Householder Reflections and Givens Rotations
Matrix Computations — CPSC 5006 E

Julien Dompierre

Department of Mathematics and Computer Science
Laurentian University

Sudbury, November 15, 2010

Outline

QR Decomposition with Householder Reflections and Givens Rotations

- Matrix of a Linear Transformation
- Householder Reflections
- Givens Rotations
- QR decomposition
- Section 5.1 and 5.2 of the textbook

Matrix Representation of a Linear Transformation

Theorem

Let $T : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation. Then there exists a unique matrix P called **standard matrix for the linear transformation**, *such that*

$$T(x) = Px \quad \text{for all } x \text{ in } \mathbb{R}^n.$$

In fact, P is the $m \times n$ matrix whose jth column is the vector $T(e_j)$, where e_j is the jth column of the identity matrix in $\mathbb{R}^n$.

$$P = \begin{bmatrix} T(e_1) & T(e_2) & \cdots & T(e_n) \end{bmatrix}.$$

All matrix transformation from $\mathbb{R}^n$ to $\mathbb{R}^m$ is a linear transformation and conversely, all linear transformation from $\mathbb{R}^n$ to $\mathbb{R}^m$ is a matrix transformation.

Proof

Proof.

Let $T : \mathbb{R}^n \to \mathbb{R}^m$ be a linear transformation, $\{e_1, e_2, \dots, e_n\}$ be a standard basis for $\mathbb{R}^n$, and u be an arbitrary vector in $\mathbb{R}^n$. Write the vectors as column vectors:

$$e_1 = \begin{bmatrix} 1 \\ \vdots \\ 0 \end{bmatrix}, \dots, e_n = \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix}, \text{and} \quad u = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix}.$$

Express u in terms of the basis

$$u = a_1 e_1 + \cdots + a_n e_n.$$

Continued next slide... $\square$

Proof (continued)

Proof.

Since T is a linear transformation

$$\begin{aligned} T(u) &= T(a_1 e_1 + \cdots + a_n e_n) \\ &= a_1 T(e_1) + \cdots + a_n T(e_n) \\ &= [T(e_1) \cdots T(e_n)] \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \end{aligned}$$

where $[T(e_1) \cdots T(e_n)]$ is a matrix with columns $T(e_1)$, ..., $T(e_n)$. Thus the linear transformation T is defined by the matrix

$$P = [T(e_1) \cdots T(e_n)].$$

$\square$

Linear Transformation of a Line is a Line

The segment through vectors p and q in $\mathbb{R}^n$ may be written in the parametric form

$$x = (1-t)p + tq, \quad 0 \leq t \leq 1.$$

If we apply the linear application $T : \mathbb{R}^n \to \mathbb{R}^m$ to this segment, we get

$$T(x) = T((1-t)p + tq) = (1-t)T(p) + tT(q), \quad 0 \leq t \leq 1,$$

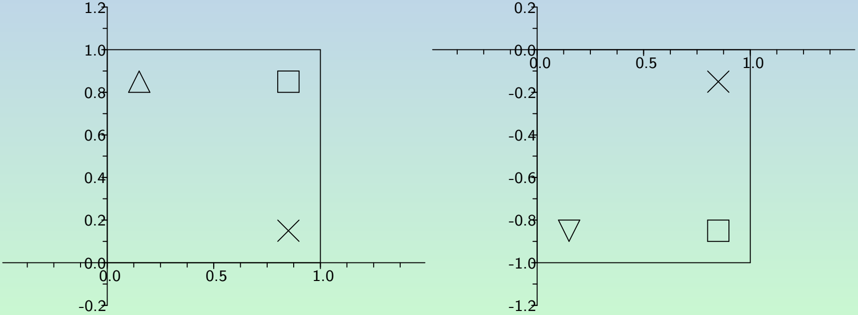
which is the equation of a segment through vectors $T(p)$ and $T(q)$ in $\mathbb{R}^m$

Reflection through the x_1-axis

The vector $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is transformed into $T(e_1) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

The vector $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is transformed into $T(e_2) = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$.

The standard matrix of the transformation is $P = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$.

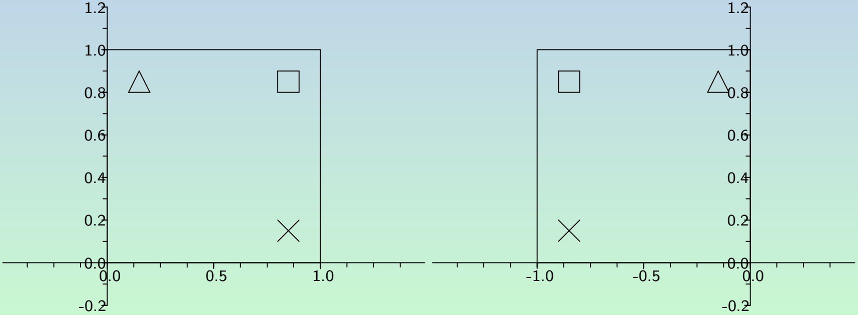


Reflection through the x_2-axis

The vector $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is transformed into $T(e_1) = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$.

The vector $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is transformed into $T(e_2) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

The standard matrix of the transformation is $P = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$.

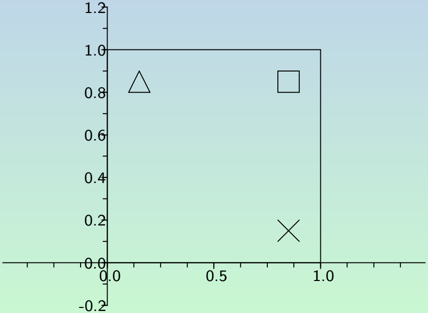


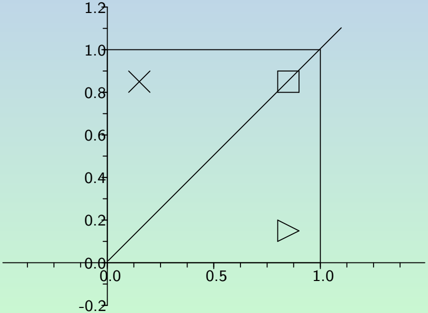
Reflection through the line $x_2 = x_1$

The vector $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is transformed into $T(e_1) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$.

The vector $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is transformed into $T(e_2) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$.

The standard matrix of the transformation is $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.



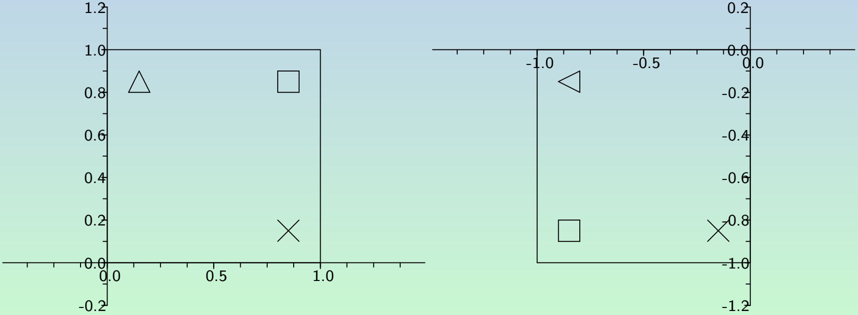


Reflection through the line $x_2 = -x_1$

The vector $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is transformed into $T(e_1) = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$.

The vector $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is transformed into $T(e_2) = \begin{bmatrix} -1 \\ 0 \end{bmatrix}$.

The standard matrix of the transformation is $P = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$.



Rotation Around the Origin O of an Angle θ

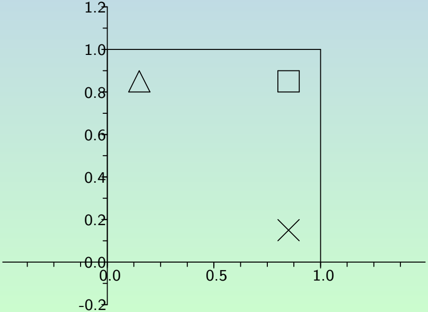
The vector $e_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is transformed into $T(e_1) = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$.

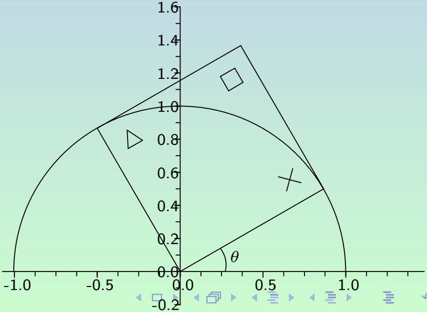
The vector $e_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is transformed into $T(e_2) = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$.

The standard matrix of the transformation is

$$P = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}.$$

In this case, $\theta = 30^\circ$.





Rotation in Three Dimensions

Counterclockwise rotation of angle θ around the x_3-axis:

$$P = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Counterclockwise rotation of angle ϕ around the x_1-axis:

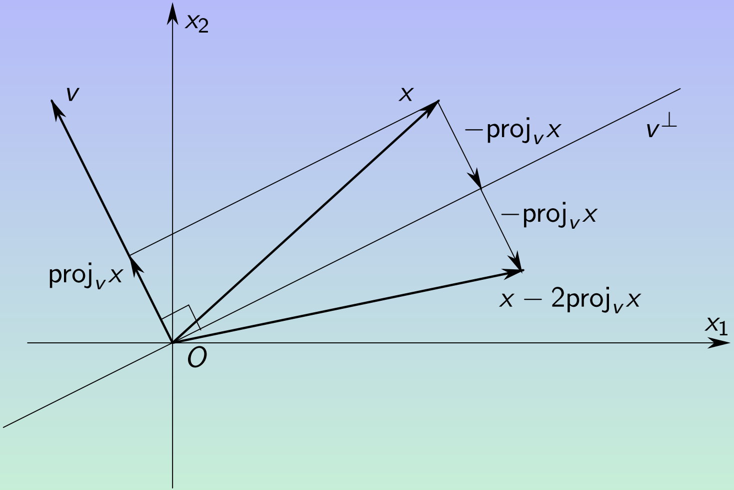
$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

Counterclockwise rotation of angle ψ around the x_2-axis:

$$P = \begin{bmatrix} \cos\psi & 0 & -\sin\psi \\ 0 & 1 & 0 \\ \sin\phi & 0 & \cos\phi \end{bmatrix}$$

Alston Scott Householder. Born: 5 May 1904 in Rockford, Illinois, USA. Died: 4 July 1993 in Malibu, California, USA.

Householder Reflection



$$\text{proj}_v x = \left(\frac{x \cdot v}{v \cdot v} \right) v.$$

Householder Reflection as a Matrix

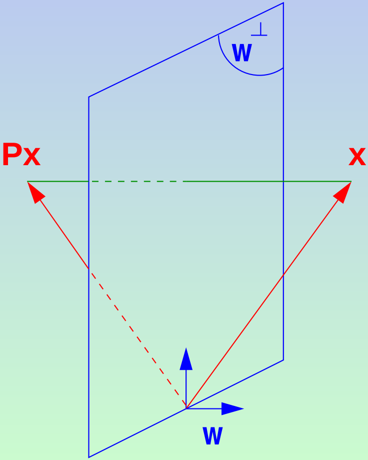
$$
\begin{aligned}
x - 2\mathrm{proj}_v x &= x - 2\left(\frac{x \cdot v}{v \cdot v}\right) v = x - 2\left(\frac{v \cdot x}{v \cdot v}\right) v \\
&= x - \frac{2}{v^T v}(v^T x) v = x - \frac{2}{v^T v} v (v^T x) \\
&= x - \frac{2}{v^T v}(v\, v^T) x = \underbrace{\left(I - \frac{2}{v^T v}(v\, v^T)\right)}_{P} x \\
&= \left(I - \frac{2}{\|v\|_2^2}(v\, v^T)\right) x = \left(I - 2\frac{v}{\|v\|_2}\frac{v^T}{\|v\|_2}\right) x \\
&= \underbrace{\left(I - 2w\, w^T\right)}_{P} x = Px
\end{aligned}
$$

where $w = v/\|v\|_2$ is a unit vector in 2-norm.

Householder reflectors are matrices of the form

$$P = I - 2w\,w^T,$$

where w is a unit vector (a vector of 2-norm unity).



Geometrically, Px represents a mirror image of x with respect to the hyperplane $\text{span}\{w\}^{\perp}$.

A Few Simple Properties

- P is symmetric, i.e. $P = P^T$.
- P is unitary. i.e. $PP^T = I$.
- In the complex case $P = I - 2w\,w^H$ is Hermitian $(P = P^H)$ and unitary $(PP^H = I)$.
- P can be written as $P = I - \beta v\,v^T$ with $\beta = 2/\|v\|_2^2$, where v is a multiple of w. So we don't need to store the matrix P, but only the vector v and the scalar β.

A Few Simple Properties (p. 211)

- Px (operation count?) can be evaluated $x - \beta(x^T v) \times v$ (operation count?)
- Similarly, if $A \in \mathbb{R}^{m \times n}$ and $P = I - \beta v\, v^T \in \mathbb{R}^{m \times m}$, then PA (operation count?) can be evaluated as

$$PA = (I - \beta v\, v^T)A = A - vz^T$$

  where $z^T = \beta v^T A$ (operation count?)
- Likewise, if $A \in \mathbb{R}^{m \times n}$ and $P = I - \beta v\, v^T \in \mathbb{R}^{n \times n}$, then AP (operation count?) can be evaluated as

$$AP = A(I - \beta v\, v^T) = A - zv^T$$

  where $z = \beta A v$ (operation count?)

Householder Vector

Problem 1: Given a vector $x \neq 0$, find w such that the Householder reflection will zero out all but the first entry of x, i.e.

$$Px = (I - 2w\,w^T)x = [\alpha, 0, 0, \cdots, 0] = \alpha e_1,$$

where α is a (free) scalar.
Writing $(I - \beta v\,v^T)x = \alpha e_1$ yields

$$\beta(v^T x)\ v = x - \alpha e_1 \quad \rightarrow \quad v = \frac{1}{\beta(v^T x)}(x - \alpha e_1)$$

Desired v is a multiple of $x - \alpha e_1$, i.e., we can take

$$v = x - \alpha e_1$$

To determine α we just recall that

$$\|Px\| = \|(I - 2w\,w^T)x\|_2 = \|x\|_2 = \|\alpha e_1\|.$$

As a result: $|\alpha| = \|x\|_2$, or

$$\alpha = \pm\|x\|_2$$

Vector Sign

Should verify that both signs work, i.e., that in both cases we indeed get $Px = \alpha e_1$ [exercise].

Which sign is best? To reduce cancellation, the resulting $x - \alpha e_1$ should not be small. So $\alpha = -\mathrm{sign}(x_1)\|x\|_2$, then

$$v = x - \alpha e_1 = x + \mathrm{sign}(x_1)\|x\|_2 e_1 \quad \text{and} \quad w = v/\|v\|_2.$$

$$v = \begin{bmatrix} \hat{x}_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} \quad \text{with} \quad \hat{x}_1 = \begin{cases} x_1 + \|x\|_2 & \text{if } x_1 > 0 \\ x_1 - \|x\|_2 & \text{if } x_1 \le 0 \end{cases}$$

OK, but will yield a negative multiple of e_1 if $x_1 > 0$.

Alternative (p. 210)

Define $\sigma = \sum_{i=2}^{n} x_i^2$ and when $x_1 > 0$ use $\alpha = \|x\|_2$:

$$\hat{x}_1 = x_1 - \|x\|_2 = \frac{x_1^2 - \|x\|_2^2}{x_1 + \|x\|_2} = \frac{-\sigma}{x_1 + \|x\|_2}$$

So:

$$\hat{x}_1 = \begin{cases} \frac{-\sigma}{x_1 + \|x\|_2} & \text{if } x_1 > 0 \\ x_1 - \|x\|_2 & \text{if } x_1 \leq 0 \end{cases}$$

It is customary to compute a vector v such that $v_1 = 1$. So v is scaled by its first component.

If σ is zero, procedure will return $v = [1; x(2:n)]$ and $\beta = 0$.

Algorithm 5.1.1 Householder Vector (p. 210)

Algorithm 1 Householder Vector. Given $x \in \mathbb{R}^n$, this function computes $v \in \mathbb{R}^n$ with $v(1) = 1$ and $\beta \in \mathbb{R}$ such that $P = I_n - \beta v v^T$ is orthogonal and $Px = \|x\|_2 e_1$.

function $[v, \beta] =$ **HouseholderVector**(x)
$n =$ **length**(x)
$\sigma = x(2:n)^T x(2:n), \quad v = \begin{bmatrix} 1 \\ x(2:n) \end{bmatrix}$
if $\sigma = 0$ **then**
    $\beta = 0$
else
    $\mu = \sqrt{x(1)^2 + \sigma}$
    if $x(1) \le 0$ **then**
        $v(1) = x(1) - \mu$
    else
        $v(1) = -\sigma/(x(1) + \mu)$
    end
    $\beta = 2v(1)^2/(\sigma + v(1)^2), \quad v = v/v(1)$
end

Problem 2: Given an $m \times n$ matrix A, find Householder vectors $w_1, w_2, \ldots, w_n$ such that

$$P_n \cdots P_2 P_1 A = (I - 2w_n w_n^T) \cdots (I - 2w_2 w_2^T)(I - 2w_1 w_1^T)A = R$$

where R is upper triangular ($r_{ij} = 0$ for $i > j$).

First step is easy: Select w_1 so that the first column of A becomes αe_1, i.e., $A_1 = P_1 A = (I - 2w_1 w_1^T)A$.

Second step: Select w_2 so that the second column of A has zeros below 2nd component, i.e.
$A_2 = P_2 A_1 = P_2 P_1 A = (I - 2w_2 w_2^T)(I - 2w_1 w_1^T)A$.

Etc... After $j-1$ steps: $A_{j-1} \equiv P_{j-1} \cdots P_1 A$ has the following shape:

Householder QR Factorization

$$A_{j-1} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & \cdots & \cdots & a_{1n} \\ & a_{22} & a_{23} & \cdots & \cdots & \cdots & a_{2n} \\ & & a_{33} & \cdots & \cdots & \cdots & a_{3n} \\ & & & \ddots & \cdots & \cdots & \vdots \\ & & & & a_{jj} & \cdots & a_{jn} \\ & & & & a_{j+1,j} & \cdots & a_{j+1,n} \\ & & & & \vdots & \vdots & \vdots \\ & & & & a_{m,j} & \cdots & a_{m,n} \end{bmatrix}.$$

To do: transform this matrix into one which is upper triangular up to the j-th column...

... while leaving the previous columns untouched.

Householder QR Factorization

To leave the first $j-1$ columns unchanged, w must have zeros in positions 1 through $j-1$.

$$P_j = I - 2w_j w_j^T, \quad w_j = \frac{v}{\|v\|_2},$$

where the vector v can be expressed as a Householder vector for a shorter vector using the function **HouseholderVector**,

$$v = \begin{bmatrix} 0 \\ \textbf{HouseholderVector}(A(j:m,k)) \end{bmatrix}$$

The result is that work is done on the $(j:m, j:n)$ submatrix.

Algorithm 5.2.1 Householder QR Factorization (p. 224)

Algorithm 2 NAIVE Householder QR Factorization Given $A \in R^{m \times n}$ with $m \geq n$, the following algorithm finds Householder matrices $P_1, P_2, ..., P_n$ such that if $Q = P_1 \cdots P_n$, then $Q^T A = R$ is upper triangular.

$R = A$
$Q = I_m$
for $j = 1 : 1 : \min(m-1, n)$
  $[v, \beta] = \textbf{HouseholderVector}(R(j:m, j))$
  $v = \begin{bmatrix} 0_{j-1} \\ v \end{bmatrix}$
  $P = I_m - \beta v\, v^T$
  $R = PR$
  $Q = QP$
end

Algorithm 5.2.1 Householder QR Factorization (p. 224)

Algorithm 3 EFFICIENT Householder QR Factorization Given $A \in R^{m \times n}$ with $m \geq n$, the following algorithm finds Householder matrices $P_1, P_2, ..., P_n$ such that if $Q = P_1 \cdots P_n$, then $Q^T A = R$ is upper triangular. The upper triangular part of A is overwritten by the upper triangular part of R and components $j+1 : m$ of the jth Householder vector are stored in $A(j+1 : m, j)$, $j < m$.

1: **for** $j = 1 : 1 : \min(m-1, n)$
2: $\quad [v, \beta] = \textbf{HouseholderVector}(A(j : m, j))$
3: $\quad A(j : m, j : n) = (I_{m-j+1} - \beta v\, v^T) A(j : m, j : n)$
4: $\quad$ **if** $j < m$ **then**
5: $\qquad A(j+1 : m, j) = v(2 : m-j+1)$
6: $\quad$ **end**
7: **end**

For line 3, see §5.1.5, page 211, for efficient multiplication of PA.
This algorithm requires $2n^2(m - n/3)$ flops.

To clarify how A (in $\mathbb{R}^{6\times 5}$ for example) is overwritten by the previous algorithm, if

$$v^{(j)} = [\underbrace{0, \ldots, 0}_{j-1}, 1, v_{j+1}^{(j)}, \ldots, v_m^{(j)}]^T$$

is the jth Householder vector, then upon completion

$$A = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} & r_{15} \\ v_2^{(1)} & r_{22} & r_{23} & r_{24} & r_{25} \\ v_3^{(1)} & v_3^{(2)} & r_{33} & r_{34} & r_{35} \\ v_4^{(1)} & v_4^{(2)} & v_4^{(3)} & r_{44} & r_{45} \\ v_5^{(1)} & v_5^{(2)} & v_5^{(3)} & v_5^{(4)} & r_{55} \\ v_6^{(1)} & v_6^{(2)} & v_6^{(3)} & v_6^{(4)} & v_6^{(5)} \end{bmatrix}.$$

We need an extra vector of size n to store the β's. If the matrix $Q = P_1 \cdots P_n$ is required, then it can be accumulated using Eqs (5.1.5) page 213.

Householder QR versus Gram-Schmidt QR

The Householder QR yields the factorization

$$A = QR$$

where

$$Q = P_1 P_2 \dots P_m \in \mathbb{R}^{m \times m}$$

and

$$R = Q^T A = P_m \cdots P_1 A = A_m \in \mathbb{R}^{m \times n}.$$

The matrix R has zeros below the m-th row. Note also that this factorization always exists.

There is a **major** difference with Gram-Schmidt QR: Q is $n \times n$ and R is $m \times n$ (same as A).

Question: From the Householder QR, how to obtain $A = Q_1 R_1$ where Q_1 as the same size as A and R_1 is $n \times n$, as in (modified) Gram-Schmidt QR?

Economy Size QR

Answer: simply use the partitioning

$$A = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} \quad \to \quad A = Q_1 R_1$$

Referred to as the "thin" QR factorization (or "economy-size QR" factorization in Scilab)

How to solve a least-squares problem $A^T A x = A^T b$ using the Householder factorization $A = QR$?

Answer: Solve $R_1 x_{LS} = Q_1^T b$, the right-hand side $Q_1^T b$ can be computed without assembling Q_1. Just apply Q_1^T to b. This entails applying the successive Householder reflections to b. (See algorithm 5.3.2, p. 240)

The Rank-Deficient Case

Householder QR gives Q_1 and R_1 such that $Q_1 R_1 = A$. However, in the rank-deficient case, Q_1 and A do not necessarily span the same space because R_1 may be singular.

Remedy: Householder QR with column pivoting. Result will be:

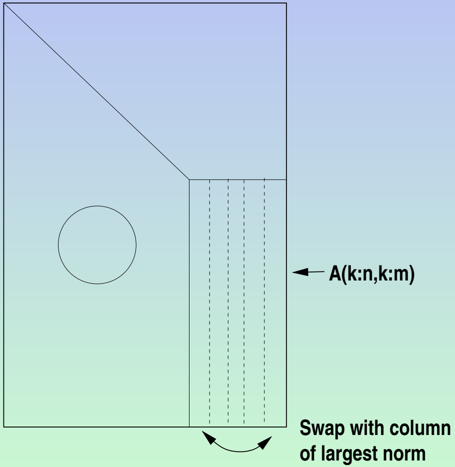
$$A\Pi \;=\; Q \begin{bmatrix} R_{11} & R_{12} \\ 0 & 0 \end{bmatrix}$$

R_{11} is non singular. So $\text{rank}(A) = \text{size of } R_{11} = \text{rank}(Q_1)$ and Q_1 and A span the same subspace.

Π permutes columns of A.

Householder with Pivoting

Algorithm: At step k, active matrix is $A(k:n, k:m)$. Swap k-th column with column of largest 2-norm in $A(k:n, k:m)$. If all the columns have zero norm, stop.



Practical Question: How to implement this (see section 5.4.1, p. 248)

Properties of the QR Factorization (p. 229)

Consider the "thin" factorization $A = QR$, (size(Q) = [n,m] = size (A)). Assume $r_{ii} > 0$, $i = 1, \ldots, m$

1. When A is of full column rank this factorization exists and is unique
2. It satisfies:

$$\text{span}\{a_1, \cdots, a_k\} = \text{span}\{q_1, \cdots, q_k\}, \quad k = 1, \ldots, m$$

3. R is identical with the Cholesky factor G^T of $A^T A$.

When A in rank-deficient and Householder with pivoting is used, then

$$\text{range}\{Q_1\} = \text{range}\{A\}$$

James Wallace Givens, Jr

James Wallace Givens, Jr. (1910 December 14 — 1993 March 5) was a mathematician and a pioneer in computer science.

Givens, Wallace. "Computation of plane unitary rotations transforming a general matrix to triangular form". J. SIAM 6(1) (1958), pp. 26 50.

From 1968 to 1970 he was fourteenth president of the Society for Industrial and Applied Mathematics.

Givens Rotations (p. 215)

Matrices of the form

$$G(i,k,\theta) = \begin{array}{c} \left[\begin{array}{ccccccc} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{array} \right] \end{array} \begin{array}{c} \\ \\ i \\ \\ k \\ \\ \\ \end{array}$$

where $c = \cos\theta$ and $s = \sin\theta$ for some θ.

Premultiplication by $G(i,k,\theta)^T$ amounts to a counterclockwise rotation of θ radians in the (i,k) coordinate plane. Givens rotations are orthogonal.

Givens Rotations

Main idea of Givens rotations: Consider $y = G(i,k,\theta)^T x$ then

$$\begin{aligned} y_i &= c\, x_i - s\, x_k \\ y_k &= s\, x_i + c\, x_k \\ y_j &= x_j \quad \text{for} \quad j \neq i, k \end{aligned}$$

Can make $y_k = 0$ by selecting

$$c = \frac{x_i}{t}; \quad s = \frac{-x_k}{t}; \quad t = \sqrt{x_i^2 + x_k^2}$$

Givens Rotations Algorithm 5.1.3 (p. 216)

Algorithm 4 Givens Rotation. Given scalar a and b (x_i and x_k in the previous slide), this function computes $c = \cos(\theta)$ and $s = \sin(\theta)$ such that

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix}.$$

function $[c, s] =$ **GivensRotation**(a, b)
if $b = 0$ **then**
    $c = 1$ and $s = 0$
else
    if $|b| > |a|$ **then**
        $\tau = -a/b$, $s = 1/\sqrt{1+\tau^2}$ and $c = s\tau$
    else
        $\tau = -b/a$, $c = 1/\sqrt{1+\tau^2}$ and $s = c\tau$
    end
end

This algorithm requires 5 flops, 1 square root and no trig functions.

Givens QR Method (p. 226)

Let A be a 4×3 matrix. Then

$$A = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \overset{(3,4)}{\longrightarrow} \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \end{bmatrix} \overset{(2,3)}{\longrightarrow} \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \overset{(1,2)}{\longrightarrow}$$

$$\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \overset{(3,4)}{\longrightarrow} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \end{bmatrix} \overset{(2,3)}{\longrightarrow} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & \times \end{bmatrix} \overset{(3,4)}{\longrightarrow}$$

$$\begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{bmatrix} = R$$

Givens QR Method (p. 226)

If t is the total number of rotations needed to transform A into an upper triangular matrix R, then

$$\begin{aligned} G_t^T \cdots G_2^T G_1^T A &= R \\ (G_1 G_2 \cdots G_t)^T A &= R \\ Q^T A &= R \end{aligned}$$

where $Q^T = (G_1 G_2 \cdots G_t)^T$, i.e. $Q = G_1 G_2 \cdots G_t$.

Algorithm 5 NAIVE Givens QR Factorization. Given $A \in \mathbb{R}^{m\times n}$ with $m \geq n$, the following algorithm overwrites A with $Q^T A = R$, where R is upper triangular and Q is orthogonal.

```
Q = I
R = A
for j = 1 : n
    for i = m : -1 : j+1
        [c, s] = GivensRotation(A(i-1, j), A(i, j))
        Create G(i-1, i, θ)
        Q = Q G(i-1, i, θ)
        R = G^T(i-1, i, θ) R
    end
end
```

Applying Givens Rotations (p. 216)

If $G(i,k,\theta) \in \mathbb{R}^{m\times m}$, then the update $A \leftarrow G(i,k,\theta)^T A$ effects just two rows of A,

$$A([i,k],:) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T A([i,k],:)$$

and requires just $6n$ flops:

$$\begin{array}{l}
\textbf{for } j = 1:n \\
\quad \tau_1 = A(i,j) \\
\quad \tau_2 = A(k,j) \\
\quad A(i,j) = c\tau_1 - s\tau_2 \\
\quad A(k,j) = s\tau_1 + c\tau_2 \\
\textbf{end}
\end{array}$$

Applying Givens Rotations (p. 216–217)

If $G(i,k,\theta) \in \mathbb{R}^{n\times n}$, then the update $Q \leftarrow QG(i,k,\theta)$ effects just two columns of Q,

$$Q(:,[i,k]) = Q(:,[i,k]) \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

and requires just $6m$ flops:

for $j = 1:m$
  $\tau_1 = Q(j,i)$
  $\tau_2 = Q(j,k)$
  $Q(j,i) = c\tau_1 - s\tau_2$
  $Q(j,k) = s\tau_1 + c\tau_2$
end

Givens QR Factorization — Algorithm 5.2.2 (p. 227)

Algorithm 6 Givens QR Factorization. Given $A \in \mathbb{R}^{m\times n}$ with $m \geq n$, the following algorithm overwrites A with $Q^T A = R$, where R is upper triangular and Q is orthogonal.

for $j = 1 : n$
    for $i = m : -1 : j+1$
        $[c, s] =$ **GivensRotation**$(A(i-1, j), A(i, j))$
        $A(i-1 : i, j : n) = \begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T A(i-1 : i, j : n)$
    end
end

This algorithm requires $3n^2(m - n/3)$ flops.

If

$$Z = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \quad \text{with } c^2 + s^2 = 1,$$

then we define the scalar ρ by

if $c = 0$ **then**
  $\rho = 1$
else if $|s| < |c|$ **then**
  $\rho = \text{sign}(c)s/2$
else
  $\rho = 2\text{sign}(s)/c$
end

Essentially, this amounts to storing $s/2$ if the sin is smaller and $2/c$ if the cosine is smaller.

With this encoding, it is possible to reconstruct $\pm Z$ as follows:

if $\rho = 1$ **then**
  $c = 0$
  $s = 1$
else if $|\rho| < 1$ **then**
  $s = 2\rho$
  $c = \sqrt{1 - s^2}$
else
  $c = 2/\rho$
  $s = \sqrt{1 - c^2}$
end