Homework Set 2, CPSC 8420, Fall 2023

Your Name

Due 10/26/2023, Thursday, 11:59PM EST

Problem 1

For PCA, from the perspective of maximizing variance, please show that the solution of $\boldsymbol{\phi}$ to maximize $\|\mathbf{X}\boldsymbol{\phi}\|_2^2, s.t.\ \|\boldsymbol{\phi}\|_2 = 1$ is exactly the first column of $\mathbf{U}$, where $[\mathbf{U}, \mathbf{S}] = svd(\mathbf{X}^T\mathbf{X})$. (Note: you need prove why it is optimal than any other reasonable combinations of $\mathbf{U}_i$, say $\hat{\boldsymbol{\phi}} = 0.8 * \mathbf{U}(:, 1) + 0.6 * \mathbf{U}(:, 2)$ which also satisfies $\|\hat{\boldsymbol{\phi}}\|_2 = 1$.)

Problem 2

Given matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ (assume each column is centered already), where n denotes sample size while p feature size. To conduct PCA, we need find eigenvectors to the largest eigenvalues of $\mathbf{X}^T\mathbf{X}$, where usually the complexity is $\mathcal{O}(p^3)$. Apparently when $n \ll p$, this is not economic when p is large. Please consider conducting PCA based on $\mathbf{X}\mathbf{X}^T$ and obtain the eigenvectors for $\mathbf{X}^T\mathbf{X}$ accordingly and use experiment to demonstrate the acceleration.

Problem 3

Let's revisit Least Squares Problem: $\underset{\boldsymbol{\beta}}{\text{minimize}}\ \frac{1}{2}\|\mathbf{y}-\mathbf{A}\boldsymbol{\beta}\|_2^2$, where $\mathbf{A}\in\mathbb{R}^{n\times p}$.

1. Please show that if $p>n$, then vanilla solution $(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}$ is not applicable any more.
2. Let's assume $\mathbf{A}=[1,2,4;1,3,5;1,7,7;1,8,9], \mathbf{y}=[1;2;3;4]$. Please show via experiment results that Gradient Descent method will obtain the optimal solution with Linear Convergence rate if the learning rate is fixed to be $\frac{1}{\sigma_{max}(\mathbf{A}^T\mathbf{A})}$, and $\boldsymbol{\beta}_0=[0;0;0]$.
3. Now let's consider ridge regression: $\underset{\boldsymbol{\beta}}{\text{minimize}}\ \frac{1}{2}\|\mathbf{y}-\mathbf{A}\boldsymbol{\beta}\|_2^2+\frac{\lambda}{2}\|\boldsymbol{\beta}\|_2^2$, where $\mathbf{A},\mathbf{y},\boldsymbol{\beta}_0$ remains the same as above while learning rate is fixed to be $\frac{1}{\lambda+\sigma_{max}(\mathbf{A}^T\mathbf{A})}$ where λ varies from $0.1,1,10,100,200$, please show that Gradient Descent method with larger λ converges faster.

Problem 4

We consider matrix completion problem. As we discussed in class, the main issue of *softImpute (Matrix Completion via Iterative Soft-Thresholded SVD)* is when the matrix size is large, conducting *SVD* is computational demanding. Let's recall the original problem where $\mathbf{X},\mathbf{Z}\in\mathbb{R}^{n\times d}$:

$$\min_{\mathbf{Z}}\frac{1}{2}\|P_\Omega(\mathbf{X})-P_\Omega(\mathbf{Z})\|_F^2+\lambda\|\mathbf{Z}\|_* \tag{1}$$

People have found that instead of finding optimal $\mathbf{Z}$, it might be better to make use of *Burer-Monteiro* method to optimize two matrices $\mathbf{A}\in\mathbb{R}^{n\times r},\mathbf{B}\in\mathbb{R}^{d\times r}(r\geq rank(\mathbf{Z}^*))$ such that $\mathbf{AB}^T=\mathbf{Z}$. The new objective is:

$$\min_{\mathbf{A},\mathbf{B}}\frac{1}{2}\|P_\Omega(\mathbf{X}-\mathbf{AB}^T)\|_F^2+\frac{\lambda}{2}(\|\mathbf{A}\|_F^2+\|\mathbf{B}\|_F^2). \tag{2}$$

- Assume $[\mathbf{U},\boldsymbol{\Sigma},\mathbf{V}]=svd(\mathbf{Z})$, show that if $\mathbf{A}=\mathbf{U}\boldsymbol{\Sigma}^{\frac{1}{2}},\mathbf{B}=\mathbf{V}\boldsymbol{\Sigma}^{\frac{1}{2}}$, then Eq. (2) is equivalent to Eq. (1).
- The *Burer-Monteiro* method suggests if we can find $\mathbf{A}^*,\mathbf{B}^*$, then the optimal $\mathbf{Z}$ to Eq. (1) can be recovered by $\mathbf{A}^*\mathbf{B}^{*T}$. It boils down to solve Eq. (2). Show that we can make use of least squares with ridge regression to update $\mathbf{A},\mathbf{B}$ row by row in an alternating minimization manner as below. Assume $n=d=2000, r=200$, please write program to find $\mathbf{Z}^*$.

$T\leftarrow 100, i\leftarrow 1$ % you can also set T to be other number instead of 100
if $i\leq T$ **then**
  update A row by row while fixing B
  update B row by row while fixing A
  $i\leftarrow i+1$
end if

Problem 1:

For matrix with self centered data, max variance:

$$\max \|X\phi\|_2^2 \qquad [\phi \rightarrow \text{vector}]$$
$$\|\phi\|_2 = 1$$

To prove: solution of ϕ for above eq^n is exactly the first column of V where $[U, S] = \text{svd}(X^TX)$

→ we can modify $\|X\phi\|_2^2$ as

$$[X\phi]^T [X\phi] = \phi^T X^T X \phi$$

As X^TX is an spd matrix,
for svd of X^TX, we have $V = U$ in $U\Sigma V^T$.
Also, singular values are absolute values of eigen values.

Also, columns of U & V are eigen vectors of A.

Consider,

Case 1: when $\phi = aU_1 + bU_2$.

where a and b are real numbers

s.t. $a^2 + b^2 = 1$

So, $\|\phi\|^2 = \phi^T \cdot \phi$

$= (aU_1 + bU_2)^T (aU_1 + bU_2)$

$= a^2 U_1^T U_1 + b^2 U_2^T U_2 + 2ab\, U_1^T U_2$

As columns of U ($U_1, U_2 \ldots$) are orthonormal vectors, $U_1^T U_2 = 0$

and $U_1^T \cdot U_1 = I$

$U_2^T U_2 = I$

So, $\|\phi\|^2 = 1$.

i.e. $\phi = aU_1 + bU_2$ $(a, b \in R)$

s.t. $a^2 + b^2 = 1$

satisfy the given constraint

If we substitute the value of ϕ in : $\phi^T X^T X \phi$

$$= \phi^T [U \Sigma U^T] \phi$$

$$= \phi^T \left[\sum_{i=1}^{p} \sigma_i u_i u_i^T\right] \phi$$

$$= \phi^T [\sigma_1 u_1 u_1^T + \sigma_2 u_2 u_2^T + \sigma_3 u_3 u_3^T + \sigma_4 u_4 u_4^T \ldots] \phi$$

$$= (a u_1 + b u_2)^T (\sigma_1 u_1 u_1^T + \sigma_2 u_2 u_2^T + \ldots)(a u_1 + b u_2)$$

$$= (a u_1^T \sigma_1 u_1 u_1^T + b u_2^T \sigma_2 u_2 u_2^T + a u_1^T \sigma_2 u_2 u_2^T + \ldots)(a u_1 + b u_2)$$

$$= (a^2 \sigma_1 u_1^T u_1 u_1^T u_1 + b^2 \sigma_2 u_2^T u_2 u_2^T u_2 + a u_1^T \sigma_2 u_2 u_2^T u_1 + \ldots)$$

As U_1 & U_2 are orthonormal vectors, all terms with $U_1^T \cdot U_2$ & $U_2^T \cdot U_1 = 0$.

So, we are left with:

$$a^2 \sigma_1 + b^2 \sigma_2$$

$$\sigma_1 \left(a^2 + b^2 \cdot \frac{\sigma_2}{\sigma_1} \right) - (1)$$

As we know that $\sigma_1 > \sigma_2$ (in Σ matrix, singular values are arranged along the diagonal in a decreasing order)

So, from eqn (1):

as $a^2 + b^2 = 1$

$$a^2 + b^2 \left(\frac{\sigma_2}{\sigma_1} \right) < 1$$

$$\Rightarrow \sigma_1 \left(a^2 + b^2 \frac{\sigma_2}{\sigma_1} \right) < \sigma_1 - (2)$$

SUNDAY 12

APRIL 2023

Case 2: Consider if $\phi = U_1$

(i.e. first column of U)

$\|\phi\|_2^2 = 1$ (orthonormal vector)

So, putting $\phi = U_1$ in:

$$\phi^T [U \Sigma U^T] \phi$$

$$= \phi^T [\sigma_1 U_1 U_1^T + \sigma_2 U_2 U_2^T + \sigma_3 U_3 U_3^T + \cdots] \phi$$

$$= U_1^T [\sigma_1 U_1 U_1^T + \sigma_2 U_2 U_2^T + \sigma_3 U_3 U_3^T + \cdots] U_1$$

$$= \sigma_1 U_1^T U_1 U_1^T U_1 + \sigma_2 U_1^T U_2 U_2^T U_1 + \sigma_3 U_1^T U_3 U_3^T U_1 + \cdots$$

As columns of U matrix are orthonormal,

$U_{i(i \neq 1)}^T \cdot U_1$ & $U_1^T \cdot U_{i(i \neq 1)} = 0$.

So, we get simplified soln:

$$= \sigma_1 U_1^T U_1 U_1^T U_1$$

$$= \sigma_1 \quad \text{— (3)}$$

Comparing $W = U$ in our argument to a general combination of vectors from U,

eq 3 gives:

$$\arg \max_{\|\phi\|_2^2 = 1} \|X\phi\|_2^2 \Rightarrow W = aU_1 + bU_2$$

(s.t. $a, b \in R$ & $a^2 + b^2 = 1$)

gives $\sigma_1 \left(a^2 + b^2 \frac{\sigma_2}{\sigma_1}\right)$.

which is less than σ_1, when $W = U_1$.

So, max. variance is achieved when ϕ = first column of U, where $U, S] = svd(X^T X)$.

APRIL 2023

Problem 2

Let $A = X^T X$ — (1)

& $B = X X^T$.

Let U be eigen vectors of B.

$$X X^T U = \lambda U$$

multiply both sides by X^T:

$$X^T X X^T U = \lambda X^T U$$

from (1), $X^T X = A$

$$A(X^T U) = \lambda (X^T U)$$

So, we can conclude that $X^T U$ are eigen vectors of A.

So, we can find eigen vectors of B, which have the computational

complexity of $O(n^3)$ instead of $O(p^3)$, (where $X \in R^{n \times p}$).
These eigen vectors can be used to find eigen vectors of (X^TX).

To convert eigen vectors of XX^T to those of X^TX

→ Multiply eigen vectors of $B(XX^T)$ by X^T and then normalize.

i.e. eigen vectors of A: X^TU
But X^TU is not normalized.
So, the matrix which gives us eigen vectors of A is X^TU with all of it's column vectors normalized.

Problem 2

```
% Experimental Setup
n = 100; % number of samples
p = 10000; % number of features

% Generate random data for matrix X (centered)
X = randn(n, p);

% Start timer for PCA using X^T X
tic;

% Compute covariance matrix X^T X
C1 = X' * X;

% Compute eigenvectors and eigenvalues of X^T X
[V1, D1] = eig(C1);

% Sort eigenvectors based on eigenvalues in descending order
[~, idx1] = sort(diag(D1), 'descend');
V1 = V1(:, idx1);

time_XTX = toc;

% Start timer for PCA using XX^T
tic;

% Compute covariance matrix XX^T
C2 = X * X';

% Compute eigenvectors and eigenvalues of XX^T
[V2, D2] = eig(C2);

% Convert eigenvectors of XX^T to those of X^T X
V2 = X' * V2;

% Normalize the eigenvectors
for i = 1:n
    V2(:,i) = V2(:,i) / norm(V2(:,i));
end

% Sort eigenvectors based on eigenvalues in descending order
[~, idx2] = sort(diag(D2), 'descend');
V2 = V2(:, idx2);

time_XXT = toc;

% Display the results
fprintf('Time for PCA using X^T X: %.4f seconds\n', time_XTX);
```

```
Time for PCA using X^T X: 71.1933 seconds
```

```
fprintf('Time for PCA using XX^T: %.4f seconds\n', time_XXT);
```

```
Time for PCA using XX^T: 0.0941 seconds
```

Problem 3 (1)

Condition: n < p

```
n = 10; % number of samples
p = 10000; % number of features

% Generate random data for matrix X (centered)
A = randn(n, p);

M = A'*A;
disp("Size of the square matrix:");
```

```
Size of the square matrix:
```

```
disp(size(M));
```

```
       10000       10000
```

```
disp("Rank of the square matrix");
```

```
Rank of the square matrix
```

```
disp(rank(M));
```

```
    10
```

Condition n > p:

```
n = 1000; % number of samples
p = 100; % number of features

% Generate random data for matrix X (centered)
A = randn(n, p);

M = A'*A;
disp("Size of the square matrix:");
```

```
Size of the square matrix:
```

```
disp(size(M));
```

```
   100   100
```

```
disp("Rank of the square matrix");
```

```
Rank of the square matrix
```

```
disp(rank(M));
```

```
   100
```

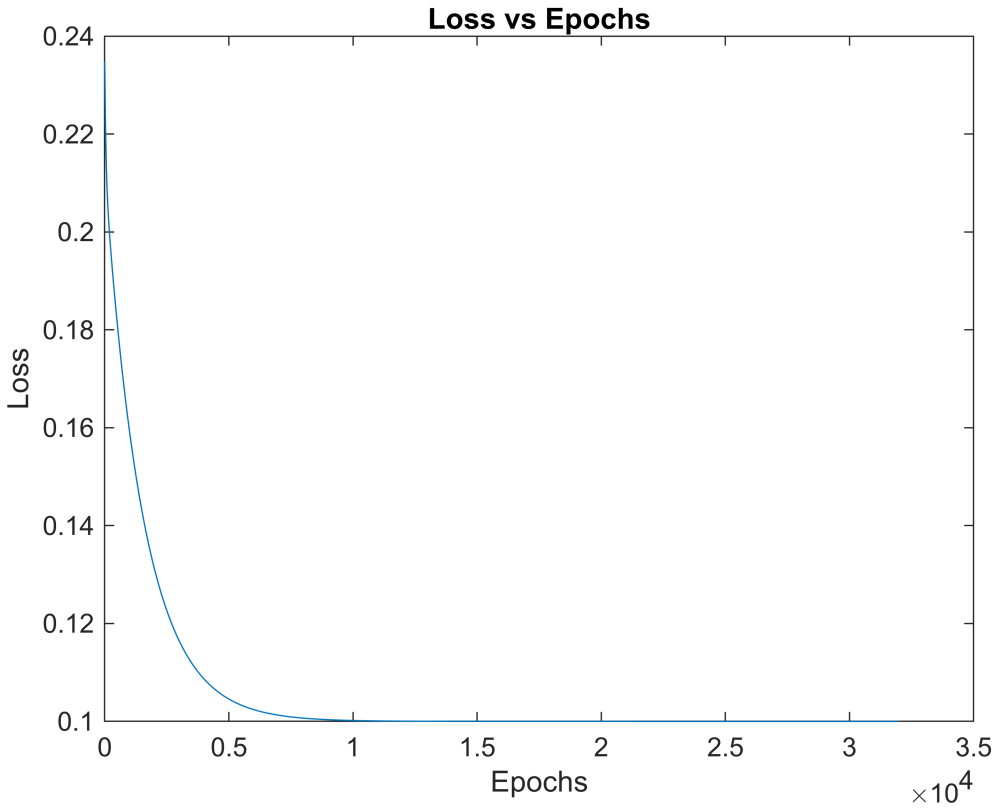
As the rank of the matrix < size of the matrix, p > n condition leads to a non invertible matrix. So, the closed form solution fails.

Problem 3 (2)

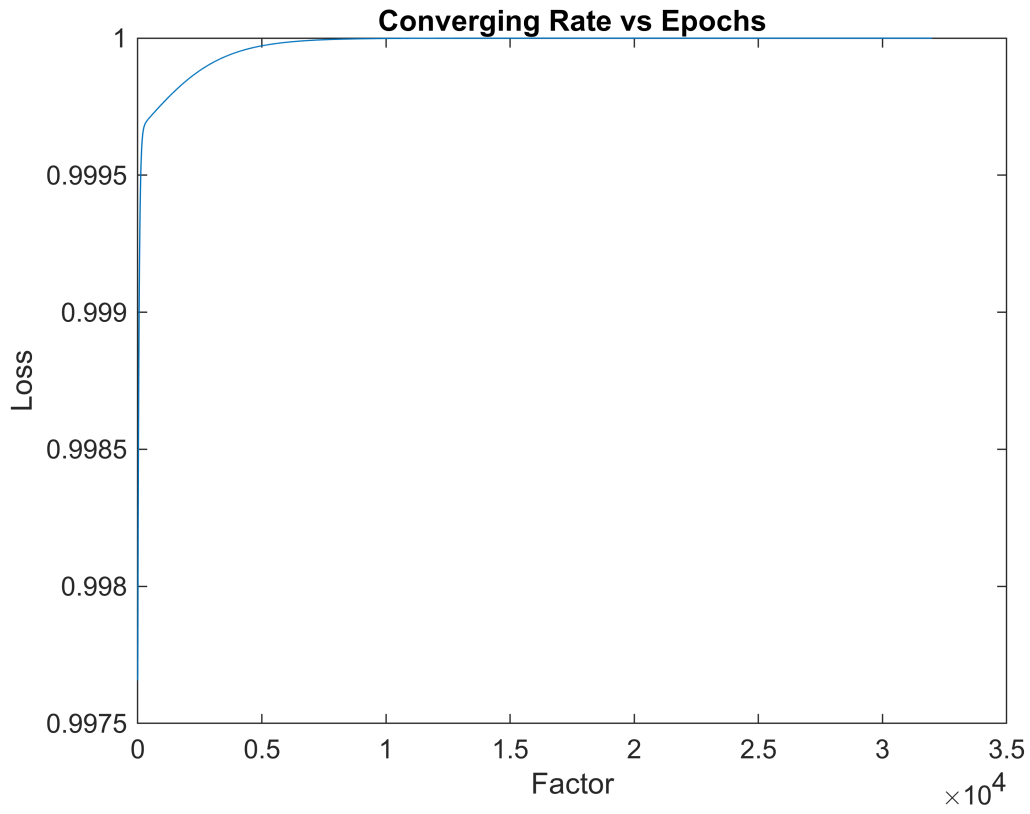
```
% Given A, y, lr and initial weights
A = [1 2 4; 1 3 5; 1 7 7; 1 8 9];
y = [1; 2; 3; 4];

% Finding the maximum singular value of A
max_singular_value = max(svd(A'*A));

% Learning rate is fixed as 1/max singular value
lr = 1/max_singular_value;
weights = [0; 0; 0];

% Iinitializing a loss row vector to track progress
loss = [];

% Implementing Vanilla Gradient Descent (32k epochs)
for i = 1:32000
    grad = A'*(A*weights - y);
    weights = weights - grad*lr;
    loss(i) = (A*weights - y)' * (A*weights - y);
end

convergence_factor = loss(2:end) ./ loss(1:end-1);
display(convergence_factor);
```

```
convergence_factor = 1×31999
    0.9977    0.9977    0.9977    0.9978    0.9978    0.9978    0.9979    0.9979 ···
```

```
% Plotting the Loss obtained with time
plot(loss);
title("Loss vs Epochs");
xlabel("Epochs");
ylabel("Loss");
```




```
% Plotting the convergence factor for each epoch
plot(convergence_factor);
title("Converging Rate vs Epochs");
xlabel("Factor");
ylabel("Loss");
```




```
%Verifying the results obtained from closed form solution
weights2 = pinv(A'*A)*A'*y;
```

```
display(weights2);
```

```
weights2 = 3×1
   -1.0000
    0.0333
    0.5333
```

```
display(weights);
```

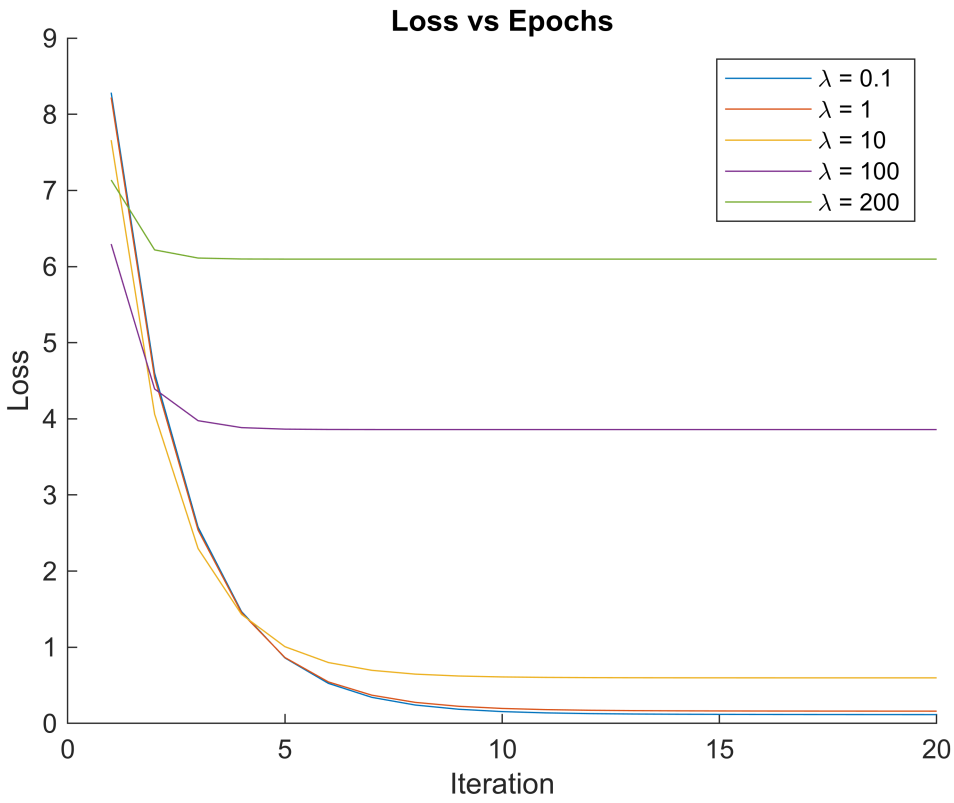
```
weights = 3×1
   -1.0000
    0.0333
    0.5333
```

From the above weight matrices, we can deduce that Gradient Descent method obtains the optimal solution with almost a Linear Convergence rate with the given learning rate.

Problem 3 (3)

```
% Define the matrix A and vector y
```

```
A = [1, 2, 4; 1, 3, 5; 1, 7, 7; 1, 8, 9];
y = [1; 2; 3; 4];

% Define a range of λ values
lambda_values = [0.1, 1, 10, 100, 200];



% Define the number of iterations
num_iterations = 20;

% Initialize variables to store losses
losses = zeros(length(lambda_values), num_iterations);


% Perform gradient descent for each λ
for k = 1:length(lambda_values)
    lambda = lambda_values(k);
    alpha = 1 / (max(max(A' * A)) + lambda_values(k));

    % Initialize θ0 to [0; 0; 0]
    theta = [0; 0; 0];
    for iteration = 1:num_iterations
        % Compute the gradient for ridge regression
        gradient = -A' * (y - A * theta) + lambda * theta;

        % Update θ using the gradient and learning rate
        theta = theta - alpha * gradient;

        % Calculate the cost (ridge regression loss)
        loss = (1 / (2 )) * norm(A * theta - y)^2 + (theta' * theta) * (lambda / 2);

        % Store the loss
        losses(k, iteration) = loss;
    end
end

% Plot the loss curves for different λ values
figure;
hold on;
for k = 1:length(lambda_values)
%     plot(1:num_iterations, losses(k, :), 'DisplayName', ['λ = ', num2str(lambda_values(k))]);
    semilogy(1:num_iterations, losses(k, :), 'DisplayName', ['\lambda = ', num2str(lambda_value
end

hold off;
xlabel('Iteration');
ylabel('Loss');
title("Loss vs Epochs");
legend;
```




```
%Verifying the results obtained from closed form solution
weights2 = pinv(A'*A + lambda_values(5)*eye(size(A, 2)))*A'*y;
```

To verify, let us compare weights2 and theta

```
display(weights2);
```

```
weights2 = 3×1
    0.0195
    0.1231
    0.1423
```

```
display(theta);
```

```
theta = 3×1
    0.0195
    0.1231
    0.1423
```

Problem 4

$$\min_{A,B} \frac{1}{2} \| P_\Omega (X - AB^T) \|_F^2 + \frac{\lambda}{2} (\|A\|_F^2 + \|B\|_F^2) \quad -(1)$$

Given: $A = U\Sigma^{1/2}$; $B = V\Sigma^{1/2}$

$$[U, \Sigma, V] = svd(Z)$$

$$P_\Omega (X - AB^T) = P_\Omega (X) - P_\Omega (AB^T)$$

$$= P_\Omega (X) - P_\Omega (Z) \quad -(2)$$

$$\|A\|_F^2 = tr(A^T A) = tr\left[(U\Sigma^{1/2})^T (U\Sigma^{1/2}) \right]$$

$$= tr\left[(\Sigma^{1/2})^T U^T U \Sigma^{1/2} \right]$$

As $U^T U = I$,

$$\|A\|_F^2 = tr\left[(\Sigma^{1/2})^T (\Sigma^{1/2}) \right] = tr\left[\|\Sigma\|_1 \right] \quad -(3)$$

Similarly, $B^T B = (V\Sigma^{1/2})^T (V\Sigma^{1/2})$

$$\|B\|_F^2 = tr(B^T B) = tr\left[\|\Sigma\|_1 \right] \quad -(4)$$

Substituting these values from 2, 3 & 4. in (1),

$$\min_{A,B} \frac{1}{2} \| P_R(X) - P_R(Z) \|_F^2 + \frac{\lambda}{2} (tr(\Sigma) + tr(\Sigma))$$

$$= \min_{A,B} \frac{1}{2} \| P_R(X) - P_R(Z) \|_F^2 + \lambda \, tr(\Sigma)$$

$tr(\Sigma)$ is the trace of Σ matrix, i.e. sum of singular values
This is also = nuclear norm of Z.

So, above eq^n can be re written as:

$$\min_{A,B} \frac{1}{2} \| P_R(X) - P_R(Z) \|_F^2 + \lambda \| Z \|_*$$

Problem 4 (2)

```
% Given A, y, lambda, learning rate and initial weights
% Parameters
n = 2000;
d = 2000;
r = 200;

% Generate a random X matrix for testing
X = randn(n, d);

% Set the regularization parameter
lambda = 0.1;

% Initialize A and B
A = randn(n, r);
B = randn(d, r);

% Maximum number of iterations
T = 100;

num_iterations = 100;

% LEARNING RATE FORMULA TBD
lr = 1 / (max(max(A' * A)) + lambda);%1/(max_singular_value + lambda(k));
for i = 1:T
    A = A + lr * ((X  - A*B')*B - A.*lambda);
    B = B + lr * ((X  - A*B')*A - B.*lambda);
end
```