Table of Contents

Eric Wan - ezw23@drexel.edu - HW9

```
format long
```

Problem 1

```
k = 1;
% part a
u1 = [1; 2; 3]; % setting first vector of set U
u2 = [-2; -3; 1]; % setting second vector of set U
% dot = transpose(u1) * u2 also works for reference
dota = u1.' * u2; % inner product (dot product)
n1 = norm(u1);
n2 = norm(u2);
check1 = eq(n1,k);
check2 = eq(n2,k);
% Not orthogonal, Not orthonormal

% part b
u1 = [1; 0; -2];
u2 = [0; 1; 0];
u3 = [2; 0; 1];
dotb = transpose(u1) * u2;
dotb = u1.' * u3;
dotb = transpose(u2) * u3;
n1 = norm(u1);
n2 = norm(u2);
n3 = norm(u3);
check1 = eq(n1,k);
check2 = eq(n2,k);
check3 = eq(n3,k);
% Is orthogonal, Not orthonormal

% part c
u1 = [1/(2^0.5); 1/(2^0.5); 1/(2^0.5); 1/(2^0.5)];
u2 = [-1; 0; 0; 0];
u3 = [0; 1/(2^0.5); 1/(2^0.5); 0];
u4 = [0; 0; 0; 1];
dotc = transpose(u1) * u2
dotc = u1.' * u3
dotc = transpose(u1) * u4
```

```
n1 = norm(u1);
n2 = norm(u2);
n3 = round(norm(u3));
n4 = norm(u4);
check1 = eq(n1,k)
check2 = eq(n2,k)
check3 = eq(n3,k)
check4 = eq(n4,k)
% Not orthogonal, Not orthonormal

% part d
u1 = [1; 0; 1; 0; 1];
u2 = [0; 2; 0; 2; 1];
dotd = transpose(u1) * u2;
n1 = norm(u1);
n2 = norm(u2);
check1 = eq(n1,k);
check2 = eq(n2,k);
% Not orthogonal, Not orthonormal


dotc =

  -0.707106781186547


dotc =

   1.000000000000000


dotc =

   0.707106781186547


check1 =

  logical

   0


check2 =

  logical

   1


check3 =

  logical
```

```
   1


check4 =

  logical

   1
```

Problem 2

u dot v = |u|*||v|| cosd(theta in degrees) part a

```
%{
u1 = [1; 2]
u2 = [a; -1];
angle = cosd(-45);
(1)(a) + (2)(-1) = ((1)^2 + (2)^2)^0.5 * ((a)^2 + (-1)^2)^0.5 * angle
a - 2 = (5^0.5) * (a^2 +1)^0.5 * cosd(-45)
(a - 2) / (5a^2 +5)^0.5 = (1/2)^0.5
2^0.5(a - 2) = (5a^2 +5)^0.5
2(a - 2)^2 = (5a^2 +5)
a = -3, 1/3
%}
a = -3;
u1 = [1; 2];
u2 = [a; -1];
dota = transpose(u1) * u2
a = 1/3;
u1 = [1; 2];
u2 = [a; -1];
dota = transpose(u1) * u2

% part b
%{
u1 = [-2; a];
u2 = [a; 3];
angle = cosd(75);
(-2)(a) + (a)(3) = ((-2)^2 + (a)^2)^0.5 * ((a)^2 + (3)^2)^0.5 * angle
a = ((a^2 +4)^0.5) * (a^2 +9)^0.5 * cosd(75)
a = complex number...?
%}

% part c
%{
u1 = [a^2; 1];
u2 = [-1; 0];
angle = cosd(90);
(a^2)(-1) + (1)(0) = ((a^2)^2 + (1)^2)^0.5 * ((-1)^2 + (0)^2)^0.5 *
 angle
-(a^2) = ((a^4 +1)^0.5) * (1)^0.5 * cosd(90)
a = 0
```

```
%}
a = 0;
u1 = [a^2; 1];
u2 = [-1; 0];
dotc = transpose(u1) * u2
```

```
dota =

    -5


dota =

  -1.666666666666667


dotc =

     0
```

Problem 3

|u + v| ? |u| + |v| part a

```
u1 = [1; 2; -5; 0; 3];
n1u1 = norm(u1,1)
n2u1 = norm(u1)

% part b
u2 = [-2; 3; 1; 4; -2];
n1u2 = norm(u2,1)
n2u1 = norm(u2)

% part c
u = u1 + u2;
nu = norm(u)
n12 = n2u1 + n2u1
```

```
n1u1 =

    11


n2u1 =

   6.244997998398398


n1u2 =

    12
```

```
n2u1 =

   5.830951894845301


nu =

   7.681145747868608


n12 =

  11.661903789690601
```

Problem 4

```
%{
rowspace = [1; 0; 0; 2; 1], [0; 1; 0; 1; 2], [0; 0; 1; 0; -2]
nullspace = [-2; -1; 0; 1; 0], [-1; -2; 2; 0; 1]
%}
u1 = [1; 0; 0; 2; 1];
u2 = [0; 1; 0; 1; 2];
u3 = [0; 0; 1; 0; -2];
u4 = [-2; -1; 0; 1; 0];
u5 = [-1; -2; 2; 0; 1];
dot = transpose(u1) * u4
dot = u1.' * u5
dot = transpose(u2) * u4
dot = u2.' * u5
dot = transpose(u3) * u4
dot = u3.' * u5
%{
columnspace = [1; 0; 0; 0], [0; 1; 0; 0], [1; 2; 0; 0]
nullspace = [0; 0; 1; 0], [0; 0; 0; 1]
%}
u1 = [1; 0; 0; 0];
u2 = [0; 1; 0; 0];
u3 = [1; 2; 0; 0];
u4 = [0; 0; 1; 0];
u5 = [0; 0; 0; 1];
dot = transpose(u1) * u4
dot = u1.' * u5
dot = transpose(u2) * u4
dot = u2.' * u5


dot =

     0
```

```
dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0


dot =

     0
```

Problem 5

part a

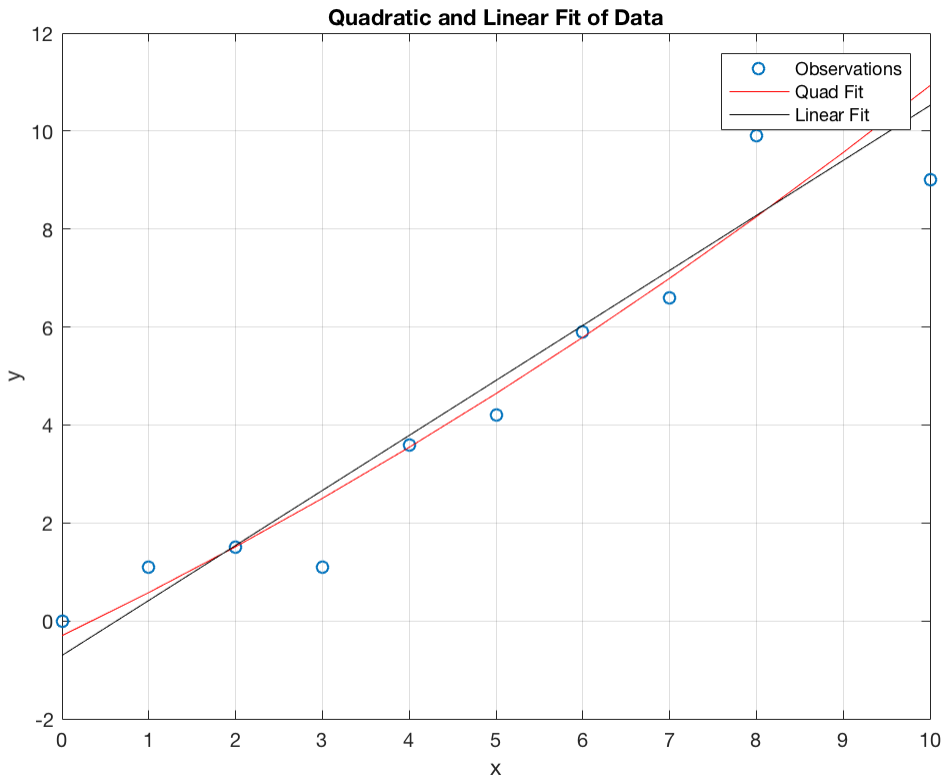
```
u1 = [1; -1; 0];
u2 = [1; 1; 0];
u3 = [0; 0; 1];
```

```
dota = transpose(u1) * u2
dota = u1.' * u3
dota = transpose(u2) * u3
% all dot products = 0 implying all vectors are orthogonal to one
 another

% part b
y = [-2; 4; 1];
% c = [-3 1 1]
sol = -3 * u1 + 1 * u2 + 1 * u3;
check = isequal(y, sol)

% part c
sol = -3 * u1 + 1 * u2

% part d
d = y - sol;
norm(d)

% part e
d = y - sol;
norm(d)


dota =

     0


dota =

     0


dota =

     0


check =

  logical

   1


sol =

    -2
     4
     0


ans =
```

```
     1


ans =

     1
```

Problem 6

```
load('pts.mat')
X = ptsMixA(1,:); % setting X
Y = ptsMixA(2,:); % setting Y
[N, XT, D, YT, beta_est, Y_est] = quadfit(X, Y); % running quadfit
err = YT - Y_est; % calculating error of each Y value
RMSEQ = (err'*err/N)^0.5 % calculating RMS error

plot(X, Y, 'o'), hold on, grid on % plotting data points of
 pts_setA(1)
plot(X, Y_est, 'r') % plotting line of best fit
xlabel('x')
ylabel('y')
title('Quadratic and Linear Fit of Data') % labeling title
hold on

[N, XT, D, YT, beta_est, Y_est] = linefit(X, Y); % running linefit
err = YT - Y_est; % calculating error of each Y value
RMSEL = (err'*err/N)^0.5 % calculating RMS error
plot(X, Y_est, 'k') % plotting line of best fit
legend('Observations','Quad Fit', 'Linear Fit') % labeling legend

% quadratic fit is better than linear fit


RMSEQ =

   1.022045615880660


RMSEL =

   1.049340603655142
```



Published with MATLAB® R2017a