Lab 6

Gram - Schmidt

```
zero = [0;0;0];
```

Define our basis

```
a = [2;1;0]
```

```
a = 3×1
     2
     1
     0
```

```
b = [0;0;4]
```

```
b = 3×1
     0
     0
     4
```

```
c = [2.5;3;0]
```

```
c = 3×1
    2.5000
    3.0000
         0
```

Current basis

```
Old_BASIS = rats([a b c])
```

```
Old_BASIS = 3×42 char array
    '       2              0              5/2     '
    '       1              0               3      '
    '       0              4               0      '
```

```
% det_old_bas = det([a b c])
```

Let's check, is it orthogonal?

```
dot(a,b)
```

```
ans = 0
```

```
dot(b,c)
```

```
ans = 0
```

```
dot(a,c)
```

```
ans = 8
```

As it can be seen, it is not (it should be equal to 0)

For checking the concepts, let's consider some vector, which will be a linear combination of our basis.

```
new_vec_el = [1;2;3];
new_vec_old_bas = new_vec_el(1)*a + new_vec_el(2)*b + new_vec_el(3)*c
```

```
new_vec_old_bas = 3×1
    9.5000
   10.0000
    8.0000
```

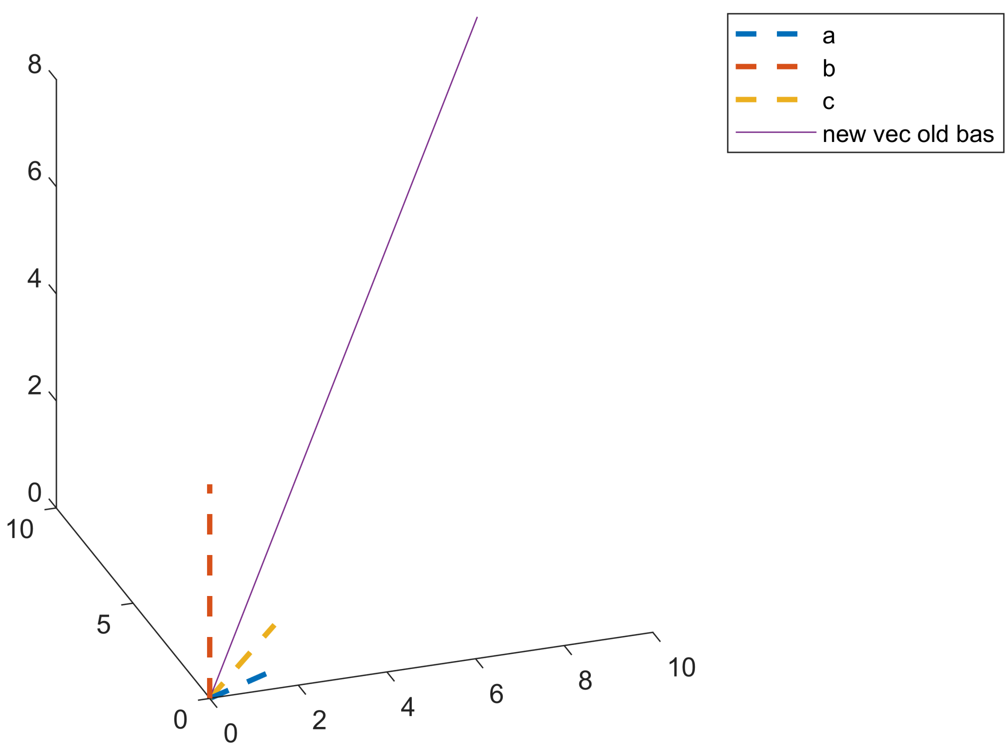
```
new_vec_old_bas_norm = new_vec_el(1)*a./norm(a) + new_vec_el(2)*b./norm(b) + new_vec_el(3)*c./r
```

```
new_vec_old_bas_norm = 3×1
    2.8150
    2.7519
    2.0000
```

```

figure
plot3([zero(1) a(1)], [zero(2) a(2)],[zero(3) a(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) b(1)], [zero(2) b(2)],[zero(3) b(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) c(1)], [zero(2) c(2)],[zero(3) c(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) new_vec_old_bas(1)], [zero(2) new_vec_old_bas(2)],[zero(3) new_vec_old_bas(3)])
legend('a','b','c','new vec old bas')

view([-19.1 25.2])
```



There is a `Gram-Schmidt` algorithm, there is a link, which is understandable.

We define the projection operator by

$$\mathrm{proj}_{\mathbf{u}}(\mathbf{v}) = \frac{\langle \mathbf{u}, \mathbf{v} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u},$$

The Gram–Schmidt process then works as follows:

$$\begin{aligned}
\mathbf{u}_1 &= \mathbf{v}_1, & \mathbf{e}_1 &= \frac{\mathbf{u}_1}{\|\mathbf{u}_1\|} \\
\mathbf{u}_2 &= \mathbf{v}_2 - \mathrm{proj}_{\mathbf{u}_1}(\mathbf{v}_2), & \mathbf{e}_2 &= \frac{\mathbf{u}_2}{\|\mathbf{u}_2\|} \\
\mathbf{u}_3 &= \mathbf{v}_3 - \mathrm{proj}_{\mathbf{u}_1}(\mathbf{v}_3) - \mathrm{proj}_{\mathbf{u}_2}(\mathbf{v}_3), & \mathbf{e}_3 &= \frac{\mathbf{u}_3}{\|\mathbf{u}_3\|} \\
\mathbf{u}_4 &= \mathbf{v}_4 - \mathrm{proj}_{\mathbf{u}_1}(\mathbf{v}_4) - \mathrm{proj}_{\mathbf{u}_2}(\mathbf{v}_4) - \mathrm{proj}_{\mathbf{u}_3}(\mathbf{v}_4), & \mathbf{e}_4 &= \frac{\mathbf{u}_4}{\|\mathbf{u}_4\|} \\
&\vdots & &\vdots \\
\mathbf{u}_k &= \mathbf{v}_k - \sum_{j=1}^{k-1} \mathrm{proj}_{\mathbf{u}_j}(\mathbf{v}_k), & \mathbf{e}_k &= \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}.
\end{aligned}$$

```
A = a;
B = b - (A'*b)./(A'*A)*A
```

```
B = 3×1
     0
     0
     4
```

```
C = c - (A'*c)./(A'*A)*A - (B'*c)./(B'*B)*B
```

```
C = 3×1
   -0.7000
    1.4000
         0
```

There is a new basis

```
New_BASIS = rats([A B C])
```

```
New_BASIS = 3×42 char array
    '       2              0            -7/10      '
    '       1              0             7/5       '
    '       0              4              0        '
```

```
% det_new_bas = det([A B C])
```

And it is `orthogonal`, because all of our dot products are equal to 0

```
dot(A,B)
```

```
ans = 0
```

```
round(dot(B,C))
```

```
ans = 0
```

```
round(dot(A,C))
```

```
ans = 0
```

Let's put our old vector scalar values in new basis

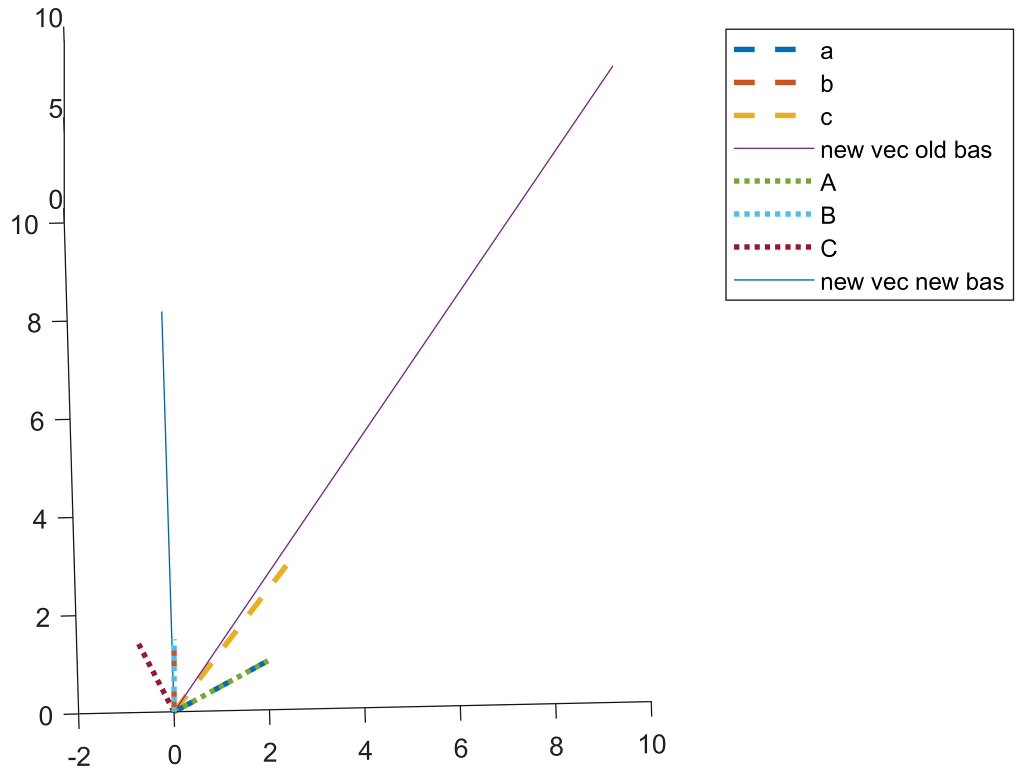
```
new_vec_new_bas = new_vec_el(1)*A + new_vec_el(2)*B + new_vec_el(3)*C
```

```
new_vec_new_bas = 3×1
   -0.1000
    5.2000
    8.0000
```

```

figure
plot3([zero(1) a(1)], [zero(2) a(2)],[zero(3) a(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) b(1)], [zero(2) b(2)],[zero(3) b(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) c(1)], [zero(2) c(2)],[zero(3) c(3)],'--',"LineWidth",2)
hold on
plot3([zero(1) new_vec_old_bas(1)], [zero(2) new_vec_old_bas(2)],[zero(3) new_vec_old_bas(3)])
hold on
plot3([zero(1) A(1)], [zero(2) A(2)],[zero(3) A(3)],':',"LineWidth",2)
hold on
plot3([zero(1) B(1)], [zero(2) B(2)],[zero(3) B(3)],':',"LineWidth",2)
hold on
plot3([zero(1) C(1)], [zero(2) C(2)],[zero(3) C(3)],':',"LineWidth",2)
hold on
plot3([zero(1) new_vec_new_bas(1)], [zero(2) new_vec_new_bas(2)],[zero(3) new_vec_new_bas(3)])
legend('a','b','c','new vec old bas','A','B','C','new vec new bas')

view([-1.45 69.72])
```



As it can be seen, there are not in the same position.

```
New_BASIS_NORM = ([A./norm(A) B./norm(B) C./norm(C)]);
```

Gram - Schmidt is complete, there is our **OrthoNormal basis**

```
New_BASIS_NORM_char = rats(New_BASIS_NORM)
```

```
New_BASIS_NORM_char = 3×42 char array
    '    305/341          0         -305/682   '
    '    305/682          0          305/341   '
    '       0             1             0      '
```

```
% det_new_bas_norm = det(New_BASIS_NORM)
```

Even with norm values, our vector is not equal from the old basis. That means, that using this tech we can only assume that we have the same space with some linear mapping between them.

```
new_vec_new_bas_norm = new_vec_el(1)*A./norm(A) + new_vec_el(2)*B./norm(B) + new_vec_el(3)*C./r
```

```
new_vec_new_bas_norm = 3×1
   -0.4472
    3.1305
    2.0000
```

Solving simple linear equation, we can find the same vector in new basis

```
new_vec_el_in_new_bas = linsolve(New_BASIS_NORM,new_vec_old_bas)
```

```
new_vec_el_in_new_bas = 3×1
   12.9692
    8.0000
    4.6957
```

```
new_vec_new_bas_norm_correct_el = new_vec_el_in_new_bas(1)*A./norm(A) + new_vec_el_in_new_bas(2
```

```
new_vec_new_bas_norm_correct_el = 3×1
    9.5000
   10.0000
    8.0000
```

```
new_vec_old_bas
```

```
new_vec_old_bas = 3×1
    9.5000
   10.0000
    8.0000
```