Max Kramer

I affirm that I have adhered to the honor code on this assingnment.

Hello again, scientist! I'll do all my writing in italics, and problems for you will be in **bold.** *Comment your code, and* explain your ideas in plaintext. *As a general rule, I expect you to do at least as much writing as I do. Code should be part of your solution, but I expect variables to be clear and explanation to involve complete sentences. Cite your sources; if you work with someone in the class on a problem, that's an extremely important source. Don't work alone.*

Problem F.01: Gram the Schmidt.

MATLAB will do the Gram-Schmidt process for you if you ask it nicely.

```
v1 = [-10;2;-6;16;2]; v2 = [13;1;3;-16;1];
v3 = [7;-5;13;-2;-5]; v4 = [-11;3;-3;5;-7]; v5 = [1;1;1;1;1];

A = [v1 v2 v3 v4 v5]
```

```
A =

   -10    13     7   -11     1
     2     1    -5     3     1
    -6     3    13    -3     1
    16   -16    -2     5     1
     2     1    -5    -7     1
```

Check that Col([v1 v2 v3 v4]) is a 4-dimensional subspace of R^5.

```
B = A(:,1:end-1)
rref(B)
```

```
B =

   -10    13     7   -11
     2     1    -5     3
    -6     3    13    -3
    16   -16    -2     5
     2     1    -5    -7


ans =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
     0     0     0     0
```

The column space of the matrix B = [v1 v2 v3 v4] is all of its pivot columns. All of the columns of B are pivot columns, and all of the columns are linearly independent. The column space of an m x n matrix is a subspace of R^m, so the span of the four columns of B are a 4-dimensional subspace of R^5.

Use Gram-Schmidt to find an orthonormal basis for Span{v1,v2}. *Do this "manually," as in without any fancy MATLAB commands. Just matrix multiplication and subtraction for this one, please.*

```
dot(v1,v2) % NOT ORTHOGONAL
vv1 = v1;
vv2 = v2 - ((v2'*vv1)/(vv1' * vv1)) * vv1;
vv2' * vv1 % ORTHOGONAL, BUT NOT ORTHONORMAL
vv1 = vv1/norm(vv1);
vv2 = vv2/norm(vv2);
vv2'*vv1
```

```
ans =

  -400


ans =

     0


ans =

   1.3878e-17
```

The dot product of v1 and v2 is nonzero, so they are not orthogonal. The basis is transformed to an orthogonal basis by applying the Gram-Schmidt process to v2, creating an orthogonal basis {vv1,vv2}. This is not an orthonormal basis as the vectors are not unit length, so a quick division by the norm produces the orthonormal basis.

Wow that was terrible, right? Check this out.

```
[Q,R]=qr(A);
```

Check that Q is orthogonal, R is upper triangular, and Q*R = A. *This is called the QR decomposition. It's not useful to know how to construct it by hand, but it's extremely useful to know what it is.*

```
Q' * Q
istriu(R)
Q * R
A
```

```
ans =

    1.0000   -0.0000   -0.0000    0.0000    0.0000
   -0.0000    1.0000   -0.0000    0.0000    0.0000
   -0.0000   -0.0000    1.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    1.0000    0.0000
```

```
    0.0000    0.0000    0.0000    0.0000    1.0000


ans =

  logical

   1


ans =

  -10.0000   13.0000    7.0000  -11.0000    1.0000
    2.0000    1.0000   -5.0000    3.0000    1.0000
   -6.0000    3.0000   13.0000   -3.0000    1.0000
   16.0000  -16.0000   -2.0000    5.0000    1.0000
    2.0000    1.0000   -5.0000   -7.0000    1.0000


A =

   -10    13     7   -11     1
     2     1    -5     3     1
    -6     3    13    -3     1
    16   -16    -2     5     1
     2     1    -5    -7     1
```

If a matrix is orthogonal, then that matrix multiplied by its own transpose should return the identity matrix. The istriu() command tests if a matrix is in upper triangular form. Q * R does in fact return the matrix A.

Cool, so how does this have anything to do with Gram-Schmidt?

```
A*R^(-1) % since Q*R = A, Q = A*R^(-1)
```

```
ans =

   -0.5000   -0.5000   -0.5774   -0.0000   -0.4082
    0.1000   -0.5000    0.0000    0.7071    0.4899
   -0.3000    0.5000   -0.5774    0.0000    0.5715
    0.8000         0   -0.5774    0.0000   -0.1633
    0.1000   -0.5000    0.0000   -0.7071    0.4899
```

Well hey! If you did your Gram-Schmidt correctly above, the columns you got should be the same as the first two columns of this. That means that the columns of R^(-1) are actually exactly the Gram-Schmidt information!

```
R^(-1)
```

```
ans =

    0.0500   -0.1667   -0.1764    0.2003   -0.4922
```

```
         0   -0.1667   -0.1283    0.2357   -0.3969
         0         0   -0.0962    0.0707   -0.1701
         0         0         0    0.1414    0.0000
         0         0         0         0    1.0206
```

Use R^(-1) to write down a linear combination of v1, v2, v3, and v4 that produces an orthonormal vector which is orthogonal to v1, v2, and v3. Check your answer.

```
Rinv = R^(-1);
lincombo = A * Rinv(:,4)
norm(lincombo)
lincombo' * v1
lincombo' * v2
lincombo' * v3
```

```
lincombo =

   -0.0000
    0.7071
    0.0000
    0.0000
   -0.7071


ans =

     1


ans =

   9.1038e-15


ans =

  -1.2768e-14


ans =

  -8.8818e-16
```

The columns of R^(-1) represent linear combinations of v1 through v5 that produce orthonormal vectors that are orthogonal to a set of the vectors v1 through v5. To find a combination that produces a vector orthogonal to v1 through v3, we multiply the original matrix A (the columns v1 through v5) by the 4th column of R^(-1). The resultant linear combination is 0.7071 * v2 - 0.7071 * v5. The norm of the resulting vector is 1 and the dot product of the vector and v1 through v3 is 0.

Published with MATLAB® R2019b