Unit 3: Gram - Schmidt orthogonalization

Given a set of linearly independent vectors $\{a_0 \dots a_{n-1}\}$, the Gram-Schmidt process computes an orthonormal $\{q_0, \dots q_{n-1}\}$ that spans the same subspaces as the original vectors

$$\text{Span}(\{a_0, \dots a_{n-1}\}) = \text{Span}(\{q_0 \dots q_{n-1}\})$$

The process as follows:

1. Compute unit length vector q_0
   - Compute $e_{0,0} = \|a_0\|_2$
   - Compute $q_0 = a_0 / e_{0,0}$
2. Compute the component orthogonal to q_0 ($a_1^\perp$ of q_1)
   - Compute $e_{0,1} = q_0^H a_1$
   - Compute $a_1^\perp = a_1 - e_{0,1} q_0$
3. Compute unit length vector q_1
   - Compute $e_{1,1} = \|a_1^\perp\|_2$
   - Compute $q_1 = a_1^\perp / e_{1,1}$
4. Continue with $a_2, a_3, \dots$

Another way of looking at it

$$a_1 - e_{0,1} q_0 = a_1 - q_0^H a_1 q_0 = \underbrace{(I - q_0 q_0^H)}_{\text{matrix that is perpendicular to subspace spanned by } q_0} a_1$$

And another way of looking at it

We have: $\text{Span}(\{a_0\}) = \text{Span}(\{q_0\})$

$\text{Span}(\{a_0, a_1\}) = \text{Span}(\{q_0, q_1\})$

$\vdots$ etc

$\text{Span}(\{a_0, \dots a_{k-1}, a_k\}) = \text{Span}(\{q_0, \dots q_{k-1}, q_k\})$

We can conclude that:

(1) $a_k = e_{0,k} q_0 + e_{1,k} q_1 + \dots e_{k-1,k} q_{k-1} + e_{k,k} q_k$

If we calculate $q_0^H a_k$, we'd have:

$q_0^H a_k = e_{0,k} q_0^H q_0 + e_{1,k} q_0^H q_1 + \dots e_{k-1,k} q_0^H q_{k-1} + e_{k,k} q_0^H q_k$

$= e_{0,k}(1) + 0 + \dots 0 + 0$

Therefore, we conclude that:

(1a) $e_{i,k} = q_i^H a_k$

And if we transform (1), we can see that

(2) $\underbrace{a_k - e_{0,k} q_0 - e_{1,k} q_1 - \dots e_{k-1,k} q_{k-1}}_{a_k^\perp} = e_{k,k} q_k$

QR factorization

$$(a_0 | a_1 | \dots | a_k | \dots | a_{n-1}) = (q_0 | \dots | q_k | \dots | q_{n-1}) \cdot \begin{pmatrix} e_{0,0} & \dots & e_{0,k} & \dots & e_{0,n-1} \\ & \ddots & \vdots & & \vdots \\ & & e_{k,k} & \dots & e_{k,n-1} \\ & & & \ddots & \vdots \\ & & & & e_{n-1,n-1} \end{pmatrix}$$

QR algorithm (CGS)

while $n(A_L) < n(A)$

$(A_L | A_R) \rightarrow (A_0 | a_1 | A_2)$, $(Q_L | Q_R) \rightarrow (Q_0 | q_1 | Q_2)$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array}\right)$$

$r_{01} := Q_0^H a_1$

$a_1^\perp := a_1 - Q_0 r_{01}$

$\rho_{11} := \| a_1^\perp \|_2$

$q_1 := a_1^\perp / \| a_1^\perp \|_2 = \rho_{11}$

$(A_L | A_R) \leftarrow (A_0\ a_1 | A_2)$, $(Q_L | Q_R) \leftarrow (Q_0\ q_1 | Q_2)$

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array}\right)$$

end while

Pseudo code for QR factorization

Directly translate from the algorithm

```
for k := 0 ... n-1
    a_k^⊥ = a_k
    for i := 0 ... k-1
        e_{i,k} = q_i^H a_k^⊥
        a_k^⊥ -= e_{i,k} q_i
    end
    e_{k,k} = ||a_k^⊥||_2
    q_k = a_k^⊥ / e_{k,k}
end
```

Same code but now we store the result of Q directly on A : (Modified Gram Schmidt)

```
for k := 0 ... n-1
    for i := 0 ... k-1
        e_{i,k} = a_i^H a_k
        a_k -= e_{i,k} a_i
    end
    e_{k,k} = ||a_k||_2
    a_k = a_k / e_{k,k}
```

MGS

Variation that eagerly update the remaining vectors $\{a_{k+1}, \dots a_{n-1}\}$ right after computing a_k (or q_k, since q_k is overwrite a_k)

```
for k = 0 ... n-1
    e_{k,k} = ||a_k||_2
    a_k = a_k / e_{k,k}
    for j = k+1 ... n-1
        e_{k,j} = a_k^H a_j
        a_j -= e_{k,j} a_k
    end
end
```

or we can think of this as vector-matrix multiplication

```
for k = 0 ... n-1
    e_{k,k} = ||a_k||_2
    a_k = a_k / e_{k,k}
    (e_{k,k+1} | ... | e_{k,n-1}) = a_k^H . (a_{k+1} | ... | a_{n-1})
    (a_{k+1} | ... | a_{n-1}) = (a_{k+1} - e_{k,k+1} a_k | ... | a_{n-1} - e_{k,n-1} a_k)
end
```


3.2.4.2

$A \to (A_L \mid A_R)$, $Q \to (Q_L \mid Q_R)$, $R \to \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right)$

We have these properties:

- $A_L = Q_L R_{TL}$
- $A_R = Q_L R_{TR} + Q_R R_{BR}$
- $C(A_L) = C(Q_L)$
- $R_{TR} = Q_L^H A_R$
- $(A_R - Q_L R_{TR})^H Q_L = 0$
- $C(A_R - Q_L R_{TR}) = C(Q_R)$
- $A_R - Q_L R_{TR} = Q_R R_{BR}$

Letting $\hat{A}$ denote the original contents of A, at a typical point:

- A_L has been updated with Q_L
- R_{TL} and R_{TR} have been computed
- $A_R = \hat{A}_R - Q_L R_{TR}$

Gram Schmidt

$A \to (A_L \mid A_R)$, $R \to \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right)$

while $n(A_L) < n(A)$

$(A_L \mid A_R) \to (A_0 \mid a_1 \mid A_2)$, $\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right) \to \left(\begin{array}{c|cc} R_{00} & r_{01} & R_{02} \\ \hline 0 & \epsilon_{11} & r_{12}^T \\ 0 & 0 & R_{22} \end{array}\right)$

CGS

$r_{01} = A_0^H a_1$

$a_1 = a_1 - A_0 r_{01}$

$\epsilon_{11} = \|a_1\|_2$

$a_1 = a_1 / \epsilon_{11}$

MGS

$[a_1, r_{01}] = \text{Proj} \perp_{to} \mathcal{C}(A_0 a_1)$

$\epsilon_{11} = \|a_1\|_2$

$a_1 = a_1 / \epsilon_{11}$

MGS (alternative)

$\epsilon_{11} = \|a_1\|_2$

$a_1 = a_1 / \epsilon_{11}$

$r_{12}^T = a_1^H A_2$

$A_2 = A_2 - a_1 r_{12}^T$

$(A_L \mid A_R) \leftarrow (A_0 \mid a_1 \mid A_2)$, $\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline 0 & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|cc} R_{00} & r_{01} & R_{02} \\ \hline 0 & \epsilon_{11} & r_{12}^T \\ 0 & 0 & R_{22} \end{array}\right)$

Thm 3.2.6.1

CGS cost $2mn^2$ flops with $A \in \mathbb{C}^{m \times n}$

Thm 3.2.6.2

MGS cost $2mn^2$ flops with $A \in \mathbb{C}^{m \times n}$

Using unitary matrices (Householder QR fac..)

We can essentially avoid errors if $A = QR$ so that Q is unitary matrice.

Householder QR factorization (motivation)

Given, $A \in \mathbb{C}^{m \times n}$, one could find a sequence of unitary matrices so that:

$H_{n-1} \cdots H_0 A = \begin{pmatrix} R \\ 0 \end{pmatrix}$, R is upper triangular

Then:

$A = \underbrace{H_0^H \cdots H_{n-1}^H}_{Q} \begin{pmatrix} R \\ 0 \end{pmatrix}$

Thm 3.3.1.1

If $A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$ and $Q = (Q_L \mid Q_R)$

$\Rightarrow A = Q_L R$

$= H_0^H \cdots H_k^H \begin{pmatrix} R \\ 0 \end{pmatrix}$

Householder transformation

Let $u \in \mathbb{C}^n$ be a vector of unit length ($\|u\|_2 = 1$).
Then:

$$H = I - 2uu^H$$

is said to be a Householder transformation or reflector

Hw 3.3.2.1

- $HH = I$ (reflecting a reflection ~~is~~ return the original vector)
- $H = H^H$
- $H^H H = HH^H = I$ (reflector is unitary)

In the other way around

Give 2 vector x, y. find the subspace that reflect x to y.

In other words, find u so that:

$$(I - 2uu^H)x = y$$

First, find v so that $v = x - y$, then the vector $u = v/\|v\|_2$

Compute QR factorization via Householder

We need to find H_0 that ~~map~~ maps x to a multiple of first unit basis vector (e_0)

$$H_0 \begin{pmatrix} a_0 & a_1 & \cdots \end{pmatrix} = \begin{pmatrix} H_0 a_0 & H_0 a_1 & \cdots \end{pmatrix}$$

$$\Leftrightarrow H_0 \left(\begin{pmatrix} \times \\ \times \\ \times \\ \times \end{pmatrix} \middle| \begin{pmatrix} \times \\ \times \\ \times \\ \times \end{pmatrix} \middle| \cdots \right) = \left(\begin{pmatrix} \times \\ 0 \\ 0 \\ 0 \end{pmatrix} \middle| \begin{pmatrix} \times \\ \times \\ \otimes \\ \otimes \end{pmatrix} \middle| \cdots \right)$$

x ; $y = \beta e_0$; $\|x\| = \|\beta\|$ since H_0 preserve length

To do that, we find $v = x - y = x - \beta e_0$

$$= x \mp \|x\|_2 e_0$$

In practice, we choose

$$v = x + \text{sign}(x_0)\|x\|_2 e_0$$

to avoid catastrophic cancellation

So we can conclude that solving this will get us the H matrix we need:

$$Hx = \mp \|x\|_2 e_0$$

where $v = x \mp \|x\|_2 e_0$

Householder QR factorization algorithm

Intuition:

$$\cdots \underbrace{\left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & I - \frac{u_1 u_1^H}{\tau_1} \end{array}\right)}_{H_1} \underbrace{\left(I - \frac{u_0 u_0^H}{\tau_0}\right)}_{H_0} A = \underbrace{\left(\begin{array}{ccc} \times & \times & \times \\ 0 & \times & \times \\ 0 & 0 & \times \\ 0 & 0 & 0 \end{array}\right)}_{R}$$

Hw 3.3.4.1

$$\left(\begin{array}{c|c} I & 0 \\ \hline 0 & I - \frac{1}{\tau}\begin{pmatrix} 1 \\ u_{21} \end{pmatrix}\begin{pmatrix} 1 \\ u_{21} \end{pmatrix}^H \end{array}\right) = I - \frac{1}{\tau}\begin{pmatrix} 0 \\ 1 \\ u_{21} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ u_{21} \end{pmatrix}^H$$

Hw 3.3.4.2

Cost of Householder QR factorization is:

$$2mn^2 - \frac{2}{3}n^3 \text{ flops}$$

Algorithm

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$ and $t \rightarrow \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix}$

A_{TL} is 0×0 and t_T has 0 elements

while $n(A_{BR}) > 0$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{cc|c} A_{00} & a_{01} & A_{02} \\ a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \text{ and } \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix} \rightarrow \begin{pmatrix} t_0 \\ \hline \tau_1 \\ t_2 \end{pmatrix}$$

$$\left[\begin{pmatrix} \alpha_{11} \\ a_{21} \end{pmatrix}, \tau_1\right] := \left[\begin{pmatrix} \rho_{11} \\ u_{21} \end{pmatrix}, \tau_1\right] = \text{Housev}\begin{pmatrix} \alpha_{11} \\ a_{21} \end{pmatrix}$$

$$\text{Update } \begin{pmatrix} a_{12}^T \\ A_{22} \end{pmatrix} := \left(I - \frac{1}{\tau_1}\begin{pmatrix} 1 \\ u_{21} \end{pmatrix}\begin{pmatrix} 1 \\ u_{21} \end{pmatrix}^H\right)\begin{pmatrix} a_{12}^T \\ A_{22} \end{pmatrix}$$

via the steps

$$w_{12}^T := \left(a_{12}^T + u_{21}^H A_{22}\right)/\tau_1$$

$$\begin{pmatrix} a_{12}^T \\ A_{22} \end{pmatrix} := \begin{pmatrix} a_{12}^T - w_{12}^T \\ A_{22} - a_{21} w_{12}^T \end{pmatrix}$$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{cc|c} A_{00} & a_{01} & A_{02} \\ a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \text{ and } \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix} \leftarrow \begin{pmatrix} t_0 \\ \hline \tau_1 \\ t_2 \end{pmatrix}$$

endwhile

Form Q Algorithm

$$A \to \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right), \quad t \to \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix}$$

A_{TL} is $n(A) \times n(A)$ and t_T has $n(A)$ elements

while $n(A_{TL}) > 0$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \to \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \quad \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix} \to \begin{pmatrix} t_0 \\ \hline t_1 \\ \hline t_2 \end{pmatrix}$$

Update $\left(\begin{array}{c|c} \alpha_{11} & a_{12}^T \\ \hline a_{21} & A_{22} \end{array}\right) = \left(I - \frac{1}{\tau_1}\begin{pmatrix} 1 \\ \hline u_{21} \end{pmatrix}\begin{pmatrix} 1 & u_{21}^H \end{pmatrix}\right)\left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & A_{22} \end{array}\right)$

via the steps:

$\alpha_{11} = 1 - 1/\tau_1$

$a_{12}^T = -(a_{21}^H A_{22})/\tau_1$

$A_{22} = A_{22} + a_{21} a_{12}^T$

$a_{21} = -a_{21}/\tau_1$

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right), \quad \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix} \leftarrow \begin{pmatrix} t_0 \\ \hline t_1 \\ \hline t_2 \end{pmatrix}$$

endwhile

Hw 3.3.5.3

Forming Q cost: $2mn^2 - \frac{2}{3}n^3$ flops

Forming Q is a expensive operation, especially when $m = n$. Therefore, we often just go straight to apply Q^H to y.

To solve: $Ax = y$

Householder QR factorization give us:

$$QRx = y$$

The next step is to find $\hat{y}$ that

$$Rx = \hat{y}$$

where $\hat{y} = Q^H y$

$$= (H_{n-1} \cdots H_0)\, y$$

Look at a specific example $H_k y$, then $\hat{y}$:

$$\left(\begin{array}{c|c} I & 0 \\ \hline 0 & I - \frac{1}{\tau_1}\begin{pmatrix} 1 \\ \hline u_{21} \end{pmatrix}\begin{pmatrix} 1 & u_{21}^H \end{pmatrix} \end{array}\right)\begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix}$$

$$= \begin{pmatrix} y_0 \\ \hline \psi_1 - \omega_1 \\ \hline y_2 - \omega_1 u_{21} \end{pmatrix} \quad \text{where } \omega_1 = \frac{(\psi_1 + u_{21}^H y_2)}{\tau_1}$$

Applying Q^H algorithm

$A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$, $t \rightarrow \begin{pmatrix} t_T \\ \hline t_B \end{pmatrix}$, $y \rightarrow \begin{pmatrix} y_T \\ y_B \end{pmatrix}$

A_{TL} is 0×0 and t_T, y_T have 0 elements

while ~~$m(A_{TL}) < m(A)$~~ $n(A_{BR}) > 0$

$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right)$, $\begin{pmatrix} t_T \\ \hline t_B \end{pmatrix} \rightarrow \begin{pmatrix} t_0 \\ \hline \tau_1 \\ \hline t_2 \end{pmatrix}$

$\begin{pmatrix} y_T \\ y_B \end{pmatrix} \rightarrow \begin{pmatrix} y_0 \\ \hline \psi_1 \\ \hline y_2 \end{pmatrix}$

Update $\begin{pmatrix} \psi_1 \\ y_2 \end{pmatrix} := \left(I - \frac{1}{\tau_1}\begin{pmatrix} 1 \\ a_{21} \end{pmatrix}\begin{pmatrix} 1 \\ a_{21} \end{pmatrix}^H\right)\begin{pmatrix} \psi_1 \\ y_2 \end{pmatrix}$

via the steps:

$\omega_1 = \left(\psi_1 + a_{21}^H y_2\right)/\tau_1$

$\begin{pmatrix} \psi_1 \\ y_2 \end{pmatrix} := \begin{pmatrix} \psi_1 - \omega_1 \\ y_2 - \omega_1 a_{21} \end{pmatrix}$

Move forward...

end while

3.3.6.1

Cost of applying Q^H to y is:

$4mn - 2n^2$

This is much cheaper than trying to form Q