Appendix

MATLAB code for P1.8

P1.8: p8_main.m

```
load('/nfs/ug/homes-2/c/chenliy5/Desktop/Matrix Algebra/PS1/PS01_dataSet/wordVecV.mat')
dsize = size(V,2);
tsize = size(V,1);

%(a)----------------------------------------
fprintf('a)\n');
min_distance(V);

%(b)----------------------------------------
fprintf('b)\n');
sum_fterm = sum(V,1);
Vd_norm = V;

for i=1:dsize
    Vd_norm(:,i) = V(:,i)/sum_fterm(i);
end
min_distance(Vd_norm);

%(c)----------------------------------------
fprintf('c)\n');
fdot=zeros(1,tsize);
for row=1:tsize
    for col=1:dsize
        if(V(row,col) > 0)
            fdot(row)= fdot(row) + 1;
        end
    end
end

Wd=V;
for row=1:tsize
    for col=1:dsize
        Wd(row,col) = (Vd_norm(row,col))*sqrt(log(dsize/fdot(row)));
    end
end
min_distance(Wd);
```

P1.8: min_distance.m

```
function [d1,d2,d3,d4] = min_distance(V)
angle=360;
distance=zeros(10,10);
d1=0;
d2=0;

%angle distance
for i=1:size(V,2)
    Vd1 = V(:,i);
    for j=1:size(V,2)
        if i~=j
            Vd2 = V(:,j);
            angle_temp = abs(acos(dot(Vd1,Vd2)/(norm(Vd1)*norm(Vd2))));
            if(angle_temp ~= 0 && angle>=angle_temp)
                angle=angle_temp;
                d1=i;
                d2=j;
            end
        end
    end
end

%Enclidean distance
for i=1:size(V,2)
    Vd1 = V(:,i);
    for j=1:size(V,2)
        if i~=j
            Vd2 = V(:,j);
            distance_temp = norm(Vd1-Vd2);
            distance(i,j) = distance_temp;
```

```
        end
    end
end

distanceCopy = (distance + diag(Inf(size(diag(distance)))));    % put Inf on diagonal
%[~,ndx] = min(distanceCopy);                        % get the linear index of the minimum value
misdistance=min(distanceCopy(:));
[d3,d4] = find(distanceCopy==misdistance);


fprintf('%dth and %dth are closest in Euclidean distance\n', d3(1), d3(2));
fprintf('The smallest Euclidean distance is %.4f\n', misdistance);
fprintf('%dth and %dth are closest in angle distance\n', d1, d2);
fprintf('The smallest angle distance is %.4f\n', angle);
end
```

MATLAB code for P1.9

P1.9: p9_main.m

```
close all; clc;

[x,y]=meshgrid(-2:0.2:3.5, -2:0.2:3.5);
f1 = 2*x+3*y+1;
f2 = x.^2+y.^2-x.*y-5;
f3 = (x-5)*cos(y-5)-(y-5)*sin(x-5);

figure(1)
plotContour(x,y,f1,1);
plotContour(x,y,f2,2);
plotContour(x,y,f3,3);

figure(2)
plotTangentPlane(x,y,f1,1);

plotTangentPlane(x,y,f2,2);

plotTangentPlane(x,y,f3,3);

function plotContour(x,y,f,subPlotInd)
subplot(1,3,subPlotInd);

contour(x,y,f,21);
hold on

[dx,dy] = gradient(f,.2,.2);
x0 = 1;
y0 = 0;
t = (x == x0) & (y == y0);
indt = find(t);
%f_grad2 = [dx(indt) dy(indt)];
gradX = linspace(1,1.5);
tangentX = linspace(0.5,1.5);

gradY = dy(indt)/dx(indt)*(gradX - 1);
tangentY = -dx(indt)/dy(indt)*(tangentX - 1);
plot(gradX,gradY);
hold on
plot(tangentX,tangentY);
axis equal
xlabel('x');
ylabel('y');
hold off;
end

function plotTangentPlane(x,y,f,subPlotInd)
subplot(1,3,subPlotInd);
surf(x,y,f)
hold on

[dx,dy] = gradient(f,.2,.2);
x0 = 1;
y0 = 0;
t = (x == x0) & (y == y0);
indt = find(t);
```

```
f_grad = [dx(indt) dy(indt)];
f_plane=f(indt)+f_grad(1)*(x-1)+f_grad(2)*y;
surf(x,y,f_plane);
xlabel('x');
ylabel('y');
zlabel('z');
end
```

MATLAB code for P1.10

P1.10: p10_main.m

```
close all; clc;

x=linspace(-2,2);
y=linspace(-1,1);
origin=[0;0];
x1=[3;-1];
b1=[-1;1];
v1=[1;2];

%(c)%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure(1)
%Draw arrow vector x1
drawArrow = @(j,k) quiver(j(1),j(1),k(1),k(2),0);
drawArrow(origin,x1);
text(2.5,-1,'x1');
hold on
%Draw the subspace V1
y=v1(2)/v1(1)*x;
l1=plot(x,y,'g--');
hold on
%Draw the affine set A1
A1=v1(2)/v1(1)*(x+1)+1;
l2=plot(x,A1,'b--');

legend([l1,l2],'Subspace V1','Affeint set A1');
hold on

%Draw projection on V1
proj_11=1/5.*v1;
q=drawArrow(origin,proj_11);
q.LineWidth=1;
q.DisplayName='projection';
text(0.2,0.2,'Proj_v_1(x1)');
%Draw projection on A1
proj_A11=b1;
q=drawArrow(origin,proj_A11);
q.LineWidth=1;
q.DisplayName='projection';
text(-1,1,'Proj_A_1(x1)');

xlim([-2 3])
ylim([-2 2])
xlabel('x');
ylabel('y');
axis equal
hold on


%(d)%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

x3=[3;0;-1;2;2];
b3=[-1;0;1;-2;1];
v1_d=[0;1;2;0;0];
v2_d=[1;3;0;1;0];
v3_d=[0;1;5;0;1];
subspace = [v1_d, v2_d, v3_d];

ortho_basis = gramschmidt(subspace);
v1_ortho = ortho_basis(:, 1);
v2_ortho = ortho_basis(:, 2);
v3_ortho = ortho_basis(:, 3);
disp('orthonormal basis')
disp(ortho_basis)
```

```
proj_x_onV = proj_v(x3, v1_ortho) + proj_v(x3, v2_ortho) + proj_v(x3, v3_ortho);
disp('projecting onto orthonormal basis')
disp(proj_x_onV)

proj_x_onA = proj_v(x3 - b3, v1_ortho) + proj_v(x3 - b3, v2_ortho) + proj_v(x3 - b3, v3_ortho) + b3;
disp('projecting onto orthonomral affine')
disp(proj_x_onA)
```

MATLAB code for P1.11

P1.11: p11_main.m

```
function [ time_pos , sq_wave , B_unnorm ] = generate_data
close all; clc;
n_comps = 30;
period = 10;
fundFreq = 1/ period ;
time_pos = 0:0.0001:2* period ;
harmonics = 2*(1: n_comps ) -1;
sq_wave = floor (0.9* sin (2* pi* fundFreq * time_pos )) +.5; %% generate the signal
B_unnorm = sin (2* pi* fundFreq *( harmonics'* time_pos )) /2; %% generate the basis

figure(1)
plot(time_pos,sq_wave);
xlabel('time ops');
ylabel('sq wave');

figure(2)
basis_1 = B_unnorm(1, :);
basis_2 = B_unnorm(2, :);
basis_3 = B_unnorm(3, :);
basis_4 = B_unnorm(4, :);
basis_5 = B_unnorm(5, :);
basis_6 = B_unnorm(6, :);
basis_30 = B_unnorm(30, :);

subplot(7, 1, 1)
plot(time_pos, basis_1);
xlabel('time ops');
ylabel('sq wave');
title('1st basis vector')
axis([0 20 -1 1]);
hold on

subplot(7, 1, 2)
plot(time_pos, basis_2);
xlabel('time ops');
ylabel('sq wave');
title('2nd basis vector')
axis([0 20 -1 1]);

subplot(7, 1, 3)
plot(time_pos, basis_3);
xlabel('time ops');
ylabel('sq wave');
title('3rd basis vectors')
axis([0 20 -1 1]);
hold on

subplot(7, 1, 4)
plot(time_pos, basis_4);
xlabel('time ops');
ylabel('sq wave');
title('4th basis vectors')
axis([0 20 -1 1]);
hold on

subplot(7, 1, 5)
plot(time_pos, basis_5);
xlabel('time ops');
ylabel('sq wave');
title('5th basis vectors')
axis([0 20 -1 1]);
```

```
hold on

subplot(7, 1, 6)
plot(time_pos, basis_6);
xlabel('time ops');
ylabel('sq wave');
title('6th basis vectors')
axis([0 20 -1 1]);
hold on

subplot(7, 1, 7)
plot(time_pos, basis_30);
xlabel('time ops');
ylabel('sq wave');
title('30th basis vectors')
axis([0 20 -1 1]);
hold on

figure(2)
norm_B_unnorm=B_unnorm;
for i=1:size(B_unnorm,1)
norm_B_unnorm(i,:)=B_unnorm(i,:)/norm(B_unnorm(i,:));
end

coeff = zeros(size(B_unnorm,1),1);
for i = 1:size(B_unnorm,1)
    coeff(i,:) = dot(sq_wave,B_unnorm(i,:)/dot(B_unnorm(i,:),B_unnorm(i,:)));
end
[~,coeff_i]=sort(coeff);

approx = 0;
for i = 1:6
    [B_ind, ~] = find(coeff_i==i);
    approx = approx+ coeff(i, :) *B_unnorm(B_ind, :);
end
[B_ind, ~] = find(30);
approx = approx+ coeff(30, :) *B_unnorm(B_ind, :);

plot(time_pos, approx);
hold on
plot(time_pos, sq_wave);
xlabel('time op');
ylabel('sq wave');
title('Approximation');
end
```

MATLAB code for P1.12

P1.12: p12_main.m

```
close all; clc;

origin=[0 0]';
x = [3 2]';
v0 = [-2 -4]'; % offset
v = [-2 5]'; % slope
t=linspace(-3,3);

A=-v.*t+v0;


figure(1);

%l2 norm----------------------------------------
xlabel('x');
ylabel('y');
plot(3,2,'o');
text(3.5,2,'x');
hold on
plot(A(1,:), A(2,:));
axis equal;
l2_norm=sqrt(dot(A-x,A-x));
[~,ind2]=min(l2_norm);
drawcircle(x(1, :), x(2, :), l2_norm(ind2))
plot(A(1,ind2),A(2,ind2),'o');
```

```
text(A(1,ind2)-2,A(2,ind2),'y_(_2_)');
axis equal;
axis([-15 15 -15 15])
title('l2 norm ball');

figure(2);
%l1 norm-------------------------------------------------------
subplot(1,2,1);
xlabel('x');
ylabel('y');
plot(3,2,'o');
text(3.5,2,'x');
hold on
plot(A(1,:), A(2,:));
axis equal;
l1_norm=sum(abs(A-x));
[~,ind1]=min(l1_norm);
drawdiamond(x(1, :), x(2, :), l1_norm(ind1))
plot(A(1,ind1),A(2,ind1),'o');
text(A(1,ind2)-2,A(2,ind2),'y_(_2_)');
axis equal;
axis([-15 15 -15 15])
title('l1 norm ball');

%linf norm----------------------------------------------------
subplot(1,2,2);
xlabel('x');
ylabel('y');
plot(3,2,'o');
text(3.5,2,'x');
hold on
plot(A(1,:), A(2,:));
axis equal;
linf_norm=max(abs(A-x));
[~,ind_inf]=min(linf_norm);
drawsquare(x(1, :), x(2, :), linf_norm(ind_inf))
plot(A(1,ind_inf),A(2,ind_inf),'o');
text(A(1,ind2)-2,A(2,ind2),'y_(_2_)');
axis equal;
axis([-15 15 -15 15])
title('linf norm ball');

%(e)---------------------------------------------------------
[y1,r]=proj_cvx(x,v0,v,1);
[yinf,r]=proj_cvx(x,v0,v,inf);
y1
yinf

%------------------------------------------------------------
function drawcircle(x,y,r)
%(x,y) -center
%r - radius
th = 0:pi/50:2*pi;
xunit = r * cos(th) + x;
yunit = r * sin(th) + y;
plot(xunit, yunit);
end

function drawdiamond(x,y,r)
%(x,y) -center
%r - from center to vertices
ver=[x-r y;x y-r;x+r y;x y+r];
pgon=polyshape(ver);
plot(pgon);
end

function drawsquare(x,y,r)
%(x,y) -center
%r - from center to 4 sides
ver=[x-r y+r;x-r y-r;x+r y-r;x+r y+r];
pgon=polyshape(ver);
plot(pgon);
end

function [y, r] = proj_cvx (x, v0 , v, nrm)%% x, v0 and v must be column vectors
objtv = @(y) norm (x-y, nrm); %% objective is L_2 norm
```

```
cvx_begin
variable y(2) %% 2-d variable we are optimizing over
variable t(1) %% real valued parameter that defines
minimize ( objtv (y)) %% defining the objective
subject to
v0 + t*v == y %% the projection y must be in set A
cvx_end
r = objtv (y); %% minimum value of the objective
end
```

MATLAB code for P1.13

P1.13: p13_main.m

```

%Gram-Schmidt
syms x;
V=[x.^0;x.^1;x.^2;x.^3;x.^4;x.^5];
U=sin(x);

%(a)
Q=V;
for i=1:6
    if(i>1)
        for j=1:i-1
            innerprod = innerprod_(Q(j),V(i));%int(V(i)*Q(j),-pi,pi);
           Q(i,1)=Q(i,1)-innerprod*Q(j);
        end
    end
    norm=sqrt(innerprod_(Q(i),Q(i)));
    Q(i)=Q(i)/norm;
end


%(b)
A=zeros(6,1);
for i=1:6
    A(i,1)=innerprod_(U,Q(i))/innerprod_(Q(i),Q(i));
end

f=0;
for i=1:6
   f= f+A(i)*Q(i);
end

hold on
fplot(f,[-pi,pi])

hold on
fplot(U,[-pi,pi])

hold on
fplot(x-x^3/6+x^5/120, [-pi,pi])

legend('Orthonomal basis approximation','sinx','Taylor approximation')

function e = innerprod_(v_,q_)
e=int(v_*q_,-pi,pi);
end
```