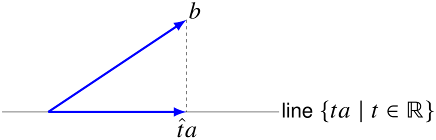
8. QR factorization

- Gram-Schmidt orthogonalization
- QR factorization
- solving linear equations
- modified Gram-Schmidt method
- Householder algorithm

Projection onto a vector

given two vectors $a, b \in \mathbb{R}^n$, with $a \neq 0$, the vector multiple ta closest to b has

$$\hat{t} = \frac{a^T b}{a^T a} = \frac{a^T b}{\|a\|^2}$$



Proof

- squared distance between ta and b is

$$\|ta - b\|^2 = (ta - b)^T (ta - b) = t^2 a^T a - 2t a^T b + b^T b$$

- derivative w.r.t. t is zero for

$$\hat{t} = \frac{a^T b}{a^T a} = \frac{a^T b}{\|a\|^2}$$

Geometric interpretation: $b - \hat{t}a \perp a$:

$$(b - \hat{t}a)^T a = 0 \Longrightarrow \hat{t} = \frac{a^T b}{\|a\|^2}$$

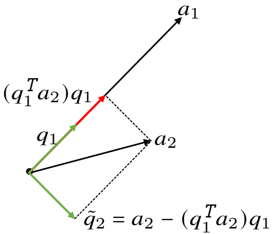
Gram-Schmidt procedure on two vectors

G-S procedure on two non-zero vectors a_1 and a_2

- normalize $q_1 = a_1/\|a_1\|$
- remove q_1 component from a_2:

$$\tilde{q}_2 = a_2 - (q_1^T a_2) q_1$$

- normalize $q_2 = \tilde{q}_2/\|\tilde{q}_2\|$



- if $\tilde{q}_2 = 0$, then the vectors a_1 and a_2 are linearly dependent
- if $\tilde{q}_2 \neq 0$, then q_1 and q_2 are orthonormal (q_1 and $\tilde{q}_2$ are orthogonal)

$$\begin{aligned} q_1^T \tilde{q}_2 = q_1^T (a_2 - (q_1^T a_2) q_1) &= q_1^T a_2 - (q_1^T a_2) q_1^T q_1 \\ &= q_1^T a_2 - q_1^T a_2 = 0 \end{aligned}$$

thus, a_1 and a_2 are linearly independent

Gram-Schmidt (G-S) procedure

given vectors $a_1, \ldots, a_n \in \mathbb{R}^m$

step 1a. $\tilde{q}_1 := a_1$

step 1b. $q_1 := \tilde{q}_1 / \|\tilde{q}_1\|$ (normalize)

step 2a. $\tilde{q}_2 := a_2 - (q_1^T a_2) q_1$ (remove q_1 component from a_2)

step 2b. $q_2 := \tilde{q}_2 / \|\tilde{q}_2\|$ (normalize)

step 3a. $\tilde{q}_3 := a_3 - (q_1^T a_3) q_1 - (q_2^T a_3) q_2$ (remove q_1, q_2 components)

step 3b. $q_3 := \tilde{q}_3 / \|\tilde{q}_3\|$ (normalize)

etc.

Gram-Schmidt (G-S) algorithm

---

given vectors $a_1, \ldots, a_n \in \mathbb{R}^m$

set $q_1 = a_1/\|a_1\|$

for $k = 2, \ldots, n$

1. *orthogonalization:* $\tilde{q}_k = a_k - (q_1^T a_k)q_1 - \cdots - (q_{k-1}^T a_k)q_{k-1}$
2. *test for linear dependence:* if $\tilde{q}_k = 0$ quit
3. *normalization:* $q_k = \tilde{q}_k/\|\tilde{q}_k\|$

- if $\tilde{q}_k = 0$ then $a_1, \ldots, a_k$ are linearly dependent
- if $a_1, \ldots, a_n$ are linearly independent, then $q_1, \ldots, q_n$ are orthonormal vectors
- a_k is a linear combination of $q_1, \ldots, q_k$
- q_k is a linear combination of $a_1, \ldots, a_k$

Example

$$a_1 = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad a_2 = \begin{bmatrix} -1 \\ 3 \\ -1 \\ 3 \end{bmatrix}, \quad a_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \end{bmatrix}$$

- $k = 1$, $\|a_1\| = 2$ and

$$q_1 = a_1/\|a_1\| = (-1/2, 1/2, -1/2, 1/2)$$

- $k = 2$, we have $q_1^T a_2 = 4$, and

$$\tilde{q}_2 = a_2 - (q_1^T a_2) q_1 = (1, 1, 1, 1)$$

normalizing, we get

$$q_2 = \tilde{q}_2/\|\tilde{q}_2\| = (1/2, 1/2, 1/2, 1/2)$$

- $k = 3$; we have $q_1^T a_3 = 2$ and $q_2^T a_3 = 8$, so

$$\tilde{q}_3 = a_3 - (q_1^T a_3)q_1 - (q_2^T a_3)q_2 = (-2, -2, 2, 2)$$

  normalizing, we get

$$q_3 = \tilde{q}_3 / \|\tilde{q}_3\| = (-1/2, -1/2, 1/2, 1/2)$$

- since no vector $\tilde{q}_i$ is zero, the vectors a_1, a_2, a_3 are linearly independent

Matrix form for Gram-Schmidt

let A be an $m \times n$ matrix with linearly independent columns

- running Gram-Schmidt on A produces orthonormal vectors $q_1, \ldots, q_n$
- we know from Gram-Schmidt algorithm that

$$\begin{aligned} a_k &= (q_1^T a_k) q_1 + \cdots + (q_{k-1}^T a_k) q_{k-1} + \|\tilde{q}_k\| q_k \\ &= R_{1k} q_1 + \cdots + R_{k-1,k} q_{k-1} + R_{kk} q_k \end{aligned}$$

  where $R_{ij} = q_i^T a_j$ and $R_{ii} = \|\tilde{q}_i\|$

- expressing this for each $k = 1, \ldots, n$,

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

$$A = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \ldots & R_{1n} \\ 0 & R_{22} & \ldots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & R_{nn} \end{bmatrix}$$

Outline

- Gram-Schmidt orthogonalization
- **QR factorization**
- solving linear equations
- modified Gram-Schmidt method
- Householder algorithm

QR factorization

if $A \in \mathbb{R}^{m \times n}$ has linearly independent columns, then it can be factored as

$$A = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & \dots & R_{1n} \\ 0 & R_{22} & \dots & R_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & R_{nn} \end{bmatrix} = QR$$

- $Q \in \mathbb{R}^{m \times n}$ has orthonormal columns ($Q^T Q = I$)
- if A is square $(m = n)$, then Q is orthogonal $\left(Q^T Q = Q Q^T = I\right)$
- $R \in \mathbb{R}^{n \times n}$ is upper triangular with nonzero diagonal, hence invertible

QR factorization via Gram-Schmidt

---

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$

set $q_1 = a_1/\|a_1\|$ and $R_{11} = \|a_1\|$

for $k = 2, \ldots, n$

1. $q_k = a_k$
2. **for** $j = 1, \ldots, k-1$

   $R_{jk} = q_j^T a_k$
   $q_k = q_k - R_{jk} q_j$
3. set

   $R_{kk} = \|q_k\|$
   $q_k = q_k / R_{kk}$

- R is generated column by column
- **complexity:** $\approx 2mn^2 \text{flops}$

Example

from calculations in last example, we have

$$R_{11} = \|\tilde{q}_1\| = 2, \quad R_{12} = q_1^T a_2 = 4$$
$$R_{22} = \|\tilde{q}_2\| = 2, \quad R_{13} = q_1^T a_3 = 2$$
$$R_{23} = q_2^T a_3 = 8, \quad R_{33} = \|\tilde{q}_3\| = 4$$

therefore,

$$\begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix} = \begin{bmatrix} q_1 & q_2 & q_3 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \end{bmatrix}$$

$$= \begin{bmatrix} -1/2 & 1/2 & -1/2 \\ 1/2 & 1/2 & -1/2 \\ -1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & 2 & 8 \\ 0 & 0 & 4 \end{bmatrix}$$

Full QR factorization

suppose $A \in \mathbb{R}^{m \times n}$ has linearly independent columns ($m \geq n$)

the *full QR factorization* or *QR decomposition* of A is

$$A = \begin{bmatrix} Q & \tilde{Q} \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}$$

- $A = QR$ is the (reduced) QR factorization as defined earlier
- $[Q \quad \tilde{Q}]$ is $m \times m$ and orthogonal; $\tilde{Q}$ has size $m \times (m - n)$
- the zero block has size $(m - n) \times n$ (size of right matrix is $m \times n$)
- given $A = QR$, we can find $\tilde{Q}$ as follows:
  - find any matrix $\tilde{A}$ such that $[A \ \ \tilde{A}]$ has linearly independent columns (*e.g.*, $\tilde{A} = I$)
  - apply Gram-Schmidt to $[A \ \ \tilde{A}]$ to find $\tilde{Q}$
- in MATLAB's: `[Q,R]=qr(A)`

Computing Cholesky factorization of Gram matrix

- suppose A is an $m \times n$ matrix with linearly independent columns
- the Gram matrix $C = A^T A$ is positive definite

two methods for computing the Cholesky factor of C, given A

1. compute $C = A^T A$, then Cholesky factorization of C

$$C = R^T R$$

2. compute QR factorization $A = QR$; since

$$C = A^T A = R^T Q^T Q R = R^T R$$

   the matrix R is the Cholesky factor of C

Example

$$A = \begin{bmatrix} 3 & -6 \\ 4 & -8 \\ 0 & 1 \end{bmatrix}, \quad C = A^T A = \begin{bmatrix} 25 & -50 \\ -50 & 101 \end{bmatrix}$$

1. Cholesky factorization:

$$C = \begin{bmatrix} 5 & 0 \\ -10 & 1 \end{bmatrix} \begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix}$$

2. QR factorization

$$A = \begin{bmatrix} 3 & -6 \\ 4 & -8 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 3/5 & 0 \\ 4/5 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 5 & -10 \\ 0 & 1 \end{bmatrix}$$

Pseudo-inverse via QR factorization

pseudo-inverse of A with linearly independent columns with $A = QR$ is

$$\begin{aligned}
A^\dagger &= (A^TA)^{-1}A^T \\
&= \left((QR)^T(QR)\right)^{-1}(QR)^T \\
&= (R^TQ^TQR)^{-1}R^TQ^T \\
&= (R^TR)^{-1}R^TQ^T \quad (Q^TQ = I) \\
&= R^{-1}R^{-T}R^TQ^T \quad (R \text{ is nonsingular}) \\
&= R^{-1}Q^T
\end{aligned}$$

- for square nonsingular A this is the inverse: $A^{-1} = (QR)^{-1} = R^{-1}Q^T$
- pseudo-inverse of A with linearly independent rows with $A^T = \tilde{Q}\tilde{R}$ is

$$A^\dagger = A^T(AA^T)^{-1} = \tilde{Q}\tilde{R}^{-T}$$

Range of a matrix

- the *span* of a collection of vectors is the set of all their linear combinations:

$$\text{span}(a_1, a_2, \ldots, a_n) = \{x_1 a_1 + x_2 a_2 + \cdots + x_n a_n \mid x \in \mathbb{R}^n\}$$

- the *range* (column space) of a matrix $A \in \mathbb{R}^{m \times n}$ is the span of its column vectors:

$$\text{range}(A) = \{Ax \mid x \in \mathbb{R}^n\}$$

Example

$$\text{range}\left(\begin{bmatrix} 1 & 0 \\ 1 & 2 \\ 0 & -1 \end{bmatrix}\right) = \left\{\begin{bmatrix} x_1 \\ x_1 + 2x_2 \\ -x_2 \end{bmatrix} \mid x_1, x_2 \in \mathbb{R}\right\}$$

Range and QR factorization

suppose A has linearly independent columns with QR factorization $A = QR$

- Q has the same range as A:

$$\begin{aligned} y \in \text{range}(A) &\iff y = Ax \text{ for some } x \\ &\iff y = QRx \text{ for some } x \\ &\iff y = Qz \text{ for some } z \\ &\iff y \in \text{range}(Q) \end{aligned}$$

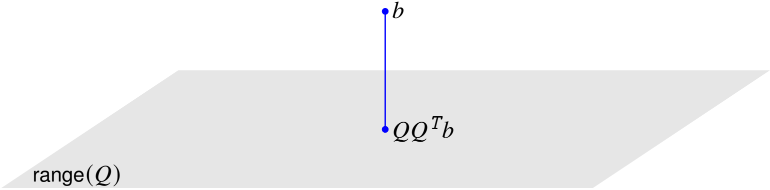
- columns of Q are an orthonormal *basis* for $\text{range}(A)$:
  they are linearly independent and $\text{span}(q_1, \ldots, q_n) = \text{range}(A)$

Projection on range of matrix with orthonormal columns

if $Q \in \mathbb{R}^{m \times n}$ has orthonormal columns $q_1, \ldots, q_n$, then the vector

$$QQ^T b$$

is the *orthogonal projection* of an m-vector b on $\mathrm{range}(Q)$



$\hat{x} = Q^T b$ satisfies $\|Q\hat{x} - b\| < \|Qx - b\|$ for all $x \neq \hat{x}$ (proof shown next on page)

Proof: the squared distance of b to an arbitrary point Qx in range(Q) is

$$\begin{aligned}\|Qx - b\|^2 &= \|Q(x - \hat{x}) + Q\hat{x} - b\|^2 \quad \text{(where } \hat{x} = Q^T b\text{)} \\ &= \|Q(x - \hat{x})\|^2 + \|Q\hat{x} - b\|^2 + 2(x - \hat{x})^T Q^T (Q\hat{x} - b) \\ &= \|Q(x - \hat{x})\|^2 + \|Q\hat{x} - b\|^2 \\ &= \|x - \hat{x}\|^2 + \|Q\hat{x} - b\|^2 \\ &\geq \|Q\hat{x} - b\|^2\end{aligned}$$

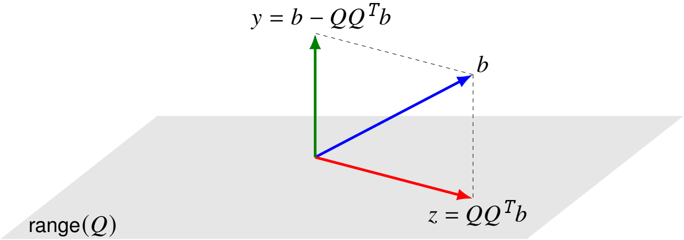
with equality only if $x = \hat{x}$

- line 3 follows because $Q^T(Q\hat{x} - b) = \hat{x} - Q^T b = 0$
- line 4 follows from $Q^T Q = I$

Orthogonal decomposition

the vector b is decomposed as a sum $b = z + y$ with

$$z \in \text{range}(Q), \quad y \in \text{range}(Q)^\perp$$



- decomposition exists and unique for every b:

$$b = Qx + y, \quad Q^Ty = 0 \quad \Longleftrightarrow \quad x = Q^Tb, \quad y = b - QQ^Tb$$

- y is orthogonal projection on range$(Q)^\perp = \{u \mid Q^Tu = 0\}$

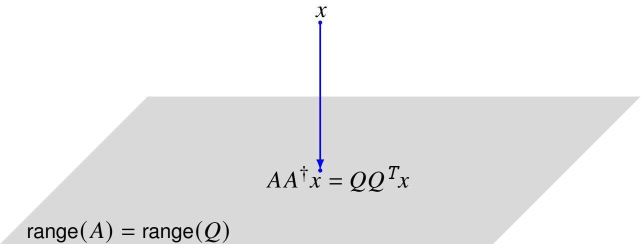
Pseudo-inverse and projection on range

- using $A = QR$ and $A^\dagger = R^{-1}Q^T$ gives

$$AA^\dagger = QRR^{-1}Q^T = QQ^T$$

note that $AA^\dagger$ and is different from $A^\dagger A = I$

- hence $AA^\dagger x = QQ^T x$ is the projection of x onto range(Q) = range(A)



Outline

- Gram-Schmidt orthogonalization
- QR factorization
- **solving linear equations**
- modified Gram-Schmidt method
- Householder algorithm

Solving linear equations

- assuming A is nonsingular, then $x = A^{-1}b$ solves $Ax = b$
- with QR factorization $A = QR$, we have $A^{-1} = (QR)^{-1} = R^{-1}Q^T$
- compute $x = R^{-1}(Q^T b)$ by back substitution

QR factorization method: to solve $Ax = b$ with nonsingular $A \in \mathbb{R}^{n \times n}$

1. factor A as $A = QR$
2. compute $Q^T b$
3. solve $Rx = y$ by back substitution

Complexity: $2n^3 + 3n^2 \approx 2n^3$

- QR factorization $2n^3$ flops
- matrix-vector product $2n^2$
- back substitution n^2

Multiple right-hand sides

consider k sets of linear equations with the same coefficient matrix A:

$$Ax_1 = b_1, \quad Ax_2 = b_2, \quad \ldots, \quad Ax_k = b_k$$

- let $X = [x_1 \ \cdots \ x_k]$ and $B = [b_1 \ \cdots \ b_k]$, each is an $n \times k$ matrix
- express equations as $AX = B$
- can be solved in $2n^3 + 3kn^2$ flops if we reuse the factorization $A = QR$
- for $k \ll n$, cost is roughly equal to cost of solving one equation $(2n^3)$

Computing the inverse

solving the matrix equation $AX = I$ gives A^{-1}

- equivalent to n equations:

$$Rx_1 = Q^T e_1, \quad Rx_2 = Q^T e_2, \quad \ldots, \quad Rx_n = Q^T e_n$$

- x_i is ith column of X and $Q^T e_i$ is the ith column of Q^T
- complexity is $2n^3 + n^3 = 3n^3$

Solving linear equations by computing the inverse

- compute inverse A^{-1} costs $3n^3$, then compute $A^{-1}b$ costs $2n^2$
- total complexity: $3n^3 + 2n^2 \approx 3n^3$
- more expensive than QR factorization method, which costs $2n^3$
- while inverse appears in many formulas, it is computed far less often

Outline

- Gram-Schmidt orthogonalization
- QR factorization
- solving linear equations
- **modified Gram-Schmidt method**
- Householder algorithm

Numerical instability of G-S

consider the following MATLAB implementation of the G-S algorithm

```
[m, n] = size(A);
Q = zeros(m,n);
R = zeros(n,n);
for k = 1:n
R(1:k-1,k) = Q(:,1:k-1)' * A(:,k);
qtilde = A(:,k) - Q(:,1:k-1) * R(1:k-1,k);
R(k,k) = norm(qtilde);
Q(:,k) = qtilde / R(k,k);
end;
```

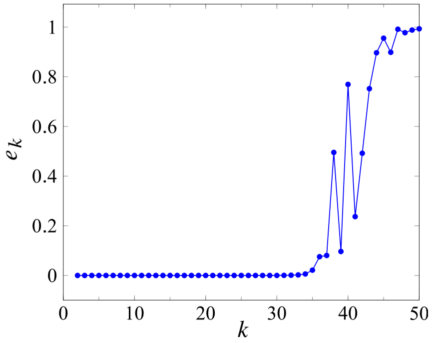
- we apply this to a square matrix A of size $m = n = 50$
- A is constructed as $A = USV$ with U, V orthogonal, S diagonal with

$$S_{ii} = 10^{-10(i-1)/(n-1)}, \quad i = 1, \ldots, n$$

Numerical instability of G-S

plot shows deviation from orthogonality between q_k and previous columns

$$e_k = \max_{1 \le i < k} |q_i^T q_k|, \quad k = 2, \ldots, n$$



loss of orthogonality is due to rounding error

Modified Gram-Schmidt

- G-S is numerically unstable if columns of A are almost linearly dependent
- this shortcoming can be alleviated by using q_k instead of a_k in the inner loop

---

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$
set $q_1 = a_1/\|a_1\|$ and $R_{11} = \|a_1\|$

for $k = 2, \ldots, n$

1. $q_k = a_k$
2. **for** $j = 1, \ldots, k-1$

   $R_{jk} = q_j^T q_k$ (reuse q_k instead of a_k)
   $q_k = q_k - R_{jk} q_j$
3. set

   $R_{kk} = \|q_k\|$
   $q_k = q_k / R_{kk}$

Modified Gram-Schmidt implementation

- rearrangement of the computation will provide an additional numerical advantage
- compute q_k then orthogonalize each of the remaining vectors against it
- generating R by rows rather than by columns

---

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$
set $Q = A$

for $k = 1, 2, \ldots, n$

1. set
$$R_{kk} = \|q_k\|$$
$$q_k = q_k / R_{kk}$$
2. **for** $j = k + 1, \ldots, n$
$$R_{kj} = q_k^T q_j$$
$$q_j = q_j - R_{kj} q_k$$

Outline

- Gram-Schmidt orthogonalization
- QR factorization
- solving linear equations
- modified Gram-Schmidt method
- **Householder algorithm**

Householder algorithm

- the most widely used algorithm for QR factorization (`qr` in MATLAB and Julia)
- less sensitive to rounding error than (modified) Gram-Schmidt algorithm
- computes a "full" QR factorization (QR decomposition)

$$A = \begin{bmatrix} Q & \tilde{Q} \end{bmatrix} \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad [Q \ \ \tilde{Q}] \text{ orthogonal}$$

- the full Q-factor is constructed as a product of orthogonal matrices

$$[Q \ \ \tilde{Q}] = H_1 H_2 \cdots H_n$$

each H_i is an $m \times m$ symmetric and orthogonal

Reflector

Reflector: an *elementary reflector* is a matrix of the form

$$H = I - 2vv^T \quad \text{with } v \text{ a unit-norm vector } \|v\| = 1$$

Properties

- a reflector matrix is symmetric
- a reflector matrix is orthogonal

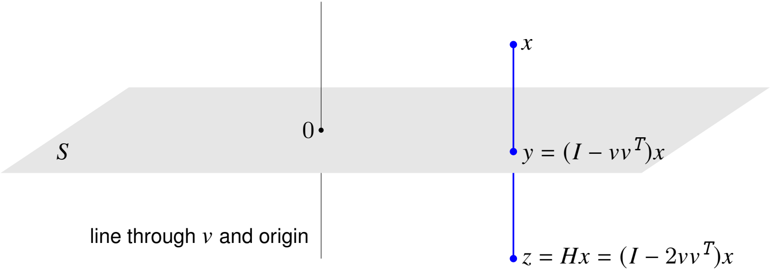
$$H^T H = (I - 2vv^T)(I - 2vv^T) = I - 4vv^T + 4vv^Tvv^T = I$$

- reflection of v: $Hv = -v$
- matrix-vector product Hx can be computed efficiently as

$$Hx = x - 2(v^Tx)v$$

  complexity is $4p$ flops if v and x have length p

Geometrical interpretation of reflector



- $S = \{u \mid v^T u = 0\}$ is the (hyper-)plane of vectors orthogonal to v
- if $\|v\| = 1$, the projection of x on S is given by (see page 8.20)

$$y = (I - vv^T)x$$

- reflection of x through the hyperplane is given by product with reflector:

$$z = y + (y - x) = (I - 2vv^T)x$$

Reflection to multiple of first unit vector

given nonzero p-vector $y = (y_1, y_2, \ldots, y_p)$, define

$$w = \begin{bmatrix} y_1 + \mathrm{sign}(y_1)\|y\| \\ y_2 \\ \vdots \\ y_p \end{bmatrix}, \quad v = \frac{1}{\|w\|} w$$

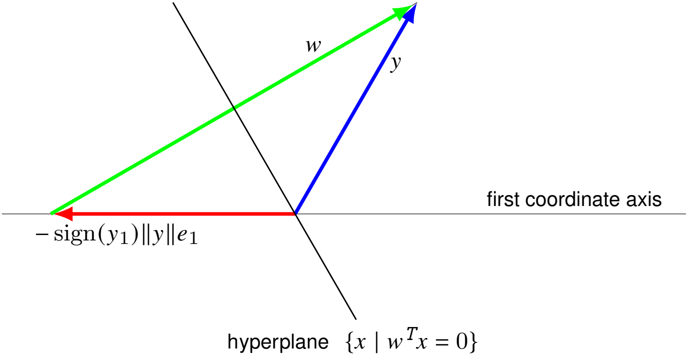
- we define $\mathrm{sign}(0) = 1$
- vector w satisfies

$$\|w\|^2 = 2(w^T y) = 2\|y\|(\|y\| + |y_1|)$$

- reflector $H = I - 2vv^T$ maps y to multiple of $e_1 = (1, 0, \ldots, 0)$:

$$Hy = y - \frac{2(w^T y)}{\|w\|^2} w = y - w = -\mathrm{sign}(y_1)\|y\| e_1$$

Geometry



the reflection through the hyperplane $\{x \mid w^T x = 0\}$ with normal vector

$$w = y + \operatorname{sign}(y_1)\|y\|e_1$$

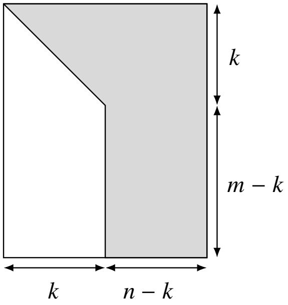
maps y to the vector $-\operatorname{sign}(y_1)\|y\|e_1$

Householder triangularization

- computes reflectors $H_1, \ldots, H_n$ that reduce A to triangular form:

$$H_n H_{n-1} \cdots H_1 A = \begin{bmatrix} R \\ 0 \end{bmatrix}$$

- after step k, the matrix $H_k H_{k-1} \cdots H_1 A$ has the following structure:



(elements in positions i, j for $i > j$ and $j \leq k$ are zero)

Householder algorithm

---

given: $m \times n$ matrix A with linearly independent columns $a_1, \ldots, a_n$

for $k = 1, 2, \ldots, n$

1. define $y = A_{k:m,k}$ and compute $(m - k + 1)$-vector v_k:

$$w = y + \operatorname{sign}(y_1)\, \|y\| e_1, \quad v_k = \frac{1}{\|w\|} w$$

2. multiply $A_{k:m,k:n}$ with reflector $I - 2v_k v_k^T$:

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

- algorithm overwrites A with $\begin{bmatrix} R \\ 0 \end{bmatrix}$
- **complexity:** $2mn^2 - \frac{2}{3}n^3$ flops (we take $2mn^2$ for the complexity of QR factorization)

Remarks

- step 2 is equivalent to multiplying A with $m \times m$ reflector

$$H_k = \begin{bmatrix} I & 0 \\ 0 & I - 2v_k v_k^T \end{bmatrix} = I - 2 \begin{bmatrix} 0 \\ v_k \end{bmatrix} \begin{bmatrix} 0 \\ v_k \end{bmatrix}^T$$

- algorithm returns the vectors $v_1, \ldots, v_n$, with v_k of length $m - k + 1$

Q-factor

$$\begin{bmatrix} Q & \tilde{Q} \end{bmatrix} = H_1 H_2 \cdots H_n$$

- usually there is no need to compute the matrix $[Q \ \ \tilde{Q}]$ explicitly
- the vectors $v_1, \ldots, v_n$ are an economical representation of $[Q \ \ \tilde{Q}]$
- products with $[Q \ \ \tilde{Q}]$ or its transpose can be computed as

$$\begin{bmatrix} Q & \tilde{Q} \end{bmatrix} x = H_1 H_2 \cdots H_n x$$

$$\begin{bmatrix} Q & \tilde{Q} \end{bmatrix}^T y = H_n H_{n-1} \cdots H_1 y$$

Example

$$A = \begin{bmatrix} -1 & -1 & 1 \\ 1 & 3 & 3 \\ -1 & -1 & 5 \\ 1 & 3 & 7 \end{bmatrix} = H_1 H_2 H_3 \begin{bmatrix} R \\ 0 \end{bmatrix}$$

we compute reflectors H_1, H_2, H_3 that triangularize A:

$$H_3 H_2 H_1 A = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ 0 & R_{22} & R_{23} \\ 0 & 0 & R_{33} \\ 0 & 0 & 0 \end{bmatrix}$$

First column of R

- compute reflector that maps first column of A to multiple of e_1:

$$y = \begin{bmatrix} -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad w = y - \|y\|e_1 = \begin{bmatrix} -3 \\ 1 \\ -1 \\ 1 \end{bmatrix}, \quad v_1 = \frac{1}{\|w\|}w = \frac{1}{2\sqrt{3}} \begin{bmatrix} -3 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

- overwrite A with product of $I - 2v_1v_1^T$ and A

$$A := (I - 2v_1v_1^T)A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & 4/3 & 8/3 \\ 0 & 2/3 & 16/3 \\ 0 & 4/3 & 20/3 \end{bmatrix}$$

Second column of R

- compute reflector that maps $A_{2:4,2}$ to multiple of e_1:

$$y = \begin{bmatrix} 4/3 \\ 2/3 \\ 4/3 \end{bmatrix}, \quad w = y + \|y\|e_1 = \begin{bmatrix} 10/3 \\ 2/3 \\ 4/3 \end{bmatrix}, \quad v_2 = \frac{1}{\|w\|}w = \frac{1}{\sqrt{30}} \begin{bmatrix} 5 \\ 1 \\ 2 \end{bmatrix}$$

- overwrite $A_{2:4,2:3}$ with product of $I - 2v_2v_2^T$ and $A_{2:4,2:3}$:

$$A := \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2v_2^T \end{bmatrix} A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & 16/5 \\ 0 & 0 & 12/5 \end{bmatrix}$$

Third column of R

- compute reflector that maps $A_{3:4,3}$ to multiple of e_1:

$$y = \begin{bmatrix} 16/5 \\ 12/5 \end{bmatrix}, \quad w = y + \|y\| e_1 = \begin{bmatrix} 36/5 \\ 12/5 \end{bmatrix}, \quad v_3 = \frac{1}{\|w\|} w = \frac{1}{\sqrt{10}} \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

- overwrite $A_{3:4,3}$ with product of $I - 2v_3 v_3^T$ and $A_{3:4,3}$:

$$A := \begin{bmatrix} I & 0 \\ 0 & I - 2v_3 v_3^T \end{bmatrix} A = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & -4 \\ 0 & 0 & 0 \end{bmatrix}$$

Final result

$$H_3H_2H_1A = \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2v_2^T \end{bmatrix} (I - 2v_1v_1^T)A$$

$$= \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & I - 2v_2v_2^T \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & 4/3 & 8/3 \\ 0 & 2/3 & 16/3 \\ 0 & 4/3 & 20/3 \end{bmatrix}$$

$$= \begin{bmatrix} I & 0 \\ 0 & I - 2v_3v_3^T \end{bmatrix} \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & 16/5 \\ 0 & 0 & 12/5 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 4 & 2 \\ 0 & -2 & -8 \\ 0 & 0 & -4 \\ 0 & 0 & 0 \end{bmatrix}$$

References and further readings

- S. Boyd and L. Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares,* Cambridge University Press, 2018.
- L. Vandenberghe. *EE133A lecture notes,* Univ. of California, Los Angeles. (`http://www.seas.ucla.edu/~vandenbe/ee133a.html`)
- U. M. Ascher. *A First Course on Numerical Methods*. Society for Industrial and Applied Mathematics, 2011.