```
%% Import Data:X is the training set,label is the label of images,y is the test vector;
tempload = load('EX7.mat');
X = tempload.X;  Z=double(X)/255.0; %Normalized
y=tempload.y;    h=double(y)/255.0; %Normalized
xlabel=tempload.label;
n=784;  d=60000;                    %Dimension
tempmatrix=Z*Z'; [~,D] = eig(tempmatrix); tempL=2*D(784,784)/n;
%% Define objective function, quadratic estimation
f=@(x) sum((h-Z*x).^2)/n;
g=@(x,lambda) lambda*sum(abs(x));
z=@(x,L) x-Z'*(Z*x-h)*2/L/n;
%F=@(x,lambda) f(x)+g(x,lambda);
F=@(x,lambda) sum((h-Z*x).^2)/n+lambda*sum(abs(x));
G=@(x,xc,L,lambda) sum((x-xc).^2)*L/2.0 + sum((h-Z*xc).^2)/n+lambda*sum(abs(x)) -2/n*(h-
Z*xc)'*Z*(x-xc) ;
%% Iterate to solve
tempF=zeros(1,2002); tempk=zeros(1,100);
fu=zeros(1,100); gu=zeros(1,100); Fu=zeros(1,100); ilambda=1; variable=zeros(1,100);
for lambda=0.002:0.002:0.2
    %初始迭代参数;
    x0=zeros(d,1);  x0(1)=1.0; k=0; %  iterL=0.5; yita=1.1;
    tempF(k+1)=F(x0,lambda);
    while k<=2000
        zk=z(x0,tempL);%求解z,便于求解w+;
        xk=argmin(zk,tempL,lambda);%求解w+;
        %ik=0;%求解最小的ik;
        %zk=z(x0,iterL);%求解z,便于求解w+;
        %xk=argmin(zk,iterL,lambda);%求解w+;
        %while F(xk,lambda)>G(xk,x0,iterL,lambda) && ik<=200
        %    ik=ik+1;
        %    iterL=yita*iterL;
        %    zk=z(x0,iterL);
        %    xk=argmin(zk,iterL,lambda);
        %end
        x0=xk;
        k=k+1;
        tempF(k+1)=F(x0,lambda);
        if tempF(k+1)>tempF(k)
            break
        end
    end
    tempk(ilambda)=k;
    variable(ilambda)=lambda; fu(ilambda)=f(x0); gu(ilambda)=g(x0,lambda);
Fu(ilambda)=fu(ilambda)+gu(ilambda);
    ilambda=ilambda+1;
end
%% Input images
scatter(variable,fu,'r','p','filled');
hold on;
scatter(variable,gu,'k','+');
hold on;
scatter(variable,Fu,'b','s');
legend('f(u)','g(u)','F(u)');
%% predicted
max=0; where=0;
for imax=1:60000
    if x0(imax)>max
```

```
        where=imax;
        max=x0(imax);
    end
end
xlabel(where)
```