SIS Scuola Interdipartimentale delle **Scienze**, dell'**Ingegneria** e della **Salute**

DiST Dipartimento di Scienze e Tecnologie

L. Magistrale in IA (ML&BD)

Scientific Computing (part 2 – 6 credits)

prof. Mariarosaria Rizzardi

Centro Direzionale di Napoli – Bldg. C4
room: n. 423 – North Side, 4th floor
phone: 081 547 6545
email: mariarosaria.rizzardi@uniparthenope.it

Contents

- Gram-Schmidt orthonormalization to check the linear independence of vectors.
- Gram-Schmidt orthonormalization and QR factorization.

What happens if GSO algorithm is applied to linearly dependent vectors?

$$A = \begin{array}{c} \begin{pmatrix} -2 & 1 & -1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} \\ \begin{array}{ccc} a_1 & a_2 & a_3 \end{array} \end{array}$$

The third column is the sum of the first two columns: the 3 columns are linearly dependent.

$$v_1 = a_1 = \begin{pmatrix} -2 \\ 1 \\ 0 \end{pmatrix} \Longrightarrow u_1 = \frac{v_1}{\|v_1\|} = \begin{pmatrix} -0.89443 \\ 0.44721 \\ 0 \end{pmatrix}$$

$$v_2 = a_2 - \langle a_2, u_1 \rangle u_1 = \begin{pmatrix} 0.2 \\ 0.4 \\ 1 \end{pmatrix} \Longrightarrow u_2 = \frac{v_2}{\|v_2\|} = \begin{pmatrix} 0.18257 \\ 0.36515 \\ 0.91287 \end{pmatrix}$$

$$v_3 = a_3 - \langle a_3, u_1 \rangle u_1 - \langle a_3, u_2 \rangle u_2 = \boxed{\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}}$$

$$u_3 = \frac{v_3}{\|v_3\|} = ?$$

NaN

NaN means "Not A Number"
(here it means "indeterminate form 0/0")

What happens if GSO algorithm is applied to linearly dependent vectors?

$$A = \begin{pmatrix} -2 & 1 & -1 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

in MATLAB

```
A=sym([-2 1 0; 1 0 1; -1 1 1]');
disp(rank(A))
     2
```

the 3rd column is the sum of the first two

linearly dependent columns

GSO

```
v(:,1)=A(:,1);
u(:,1)=v(:,1)/norm(v(:,1));
v(:,2)=A(:,2) - dot(A(:,2),u(:,1))*u(:,1);
u(:,2)=v(:,2)/norm(v(:,2));
v(:,3)=A(:,3) - dot(A(:,3),u(:,1))*u(:,1) ...
              - dot(A(:,3),u(:,2))*u(:,2);
u(:,3)=v(:,3)/norm(v(:,3));
disp(v)
[-2, 1/5, 0]
[ 1, 2/5, 0]
[ 0,   1, 0]
```

You can write the code in a more general form

zero vector

```
u=simplify(u)
u =
[-(2*5^(1/2))/5, 30^(1/2)/30, NaN]
[     5^(1/2)/5, 30^(1/2)/15, NaN]
[             0,  30^(1/2)/6, NaN]
```

What happens if GSO algorithm is applied to linearly dependent vectors? in MATLAB

```
syms x real
A=[sym(1)  x+1  x  x^2]
A =
[ 1,  x+1,  x  x^2]
```

$\underbrace{1,\ x+1,\ x\ x^2}_{\text{4 functions}}$

the third is the difference between the second and the first function.

```
dotProd=@(f,g) int(f*g,-1,1);
```

$$\langle f,g\rangle=\int_{-1}^{1} f(x)g(x)dx$$

```
v(1)=A(1);
u(1)=v(1)/sqrt(dotProd(v(1),v(1)));
v(2)=A(2)-dotProd(A(2),u(1))*u(1);
u(2)=v(2)/sqrt(dotProd(v(2),v(2)));
v(3)=A(3)-dotProd(A(3),u(1))*u(1)-dotProd(A(3),u(2))*u(2);
u(3)=v(3)/sqrt(dotProd(v(3),v(3)));
v=simplify(v)
v =
[ 1, x, 0]
disp(simplify(u))
[ 2^(1/2)/2, (6^(1/2)*x)/2, NaN]
v(4)= … NaN
```

The 3rd is the identically zero function

NaN means "Not A Number"

(here it means "indeterminate form 0/0")

Exercises

Write a MATLAB function that accepts a sequence of vectors on input, then by means of GSO Algorithm (*Gram-Schmidt Orthonormalization Algorithm*) checks their linear independence and, if so, it computes the two systems of orthogonal and orthonormal vectors.

What would change in the code if the input vectors were complex instead of real?

Write both the symbolic version of the code, suitable for function vectors too, and the numerical version of the code, only applicable to vectors of n real components.

GS Orthonormalization and QR factorization

The **GS orthonormalization**, when applied to the column vectors of a non-singular matrix $A_{(n\times n)}$, can be viewed as the QR-factorization of A, where:

- R is an upper triangular matrix;
- Q is an orthogonal matrix* $(Q^{-1}=Q^{T} \Longleftrightarrow Q^{T}Q=I)$

* If the A matrix is rectangular, then Q has orthonormal columns.

$$Q = (\underbrace{u_1, u_2, u_3, \ldots, u_n}_{\boxed{\text{orthonormal columns}}})$$

Gram-Schmidt Orthonormalization algorithm

we solve each formula w.r.t. a_k:

$$v_1 = a_1; \quad u_1 = \frac{v_1}{\|v_1\|} = \frac{a_1}{\|v_1\|}$$

$$\left.\begin{aligned} v_k &= a_k - \sum_{j=1}^{k-1} \langle a_k, u_j \rangle u_j \\ u_k &= \frac{v_k}{\|v_k\|} \end{aligned}\right\} \text{for } \; k = 2,3,\dots,n$$

$$a_1 = v_1 = \|v_1\| u_1;$$

$$\left.\begin{aligned} a_k &= \sum_{j=1}^{k-1} \langle a_k, u_j \rangle u_j + \|v_k\| u_k \\ v_k &= \|v_k\| u_k \end{aligned}\right\} \text{for } \; k = 2,3,\dots,n$$

What is the R-matrix? (1)

We rewrite the *GS orthonormalization formulas* such that they give the columns in $\boldsymbol{A}$:

$$\boldsymbol{A} = \left(\underline{\boldsymbol{a}_1} \quad \underline{\boldsymbol{a}_2} \quad \cdots \quad \underline{\boldsymbol{a}_n} \right) \longleftarrow \boxed{\text{columns in } A}$$

$$\|v_1\| u_1 = \underline{a_1}; \quad \Leftarrow \quad u_1 = \frac{v_1}{\|v_1\|} \qquad v_1 \longleftarrow a_1$$

$$\left.\begin{array}{c} \displaystyle\sum_{j=1}^{k-1} \left\langle \underline{a_k}, u_j \right\rangle u_j + \|v_k\| u_k = \underline{a_k} \\ \Uparrow \\ u_k = \dfrac{v_k}{\|v_k\|} \end{array}\right\} \quad \text{for} \quad k = 2, 3, \ldots, n$$

What is the R-matrix? (2)

$$Q = \begin{pmatrix} \underline{u_1} & \underline{u_2} & \cdots & \underline{u_n} \end{pmatrix} \leftarrow \boxed{\text{columns in Q}}$$

We get:

$$\boxed{\begin{array}{lllllll}
\|\underline{v_1}\|\underline{u_1} & + \ 0\underline{u_2} & + \ 0\underline{u_3} & + \ \cdots & + \ 0\underline{u_n} & = \underline{a_1} \\
\langle \underline{a_2}, \underline{u_1} \rangle \underline{u_1} & + \ \|\underline{v_2}\|\underline{u_2} & + \ 0\underline{u_3} & + \ \cdots & + \ 0\underline{u_n} & = \underline{a_2} \\
\langle \underline{a_3}, \underline{u_1} \rangle \underline{u_1} & + \ \langle \underline{a_3}, \underline{u_2} \rangle \underline{u_2} & + \ \|\underline{v_3}\|\underline{u_3} & + \ \cdots & + \ 0\underline{u_n} & = \underline{a_3} \\
\langle \underline{a_4}, \underline{u_1} \rangle \underline{u_1} & + \ \langle \underline{a_4}, \underline{u_2} \rangle \underline{u_2} & + \ \langle \underline{a_4}, \underline{u_3} \rangle \underline{u_3} & + \ \cdots & + \ 0\underline{u_n} & = \underline{a_4} \\
\quad \vdots & \quad \vdots & \quad \vdots & \ \ \cdots & \vdots & \quad \vdots \\
\langle \underline{a_n}, \underline{u_1} \rangle \underline{u_1} & + \ \langle \underline{a_n}, \underline{u_2} \rangle \underline{u_2} & + \ \langle \underline{a_n}, \underline{u_3} \rangle \underline{u_3} & + \ \cdots & + \ \|\underline{v_n}\|\underline{u_n} & = \underline{a_n}
\end{array}}$$

$$\underline{a_j} = \overbrace{\alpha_{1j}\underline{u_1} + \alpha_{2j}\underline{u_2} + \alpha_{3j}\underline{u_3} + \alpha_{4j}\underline{u_4} + \cdots + \alpha_{nj}\underline{u_n}}^{\text{linear combination of columns in } Q} = Q \begin{pmatrix} \alpha_{1j} \\ \alpha_{2j} \\ \vdots \\ \alpha_{nj} \end{pmatrix}$$

$\boxed{\text{j}^{\text{th}} \text{ column of } A}$ $\boxed{\text{j}^{\text{th}} \text{ column of } R}$

What is the R-matrix? (3)

$$R = \begin{pmatrix} \underline{\alpha_1} & \underline{\alpha_2} & \cdots & \underline{\alpha_n} \end{pmatrix} \longleftarrow \boxed{\text{columns in R}}$$

$$\begin{aligned}
\|v_1\| u_1 + 0u_2 + 0u_3 + \cdots + 0u_n &= a_1 \\
\langle a_2, u_1\rangle u_1 + \|v_2\| u_2 + 0u_3 + \cdots + 0u_n &= a_2 \\
\langle a_3, u_1\rangle u_1 + \langle a_3, u_2\rangle u_2 + \|v_3\| u_3 + 0u_4 + \cdots + 0u_n &= a_3 \\
\langle a_4, u_1\rangle u_1 + \langle a_4, u_2\rangle u_2 + \langle a_4, u_3\rangle u_3 + \|v_4\| u_4 + \cdots + 0u_n &= a_4 \\
&\vdots \\
\sum_{j=1}^{n-1} \langle a_n, u_j\rangle u_j + \|v_n\| u_n &= a_n
\end{aligned}$$

coefficients of u_k

$$\underline{\alpha_3} =$$

$$R = \begin{pmatrix}
\|v_1\| & \langle a_2, u_1\rangle & \langle a_3, u_1\rangle & \langle a_4, u_1\rangle & \cdots & \langle a_n, u_1\rangle \\
 & \|v_2\| & \langle a_3, u_2\rangle & \langle a_4, u_2\rangle & \cdots & \langle a_n, u_2\rangle \\
 & & \|v_3\| & \langle a_4, u_3\rangle & \cdots & \langle a_n, u_3\rangle \\
 & & & \|v_4\| & \cdots & \langle a_n, u_4\rangle \\
 & & & & \ddots & \vdots \\
 & & & & & \|v_n\|
\end{pmatrix}$$

Example 1: MATLAB qr() function

$$A=\begin{pmatrix}-2 & 1 & 0\\ 1 & 0 & 2\\ 0 & 1 & 1\end{pmatrix}$$

```
A=[-2 1 0; 1 0 1; 0 2 1]';
v(:,1)=A(:,1); u(:,1)=v(:,1)/norm(v(:,1));
v(:,2)=A(:,2)-A(:,2)'*u(:,1)*u(:,1); u(:,2)=v(:,2)/norm(v(:,2));
v(:,3)=A(:,3)-(A(:,3)'*u(:,1)*u(:,1)+A(:,3)'*u(:,2)*u(:,2));
u(:,3)=v(:,3)/norm(v(:,3));
u
u =
     -0.8944     0.1826     0.4082
      0.4472     0.3651     0.8165
           0     0.9129    -0.4082
[Q,R]=qr(A)
Q =
     -0.8944    -0.1826    -0.4082
      0.4472    -0.3651    -0.8165
           0    -0.9129     0.4082
R =
      2.2361    -0.8944     0.8944
           0    -1.0954    -1.6432
           0          0    -1.2247
```

square matrix

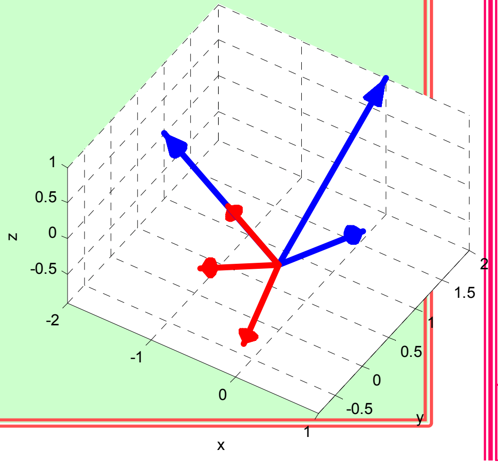
GSO

$\mathbb{R}^3$ basis

Gram-Schmidt

=

QR



Example 2: MATLAB qr() function

$$A = \begin{pmatrix} -2 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

```
A=[-2 1 0; 1 0 1]';
```
rectangular matrix

GSO
```
v(:,1)=A(:,1); u(:,1)=v(:,1)/norm(v(:,1));
v(:,2)=A(:,2)-A(:,2)'*u(:,1)*u(:,1); u(:,2)=v(:,2)/norm(v(:,2));
u
u =
      -0.8944    0.1826
       0.4472    0.3651
            0    0.9129

[Q,R]=qr(A)
Q =
      -0.8944   -0.1826   -0.4082
       0.4472   -0.3651   -0.8165
            0   -0.9129    0.4082
R =
       2.2361   -0.8944
            0   -1.0954
            0         0
```

$\mathscr{R}(A)$ basis

$\mathscr{R}(A)^{\perp}$ basis

QR = Gram-Schmidt +

build a basis of $\mathbb{R}^3$ by means of $\mathbb{R}^3 = \mathscr{R}(A) \oplus \mathscr{R}(A)^{\perp}$

Example 3: MATLAB qr() function

$$A=\begin{pmatrix}-2 & 1\\ 1 & 0\\ 0 & 1\end{pmatrix}$$

```
A=[-2 1 0; 1 0 1]';
```
rectangular matrix

GSO
```
v(:,1)=A(:,1); u(:,1)=v(:,1)/norm(v(:,1));
v(:,2)=A(:,2)-A(:,2)'*u(:,1)*u(:,1); u(:,2)=v(:,2)/norm(v(:,2));
u
u =
     -0.8944    0.1826
      0.4472    0.3651
           0    0.9129

[Q,R]=qr(A,"econ")  % economy size

Q =
     -0.8944    -0.1826
      0.4472    -0.3651
           0    -0.9129
R =
      2.2361   -0.8944
           0   -1.0954
```

$\mathscr{R}(Q)=\mathscr{R}(A)$

QR = Gram-Schmidt

```
rank([A Q])
ans =
     2
```

R: upper triangular matrix