Factorisation QR

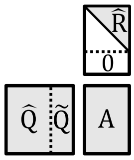
AnaNum – Chapitre 4

I. Factorisation QR

1. Les matrices orthogonales

$$Q \text{ orthogonale} \quad \Leftrightarrow \quad Q^TQ = QQ^T = I \quad \Leftrightarrow \quad Q^T = Q^{-1}$$

2. Principe



$$A = QR = \hat{Q}\hat{R} \qquad \begin{array}{l} Q \text{ orthogonale} \\ R \text{ tri. sup.} \end{array}$$

$$Q^TQ = QQ^T = \hat{Q}^T\hat{Q} = I \quad \hat{Q}\hat{Q}^T \neq I$$

a. Matrice de Householder

$$\boxed{H_v = I - \frac{2}{\underbrace{v^Tv}_{\beta}} vv^T} \qquad \text{Orthogonale, symétrique, } H_v = K_{\gamma v} \ (\gamma \in \mathbb{R}^*)$$

b. Utilisation pour QR

$$H_v x = -\alpha e_1 \quad \Rightarrow \quad \boxed{v = x + \alpha e_1} \quad \text{et} \quad \boxed{\alpha = \pm\|x\|} \quad \begin{pmatrix} \text{US :} & -\|x\| \\ \text{Fr :} & signe(x_1)\ \|x\| \end{pmatrix}$$

Note : Pour avoir $v_1 = 1$ ou peut prendre $v = v/v_1$

c. Calcul de QR

$$H_p \dots H_1 A = R \quad \Rightarrow \quad \boxed{A = \underbrace{(H_1 \dots H_p)^T}_{Q} R = QR}$$

3. Algorithmes

a. Calcul du vecteur de House

```
fonction [v, β] = vecteurDeHouse(x)

norm_x = x'*x
v = x                              % v = x + αe_1
v(1) = v(1) + ± √norm_x
β = 2 v(1)^2 / (norm_x + v(1)^2 − x(1)^2 )
v = v/v(1)
```

b. Produit House-vecteur

$$H_v x = \left(I - \frac{2}{v^Tv} vv^T\right) x = x - \frac{2}{\underbrace{v^Tv}_{\beta}} \underbrace{v^Tx}_{w} \ v$$

```
fonction x = House_vecteur(x, v, β)

w = v' * x
x = x − β * w * v
```

c. Transformation d'une matrice

```
fonction [A, β] = House_matrice(A)

[v, β] = vecteurDeHouse(A(:,1))   % calcul du vecteur de House
pour j = 1 à p faire              % calcul des produits House-vecteur
    A(:,j) = House_vecteur(A(:,j), v, β)
finpour
A(2:end,1) = v(2:end)             % stockage de v (sauf v(1) = 1)
```

d. Factorisation QR

```
fonction A = qr(A)

pour j = 1 à p faire
     [A(j:n, j:p), β] = House_matrice(A)
finpour
% A contient R et les vecteurs v
```

e. Calcul Qb

```
fonction b = Qb(A, b)

pour j = 1 à p faire
     % initialisation v
     v(j) = 1; v(j+1:n) = A(j+1:n,j)
     b(j:n) = b(j:n) - β_j * v * v' * b(j:n)
finpour
```

II. Les moindres carrés pour résoudre $Bx = d$

1. Définition et condition d'optimalité

$$\min_x \underbrace{\|Bx - d\|^2 = x^T Ax - 2x^T b}_{J(x)} \Leftrightarrow \boxed{\nabla_x J(x) = 2Ax - 2b = 0} \Leftrightarrow \boxed{Ax = b} \qquad \underline{A = B^T B} \quad \underline{b = B^T d}$$

Formules de gradient : $\nabla_x(x^T a) = a \qquad \nabla_x(x^T Ax) = Ax + A^T x$

2. Utilisation de QR

$$B = QR \qquad A = R^T R \quad \Leftrightarrow \quad Ax - b = \hat{R}^T\left(\hat{R}x - \hat{Q}^T d\right) = 0 \quad \Leftrightarrow \quad \boxed{x = \hat{R}^{-1} \underbrace{\hat{Q}^T d}_{z}}$$

```
fonction x = moindres_carres_chol(B, d)

A = B'*B
B = A'*d
L = chol(A)
z = tri_inf(L, b)
x = tri_sup(L', y)
```

```
fonction x = moindres_carres_qr(B, d)

RetV = qr(B)
z = qr_b(RetV)
x = tri_sup(RetV, b)
```

Note : QR permet d'éviter que A ne soit pas définie positive si B n'est pas de plein rang