Fatoração QR

Prof. Alexandre Salles da Cunha e Profa. Ana Paula Couto

UFMG
UNIVERSIDADE FEDERAL
DE MINAS GERAIS

UFMG - ICEx
DEPARTAMENTO DE CIÊNCIA DA
COMPUTAÇÃO

$A = QR$ onde $A \in \mathbb{R}^{m \times n}$, $Q \in \mathbb{R}^{m \times n}$, $R \in \mathbb{R}^{n \times n}$

- Hipótese inicial: A possui posto completo: $r(A) = n$ (vamos relaxar esta hipótese em breve).
- Q possui n colunas ortonormais.
- R é triangular superior e $r_{ii} \neq 0$ para qualquer i.

$\Rightarrow$ Se o sistema é quadrado ($m = n$) e precisamos resolver $Ax = b$ temos:

$\Rightarrow$ $Q(Rx) = b \rightarrow Qy = b \rightarrow y = Q^T b \rightarrow Rx = Q^T b$
(mais estável numericamente pois Q é ortogonal).

- Matrizes ortogonais são excelentes para computação científica: as quantidades computadas não crescem muito, dado que Q tem norma unitária; menores erros numéricos.

- Notação:
  - $A_i \in \mathbb{R}^m : i = 1, \ldots, n$ são as colunas de A.
  - $q_i \in \mathbb{R}^m : i = 1, \ldots, n$ são as colunas de Q.
- Subespaços são idênticos - estrutura a ser explorada no algoritmo:

$$\begin{aligned} span\{A_1\} &= span\{q_1\} \\ span\{A_1, A_2\} &= span\{q_1, q_2\} \\ \vdots & \quad \vdots \\ span\{A_1, \ldots, A_n\} &= span\{q_1, q_2, \ldots, q_n\} \end{aligned}$$

- Desejamos encontrar uma base mais conveniente para $\mathcal{C}(A)$. Mais conveniente significa "base ortonormal".
- Além disso, sabemos que:
  $span\{A_1\} \subseteq span\{A_1, A_2\} \subseteq \cdots \subseteq span\{A_1, A_2, \ldots, A_n\}$

$$\left[\begin{array}{c|c|c|c} A_1 & A_2 & \cdots & A_n \end{array}\right] = \left[\begin{array}{c|c|c|c} q_1 & q_2 & \cdots & q_n \end{array}\right] \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \vdots \\ & & & r_{nn} \end{bmatrix} \quad (1)$$

Equivalentemente, o sistema (1) corresponde a:

- $A_1 = r_{11} q_1$
- $A_2 = r_{12} q_1 + r_{22} q_2$
- $\cdots$
- $A_i = \sum_{k=1}^{i} r_{ki} q_k$
- $A_n = r_{1n} q_1 + r_{2n} q_2 + \cdots + r_{nn} q_n$

onde:

- $q_i^T q_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$
- $r_{ii} \neq 0 : i = 1, \ldots, n$ (como consequência de $r(A) = n$).

Fatoração $A = QR$: Algoritmo clássico

- Baseado na ideia de ortogonalização de bases de Gram-Schmidt.
- Do ponto de vista teórico e conceitual, a ortogonalização de Gram-Schmidt é fundamental, permitindo demonstrar diversas propriedades da fatoração QR.
- Na sequência, apresentamos o algoritmo clássico baseado em Gram-Schmidt.
- Posteriormente, discutiremos como modificá-lo visando redução de erros numéricos.

Invariante do algoritmo:

No início da iteração j:

- dispomos de $j-1$ colunas ortonormais $q_i : i = 1, \ldots, j-1$
- estas colunas satisfazem:

$$span\{A_1, \ldots, A_{j-1}\} = span\{q_1, \ldots, q_{j-1}\}$$

Nosso objetivo na iteração j é determinar q_j tal que:

- $span\{A_1, \ldots, A_j\} = span\{q_1, \ldots, q_j\}$
- $q_j \perp span\{q_1, \ldots, q_{j-1}\}$
- $\|q_j\| = 1$

O processo empregado para se determinar q_j como desejado acima é a *Ortonogalização de Gram-Schmidt*.

- Vamos formalizar o algoritmo e, em paralelo, aplicá-lo para obter a fatoração de uma matriz 3×3.
- Enfatizamos que a fatoração $A = QR$ não se destina apenas a fatorar matriz quadradas.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & -3 \\ 0 & -2 & 3 \end{bmatrix}.$$

Observe que as colunas $A_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $A_2 = \begin{bmatrix} 2 \\ 0 \\ -2 \end{bmatrix}$, $A_3 = \begin{bmatrix} 3 \\ -3 \\ 3 \end{bmatrix}$ são linearmente independentes.

Primeira iteração - $j = 1$:

- Dado A_1 desejamos $span\{q_1\} = span\{A_1\}$, $\|q_1\| = 1$.
- Então fazemos $q_1 = \frac{A_1}{\|A_1\|}$
- Ou seja, $r_{11} = \|A_1\|$

Primeira iteração - $j = 1$:

$$A_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}, \; q_1 = \frac{\sqrt{2}}{2} \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}, \; r_{11} = \sqrt{2}.$$

Iteração típica - $j \in \{2, \ldots, n\}$: Dada a nova coluna A_j e $\{q_1, q_2, \ldots, q_{j-1}\}$ satisfazendo

$$span\{A_1, \ldots, A_{j-1}\} = span\{q_1, \ldots, q_{j-1}\}$$

calculamos q_j, da seguinte forma:

1. Calculamos a projeção p_j de A_j no subespaço $span\{q_1, \ldots, q_{j-1}\}$

$$p_j = (q_1^T A_j)q_1 + (q_2^T A_j)q_2 + \cdots + (q_{j-1}^T A_j)q_{j-1}$$

2. Da expressão acima, temos: $r_{ij} = q_i^T A_j$ para $i = 1, \ldots, j-1$.
3. Calculamos o vetor "erro": $v_j = A_j - p_j$
   (recorde-se de que $(A_j - p_j) \perp span\{q_1, \ldots, q_{j-1}\}$)
4. Normalizamos o erro: $q_j = \frac{v_j}{r_{jj}}$, e $r_{jj} = \|v_j\|$

Segunda iteração - $j = 2$:

- Dados: $q_1 = \frac{\sqrt{2}}{2}\begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $A_2 = \begin{bmatrix} 2 \\ 0 \\ -2 \end{bmatrix}$
- Projeção: $p_2 = (q_1^T A_2)q_1 = \sqrt{2}q_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $r_{12} = \sqrt{2}$.
- Erro: $v_2 = A_2 - p_2 = \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}$, $\|v_2\| = r_{22} = \sqrt{6}$.
- $q_2 = \frac{\sqrt{6}}{6}\begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}$

Terceira iteração - $j = 3$:

- Dados: $q_1 = \frac{\sqrt{2}}{2}\begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $q_2 = \frac{\sqrt{6}}{6}\begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}$, $A_3 = \begin{bmatrix} 3 \\ -3 \\ 3 \end{bmatrix}$
- Projeção:

  $$p_3 = (q_1^T A_3)q_1 + (q_2^T A_3)q_2 = 3\sqrt{2}q_1 - \sqrt{6}q_2 = \begin{bmatrix} 2 \\ -4 \\ 2 \end{bmatrix},$$

  $r_{13} = 3\sqrt{2}, r_{23} = -\sqrt{6}$.
- Erro: $v_3 = A_3 - p_3 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$.
- $q_3 = \frac{\sqrt{3}}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$, $r_{33} = \sqrt{3}$.

Fatoração concluída:

$$\begin{bmatrix} 1 & 2 & 3 \\ -1 & 0 & -3 \\ 0 & -2 & 3 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{6}}{6} & \frac{\sqrt{3}}{3} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{6}}{6} & \frac{\sqrt{3}}{3} \\ 0 & -\frac{\sqrt{6}}{3} & \frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} \sqrt{2} & \sqrt{2} & 3\sqrt{2} \\ & \sqrt{6} & -\sqrt{6} \\ & & \sqrt{3} \end{bmatrix}$$

- Passo de projeção:

$$\begin{aligned} p_j =& (q_1^T A_j) q_1 + (q_2^T A_j) q_2 + \cdots + (q_{j-1}^T A_j) q_{j-1} \\ p_j =& q_1 (q_1^T A_j) + q_2 (q_2^T A_j) + \cdots + q_{j-1} (q_{j-1}^T A_j) \\ p_j =& \sum_{i=1}^{j-1} q_i q_i^T A_j \end{aligned}$$

- Calculando a diferença ou erro:

$$\begin{aligned} v_j =& A_j - p_j \\ v_j =& I A_j - \sum_{i=1}^{j-1} q_i q_i^T A_j \\ v_j =& (I - Q_{j-1} Q_{j-1}^T) A_j \end{aligned}$$

onde Q_{j-1} é à matriz $m \times (j-1)$ com as primeiras $j-1$ colunas de Q.

- $(I - Q_{j-1} Q_{j-1}^T)$ projeta A_j em $span\{q_1, \ldots, q_{j-1}\}^{\perp}$.

Atenção: Assumimos posto completo nesta implementação.

```
function [Q,R] = Fatoracao_QR_Classica(A)
    [m,n] = size(A)
    R = zeros(n,n)
    Q = zeros(m,n)
    for j = 1:n
        V = A(:,j)
        for i = 1:j-1
            R(i,j) = Q(:,i)'*A(:,j)
            V = V - R(i,j)*Q(:,i)
        end
        R(j,j) = norm(V,2)
        Q(:,j) = 1.0/R(j,j) * V
    end
endfunction
```

- Complexidade:
- O algoritmo pode falhar ?

  Se $r_{jj} = \|v_j\| = 0$, o algoritmo falha.

  $\Rightarrow$ Porém, isso só ocorre se $v_j = 0_m$ (vetor erro é identicamente nulo), o que significa que $A_j \in span\{q_1, q_2, \cdots, q_{j-1}\} = span\{A_1, A_2, \cdots, A_{j-1}\}$, contradizendo a hipótese de posto completo de A, $r(A) = n$.

  $\Rightarrow$ Equivalentemente, neste caso, $p_j = A_j$, ou seja A_j é sua própria projeção em $span\{q_1, q_2, \cdots, q_{j-1}\}$.

Vamos considerar que A não tenha posto completo: $r(A) < n$.

- Neste caso, devemos esperar que, para algum j, encontremos $v_j = 0_m$ e $r_{jj} = 0$.
- Eliminamos a coluna A_j do processo, pois ela não "adiciona posto" à A. $\mathcal{C}(A)$ não precisa de A_j (ou de um q_j associado a A_j para ser caracterizado).
- Assim, ao invés de obtermos uma nova coluna $q_j \in \mathcal{C}(A)$, linearmente independente com $\{q_1, \ldots, q_{j-1}\}$, podemos "completar" a fatoração com qualquer coluna q_j ortogonal a $\{q_1, \ldots, q_{j-1}\}$.
- Podemos estender a ideia acima de forma que Q tenha m colunas ortonormais e não apenas $r(A)$ colunas (ou $n = r(A)$ colunas quando o posto é completo) $\Rightarrow$ Fatoração QR completa.

porque Q tem as mesmas dimensões de A

- Veja que $Q \in \mathbb{R}^{m \times n}$ e $R \in \mathbb{R}^{n \times n}$.
- Q contém apenas colunas ortonormais que geram o espaço coluna de A.
- R contém, em suas colunas, os pesos que relacionam as combinações lineares de bases diferentes para o espaço coluna de A.

Porém, podemos expandir Q, adicionando a ela $m - r(A)$ novas colunas m dimensionais, que representam uma base para $\mathcal{N}(A^T)$ e introduzindo $m - r(A)$ linhas de zeros em R....

permite tratar o caso em que A possui deficiência de posto e fornecer uma base para $\mathcal{N}(A^T)$

$A = QR$, porém com: $A \in \mathbb{R}^{m\times n}$, $Q \in \mathbb{R}^{m\times m}$, $R \in \mathbb{R}^{m\times n}$

$$Q = \left[\begin{array}{c|c|c|c|c|c|c} q_1 & q_2 & \cdots & q_{r(A)} & q_{r(A)+1} & \cdots & q_m \end{array}\right] \tag{2}$$

- $q_i : i = 1, \ldots, r(A)$ fornecem base para $\mathcal{C}(A)$
- $q_j : j = r(A)+1, \ldots, m$ fornecem base $\mathcal{C}(A)^{\perp} = \mathcal{N}(A^T)$
- Completamos R com $m - r(A)$ linhas de zeros.

Exemplo: Fatoração QR (completa) de A com deficiência de rank

$$A = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 2 & 0 \\ -1 & 2 & 3 & -1 \\ 2 & 1 & 4 & 1 \end{bmatrix}.$$

Primeira iteração - $j = 1$:

$$A_1 = \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}, \; q_1 = \frac{\sqrt{6}}{6} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}, \; r_{11} = \sqrt{6}.$$

Segunda iteração - $j = 2$:

$$A_2 = \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \; q_1 = \frac{\sqrt{6}}{6} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}$$

Projeção: $p_2 = (q_1^T)A_2 q_1 = 0 q_1$ (ou seja: $q_1 \perp A_2$), $r_{12} = 0$

$$q_2 = \frac{\sqrt{6}}{6} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}, \; r_{22} = \sqrt{6}.$$

Terceira iteração - $j = 3$:

$$A_3 = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}, \; q_1 = \frac{\sqrt{6}}{6} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}, \; q_2 = \frac{\sqrt{6}}{6} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

Projeção: $p_3 = (q_1^T)A_3 q_1 + (q_2^T)A_3 q_2$

$r_{13} = q_1^T A_3 = \sqrt{6}$, $r_{23} = q_2^T A_3 = 2\sqrt{6}$, $\Rightarrow \quad p_3 = A_3$

$v_3 = 0, r_{33} = 0$

Desta forma $A_3 \in span\{q_1, q_2\}$ e a matriz A não possui posto completo.

Em q_3 na fatoração, posteriormente, vamos armazenar uma coluna ortonormal, que seja ortogonal a $\mathcal{C}(A)$. Ou seja, ainda não determinamos q_3. Adiamos, temporariamente este passo.

Quarta iteração - $j = 4$:

$$A_4 = \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \end{bmatrix}, \; q_1 = \frac{\sqrt{6}}{6} \begin{bmatrix} 1 \\ 0 \\ -1 \\ 2 \end{bmatrix}, \; q_2 = \frac{\sqrt{6}}{6} \begin{bmatrix} 0 \\ 1 \\ 2 \\ 1 \end{bmatrix}$$

Projeção: $p_4 = (q_1^T)A_4 q_1 + (q_2^T)A_4 q_2 + 0q_3$

$r_{14} = q_1^T A_4 = \frac{\sqrt{6}}{2}$, $r_{24} = q_2^T A_4 = -\frac{\sqrt{6}}{6}$, $r_{34} = 0$.

$$p_4 = \begin{bmatrix} \frac{1}{2} \\ -\frac{1}{6} \\ -\frac{5}{6} \\ \frac{5}{6} \end{bmatrix}$$

Erro: $v_4 = \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{6} \\ -\frac{1}{6} \\ \frac{1}{6} \end{bmatrix}$, $r_{44} = \frac{\sqrt{3}}{3}$, $q_4 = \sqrt{3} \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{6} \\ -\frac{1}{6} \\ \frac{1}{6} \end{bmatrix}$

- Desejamos encontrar $q_3 \in \mathcal{N}(A^T)$.
- (apenas para facilitar as contas aqui....) multiplicando q_1, q_2, q_4 respectivamente por $\frac{6}{\sqrt{6}}, \frac{6}{\sqrt{6}}, \frac{6}{\sqrt{3}}$, resolvemos o sistema linear:

$$\begin{bmatrix} 1 & 0 & -1 & 2 \\ 0 & 1 & 2 & 1 \\ -3 & 1 & -1 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- Após eliminação temos: $\begin{bmatrix} 1 & 0 & -1 & 2 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & -6 & 6 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

- Para resolvermos o sistema, arbitramos a variável livre y_4, por exemplo em $y_4 = -1$ e resolvemos.
- Após eliminação temos: $\begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & -6 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 6 \end{bmatrix}$
- Obtendo a solução $y = \begin{bmatrix} 1 & 3 & -1 & -1 \end{bmatrix}^T$
- Fazemos $q_3 = \frac{1}{\|y\|} y = \frac{\sqrt{3}}{6} \begin{bmatrix} 1 & 3 & -1 & -1 \end{bmatrix}^T$
- A terceira linha da matriz R é composta por zeros na fatoração.
- Para termos a estrutura que desejamos, fazemos um pivoteamento de colunas, trocando a 4a. com a 3a., de forma que as três primeiras entradas na diagonal de R sejam não nulas, uma vez que $r(A) = 3$.

Considere a matriz $A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}$.

- Veja que $A_3 = 2A_2 - A_1$, $A_4 = 2A_3 - A_2$, $r(A) = 2$ e que, portanto, a matriz A tem deficiência de rank.
- A fatoração baseada puramente em Gram-Schmidt tem dificuldade em identificar o rank da matriz (fruto de erros numéricos).

```
-->[q,r] = Fatoracao_QR_Classica(A)
 q  =
   0.09245     0.5392919  -0.5465825  -0.5465825
   0.4622502   0.2927584  -0.370154   -0.370154
   0.8320503   0.046225   -0.2118872  -0.2118872
   0.09245    -0.0616334   0.0397829   0.0397829
   0.2773501  -0.7858253   0.7195517   0.7195517
 r  =
   10.816654   11.926054   13.035455   14.144855
   0.          1.6641006   3.3282012   4.9923018
   0.          0.          3.209D-14  -7.650426
   0.          0.          0.          7.650426
```

1. Assim que calculamos a coluna q_1, projetamos as colunas $A_2, \ldots, A_n$ em q_1, calculamos os elementos $r_{1j} : j = 2, \ldots, n$, calculamos a diferença (ou erro) $v_j = A_j - r_{1j} q_1$, que é armazenada no lugar de A_j, para as próximas iterações.

   Repetimos a ideia para as demais colunas de Q que forem calculadas: uma vez que dispomos de q_i, projetamos a diferença A_j em q_i para todo $j = i + 1, \ldots, n$ (podemos antecipar o cálculo dos r's na linha pois os q's são ortogonais).

   Justificativa: As primeiras colunas de Q que calculamos carregam menos erros numéricos do que as últimas.

   $\Rightarrow$ Esta ideia dá origem a uma modificação simples, porém bastante mais estável, da implementação anterior.

2. Pivoteamento de colunas.

Revisada sem ainda implementar o pivoteamento de colunas.‘

```
function [Q,R] = Fatoracao_QR_Revisada(A)
// Nao inclui pivoteamento de colunas
// Assume rank completo
   [m,n] = size(A)
   R = zeros(n,n)
   Q = zeros(m,n)
   V = A
   //printf("Rows %d, Columns %d \n ",m,n)
   for i = 1:n
       R(i,i) = norm(V(:,i),2)
       Q(:,i) = V(:,i)/R(i,i)
       for j = (i+1):n
           R(i,j) = Q(:,i)'*V(:,j)
           V(:,j) = V(:,j) - R(i,j)*Q(:,i)
       end
   end
endfunction
```

Fatoração $A = QR$ revisada (sem pivoteamento de colunas)

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}.$$

```
-->[Q,R] = Fatoracao_QR_Revisada(A)
 Q  =

   0.09245     0.5392919  -0.1740118   0.2700975
   0.4622502   0.2927584  -0.4524306  -0.8757263
   0.8320503   0.046225   -0.843957    0.3894845
   0.09245    -0.0616334  -0.0957065   0.0170553
   0.2773501  -0.7858253  -0.2088141   0.0903422
 R  =
   10.816654   11.926054    13.035455   14.144855
   0.          1.6641006    3.3282012   4.9923018
   0.          0.           6.380D-15   8.836D-15
   0.          0.           0.          7.599D-16
```

```
-->[q,r] = Fatoracao_QR_Classica(A)
 q  =

   0.09245     0.5392919  -0.5465825  -0.5465825
   0.4622502   0.2927584  -0.370154   -0.370154
   0.8320503   0.046225   -0.2118872  -0.2118872
   0.09245    -0.0616334   0.0397829   0.0397829
   0.2773501  -0.7858253   0.7195517   0.7195517
 r  =

   10.816654   11.926054    13.035455   14.144855
   0.          1.6641006    3.3282012   4.9923018
   0.          0.           3.209D-14  -7.650426
   0.          0.           0.          7.650426
```

Na implentação clássica:

- Na iteração j, referente à coluna A_j, calculamos a projeção $p_j = P_j A_j$, por meio do projetor P_j. Este projetor é calculado como a soma de $j-1$ matrizes de rank 1.
- E na sequência, calculamos a diferença, $v_j = A_j - p_j$, por meio do projetor $(I - P_j)$. Este transformação linear, dada por $I - P_j$, é uma operação de um único projetor de rank $m - (j-1)$.

Na implementação revisada:

- Assim que q_i é conhecida, aplicamos o projetor no espaço ortogonal a $span\{q_i\}$ nas colunas $V(:, j)$ para $j > i$.
- Mais cedo este projetor sendo aplicado nas nas colunas $V(:, j)$ para $j > i$, menor o efeito acumulativo dos erros numéricos.
- Algebricamente, estamos calculando v_j como uma sequência de $j-1$ projeções de rank $m-1$.

- A fatoração $A = QR$ produzida pela nossa implementação (inocente, que assume posto completo), baseada exclusivamente na ideia de Gram-Schmidt, sem a revisão discutida, erradamente sugere que $r(A) = 3$.
- Um refinamento indispensável é o pivoteamento de colunas.
- Assim como na fatoração LU precisamos pivotear linhas para evitar erros numéricos, precisamos pivotear colunas na fatoração QR.
- Assim sendo, usaremos uma matriz de permutações P para representar as trocas de colunas $AP = QR$.

Resultados de algoritmos QR distintos

```
-->[q,r,P] = qr(A)
 q  =
  -0.2666667   0.4777778   0.501608    0.6688727   0.0401987
  -0.5333333   0.1222222   0.4967688  -0.6666392  -0.0971093
  -0.8        -0.2333333  -0.503097    0.2268869  -0.0311653
  -0.0666667  -0.0888889   0.0565821  -0.0650198   0.9900632
   0.         -0.8333333   0.4952798   0.2291204  -0.0880759
 r  =
  -15.  -10.2  -11.8       -13.4
   0.   -3.6   -2.4        -1.2
   0.    0.    -1.617D-15  -8.083D-16
   0.    0.     0.         -2.465D-32
   0.    0.     0.          0.
 P  =
   0.   1.   0.   0.
   0.   0.   1.   0.
   0.   0.   0.   1.
   1.   0.   0.   0.

-->[q,r] = Fatoracao_QR_Classica(A)
 q  =

   0.09245     0.5392919  -0.5465825  -0.5465825
   0.4622502   0.2927584  -0.370154   -0.370154
   0.8320503   0.046225   -0.2118872  -0.2118872
   0.09245    -0.0616334   0.0397829   0.0397829
   0.2773501  -0.7858253   0.7195517   0.7195517
 r  =

   10.816654   11.926054   13.035455   14.144855
   0.          1.6641006   3.3282012   4.9923018
   0.          0.          3.209D-14  -7.650426
   0.          0.          0.          7.650426
```

- Na primeira iteração ($j = 1$)

  A coluna de maior norma de A gera a primeira coluna q_1 de Q.

- Nas demais iterações $j \geq 2$

  Consideramos as colunas $q_1, \ldots, q_{j-1}$ geradas até aquela iteração e fazemos o seguinte:

  - Seja $\mathcal{K}^j \subset \{1, 2, \ldots, n\}$ o conjunto de índices de colunas de A que não foram empregadas para gerar colunas de Q até a iteração j.
  - Para cada coluna $A_k : k \in \mathcal{K}^j$, calculamos p_k, a projeção da coluna A_k em $span\{q_1, \ldots, q_{j-1}\}$ e, na sequência, calculamos a diferença $v_k = A_k - p_k$.
  - A coluna $A_j : j \in \mathcal{K}^j$ que gera a coluna q_j de Q satisfaz:

$$j = \arg\max \|v_k\|_2 : k \in \mathcal{K}^j$$

Atenção: ilustramos a introdução do pivoteamento na implementação clássica pois aqui, vamos usar aritmética de precisão infinita.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}.$$

Primeira coluna:

- A coluna de maior norma Euclideana é A_4, com $\|A_4\| = 15$.
- $r_{11} = 15$, $pivot(1) = 4, pivot(4) = 1$
- $q_1 = \frac{1}{15} \begin{bmatrix} 4 \\ 8 \\ 12 \\ 1 \end{bmatrix}$

Segunda coluna: $j = 2$

- Colunas candidatas:
$$\begin{array}{ccc} A_1 & A_2 & A_3 \\ \hline 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 10 & 11 \\ 1 & 1 & 1 \\ 3 & 2 & 1 \end{array}, \quad q_1 = \frac{1}{15}\begin{bmatrix} 4 \\ 8 \\ 12 \\ 1 \\ 0 \end{bmatrix}$$
- Calculamos a projeção p_k^{j-1} de cada coluna candidata $A_k : k \in \{1, 2, 3\}$ em $span\{q_1\}$.
- Calculamos a diferença $v_k^{j-1} = A_k - p_k^{j-1} : k \in \{1, 2, 3\}$
- Calculamos $\|v_k^{j-1}\| : k \in \{1, 2, 3\}$
- Escolhemos a coluna A_k que maximiza a norma $\|v_k^{j-1}\|$

Segunda coluna: $j = 2$

- Colunas candidatas:
$$\left[\begin{array}{ccc} A_1 & A_2 & A_3 \\ \hline 1 & 2 & 3 \\ 5 & 6 & 7 \\ 9 & 10 & 11 \\ 1 & 1 & 1 \\ 3 & 2 & 1 \end{array}\right], \quad q_1 = \frac{1}{15}\begin{bmatrix} 4 \\ 8 \\ 12 \\ 1 \\ 0 \end{bmatrix}$$
- Produtos internos: $q_1^T A_1 = \frac{153}{15}$, $q_1^T A_2 = \frac{177}{15}$, $q_1^T A_3 = \frac{201}{15}$.
- Projeções: $p_1^1 = \frac{153}{15} q_1$, $p_2^1 = \frac{177}{15} q_1$, $p_3^1 = \frac{201}{15} q_1$
- Diferenças: $v_1^1 = A_1 - \frac{153}{15} q_1$, $v_2^1 = A_2 - \frac{177}{15} q_1$, $v_3^1 = A_3 - \frac{201}{15} q_1$

Segunda coluna: $j = 2$

- $v_1^1 = \frac{1}{225}\begin{bmatrix} -387 \\ -99 \\ 189 \\ 72 \\ 675 \end{bmatrix}$, $v_2^1 = \frac{1}{225}\begin{bmatrix} -162 \\ 126 \\ 414 \\ 72 \\ 450 \end{bmatrix}$, $v_3^1 = \frac{1}{225}\begin{bmatrix} 63 \\ 351 \\ 639 \\ 72 \\ 225 \end{bmatrix}$
- $\|v_1^1\| = \frac{810}{225}$, $\|v_2^1\| < \frac{649}{225}$, $\|v_3^1\| < \frac{769}{225}$
- Logo, q_2 vem de v_1^1, $r_{22} = \frac{810}{225} = 3.6$ $(pivot(2) = 1)$
- $q_2 = v_1^1 : \left(\frac{810}{225}\right) = \frac{1}{810}\begin{bmatrix} -387 \\ -99 \\ 189 \\ 72 \\ 675 \end{bmatrix}$
- Como q_2 veio de A_1, $r_{12} = q_1^T A_1 = \frac{153}{15} = 10.2$

Terceira coluna: $j = 3$

- Candidatas: $\left[\begin{array}{cc} A_2 & A_3 \\ \hline 2 & 3 \\ 6 & 7 \\ 10 & 11 \\ 1 & 1 \\ 2 & 1 \end{array}\right]$, $q_1 = \frac{1}{15}\begin{bmatrix} 4 \\ 8 \\ 12 \\ 1 \\ 0 \end{bmatrix}$, $q_2 = \frac{1}{810}\begin{bmatrix} -387 \\ -99 \\ 189 \\ 72 \\ 675 \end{bmatrix}$
- Produtos internos já calculados: $q_1^T A_2 = \frac{177}{15}$, $q_1^T A_3 = \frac{201}{15}$.
- Novos produtos internos: $q_2^T A_2 = \frac{12}{5}$, $q_2^T A_3 = \frac{6}{5}$
- Projeções: $p_2^2 = (q_1^T A_2)q_1 + (q_2^T A_2)q_2 = \frac{177}{15}q_1 + \frac{12}{5}q_2$, $p_3^2 = (q_1^T A_3)q_1 + (q_2^T A_3)q_2 = \frac{201}{15}q_1 + \frac{6}{5}q_2$
- Diferenças: $v_2^2 = A_2 - p_2^2$ e $v_3^2 = A_3 - p_3^2$

Terceira coluna: $j = 3$

- Projeções: $p_2^2 = (q_1^T A_2)q_1 + (q_2^T A_2)q_2 = \frac{177}{15} q_1 + \frac{12}{5} q_2$, $p_3^2 = (q_1^T A_3)q_1 + (q_2^T A_3)q_2 = \frac{201}{15} q_1 + \frac{6}{5} q_2$
- $p_2^2 = \begin{bmatrix} 2 \\ 6 \\ 10 \\ 1 \\ 2 \end{bmatrix} = A_2 \rightarrow v_2^2 = 0$
- $p_3^2 = \begin{bmatrix} 3 \\ 7 \\ 11 \\ 1 \\ 1 \end{bmatrix} = A_3 \rightarrow v_3^2 = 0$
- Neste momento, sabemos que o $r(A) = 2$ e concluímos a fatoração reduzida.
- As demais colunas na fatoração completa devem ser resolvidas via eliminação.

- A primeira e segunda colunas de Q vieram respectivamente da quarta e primeira colunas de A.
- A terceira e quarta colunas de Q não foram calculadas.
- Veja que $r(A) = 2$ (# elementos não nulos na diagonal de R).
- As linhas de índice $\geq (r(A) + 1)$ de R (tanto na fatoração completa quanto na reduzida) serão nulas.

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} =$$

$$\begin{bmatrix} | & | & | & | \\ q_1 & q_2 & q_3 & q_4 \\ | & | & | & | \end{bmatrix} \begin{bmatrix} 15 & \frac{153}{15} & \frac{177}{15} & \frac{201}{15} \\ 0 & 3.6 & \frac{12}{5} & \frac{6}{5} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

- Na fatoração completa teríamos uma quinta coluna $q_5 \in \mathcal{N}(A^T)$ e $R : 5 \times 4$, com 3 linhas de zeros.

Pivoteamento implementado na versão revisada. Ainda assume rank completo, mas detecta rank numérico.

```
function [Q,R,pivot] = Fatoracao_QR_TrocaColunas(A)
   [m,n] = size(A)
   nainf = norm(A,'inf'); eps = 1.0E-14
   pivot = zeros(n); R = zeros(n,n); Q = zeros(m,n)
   posto = n;
   for i=1:n
       pivot(i) = i
   end
   V = A
   for i = 1:n
       [p,nmax] = DeterminaNormaMaxima(V,i,n)
       if (p <> i) then
           [R,V,pivot] = TrocaConteudoColunas(i,p,R,V,pivot)
       end
       R(i,i) = nmax
       if ((nmax < eps * nainf) & (posto == n)) then
           posto = i-1
       end
       Q(:,i) = V(:,i)/R(i,i)
       for j = (i+1):n
           R(i,j) = Q(:,i)'*V(:,j)
           V(:,j) = V(:,j) - R(i,j)*Q(:,i)
       end
   end
   printf("Rank numérico detectado: %d \n",posto)
endfunction
```

Fatoração $A = QR$ com pivoteamento de colunas

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}.$$

```
-->[Q,R] = Fatoracao_QR_Revisada(A)
 Q  =

   0.09245     0.5392919  -0.1740118   0.2700975
   0.4622502   0.2927584  -0.4524306  -0.8757263
   0.8320503   0.046225   -0.843957    0.3894845
   0.09245    -0.0616334  -0.0957065   0.0170553
   0.2773501  -0.7858253  -0.2088141   0.0903422
 R  =
   10.816654   11.926054    13.035455   14.144855
   0.          1.6641006   3.3282012   4.9923018
   0.          0.          6.380D-15   8.836D-15
   0.          0.          0.          7.599D-16
```

```
[Q,R,pivot] = Fatoracao_QR_TrocaColunas(A)
Rank numérico detectado: 2
 Q  =
   0.2666667  -0.4777778  -0.281048    0.0586473
   0.5333333  -0.1222222  -0.562096    0.6266858
   0.8         0.2333333  -0.7728819  -0.4936518
   0.0666667   0.0888889  -0.0878275   0.1456751
   0.          0.8333333   0.          0.5821615
 R  =
   15.   10.2   11.8         13.4
   0.    3.6    2.4         1.2
   0.    0.     1.580D-15   4.746D-16
   0.    0.     0.          3.814D-16
 pivot  =
   4.
   1.
   2.
   3.
```

- Vamos assumir aqui que a matriz a ser fatorada possui posto completo, $r(A) = n$.
- Ao final do processo de fatoração, devemos ter $I - Q^T Q = 0$.
- Uma medida do efeito de erros numéricos pode ser obtida calculando-se $\|I - Q^T Q\|_2$.
- Idealmente $\|I - Q^T Q\|_2 \approx c\epsilon$, onde c é uma constante pequena e ϵ é a precisão da máquina.

Vamos conduzir o seguinte experimento numérico com a fatoração de uma matriz de Vandermonde retangular:

- Vamos fatorar $V = QR$ para uma matriz de Vandermonde V, retangular $m \times n$, para diversos valores de m, n, usando os três algoritmos que discutimos aqui, além de uma implementação mais sofisticada, existente nos pacotes de Álgebra Linear Numérica, como `Scilab`, `MATLAB`, `NumPy`.

- Os elementos da matriz de Vandermonde considerada são

$$v_{ij} = \left(\frac{j}{n}\right)^{i-1}.$$

- Para cada algoritmo, avaliamos o indicador $\|I - Q^T Q\|_2$ obtido com a matriz Q produzido pelo algoritmo.

A tabela apresenta valores para $\|I - Q^T Q\|_2$ para quatro implementações distintas da fatoração QR de V.

m	n	$\kappa(V)$	Algoritmo empregado			
			QR Clássico	QR Rev	QR Rev + Troca	qr Scilab
6	4	1.066D+02	5.243D-15	4.696D-15	7.226D-15	9.174D-16
9	6	2.752D+03	8.253D-11	1.283D-13	1.363D-13	6.753D-16
12	8	7.280D+04	0.0000026	3.274D-12	2.468D-12	9.491D-16
15	10	1.952D+06	0.1265365	8.151D-11	3.466D-11	6.636D-16
18	12	5.280D+07	2.6409387	1.037D-09	1.669D-09	8.429D-16
25	20	3.243D+14	11.387547	0.0079543	0.0083214	1.314D-15

- $\kappa_p(V)$ é o número de condição de uma matriz retangular, $\kappa := \frac{\max_{x \neq 0} \|Vx\|_p}{\min_{x \neq 0} \|Vx\|_p}$
- O método revisado possui $\|I - Q^T Q\|_2 \approx k_2(V)\epsilon$.
- $\epsilon \approx 1.11 \times 10^{-16}$
- O algoritmo implementado no Scilab utiliza ortogonalização por *refletores de Householder*, uma técnica a ser discutida na sequência.

Se as colunas de Q não forem suficientemente ortogonais, podemos reortogonalizar Q. No exemplo abaixo, isso é feito *a posteriori*, apenas para ilustrar o efeito do processo. Mas na prática, a segunda (ou as múltiplas) ortogonalização(ões) é (são) feita(s) internamente.

```
function [Q,R] = Fatoracao_Revisada_Reortogonalizacao(A)
   [m,n] = size(A)
   [Q,R] = Fatoracao_QR_Revisada(A)
   n2 = norm(eye(n,n)-Q'*Q,2)
   printf("%4.3E \n",n2)
   while (n2 > 100*%eps)
     [Qn,Rn] = Fatoracao_QR_Revisada(Q)
      n2 = norm(eye(n,n)-Qn'*Qn,2)
      printf("%4.3E \n",n2)
      R = Rn*R
   end
endfunction
```

Considerando a fatoração de V de ordem 25×20 anterior:

```
-->V = GeraVandermondeMod(25,20);

-->[Q,R] = Fatoracao_Revisada_Reortogonalizacao(V);
7.954E-03
4.572E-16

-->norm(V-Q*R,'inf')
 ans  =

   1.634D-12
```

1. É uma técnica que usa o conceito de *refletor* (de Householder).

2. Dentre outros propósitos, é empregada para se produzir uma fatoração $A = QR$ bastante estável.

3. Também reúne ideias importantes para o cálculo de autovalores e autovetores, podendo ser empregada em etapas de algoritmos para a decomposição em valores singulares de A.

- Na primeira iteração de Gram-Schmidt (revisada), inicializamos $v_k^1 \leftarrow A_k : k = 1, \ldots, n$, e ortogonalizamos a primeira:

$$\left[\begin{array}{c|c|c|c|c} v_1^1 & v_2^1 & \cdots & v_n^1 \end{array}\right] \begin{bmatrix} \frac{1}{r_{11}} & \frac{-r_{12}}{r_{11}} & \frac{-r_{13}}{r_{11}} & \cdots & \frac{-r_{1n}}{r_{11}} \\ & 1 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c} q_1 & v_2^2 & \cdots & v_n^2 \end{array}\right] \tag{3}$$

lembrando que $v_k^i = A_k - \sum_{l=1}^{i-1} (q_l^T A_k) q_l$

- Na segunda iteração:

$$\left[\begin{array}{c|c|c|c|c} q_1 & v_2^2 & v_3^2 & \cdots & v_n^2 \end{array}\right] \begin{bmatrix} 1 & & & & \\ & \frac{1}{r_{22}} & \frac{-r_{23}}{r_{22}} & \cdots & \frac{-r_{2n}}{r_{22}} \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c} q_1 & q_2 & v_3^3 & \cdots & v_n^3 \end{array}\right] \tag{4}$$

- Ou seja, a cada iteração $i = 1, \ldots, n$ multiplicamos a matriz (parcialmente) ortogonalizada que dispunhamos na iteração anterior, por uma matriz triangular superior R_i.
- Ao final de n iterações:

$$AR_1R_2 \ldots R_n = Q$$

- O produto $R_1R_2 \ldots R_n$ é também triangular superior, não singular (assumindo $r(A) = n$).
- O fator R na fatoração $A = QR$ satisfaz a relação:

$$R^{-1} = R_1R_2 \ldots R_n$$

- Ou seja, Gram-Schmidt faz uma ortogonalização triangular, isto é, ortogonaliza as colunas de A, por meio da multiplicação de A por matrizes R_i triangulares superiores, convenientemente escolhidas.

- Gram-Schmidt faz ortogonalização triangular: usa matrizes triangulares superiores, multiplicando à direita de A, para obter uma matriz Q ortogonal.

- Householder faz uma triangularização ortogonal: usa matrizes ortogonais (construídas por meio de refletores), pré-multiplicando A, para transformar A em uma matriz triangular superior.

- A ideia é construir matrizes ortogonais Q_k (unitárias, no caso complexo) tais que $Q_n Q_{n-1} \dots Q_2 Q_1 A = R$, onde R é uma matriz triangular superior.

- Recordando: o produto de matrizes ortogonais é uma matriz ortogonal, portanto $A = Q_1^T \dots Q_n^T R$ é uma fatoração QR de A (ortogonal por triangular superior).

- Veja o efeito desejado de Q_1, Q_2, $Q_3, \dots$

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{Q_1} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{Q_2} \begin{bmatrix} x & x & x \\ & x & x \\ & 0 & x \\ & 0 & x \\ & 0 & x \end{bmatrix} \xrightarrow{Q_3} \begin{bmatrix} x & x & x \\ & x & x \\ & & x \\ & & 0 \\ & & 0 \end{bmatrix}$$

- A $k-$ésima matriz Q_k é escolhida para introduzir zeros na $k-$ésima coluna de $Q_{k-1} \ldots Q_1 A$, preservando os zeros gerados nas colunas de índice igual ou inferior a $k-1$, nas iterações anteriores.

$$\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} \xrightarrow{Q_1} \begin{bmatrix} x & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \\ 0 & x & x \end{bmatrix} \xrightarrow{Q_2} \begin{bmatrix} x & x & x \\ & x & x \\ & 0 & x \\ & 0 & x \\ & 0 & x \end{bmatrix} \xrightarrow{Q_3} \begin{bmatrix} x & x & x \\ & x & x \\ & & x \\ & & 0 \\ & & 0 \end{bmatrix}$$

Refletor de Householder

Dado um vetor $v \in \mathbb{R}^n$ e $u = \frac{v}{\|v\|}$ sua normalização, a matriz

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

é chamada de refletor de Householder.

Propriedades de F:

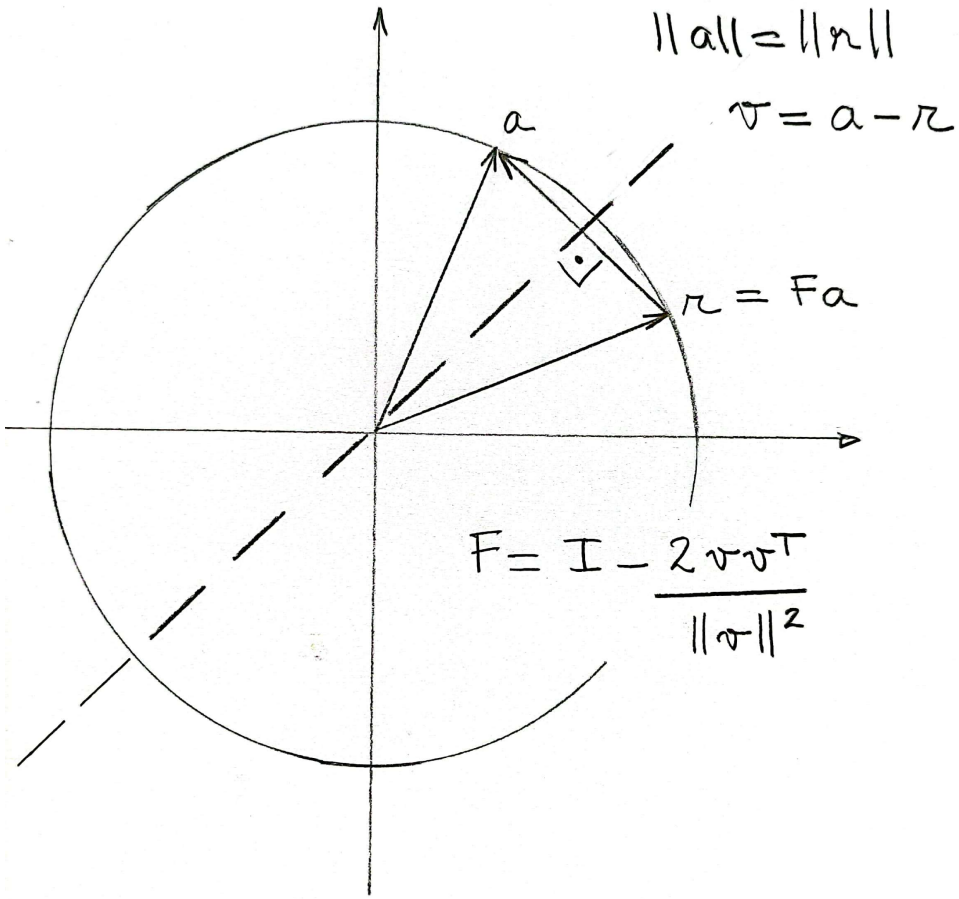
- Simétrica.
- Ortogonal:

$$\begin{aligned}(I - 2uu^T)^T(I - 2uu^T) &= I - 4uu^T + (2uu^T)^T(2uu^T) \\ &= I - 4uu^T + 4uu^T uu^T \\ &= I\end{aligned}$$

- Não singular.

Dados dois vetores $a, r \in \mathbb{R}^n$ tais que $\|a\| = \|r\|$ e $v = a - r$, a matriz $F = I - 2\frac{vv^T}{\|v\|^2}$ aplicada em a satisfaz:

$$Fa = r$$



$$
\begin{aligned}
Fa &= \left(I - 2\frac{(a-r)(a-r)^T}{(a-r)^T(a-r)} \right) a \\
&= a - 2\frac{(a-r)(a^Ta - r^Ta)}{(a-r)^T(a-r)} \\
&= a - 2\frac{(a-r)(a^Ta - r^Ta)}{a^Ta - 2r^Ta + r^Tr} \\
&= a - 2\frac{(a-r)(a^Ta - r^Ta)}{2a^Ta - 2r^Ta} \\
&= a - 2\frac{(a-r)(a^Ta - r^Ta)}{2(a^Ta - r^Ta)} \\
&= r
\end{aligned}
$$

Consequência: como $F = F^T$ e $F^TF = I$, temos que $F^T(Fa) = F^Tr = Fr$, logo $a = Fr$ (reflexão).

- Invariante: No início da iteração k, há um bloco de vetores de zeros nas colunas $k-1$ anteriores.

- Na iteração k, a aplicação de Q_k em $Q_{k-1}\ldots Q_1 A$, combina as linhas de $Q_{k-1}\ldots Q_1 A$. A combinação linear das entradas zero permanece zero.

- Para se garantir esta propriedade (preservar as entradas zero já criadas nas colunas de índice $k-1$ ou menor), escolhemos a matriz Q_k ortogonal (ou unitária), cujo particionamento em blocos é:

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

onde I é a matriz identidade de ordem $k-1$ e F é uma matriz ortogonal de ordem $m-(k-1)$.

- O refletor F é responsável por produzir zeros nas posições corretas da coluna k.

- Vamos assumir que no início da iteração k, as entradas nas linhas $k, k+1, \ldots, m$ da coluna k de $Q_{k-1} \ldots Q_1 A$ sejam respresentadas pelo vetor $x \in \mathbb{R}^{m-(k-1)}$.

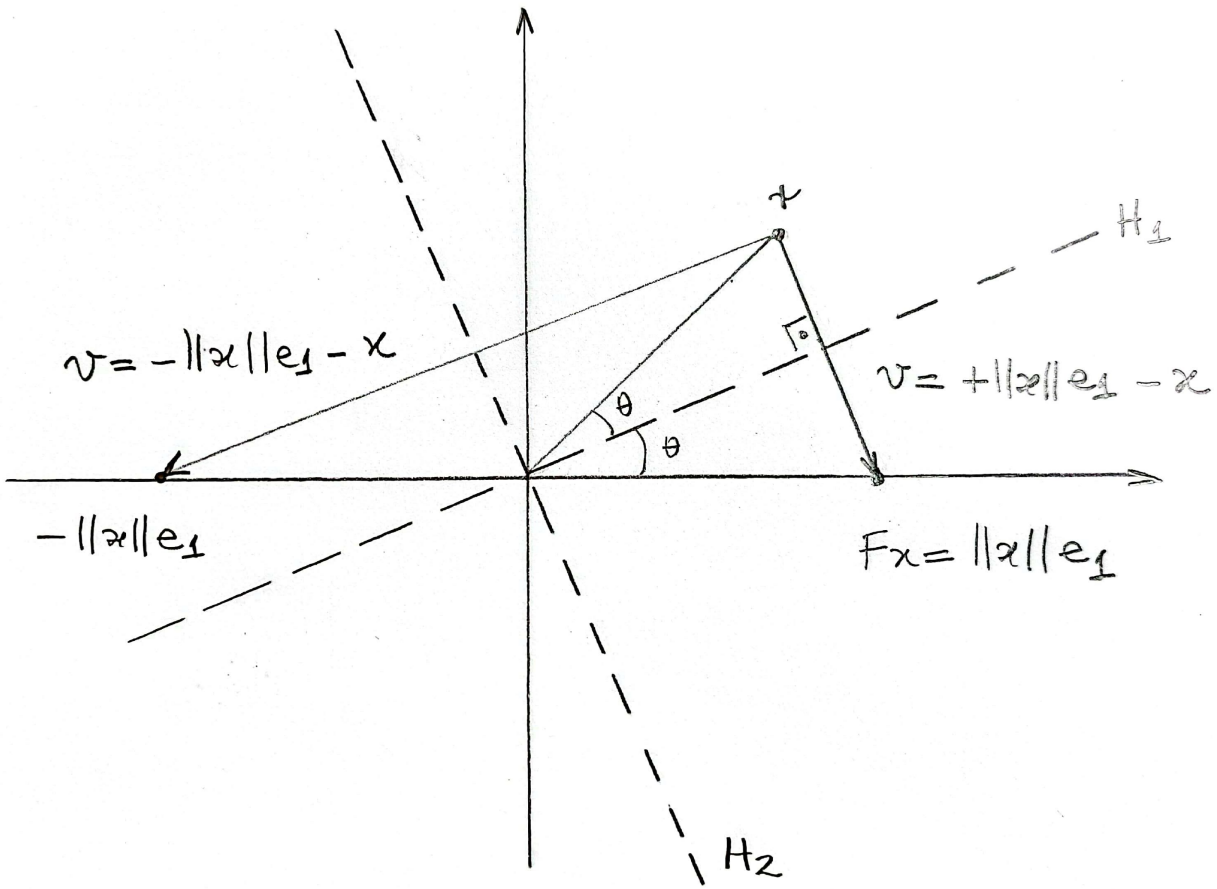
- O refletor F aplicado em x deve produzir a transformação linear:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{m-(k-1)} \end{bmatrix} \xrightarrow{F} Fx = \begin{bmatrix} \|x\| \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \|x\| e_1$$

onde e_1 é um vetor de zeros, exceto pela primeira posição que é 1.

Interpretação geométrica da ação de F em x

- Na forma como definimos, F reflete o espaço $\mathbb{R}^{m-(k-1)}$ em torno do hiperplano H_1.
- O vetor $v = \|x\|e_1 - x$ é perpendicular ao hiperplano H.
- Porém, há mais de uma escolha para F. Veja o subespaço H_2.



Vamos considerar a figura anterior e a escolha de F tal que $Fx = +\|x\|e_1$.

- O vetor $v = +\|x\|e_1 - x$ é perpendicular ao subespaço H_1 (H_1 é um hiperplano que passa pela origem, por isso é um subespaço).
- Veja que, a partir de x, o ponto em que o vetor v intercepta H_1 está na metade do segmento até o ponto Fx, resultado da transformação linear por x.
- Sabemos que se desejamos projetar um vetor qualquer y em $span\{v\}^{\perp}$ (isto é, no subespaço definido por H_1) precisamos aplicar o projetor

$$P = \left(I - \frac{vv^T}{v^T v}\right),$$

em y, uma vez que o projetor $\frac{vv^T}{v^T v}$ projeta y em $span\{v\}$.

Vamos considerar a figura anterior e a escolha de F tal que $Fx = +\|x\|e_1$.

- Se aplicarmos o deslocamento $\left(I - \frac{vv^T}{v^T v}\right)$ em x, obtemos o ponto $\left(I - \frac{vv^T}{v^T v}\right)x$ que é sua projeção em H_1 e está na metade do segmento que nos leva até o ponto $+\|x\|e_1$ desejado.
- Portanto, para obtermos o ponto $Fx = +\|x\|e_1$, devemos aplicar em x a matriz $\left(I - 2\frac{vv^T}{v^T v}\right)$.
- Por esta razão, o refletor de Householder é $F = \left(I - 2\frac{vv^T}{v^T v}\right)$.
- Já mostramos que $F^T F = I$. Logo $Q_k^T Q_k = I$.

$$A = \begin{bmatrix} -4 & 1 & 1 \\ 2 & 1 & -1 \\ 4 & 1 & 1 \end{bmatrix}.$$

Primeira iteração:

- $x = \begin{bmatrix} -4 & 2 & 4 \end{bmatrix}^T$, $\|x\| = 6$.
- $v^T = \begin{bmatrix} 6 & 0 & 0 \end{bmatrix}^T - \begin{bmatrix} -4 & 2 & 4 \end{bmatrix}^T = \begin{bmatrix} 10 & -2 & -4 \end{bmatrix}^T$, $v^T v = 120$.
- $F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{2}{120} \begin{bmatrix} 10 \\ -2 \\ -4 \end{bmatrix} \begin{bmatrix} 10 & -2 & -4 \end{bmatrix}$

$$F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \frac{2}{120} \begin{bmatrix} 100 & -20 & -40 \\ -20 & 4 & 8 \\ -40 & 8 & 16 \end{bmatrix}$$

$$F = \begin{bmatrix} -2/3 & 1/3 & 2/3 \\ 1/3 & 14/15 & -2/15 \\ 2/3 & -2/15 & 11/15 \end{bmatrix}$$

Primeira iteração (continua):

- $FA = \begin{bmatrix} -2/3 & 1/3 & 2/3 \\ 1/3 & 14/15 & -2/15 \\ 2/3 & -2/15 & 11/15 \end{bmatrix} \begin{bmatrix} -4 & 1 & 1 \\ 2 & 1 & -1 \\ 4 & 1 & 1 \end{bmatrix} =$
  $\begin{bmatrix} 6 & 1/3 & -1/3 \\ 0 & 17/15 & -11/15 \\ 0 & 19/15 & 23/15 \end{bmatrix}$

- Para a primeira iteração, temos $Q_1 = F$.

Segunda iteração:

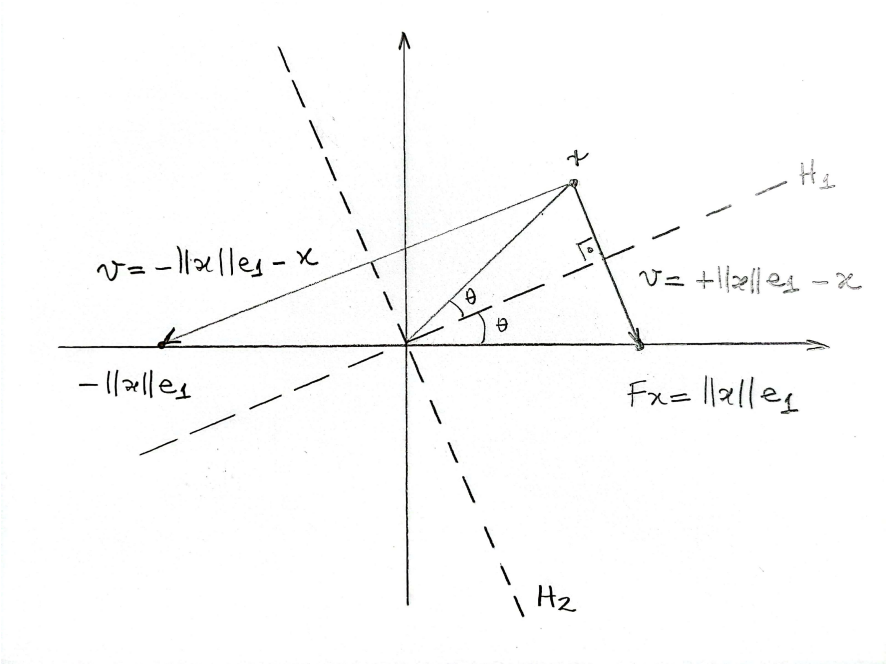
- $Q_1 A = \begin{bmatrix} 6 & 1/3 & -1/3 \\ 0 & 17/15 & -11/15 \\ 0 & 19/15 & 23/15 \end{bmatrix}$
- Q_2 tem a forma $\left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & F_2 \end{array}\right]$, onde $F_2 \in \mathbb{R}^{2\times 2}$ é o segundo refletor.
- F_2 será contruído a partir de $x = \begin{bmatrix} 17/15 & 19/15 \end{bmatrix}^T$, $\|x\| = \frac{\sqrt{26}}{3}$.
- $v = \frac{\sqrt{26}}{3}\begin{bmatrix} 1 & 0 \end{bmatrix}^T - \begin{bmatrix} 17/15 & 19/15 \end{bmatrix}^T = \begin{bmatrix} \frac{5\sqrt{26}-17}{15} & -\frac{19}{15} \end{bmatrix}^T$, $v^T v \approx 1.9251853$.
- $F_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - 2/1.9251853 \begin{bmatrix} \frac{5\sqrt{26}-17}{15} \\ -\frac{19}{15} \end{bmatrix} \begin{bmatrix} \frac{5\sqrt{26}-17}{15} & -\frac{19}{15} \end{bmatrix}$

  $F_2 = \begin{bmatrix} 0.6667949 & 0.7452413 \\ 0.7452413 & -0.6667949 \end{bmatrix}$

Segunda iteração (continua):

- $Q_2 Q_1 A = R = \begin{bmatrix} 6 & 1/3 & -1/3 \\ 0 & 1.6996732 & 0.6537205 \\ 0 & 0 & -1.5689291 \end{bmatrix}$
- $Q_2 Q_1 = \begin{bmatrix} -0.6666667 & 0.3333333 & 0.6666667 \\ 0.7190925 & 0.5229764 & 0.4576043 \\ -0.1961161 & 0.7844645 & -0.5883484 \end{bmatrix}$
- Fatoração QR resultante: $A = (Q_2 Q_1)^T R$
- Veja que não dispomos explicitamente do fator $(Q_2 Q_1)^T$.

- e $A \in \mathbb{C}^{m\times n}$, temos inúmeras escolhas de pontos de reflexão: pode ser qualquer $z\|x\|e_1$ para um complexo $z \in \mathbb{C}, |z| = 1$.
- No caso de $A \in \mathbb{R}^{m\times n}$, temos duas escolhas. Veja abaixo:



- Para aumentar a estabilidade numérica, devemos escolher o ponto de reflexão que promove o máximo deslocamento, isto é, $z = -sinal(x_1)$ ($sinal(x_1)$ retorna -1 se $x_1 < 0$ ou 1, cc).
- Na figura: escolhemos $-\|x\|e_1$ e a direção $v = -\|x\|e_1 - x$.

Escolhendo o ponto de reflexão

- Opção 1: $v = \|x\|e_1 - x$
- Opção 2: $v = -\|x\|e_1 - x$
- A escolha mais estável: $v = -sinal(x_1)\|x\|e_1 - x$, que para efeito do cálculo de F é equivalente a $sinal(x_1)\|x\| + x$

```
function [Q,R] = QR_Householder(A)
    [m,n] = size(A)
    R = A
    P = eye(m,m)
    for k = 1:n
        x = R(k:m,k)

        vk = sign(x(1))*norm(x,2) * eye(m-k+1,1) + x
        vk = 1.0 / norm(vk,2) * vk

        R(k:m,k:n) = R(k:m,k:n)-2.0*vk*(vk'*R(k:m,k:n))

        Pa = eye(m,m)
        Pa(k:m,k:m) = eye(m-k+1,m-k+1)-2.0*vk*vk'*Pa(k:m,k:m)
        P = Pa*P

    end
    Q = P'
endfunction
```

1. Vimos que para resolver

$$\min \|Ax - b\|,$$

podemos formular e resolver o sistema de equações normais:

$$A^T A\hat{x} = A^T b$$

que nos fornece o vetor $\hat{x}$ tal que $p = A\hat{x}$ está em $\mathcal{C}(A)$ e $\hat{x}$ é o vetor de parâmetros que melhor explica os dados.

2. A matriz $A^T A$ é usualmente malcondicionada e seu cálculo deve ser evitado.

3. Podemos usar a fatoração QR de A para resolver o problema, sem explicitamente calcular $A^T A$.

Supondo que a fatoração $A = QR$ (reduzida, R é quadrada de ordem $r(A)$) seja disponível e que $P = A(A^TA)^{-1}A^T$ denote o projetor ortogonal em $\mathcal{C}(A)$. Assumimos que as colunas *ld* de A foram removidas.

$$\begin{aligned} P &= A(A^TA)^{-1}A^T \\ &= (QR)(R^TQ^TQR)^{-1}(R^TQ^T) \\ &= (QR)(R^TR)^{-1}(R^TQ^T) \\ &= (QR)(R^{-1}R^{-T})(R^TQ^T) \\ &= QQ^T \end{aligned}$$

1. Com o projetor calculamos $p = Pb$.
2. Como $p \in \mathcal{C}(A)$, o sistema linear $Ax = p$ possui uma solução.
3. Combinando temos: $Ax = Pb \rightarrow QRx = Pb = QQ^T b$.
4. Então, mais precisamente:
    1. Calculamos a fatoração $A = QR$ (reduzida), calculamos o vetor $Q^T b$ (se dispomos de Q explícita)
    2. Ou, se não desejamos armazenar Q explicitamente (alguma implementação de Householder que por escolha não armazene Q), fatoramos $[A|b]$ na forma QR. Ao final, obtemos a matriz $[R|Q^T b]$.
    3. Resolvemos o sistema triangular superior $Rx = Q^T b$, para obter o vetor de parâmetros ótimo x da regressão linear.

$$b = \alpha x^{\beta} \rightarrow \ln b = \ln \alpha + \beta \ln x.$$

Velocidade (m/s)	10	20	30	40	50	60	70	80
Força (N)	25	70	380	550	610	1220	830	1450

A matriz $[A|b]$ correspondente à linearização é:

```
Ab  =
    1.    2.3025851    3.2188758
    1.    2.9957323    4.2484952
    1.    3.4011974    5.9401713
    1.    3.6888795    6.3099183
    1.    3.912023     6.413459
    1.    4.0943446    7.1066061
    1.    4.2484952    6.7214257
    1.    4.3820266    7.2793188
```

Exemplo: Ajuste força vs velocidade

- Resolvendo via Fatoração de Householder.
- Fatoramos a matriz $[A|b]$ acima, obtendo Q, R.
- A terceira coluna da matriz R fornecida é, na verdade $Q^T b$.
- Resolvemos $Rx = Q^T b$ e obtemos $\ln(\alpha), \beta$.

```
[Q,R] = QR_Householder(Ab);
x = inv(R(1:2,1:2))*R(1:2,3)
x  =
  -1.294126
   1.9841763
```

Então $\alpha = e^{-1.294126} = 0.2741373$