Better of Two Reflectors

In fact we can have two reflectors… is there a difference?

Two possible images on the axis, $\pm\,\|x\|e_1$. We'll always want the furthest one from x. Why?

It turns out that we will calculate $v = \pm\|x\|e_1 - x$. If they are close the result is very sensitive to rounding errors.



Householder QR

Given a matrix $A \in \mathbb{R}^{m \times n}, m \geq n$, the following algorithm computes a QR factorization of A. The upper triangular part of R will overwrite A. The Q factor is stored as Householder vectors $v_1, \dots, v_n$.
We can further store the Householder vectors in lower triangular part of A so that we completely overwritten A and need no extra memory.

Algorithm 10.1. Householder QR Factorization

for $k = 1$ **to** n

$x = A_{k:m,k}$

$v_k = \operatorname{sign}(x_1)\|x\|_2 e_1 + x$

$v_k = v_k / \|v_k\|_2$

$A_{k:m,k:n} = A_{k:m,k:n} - 2v_k(v_k^* A_{k:m,k:n})$

Applying or Forming Q

OK, but how do I get my good Q from the Householder vectors, you might ask?

It turns out that, you probably can get away without forming Q, if all you need to to multiply Q or Q^T by something (applying).

But if you insist, you can form the explicit Q by applying Q to the identity matrix I…

Let’s see.

Applying Q and Q^T

We know that:

$$Q^T = Q_n \cdots Q_2 Q_1$$

and its conjugate:

$$Q = Q_1 Q_2 \cdots Q_n$$

(why there's no transpose?)

Algorithm 10.2. Implicit Calculation of a Product Q^*b

for $k = 1$ **to** n

$$b_{k:m} = b_{k:m} - 2v_k(v_k^* b_{k:m})$$

Algorithm 10.3. Implicit Calculation of a Product Qx

for $k = n$ **downto** 1

$$x_{k:m} = x_{k:m} - 2v_k(v_k^* x_{k:m})$$

Forming Q

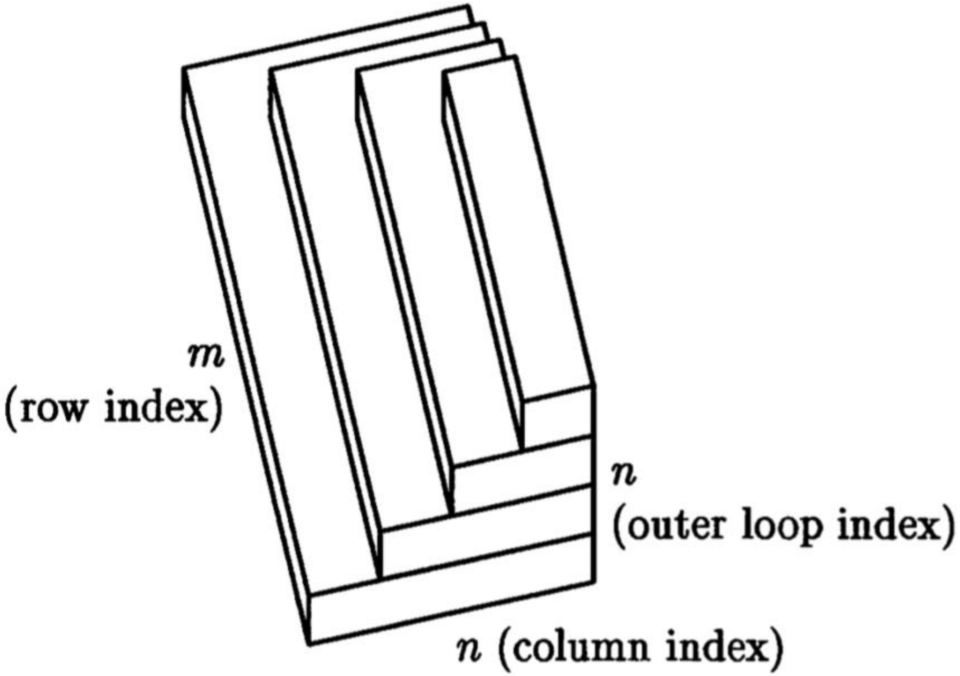
A bunch of ways:

1. QI
2. $Q^T I$ then transpose
3. IQ

The first method is the best. Why?

Operation Count

Householder QR:



$\cdots$

$$\sim 2mn^2 - \frac{2}{3}n^3 \text{ flops}$$

Classical vs. Modified Gram-Schmidt

Numerical stability between CGS and MGS.

First we create a matrix with a widely varying singular values spaced by factor of 2 between 2^{-1} and 2^{-80} using Matlab:

`[U,X] = qr(randn(80));`	Set U to a random orthogonal matrix.
`[V,X] = qr(randn(80));`	Set V to a random orthogonal matrix.
`S=diag(2.^(-1:-1:-80));`	Set S to a diagonal matrix with exponentially graded entries.
`A = U*S*V;`	Set A to a matrix with these entries as singular values.

CGS and MGS to do QR

We plot the diagonal elements r_{ii}. Since $r_{ii} = \|P_i a_i\|$, we would estimate that r_{ii} closes matches the i-th singular value $\sigma_i = 2^{-i}$ within a constant factor.

Here's the result of the r_{ii} produced by CGS and MGS QR:

r_{ii} of CGS and MGS