CS/ECE/ME 532
Homework 5: The SVD

In this homework set you will work with an analyze the dasaset `jesterdata.mat`, which is available on the moodle site. The dataset contains an $m = 100$ by $n = 7200$ dimensional matrix X. Each row of X corresponds to a joke, and each column corresponds to a user. Each of the users rated the quality of each joke on a scale of $[-10, 10]$.

1. Suppose that you work for a company that makes joke recommendations to customers. You are given a large dataset X of jokes and ratings. It contains n reviews for each of m jokes. The reviews were generated by n users who represent a diverse set of tastes. Each reviewer rated every movie on a scale of $[-10, 10]$. A new customer has rated $k = 25$ of the jokes, and the goal is to predict another joke that the customer will like based on her k ratings. Use the first $n = 20$ columns of X for this prediction problem (so that the problem is overdetermined). Her ratings are contained in the file `newuser.mat`, also on moodle, in a vector b. The jokes she didn't rate are indicated by a (false) score of -99. Compare your predictions to her complete set of ratings, contained in the vector `trueb`. Her actual favorite joke was number 29. Does it seem like your predictor is working well?

2. Repeat the prediction problem above, but this time use the entire X matrix. Note that now the problem is underdetermined. Explain how you will solve this prediction problem and apply it to the data. Does it seem like your predictor is working? How does it compare to the first method based on only 20 users?

3. Propose a method for finding one other user that seems to give the best predictions for the new users. How well does this approach perform? Now try to find the best two users to predict the new user.

4. Use the Matlab function `svd` with the 'economy size' option to compute the SVD of $X = U\Sigma V^T$. Plot the sprectrum of X. What is the rank of X? How many dimensions seem important? What does this tell us about the jokes and users?

5. Visualize the dataset by projecting the columns and rows on to the first three principle component directions. Use the `rotate` tool in the Matlab plot to get different views of the three dimensional projections. Discuss the structure of the projections and what it might tell us about the jokes and users.

6. One easy way to compute the first principle component for large datasets like this is the so-called power method (see `http://en.wikipedia.org/wiki/Power_iteration`). Explain the power method and why it works. Write your own code to implement the power method in Matlab and use it to compute the first column of U and V in the SVD of X. Does it produce the same result as Matlab's built-in `svd` function?

7. The power method is based on an initial starting vector. Give one example of a starting vector for which the power method will fail to find the first left and right singular vectors in this problem.

```
%% Homework 5
clear all; close all; clc;
% First load all the relevant files
load jesterdata.mat;
load newuser.mat;

%% Part 1
% Lets begin by finding the labeled jokes

fprintf(' ---------- Part 1 ------------ \n');

unlabeledIndices = find(b == -99);
labeledIndices = setdiff(1:100, unlabeledIndices);

% labels of answered questions
bAnswered = b(labeledIndices);

% Rows corresponding to answered questions
Xanswered = X(labeledIndices,:);

% Now use the matrix only for the first 20 users and those columns which
% have been labeled to estimate a weight vector

Xsmall = X(labeledIndices, 1:20);

beta1 = inv(Xsmall'*Xsmall)*Xsmall'*bAnswered;

% The estimate for our whole data will be X*beta1
predictions1 = X(:,1:20)*beta1;

fprintf('The prediction error is %.2f per joke\n', norm(trueb - predictions1)/100);

fprintf('The predicted favorite joke is number %d \n', find(predictions1 == max(predictions1),1));

%% Part 2
% We can try to find different combinations of 25 users that gives us the
% best results. This is a really costly computations as there are
% (7200 choose 25) = 1.68 e71 different combinations of columns. Therefore
% and exhaustive search will not be possible

fprintf('\n  ---------- Part 2 ------------ \n');

nTries  = 10000;
bestErr = Inf;
bestCols = zeros(1,25);

for nIter = 1:nTries
    pickedCols = sort(randsample(7200, 25));
    Xp = Xanswered(:,pickedCols);
    betap = inv(Xp'*Xp)*Xp'*bAnswered;
    bpred = X(:,pickedCols)*betap;
    currErr = norm(bpred - trueb)/100;
    if currErr < bestErr
        bestErr = currErr;
        bestCols = pickedCols;
        bestbeta = betap;
    end
end

fprintf('After %d 25-tuples searched, best error found is %3.2f \n', nTries, bestErr);

predictions2 = X(:,bestCols)*bestbeta;

fprintf('The predicted favorite joke is number %d \n', find(predictions2 == max(predictions2),1));
```

```
%% Part 3a
% Let's first search over all the 7200 columns to see which one best
% predicts the users tastes

fprintf('\n  ---------- Part 3a ------------ \n');


bestErr = Inf;
for col = 1:7200
   betacol = inv(Xanswered(:,col)'*Xanswered(:,col))*Xanswered(:,col)'*bAnswered;
   colpred = X(:,col)*betacol;
   currErr = norm(colpred - trueb)/100;
   if currErr < bestErr
       bestErr = currErr;
       bestCol = col;
       bestbeta = betacol;
   end
end

fprintf('After searching all columns, best error found is %3.2f, for column %d\n', bestErr, bestCol);

predictions3a = X(:,bestCol)*bestbeta;

fprintf('The predicted favorite joke is number %d \n', find(predictions3a == max(predictions3a),1));

%% Part 3b
% Now we seraching through all (7200 choose 2) columns is still expensive.
% Instead, let's use a greedy method and find the best additional column to
% the one already found in part 3a

fprintf('\n  ---------- Part 3b ------------ \n');

bestErr = Inf;
for col = 1:7200
    inds = [col, bestCol];
    betacol = inv(Xanswered(:,inds)'*Xanswered(:,inds))*Xanswered(:,inds)'*bAnswered;
    colpred  = X(:, inds)*betacol;
    currErr = norm(colpred - trueb)/100;
    if currErr < bestErr
        bestErr = currErr;
        bestCol2 = col;
        bestbeta = betacol;
    end
end

inds  = [bestCol, bestCol2];

fprintf('After greedy search, best error found is %3.2f, for columns %d and %d\n', bestErr, bestCol2,
bestCol);

predictions3b = X(:,inds)*bestbeta;

fprintf('The predicted favorite joke is number %d \n', find(predictions3b == max(predictions3b),1));

%% Part 4
fprintf('\n  ---------- Part 4 ------------ \n');

[U,S,V] = svd(X, 'econ');
eigvals = diag(S);
Slow = zeros(size(S));
for k = 1:3
    Slow(k,k) = S(k,k);
end
rankX = length( find(abs(eigvals) > 0));
```

```
fprintf('The rank of X is %d\n', rankX);

figure;
plot(eigvals);
title('Eigenvalues of X in descending order');

fprintf('From the figure, it looks to be approximately a rank 6 matrix \n');

%% Part 5
% Now, we already have the SVD of X, project onto the first three columns
% of U

fprintf('\n  ---------- Part 5------------- \n');

X3drows = S(1:3,1:3)*V(:,1:3)';
X3dcols = U(:,1:3)*S(1:3,1:3);
X3dcols = X3dcols';
figure;
scatter3(X3drows(1,:), X3drows(2,:), X3drows(3,:), 0.1);
figure;
scatter3(X3dcols(1,:), X3dcols(2,:), X3dcols(3,:));


%% Part 6 and 8
% X is not square but note the the eigenvalues of X*X' are the square of
% those of X, so we can proceed with X2 = X*X'

fprintf('\n  ---------- Parts 6 and 8 ------------- \n');

% Random starting point
b = normrnd(0,1,100,1);
% Initialized with an already known eigenvector: this will lead to not
% finding the correct eignenvalue
%b = U(:,2);
b0 = zeros(100,1);
nIters = 0;
MAXITERS = 1e4;
TOL = 1e-5;
X2 = X*X';
while ( norm(b - b0) > TOL && nIters <= MAXITERS)
    b0 = b;
    temp = X2*b;
    b = temp./norm(temp);
    nIters = nIters + 1;
end

if nIters > MAXITERS && norm(b - b0) > TOL
    fprintf('The power iteration did not converge\n');
else
    fprintf('The power iteration converged in %d iterations\n', nIters);
    fprintf('The estimated eigenvalue is %3.2f, the true eigenvalue is %3.2f\n', sqrt(mean(X2*b./b)),
S(1,1));
end
```