QR Decomposition: Projection, Reflection, Rotation

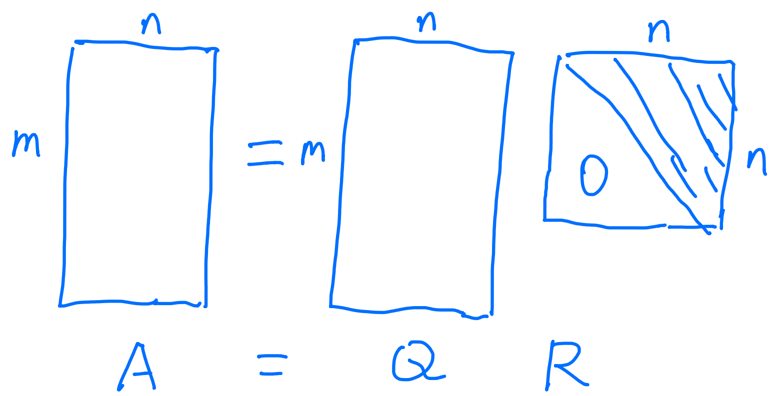
QR decomposition: Let $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$. Assume that A has **full column rank**. Then there exists a decomposition

$$A = QR,$$

where

$Q \in \mathbb{R}^{m\times n}$ is **orthogonal**, i.e., $Q^TQ = I$, and

$R \in \mathbb{R}^{n\times n}$ is **upper triangular** with non-zero diagonals



Remark:

- The diagonals of R has to be non-zero, because otherwise $\det(R) = r_{11}r_{22}\cdots r_{nn} = 0$, (Q: Prove the determinate of a triangular matrix is the product of diagonals) which implies $\operatorname{rank}(R) < n$.

  However, $n = \operatorname{rank}(A) \leq \min\{\operatorname{rank}(Q), \operatorname{rank}(R)\} < n$. Contradiction.
  (Q: prove $\operatorname{rank}(AB) \leq \min\{\operatorname{rank}(A), \operatorname{rank}(B)\}$)

- The QR decomposition is unique when the diagonals of R are positive.

- We can use QR decomposition to solve $Ax = b$ with $A \in \mathbb{R}^{n\times n}$ nonsingular. Because $\operatorname{rank}(A) = n$, let $A = QR$ be the QR decomposition of A. So,

$$Ax = b \iff QRx = b \underset{\substack{\uparrow \\ Q^TQ = I}}{\iff} Rx = Q^Tb$$

Solved by back substitution because R is upper triangular.

Geometry of QR decomposition

Write $A = [a_1, a_2, \cdots, a_n]$ and $Q = [q_1, q_2, \cdots, q_n]$. Then

$$\mathrm{Ran}(A) = \{Ax \mid x \in \mathbb{R}^n\} = \{QRx \mid x \in \mathbb{R}^n\} = \{Qy \mid y \in \mathbb{R}^n\}$$

$$= \mathrm{Ran}(Q)$$

because A is of full col rank and so R is invertible.

Recall that

$$\mathrm{Ran}(A) = \{\text{linear combinations of columns of } A\}$$

$$= \mathrm{span}\{a_1, a_2, \cdots, a_n\}$$

(means linear combinations of the vectors $a_1, \cdots, a_n$)

Therefore,

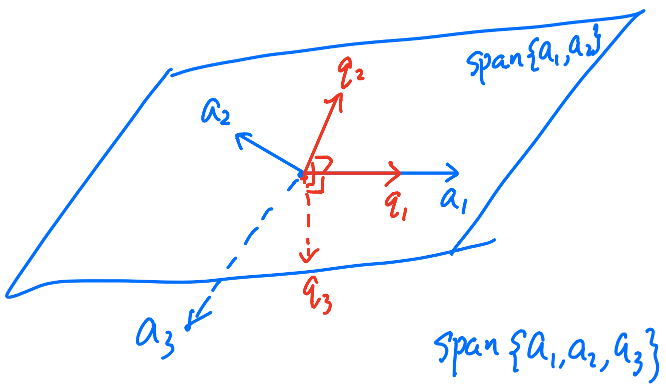
$$\mathrm{span}\{a_1, a_2, \cdots, a_n\} = \mathrm{span}\{q_1, q_2, \cdots, q_n\}$$

$\boxed{\text{QR decomposition finds an orthonormal basis of } \mathrm{span}\{a_1, \cdots, a_n\}}$ $\boxed{\mathrm{Ran}(A)}$

Furthermore, because $R \in \mathbb{R}^{n\times n}$ is upper triangular, we have

$$\begin{cases} a_1 = r_{11} q_1 \\ a_2 = r_{21} q_1 + r_{22} q_2 \\ \quad\vdots \\ a_k = r_{k1} q_1 + r_{k2} q_2 + \cdots + r_{kk} q_k \\ \quad\vdots \end{cases}$$

$$\boxed{\text{So, } \quad \mathrm{span}\{a_1, a_2, \cdots, a_k\} = \mathrm{span}\{q_1, q_2, \cdots, q_k\}. \quad \forall k = 1, 2, \cdots, n.}$$



* Algorithms

Gram-Schmidt Procedure

Find the Q matrix column by column

and R matrix column by column

Step 1: We rewrite $Q = [Q(1:m,1) \;\; Q(1:m,2:n)]$

and $R = \begin{bmatrix} R(1,1) & R(1,2:n) \\ 0 & R(2:n,2:n) \end{bmatrix}$ — to be computed.

Therefore $QR = [Q(1:m,1)R(1,1) \quad * \;]$

Compare it with $A = [A(1:m,1) \quad * \;]$

We have $Q(1:m,1)\,R(1,1) = A(1:m,1) \qquad (S_1)$

Taking 2-norms on both sides of (S_1)

$$\left.\begin{array}{l} |R(1,1)|\;\|Q(1:m,1)\|_2 = \|A(1:m,1)\|_2 \\ \text{Since } Q \text{ is orthogonal, } \|Q(1:m,1)\|_2 = 1 \end{array}\right\} \Rightarrow |R(1,1)| = \|A(1:m,1)\|_2$$

Since we have assumed diagonals of R are positive

$$\Longrightarrow \boxed{R(1,1) = \|A(1:m,1)\|_2}$$

Also, $(S1) \Longrightarrow \boxed{Q(1:m,1) = A(1:m,1)/R(1,1)}$

$\vdots$

Step k: Assume $Q(1:m,1:k-1)$ and $R(1:k-1,1:k-1)$ are computed from previous steps.

Then we partition

$$Q = [Q(1:m,1:k-1) \;\; Q(1:m,k) \;\; *]$$ — to be computed

$$R = \begin{bmatrix} R(1:k-1,1:k-1) & R(1:k-1,k) & * \\ 0 & R(k,k) & * \\ 0 & 0 & * \end{bmatrix}$$ — computed

So that

$$QR = [* \quad Q(1:m,1:k-1)R(1:k-1,k) + Q(1:m,k)R(k,k) \quad *]$$

Compare it with

$$A = [* \quad A(1:m,k) \quad *]$$

It holds that

$$A(1:m,k) = Q(1:m,1:k-1)R(1:k-1,k) + Q(1:m,k)R(k,k) \quad \cdots\cdots$$

Left multiply both sides by $(Q(1:m,1:k-1))^T$

$$(Q(1:m,1:k-1))^T A(1:m,k) = \underbrace{(Q(1:m,1:k-1))^T Q(1:m,1:k-1)}_{= I \in \mathbb{R}^{(k-1)\times(k-1)}} R(1:k-1,k) + \underbrace{(Q(1:m,1:k-1))^T Q(1:m,k)}_{= 0 \in \mathbb{R}^{(k-1)\times 1}} R(k,k)$$

$$\Rightarrow \boxed{R(1:k-1,k) = (Q(1:m,1:k-1))^T A(1:m,k)}$$

Also, $\|Q(1:m,k)\|_2 = 1$. We assumed $R(k,k) \geq 0$.

Then $R(k,k) = \| A(1:m,k) - Q(1:m,1:k-1)R(1:k-1,k)\|_2$

and $Q(1:m,k) = \big(A(1:m,k) - Q(1:m,1:k-1)R(1:k-1,k)\big)/R(k,k)$

We obtain k-th columns of Q and R respectively.

⋮

The full algorithm of Gram-Schmidt procedure for QR decomposition

```
for k = 1:n
    R(1:k-1,k) = (Q(1:m,1:k-1))^T A(1:m,k)
    Q(1:m,k) = A(1:m,k) - Q(1:m,1:k-1) R(1:k-1,k)
    R(k,k) = ||Q(1:m,k)||_2
    Q(1:m,k) = Q(1:m,k)/R(k,k)
end
```

Geometry of Gram-Schmidt.

If we write

$$A = [a_1 \ a_2 \ \cdots \ a_n] \qquad Q = [q_1 \ q_2 \ \cdots \ q_n]$$

and

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ & r_{22} & r_{23} & \cdots & r_{2n} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & \vdots \\ & & & & r_{nn} \end{bmatrix}$$

where $a_i \in \mathbb{R}^n$, $q_i \in \mathbb{R}^n$ are i-th column of A and Q respectively.

Then the Gram-Schmidt can be rewritten as

for $k = 1:n$

$r_{ik} = q_i^T a_k, \quad i = 1, 2, \cdots, k-1$ ①

$q_k = a_k - \sum_{i=1}^{k-1} r_{ik} q_i$ ②

$r_{kk} = \|q_k\|_2$ ③

$q_k = q_k / r_{kk}$ ④

end.

① ② projection and obtain the residual

③ ④ normalization.

We have

$$q_k \underset{③}{=} \frac{q_k}{r_{kk}} \underset{②}{=} \frac{a_k - \sum_{i=1}^{k-1} r_{ik} q_i}{r_{kk}} \underset{①}{=} \frac{a_k - \sum_{i=1}^{k} (q_i^T a_k) q_i}{r_{kk}}$$

$$\Rightarrow \quad r_{kk} q_k = a_k - \sum_{i=1}^{k-1} (q_i^T a_k) q_i = a_k - \left(\sum_{i=1}^{k-1} q_i q_i^T \right) a_k$$

$$= a_k - [q_1 \ q_2 \cdots q_{k-1}] \begin{bmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_{k-1}^T \end{bmatrix} a_k$$

Define $\quad Q_{k-1} = [q_1 \ q_2 \ \cdots \ q_{k-1}]$.

Then the Gram-Schmidt gives r_{kk} and q_k satisfying

$$r_{kk} q_k = a_k - Q_{k-1} Q_{k-1}^T a_k$$

Let

$$S_{k-1} \equiv \text{span}\{a_1, a_2, \cdots, a_{k-1}\}$$

Then

$$S_{k-1} = \text{span}\{r_{11}q_1,\ r_{21}q_1 + r_{22}q_2,\ \cdots,\ \sum_{j=1}^{k-1} r_{k-1,j}\, q_j\}$$

$$= \text{span}\{q_1, q_2, \cdots, q_{k-1}\} = \text{Ran}(Q_{k-1})$$

Theorem: Let $v_k = Q_{k-1}Q_{k-1}^T a_k$. Then

$$v_k = \underset{v \in S_{k-1}}{\arg\min} \|a_k - v\|_2^2$$

Call v_k the orthogonal projection of a_k on S_{k-1}.

Proof.

$$v_k = \underset{v \in S_{k-1}}{\arg\min} \|a_k - v\|_2^2$$

$$\Updownarrow$$

$$\|a_k - (v_k + w)\|_2^2 \geq \|a_k - v_k\|_2^2 \quad \forall w \in S_{k-1} \qquad \left(\text{Because } v_k + w \in S_{k-1}\right)$$

$$\Updownarrow$$

$$\|(a_k - v_k) - w\|_2^2 = \|a_k - v_k\|_2^2 - 2\langle a_k - v_k, w\rangle + \|w\|_2^2 \geq \|a_k - v_k\|_2^2 \quad \forall w \in S_{k-1}$$

$$\Updownarrow$$

$$\langle a_k - v_k, w\rangle \leq \frac{\|w\|_2^2}{2} \qquad \forall w \in S_{k-1}$$

Now we prove

$$\langle a_k - v_k, w\rangle \leq \frac{\|w\|_2^2}{2} \ \forall w \in S_{k-1} \iff \langle a_k - v_k, w\rangle = 0 \ \forall w \in S_{k-1}.$$

- "$\Rightarrow$". Assume $\langle a_k - v_k, w\rangle \leq \frac{\|w\|_2^2}{2}$ $\forall w \in S_{k-1}$.

  For any $w \in S_{k-1}$, Since $tw \in S_{k-1}$ $\forall t \in \mathbb{R}$,

  $$\langle a_k - v_k, tw\rangle \leq \frac{\|tw\|_2^2}{2} = \frac{t^2\|w\|_2^2}{2}$$

  Choose $t > 0$, $\langle a_k - v_k, w\rangle \leq \frac{t\|w\|_2^2}{2} \Rightarrow \langle a_k - v_k, w\rangle \leq 0$ (as $t \to 0_+$)

  Choose $t < 0$, $\langle a_k - v_k, w\rangle \geq \frac{t\|w\|_2^2}{2} \Rightarrow \langle a_k - v_k, w\rangle \geq 0$ (as $t \to 0_-$)

  Therefore, $\langle a_k - v_k, w\rangle = 0$ $\forall w \in S_{k-1}$

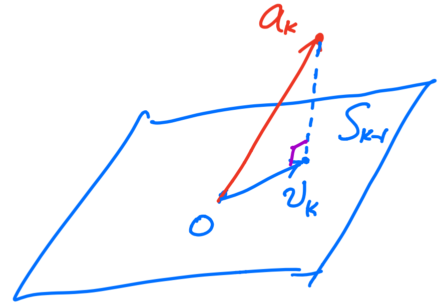
- "$\Leftarrow$". Assume $\langle a_k - v_k, w\rangle = 0$ $\forall w \in S_{k-1}$.

  Then $\frac{\|w\|_2^2}{2} \geq 0 = \langle a_k - v_k, w\rangle$ $\forall w \in S_{k-1}$.

Finally, $v_k = \arg\min_{v \in S_{k-1}} \|a_k - v\|_2^2$

$$\Updownarrow$$

$$\langle a_k - v_k, w \rangle = 0 \quad \forall w \in S_{k-1}$$



Let's check $v_k = Q_{k-1}Q_{k-1}^T a_k$

indeed satisfies $\langle a_k - v_k, w\rangle = 0, \forall w \in S_{k-1}$, as in the following:

$\forall w \in S_{k-1} = \text{Ran}(Q_{k-1})$, $\exists c$ s.t. $w = Q_{k-1}c$, and

$$\langle a_k - Q_{k-1}Q_{k-1}^T a_k, w\rangle = \langle a_k, w\rangle - \langle a_k, Q_{k-1}Q_{k-1}^T w\rangle$$

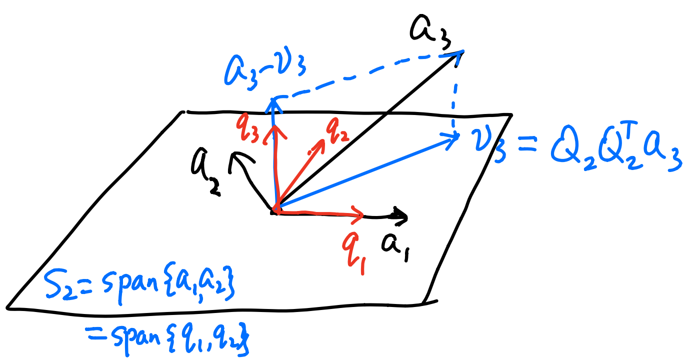
$$= \langle a_k, w\rangle - \langle a_k, Q_{k-1}Q_{k-1}^T Q_{k-1} c\rangle$$

$$= \langle a_k, w\rangle - \langle a_k, Q_{k-1}c\rangle = \langle a_k, w\rangle - \langle a_k, w\rangle = 0 \quad \boxtimes$$

Therefore, at step k of Gram-Schmidt, we do the following.

(i) Project a_k onto $\text{span}\{q_1, \cdots, q_{k-1}\}$ orthogonally. (Line 1 and 2)

(ii) Normalize the residual to get q_k. (Line 3 and 4)

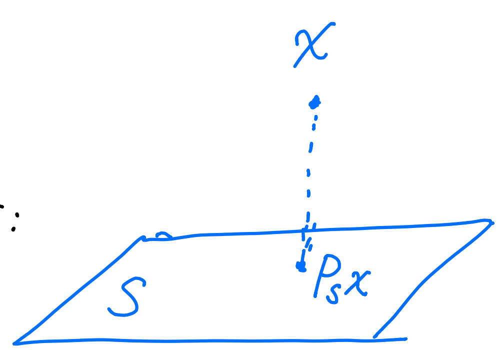


General Theory: Orthogonal Projector

Let $S \subseteq \mathbb{R}^n$ be a subspace of $\mathbb{R}^n$.

The orthogonal projection of $x \in \mathbb{R}^n$ onto S is:

$$P_S x,$$



where $P_S = UU^T \in \mathbb{R}^{n\times n}$ is the orthogonal projector with $U=[u_1,\cdots,u_r]$ an orthonormal basis of S.

Properties of orthogonal projection:

① $\langle x-P_S x, s\rangle = 0 \quad \forall x\in\mathbb{R}^n, \ s\in S$

② $P_S x = \arg\min_{s\in S} \|x-s\|_2^2$

③ $I-P_S$ is the orthogonal projector on $S^\perp$.

④ $P_S^2 = P_S$ and $P_S^T = P_S$

Standard GS is NOT numerically stable.

Example:

$$a_1 = \begin{pmatrix} 1 \\ \varepsilon \\ 0 \\ 0 \end{pmatrix} \qquad a_2 = \begin{pmatrix} 1 \\ 0 \\ \varepsilon \\ 0 \end{pmatrix} \qquad a_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \varepsilon \end{pmatrix}$$

Classical GS:

$q_1 \leftarrow (1,\varepsilon,0,0)^T, \quad r_{11} = \sqrt{1+\varepsilon^2} \approx 1, \quad q_1 \leftarrow \frac{q_1}{1} = (1,\varepsilon,0,0)^T$

$q_2 \leftarrow (1,0,\varepsilon,0)^T, \quad r_{12} = q_1^T a_2 = 1, \quad q_2 \leftarrow q_2 - 1\cdot q_1 = (0,-\varepsilon,\varepsilon,0)^T,$

$r_{22} = \sqrt{2}\varepsilon, \quad q_2 \leftarrow \frac{q_2}{r_{22}} = (0,-1,1,0)^T/\sqrt{2}$

$q_3 \leftarrow (1,0,0,\varepsilon)^T, \quad r_{13} = q_1^T a_3 = 1, \quad q_3 \leftarrow q_3 - 1\cdot q_1 = (0,-\varepsilon,0,\varepsilon)^T$

$r_{23} = q_2^T a_3 = 0, \quad q_3 \leftarrow q_3 - 0\cdot q_2 = (0,-\varepsilon,0,\varepsilon)^T$

$r_{33} = \sqrt{2}\varepsilon, \quad q_3 \leftarrow q_3/r_{33} = (0,-1,0,1)^T/\sqrt{2}$

Check the orthogonality:

$q_1^T q_2 = -\varepsilon/\sqrt{2}, \quad q_2^T q_3 = 1/2$ too large.

q_2 and q_3 are not orthogonal at all due to the round-off error in calculation.

Modified G-S

At step k,

$$r_{kk} q_k = a_k - Q_{k-1} Q_{k-1}^T a_k$$

$$= a_k - \sum_{i=1}^{k-1} (q_i^T a_k) q_j \qquad r_{ik}$$

Compute only $q_j^T a_k$. The error in q_i, $i=1,\cdots,k-1$, will pass to q_k, resulting large error in $q_i^T q_k$.

In order to cancel the error, we use the fact $q_i^T q_j = \begin{cases} 0 & i \neq j \\ 1, & i=j \end{cases}$ in the computation of $a_k - \sum_{i=1}^{k-1} q_i^T a_k q_i$.

Notice that

$$r_{1k} = q_1^T a_k$$

$$r_{2k} = q_2^T a_k = q_2^T (a_k - q_1^T a_k q_1) \qquad (\text{Because } q_2^T q_1 = 0)$$

$$r_{3k} = q_3^T a_k = q_3^T (a_k - q_1^T a_k q_1 - q_2^T a_k q_2) \qquad (\text{Because } q_3^T q_2 = q_3^T q_1 = 0)$$

$$\vdots$$

$$r_{k-1,k} = q_{k-1}^T a_k = q_{k-1}^T (a_k - q_1^T a_k q_1 - \cdots - q_{k-2}^T a_k q_{k-2}) \qquad (\text{Becaus } q_{k-1}^T q_i = 0 \ \forall i-1<k)$$

So we use the following

```
for k = 1:n
    q_k = a_k
    for i = 1:k-1
        r_ik = q_i^T q_k
        q_k = q_k - r_ik q_i
    end
    r_kk = ||q_k||_2
    q_k = q_k / r_kk
end
```

modified G-S

For the same example,

Modified GS:

$$q_1 \leftarrow (1, \varepsilon, 0, 0)^T, \quad r_{11} = \sqrt{1+\varepsilon^2} \approx 1, \quad q_1 \leftarrow \frac{q_1}{1} = (1, \varepsilon, 0, 0)^T$$

$$q_2 \leftarrow (1, 0, \varepsilon, 0)^T, \quad r_{12} = q_1^T q_2 = 1, \quad q_2 \leftarrow q_2 - 1 \cdot q_1 = (0, -\varepsilon, \varepsilon, 0)^T$$

$$r_{22} = \sqrt{2}\,\varepsilon, \quad q_2 \leftarrow \frac{q_2}{r_{22}} = (0, -1, 1, 0)^T/\sqrt{2}$$

$$q_3 \leftarrow (1, 0, 0, \varepsilon)^T, \quad r_{13} = q_1^T q_3 = 1, \quad q_3 \leftarrow q_3 - 1 \cdot q_1 = (0, -\varepsilon, 0, \varepsilon)^T$$

$$r_{23} = q_2^T q_3 = \varepsilon/\sqrt{2}, \quad q_3 \leftarrow q_3 - \frac{\varepsilon}{\sqrt{2}} q_2 = \left(0, -\frac{\varepsilon}{2}, -\frac{\varepsilon}{2}, \varepsilon\right)^T$$

$$r_{33} = \sqrt{6}\,\varepsilon/2, \quad q_3 \leftarrow q_3/r_{33} = (0, -1, -1, 2)^T/\sqrt{6}$$

We can check that

$$q_1^T q_2 = -\varepsilon/\sqrt{2}, \quad q_2^T q_3 = 0 \quad \text{(Much better than classical GS)}$$

Conclusion:

The classical GS and modified GS are theoretically equivalent, but numerically not equivalent.

Operation counting:

$$\sum_{k=1}^{n} \left(\sum_{i=1}^{k-1} (2m + 2m) + 2m + m \right) = \left(\sum_{k=1}^{m} 4m(k-1) + 3m \right)$$

$$= 4m \sum_{k=1}^{n} k - m \sum_{k=1}^{n} 1 = 4m \cdot \frac{n(n+1)}{2} - mn$$

$$= 2mn^2 + \text{lower-order terms}$$

$$\sim O(mn^2)$$

Though modified G-S is more stable than the classical one, it is still unsatisfactory under some circumstances. The reason is that the Q matrix is computed column-by-column, and their orthogonality are not fully exploited.

In the following, we present algorithms where Q is computed as a whole matrix. By this way, the columns of Q are guaranteed orthogonal.

To this end, we first present a "full" QR decomposition by completing the Q matrix to a full square orthogonal matrix.

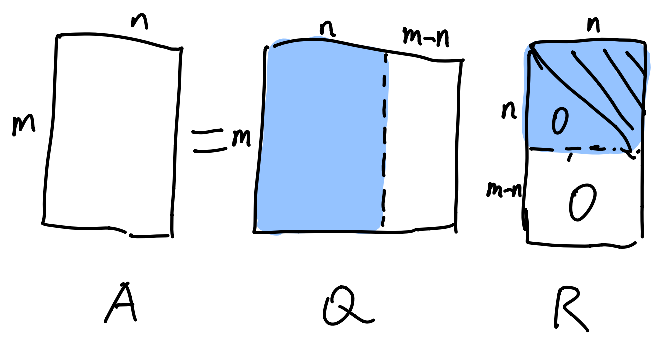
Full QR decomposition

Let $A \in \mathbb{R}^{m\times n}$ $(m \geq n)$ and $\text{rank}(A) = n$. Then, there exist a decomposition

$$A = QR,$$

where $Q \in \mathbb{R}^{m\times m}$ is square and orthogonal (i.e., $QQ^T = Q^TQ = I$)

and $R \in \mathbb{R}^{m\times n}$ is upper triangular (i.e., $r_{ij} = 0$ if $i > j$)



- If we have the full QR decomposition, then, by taking only the blue parts in Q and R in the figure above, we obtain the QR decomposition discussed previously. Since the QR decomposition discussed before involves only part of the matrices in the full QR decomposition, we call it economic QR decomposition in the following discussions.
- If we have the economic QR decomposition $A = QR$, where $Q \in \mathbb{R}^{m\times n}$ satisfies $Q^TQ = I$ and $R \in \mathbb{R}^{n\times n}$ is upper triangular, then we can
  - — Completing Q to a full square orthogonal matrix by adding $(m-n)$ columns that are orthogonal to columns of Q,

  and

  - — adding a $(m-n)\times n$ 0 matrix below R

  to obtain the full QR decomposition.

Both the following algorithms compute the full QR decomposition.

* QR by Orthogonal Triangularization.

We want to compute the full QR decomposition

$$A = QR$$

Since $QQ^T = Q^TQ = I$, Q is invertible with inverse Q^T.

So, $$A = QR \iff Q^TA = R.$$

We then need to find an square orthogonal matrix Q^T that reduces A to an upper triangular matrix.

Idea: Reduce A to an upper triangular matrix column by column by an orthogonal transforms

$$Q_1A = \begin{bmatrix} x & x & \cdots & x \\ 0 & x & & \vdots \\ 0 & x & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & x & & x \end{bmatrix} \qquad Q_2Q_1A = \begin{bmatrix} x & x & x & \cdots & x \\ 0 & x & x & \cdots & x \\ 0 & 0 & x & \cdots & x \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & x & \cdots & x \end{bmatrix} \qquad \cdots$$

$$\underbrace{Q_nQ_{n-1}\cdots Q_2Q_1}_{Q^T\in\mathbb{R}^{m\times m}}A = \begin{array}{c} \\ n\left\{\begin{array}{c}\\ \\ \\ \\ \end{array}\right. \\ \\ \\ \\ \end{array} \overbrace{\begin{bmatrix} x & x & \cdots & x \\ & x & & \vdots \\ & & \ddots & \vdots \\ & 0 & & x \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}}^{n} \equiv R$$

Then: ① $Q = Q_1^TQ_2^T\cdots Q_n^T$ so that

$$\left.\begin{array}{l} Q^TQ = Q_n\cdots \underbrace{Q_2\underbrace{Q_1Q_1^T}_{I}Q_2^T}_{I}\cdots Q_n^T = I \\ \text{and } QQ^T = Q_1^T\cdots \underbrace{Q_{n-1}^T\underbrace{Q_n^TQ_n}_{I}Q_{n-1}}_{I}\cdots Q_1 = I \end{array}\right\} \Rightarrow Q \text{ is a square full orthogonal matrix}$$

② $$Q^TA = R \Rightarrow QQ^TA = QR \Rightarrow A = QR.$$

Thus, we find the full QR decomposition.

Question: Which $Q_1, Q_2, \cdots, Q_n$ reduce A to upper triangular?

Since

$$Q_1 A = \begin{bmatrix} x & x & \cdots & x \\ 0 & x & \cdots & x \\ 0 & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & x & \cdots & x \end{bmatrix}, \quad Q_1 a_1 = \begin{bmatrix} x \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} \equiv c e_1 \text{ for some } c \in \mathbb{R}$$

Because Q_1 is an orthogonal matrix, which preserve the length,

$$\|a_1\|_2 = \|Q_1 a_1\|_2 = \|c e_1\|_2 = |c|$$

So $\quad Q_1 a_1 = \alpha \|a_1\|_2 e_1$, where $\alpha = 1$ or -1.

Householder transform (Householder reflector)

$$H_v = I - 2uu^T,$$

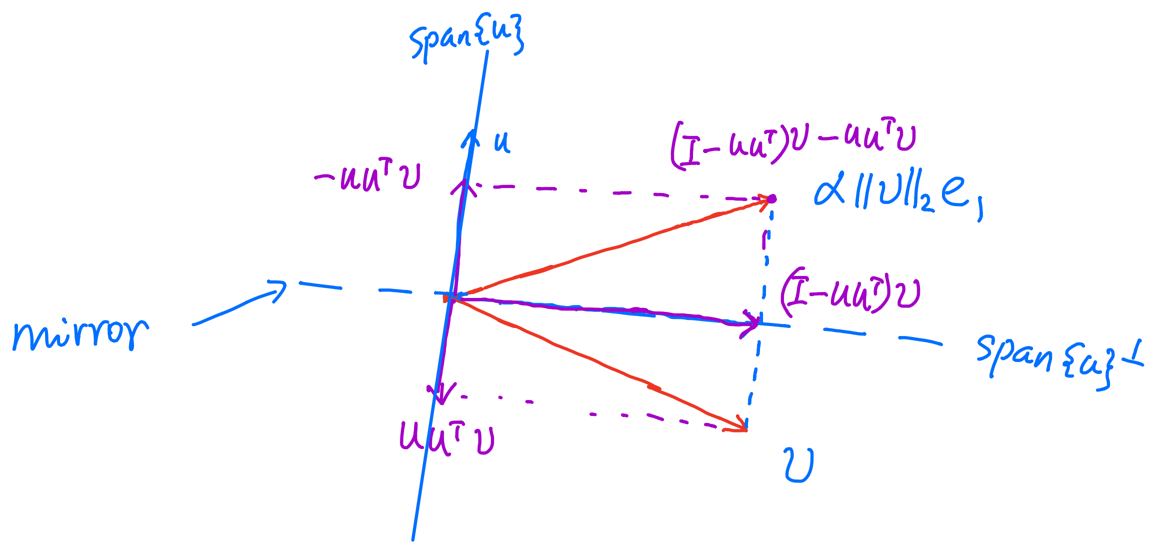
where u is the unit direction of $v - \alpha\|v\|_2 e_1$

— H_v is orthogonal:

$$\begin{aligned} H_v^T H_v &= (I - 2uu^T)(I - 2uu^T) \\ &= I - 2uu^T - 2uu^T + 2uu^T 2uu^T \\ &= I - 4uu^T - 4u\overset{1}{(u^Tu)}u^T = I \end{aligned}$$

— $H_v v = \alpha \|v\|_2 e_1$

$$H_v v = (I - 2uu^T)v = \underbrace{(I - uu^T)v}_{\text{projection of } v \text{ onto } \operatorname{span}\{u\}^\perp} - \underbrace{uu^Tv}_{\text{projection of } v \text{ onto } \operatorname{span}\{u\}} = \alpha\|v\|_2 e_1$$



We see that $H_v v$ is the reflection of v about the mirror span$\{u\}^\perp$.

Thus, Householder transform is also known as Householder reflector.

— For numerical stability, we usually choose

$$\alpha = -\mathrm{sgn}(v_1)$$

We should avoid the difference of two numbers that are close due to the cancellation, e.g.. $\underset{\text{5 correct digits}}{1.00002} - \underset{\text{5 digits}}{1.00001} = \underset{\text{1 digits.}}{0.00001}$

loss of 4 digits.

Therefore, we choose $Q_1 = H_{a_1}$, and

$$Q_1 A = \begin{bmatrix} x & x & x & & x \\ 0 & \boxed{x} & x & \cdots & x \\ \vdots & x & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & x & x & & x \end{bmatrix} \equiv A^{(1)}.$$

$a_2^{(1)}$

Let $a_2^{(1)} = A^{(1)}(2:m,2)$ and let

$$Q_2 = \begin{bmatrix} 1 & \\ & H_{a_2^{(1)}} \end{bmatrix}.$$

Then

$$Q_2 Q_1 A = Q_2 A^{(1)}$$

$$= \begin{bmatrix} 1 & \\ & H_{a_2^{(1)}} \end{bmatrix} \begin{bmatrix} x & x & x & \cdots & x \\ 0 & x & x & \cdots & x \\ 0 & x & x & \cdots & x \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & x & x & \cdots & x \end{bmatrix} = \begin{bmatrix} x & x & x & \cdots & x \\ 0 & x & x & \cdots & x \\ \vdots & 0 & x & \cdots & x \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & x & \cdots & x \end{bmatrix} \equiv A^{(2)}$$

$a_2^{(1)}$

$-\mathrm{sgn}(a_2^{(1)}(1))\|a_2^{(1)}\|_2$

unchanged from $A^{(1)}$

modified from the corresponding part of $A^{(1)}$.

Let $a_3^{(2)} = A^{(2)}(3:m,3)$ and let

$$Q_3 = \begin{bmatrix} 1 & & \\ & 1 & \\ & & H_{a_3^{(2)}} \end{bmatrix}$$

Then $Q_3 Q_2 Q_1 A = \begin{bmatrix} x & x & x & x & \cdots & x \\ & x & x & x & & x \\ & & x & x & & x \\ & & & x & & \vdots \end{bmatrix}$

$$\begin{bmatrix} & \vdots & \vdots \\ & \times & \times \end{bmatrix}.$$

At step k, choose

$$Q_k = \begin{bmatrix} \overbrace{\begin{matrix}1 & & \\ & \ddots & \\ & & 1\end{matrix}}^{k-1} & \\ & H_{a_k^{(k-1)}} \end{bmatrix}$$

Finally, we have

$$Q_n Q_{n-1} \cdots Q_1 A = \begin{bmatrix} \left.\begin{matrix} \times & \times & \cdots & \times \\ & \times & & \vdots \\ & & \ddots & \times \\ & & & \times \end{matrix}\right\} n \\ 0 \end{bmatrix}$$

Algorithm (QR by Householder transforms)

for $k = 1:n$

$v_k = A(k:m, k)$

$u_k = v_k + \mathrm{sgn}(v_k(1)) \|x\|_2 e_1$

$u_k = u_k / \|u_k\|_2$

$A(k:m, k:n) = A(k:m, k:n) - 2u_k(u_k^T A(k:m, k:n))$

end

Operation count:

- $2(m-k)(n-k)$ flops for $u_k^T A(k:m, k:n)$
- $(m-k)(n-k)$ flops for $u_k^T(\cdots)$
- $(m-k)(n-k)$ flops for $A(k:m, k:n) - \cdots$
- Other operations are in $O(m-k)$ (neglectable)

$$\sum_{k=1}^{n} 4(m-k)(n-k) = 4\sum_{k=1}^{n}(mn - k(m+n) + k^2) \approx 2mn^2 - \frac{2}{3}n^3$$

The Q matrix is stored implicitly by $u_1, u_2, \cdots, u_n$.

Algorithm (Implicit Calculation of $Q^T x$) $\quad Q^T x = Q_n Q_{n-1} \cdots Q_1 x$

for $k = 1:n$

$\quad x(k:m) = x(k:m) - 2\left(u_k^T x(k:m)\right) u_k$

end.

Algorithm (Implicit calculation of Qx) $\quad Qx = (Q_n Q_{n-1} \cdots Q_1)^T x$

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

$= Q_1 Q_2 \cdots Q_n x$

for $x = n:-1:1$

$\quad x(k:m) = x(k:m) - 2(u_k^T x(k:m)) u_k$

end

We need the following algorithm for forming the Q factor explicitly.

$$Q^T = Q_n Q_{n-1} \cdots Q_1 \Rightarrow Q = Q_1^T Q_2^T \cdots Q_n^T = Q_1 \cdots Q_n I$$

Algorithm: $\quad Q = I$

for $k = n:-1:1$

$\quad Q(k:m, 1:m) = Q(k:m, 1:m) - 2u_k\left(u_k^T Q(k:m, 1:m)\right)$

end

The operating counting is obtained similar to Householder QR case.

$$\sum_{k=1}^{n} 4(m-k)^2 = 4 \sum_{k=1}^{n}(m^2 - 2mk + k^2) \simeq 4m^2 n - 4mn^2 + \frac{4n^3}{3}$$

$$= 4(m-n)mn + \frac{4}{3}n^3$$

Therefore, the total computational cost of full QR decomposition via Householder transform is $O(m^2 n)$.

In contrast, if only economic QR is needed, we only compute the first n columns of the Q factor in the full QR decomposition. Thus, the Q factor in economic QR is computed via:

$$Q_1 Q_2 \cdots Q_n I(1:m, 1:n)$$

Algorithm: $Q = I(1:m, 1:n)$

for $k = n:-1:1$

$$Q(k:m, 1:n) = Q(k:m, 1:n) - 2u_k(u_k^T Q(k:m, 1:n))$$

end

Similarly, it can be checked the computational cost is $O(mn^2)$ for the Q factor. Thus, the total computational cost of economic QR decomposition is $O(mn^2)$. This is the same order as the computational cost by Gram-Schmidt, and Householder QR has a bigger constant before mn^2 than G-S QR.

① Householder QR is more computationally stable than GS QR.

② Householder QR is more computationaly expensive than GS QR, though both computational costs are in the same order ($O(mn^2)$) for economic QR.

③ Householder QR can be used to compute both full and economic QR, but G-S QR can be used to compute only the economic QR.

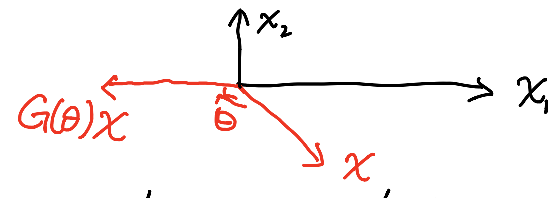
– QR by Givens Rotations

In many application and other problems, the A matrix contain many 0. However, Householder QR ignores existing 0's in A in triangalarization. We present Given Rotation QR to utilize 0's in A.

- Rotation matrix in $\mathbb{R}^2$

$$G(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \end{bmatrix}$$

$$G(\theta) = \begin{bmatrix} & \\ \sin\theta & \cos\theta \end{bmatrix},$$



which rotate $x \in \mathbb{R}^2$ anticlockwisely by θ

To set an element to 0, choose $\sin\theta$ and $\cos\theta$ so that

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_i \\ x_j \end{bmatrix} = \begin{bmatrix} \sqrt{x_i^2 + x_j^2} \\ 0 \end{bmatrix},$$

where $\cos\theta = \dfrac{x_i}{\sqrt{x_i^2 + x_j^2}}$ $\quad \sin\theta = -\dfrac{x_j}{\sqrt{x_i^2 + x_j^2}}$

QR by Givens Rotation:

Introducing 0's in column from bottom and up.

$$\underbrace{\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & c & -s \\ & & s & c \end{bmatrix}}_{G_1^{(1)}} \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix}}_{A} \longrightarrow \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix}}_{A_1^{(1)}}$$

Elements in red mean "the elements are changed"

$$\underbrace{\begin{bmatrix} 1 & & & \\ & c & -s & \\ & s & c & \\ & & & 1 \end{bmatrix}}_{G_2^{(1)}} \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix}}_{A_1^{(1)}} \longrightarrow \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix}}_{A_2^{(1)}}$$

$$\underbrace{\begin{bmatrix} c & -s & & \\ s & c & & \\ & & 1 & \\ & & & 1 \end{bmatrix}}_{G_3^{(1)}} \underbrace{\begin{bmatrix} x & x & x \\ x & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix}}_{A_2^{(1)}} \longrightarrow \underbrace{\begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix}}_{A_3^{(1)} = A_0^{(2)}}$$

$$\begin{bmatrix} 1 & \cdots \end{bmatrix} \begin{bmatrix} x & x & x \\ \cdots \end{bmatrix} \quad \begin{bmatrix} x & x & x \\ \cdots \end{bmatrix}$$

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & c & -s \\ & & s & c \end{bmatrix} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \longrightarrow \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & 0 & x \end{bmatrix}$$

$$G_1^{(2)} \qquad A_0^{(2)} \qquad A_1^{(2)}$$

$$\begin{bmatrix} 1 & & & \\ & c & -s & \\ & s & c & \\ & & & 1 \end{bmatrix} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & 0 & x \end{bmatrix} \longrightarrow \begin{bmatrix} \boxed{\begin{matrix} x \\ 0 \end{matrix}} & \boxed{\begin{matrix} x \\ x \end{matrix}} & \begin{matrix} x \\ x \end{matrix} \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix}$$

$$G_2^{(2)} \qquad A_1^{(2)} \qquad A_2^{(2)} \equiv A_0^{(3)}$$

We can not further rotate the vector in red box to $\begin{pmatrix} x \\ 0 \end{pmatrix}$, because it will cause the vector in blue $\begin{pmatrix} x \\ x \end{pmatrix}$

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & c & -s \\ & & s & c \end{bmatrix} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & x \end{bmatrix} \longrightarrow \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \end{bmatrix} \equiv \begin{bmatrix} R \\ 0 \end{bmatrix}$$

$$G_1^{(3)} \qquad A_0^{(3)} \qquad A_1^{(3)}$$

So $\underbrace{G_1^{(3)} G_2^{(2)} G_1^{(2)} G_3^{(1)} G_2^{(1)} G_1^{(1)}}_{Q} A = \begin{bmatrix} R \\ 0 \end{bmatrix}$

For general matrix $A \in \mathbb{R}^{m \times n}$ with $m \geq n$, the QR with Givens rotation needs $3mn^2 - n^3$ flops, which is about 50% more than Householder QR.

- Givens QR is mor suitable for parallel computing than Householder QR.

Example: $\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{G_1} \begin{bmatrix} x & x & x \\ x & x & x \\ 0 & x & x \end{bmatrix}$ The computation of the blue part

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{G_2} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \end{bmatrix}$$

and red part can be done parallelly.

- Givens QR is more suitable for sparse A than Householder QR.

Example:
$$\begin{bmatrix} \times & \times & \times \\ 0 & \times & 0 \\ \times & \times & \times \\ 0 & \times & 0 \\ \times & \times & \times \end{bmatrix}$$
no need to reduce the leading elements of these two rows to 0.

Different forms of QR decomposition.

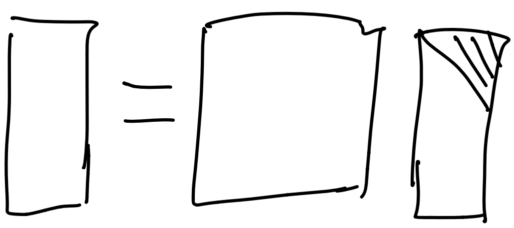
Let $A \in \mathbb{R}^{m\times n}$

- We first assume $m \geq n$ and $\operatorname{rank}(A) = n$.

① Full QR: $A = QR$,

where $Q \in \mathbb{R}^{m\times m}$ satisfying $QQ^T = Q^TQ = I$

$R \in \mathbb{R}^{m\times n}$ upper triangular



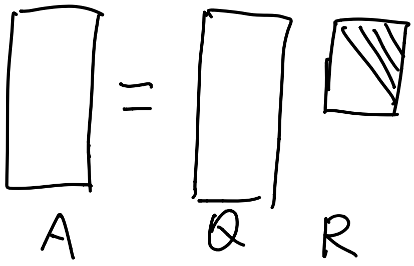
A Q R

Full QR can be computed via Householder QR and Givens QR, both costs $O(m^2n)$.

② Economic QR: $A = QR$

where $Q \in \mathbb{R}^{m\times n}$ satisfying $Q^TQ = I$ $\left(QQ^T \neq I \text{ if } m > n \right)$

$R \in \mathbb{R}^{n\times n}$ upper triangular

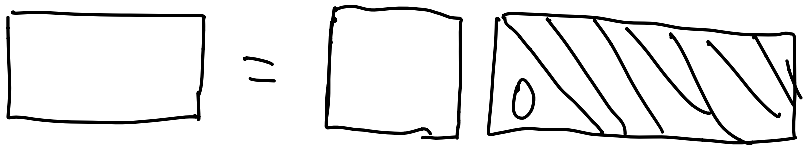


The economic QR can be computed via Gram-Schmidt QR, Householder QR, and Givens QR. All algorithms cost $O(mn^2)$.

- We then assume $m \le n$

③ Let $A \in \mathbb{R}^{m\times n}$ with $m \le n$ with the first m columns linearly independent.
Then there exists a decomposition

$$A = QR$$



where $Q \in \mathbb{R}^{m\times m}$ satisfying $QQ^T = Q^TQ = I$.
$R \in \mathbb{R}^{m\times n}$ is upper triangular.

- We finally consider $m \ge n$ but the rank of A is arbitrary.

④ Rank-Revealing QR (RRQR)
(To handle the case where columns of A are linearly dependent or they are close to linearly dependent)

Let $A \in \mathbb{R}^{m\times n}$ with $m \ge n$. There exists a decomposition

$$AP = QR$$

where $P \in \mathbb{R}^{n\times n}$ a permutation (on columns of A)
$Q \in \mathbb{R}^{m\times m}$ orthogonal (i.e., $QQ^T = Q^TQ = I$)
$R \in \mathbb{R}^{m\times n}$ is in the form of

$$R = \left[\begin{array}{c|c} R_{11} & R_{12} \\ \hline 0 & R_{22} \\ \hline 0 & 0 \end{array}\right] \begin{array}{l} \}\, K \\ \}\, n-K \\ \end{array}$$

$R_{11} \in \mathbb{R}^{K\times K}$ is upper triangular
$R_{22} \in \mathbb{R}^{(n-K)\times(n-K)}$ is small (e.g., $\|R_{22}\|_F$ is small)

$$\overset{AP}{\underset{\quad k \qquad n-k}{\begin{bmatrix} A_1 & A_2 \end{bmatrix}}} = \overset{Q}{\underset{k \quad n-k}{\begin{bmatrix} Q_1 & Q_2 & Q_3 \end{bmatrix}}} \overset{R}{\underset{k \quad n-k}{\begin{array}{c} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \\ 0 & 0 \end{bmatrix} \end{array}}} \begin{array}{l} k \\ n-k \\ \end{array} \qquad \boxed{R_{22} \text{ is small}}$$

$$\begin{cases} A_1 = Q_1 R_{11} \\ A_2 = Q_1 R_{12} + Q_2 R_{22} \underset{\uparrow}{\approx} Q_1 R_{12} \end{cases} \Longrightarrow \begin{cases} Q_1 \text{ forms an orthonormal basis of } \operatorname{span}\{A_1\} \\ \operatorname{Ran}(A_1) = \operatorname{Ran}(Q_1) \end{cases}$$

R_{22} is small

$$\underbrace{\overset{k \qquad n-k}{\left[A_1 \;\; A_2\right]}}_{AP} \approx \overset{k}{Q_1} \left[R_{11} \;\; R_{12}\right]$$

So, A_1 are "important" columns of A

(A_1 are strongly linearly independent columns of A)

A_2 are less important columns of A.

(A_2 is close to linearly dependent on A_1)

k is called numerical rank of A.

RRQR can be widely used for selection of "representative" columns of A.

However, the computation of RRQR is challenging, though there are many available software packages for RRQR.