Homework 7 Solutions
EE 263 Stanford University Summer 2019
August 8, 2019

1. Properties of symmetric matrices. In this problem $P$ and $Q$ are symmetric matrices. For each statement below, either give a proof or a specific counterexample.

   a) If $P \geq 0$ then $P + Q \geq Q$.

   b) If $P \geq Q$ then $-P \leq -Q$.

   c) If $P > 0$ then $P - 1 > 0$.

   d) If $P > Q > 0$ then $P - 1 \leq Q - 1$.

   e) If $P \geq Q$ then $P^2 \geq Q^2$.

*Hint:* you might find it useful for part (d) to prove $Z \geq I$ implies $Z^{-1} \leq I$.

**Solution.**

   a) By definition, $A \geq B$ if and only if $A - B \geq 0$. So, if $P \geq 0$, then $P + Q - Q \geq 0$ and therefore $P + Q \geq Q$.

   b) If $P \geq Q$ then $P - Q \geq 0$, and by definition $-(P - Q) \leq 0$ or $-P + Q \leq 0$ so finally $-Q \geq -P$.

   c) If $P > 0$ then all eigenvalues of $P$ are strictly positive and $P^{-1}$ exists. If $\lambda_1, \ldots, \lambda_n > 0$ are the eigenvalues of $P$ then the eigenvalues of $P^{-1}$ are $1/\lambda_1, \ldots, 1/\lambda_n$. Since $\lambda_i > 0$ then $1/\lambda_i > 0$ so the eigenvalues of $P^{-1}$ are all positive and therefore $P^{-1} > 0$.

   d) First we prove the hint, *i.e.*, if $Z \geq I$ then $Z^{-1} \leq I$. Suppose the eigenvalues of $Z \in \mathbb{R}^{n \times n}$ are $\lambda_1, \ldots, \lambda_n$. Then the eigenvalues of $Z - I$ are $\lambda_1 - 1, \ldots, \lambda_n - 1$ because if $v_i$ is the eigenvector associated with $\lambda_i$ then

   $$(Z - I)v_i = Zv_i - v_i = \lambda_i v_i - v_i = (\lambda_i - 1)v_i$$

   which means that $\lambda_i - 1$ is an eigenvalue of $Z - I$. Since $Z \geq I$ or $Z - I \geq 0$ then all eigenvalues of $Z - I$ are nonnegative or $\lambda_i \geq 1$. The eigenvalues of $Z^{-1}$ are $1/\lambda_1, \ldots, 1/\lambda_n$ and from $\lambda_i \geq 1$ we conclude that $1/\lambda_i \leq 1$ or the eigenvalues of $Z^{-1}$ are all less than or equal to 1. The eigenvalues of $Z^{-1} - I$ are $1/\lambda_1 - 1, \ldots, 1/\lambda_n - 1$ and therefore are all nonpositive. Hence $Z^{-1} - I \leq 0$ or $Z^{-1} \leq I$ and we are done. Now we prove that $P \geq Q > 0$ implies that $P^{-1} \leq Q^{-1}$ or $P^{-1} - Q^{-1} \leq 0$. Suppose that $Q = U\Lambda U^T$ is an
eigenvalue decomposition of $Q$. Since $Q > 0$ then $\Lambda > 0$ and therefore $Q^{-1/2} = Q^{-T/2} = U\Lambda^{-1/2}U^T$ exists. By congruence, $P - Q \geq 0$ implies that

$$Q^{-T/2}(P - Q)Q^{-1/2} \geq 0$$

or

$$Q^{-T/2}PQ^{-1/2} - Q^{-T/2}QQ^{-1/2} \geq 0$$

and therefore

$$Q^{-T/2}PQ^{-1/2} - I \geq 0.$$  

Now according to the hint (take $Z = Q^{-T/2}PQ^{-1/2}$) we have

$$(Q^{-T/2}PQ^{-1/2})^{-1} - I \leq 0$$

or

$$Q^{1/2}P^{-1}Q^{T/2} - I \leq 0.$$  

Again by congruence this implies

$$Q^{-1/2}(Q^{1/2}P^{-1}Q^{T/2} - I)Q^{-T/2} \leq 0$$

or

$$P^{-1} - Q^{-1/2}Q^{-T/2} \leq 0$$

and finally

$$P^{-1} - Q^{-1} \leq 0.$$  

e) The statement is false. A simple counterexample is $P = -1$ and $Q = -2.$

2. **Drawing a representation of a graph.** Consider an undirected graph with $n$ nodes, and $m$ edges. We want to draw a representation of this graph in the plane, which means assigning coordinates $(x_i, y_i) \in \mathbb{R}^2$ to node $i$ for $i = 1, \ldots, n$. Let $x \in \mathbb{R}^n$ be the vector of $x$-coordinates of the nodes, and $y \in \mathbb{R}^n$ be the vector of $y$-coordinates of the nodes. One desirable property of our representation is that nodes that are connected by an edge should not be too far apart. We can try to find a representation that has this property by minimizing the objective

$$J = \sum_{(i,j) \in E} ((x_i - x_j)^2 + (y_i - y_j)^2),$$

where $E$ is the edge set of the graph. Note that $J$ is the sum of the squares of the lengths of all of the edges in the graph. Just minimizing $J$ does not usually yield a sensible solution, so we need to introduce some additional constraints. Since the objective $J$ is not affected by shifting all of the coordinates by some fixed amount, we will assume that the coordinates are centered: that is,

$$\sum_{i=1}^n x_i = 0, \quad \text{and} \quad \sum_{i=1}^n y_i = 0.$$  

Another problem with just minimizing $J$ is that there is a trivial solution: set $x = y = 0$. To force the nodes to spread out, we impose the constraints
\[
\sum_{i=1}^{n} x_i^2 = 1, \quad \sum_{i=1}^{n} y_i^2 = 1, \quad \text{and} \quad \sum_{i=1}^{n} x_i y_i = 0.
\]
The first constraint says that the $x$-coordinates have unit variance, while the second constraint says that the $y$-coordinates have unit variance; the third constraint says that the $x$-coordinates and $y$-coordinates are uncorrelated. Even with all of these additional constraints, there is still not a unique set of coordinates that minimize $J$. For example, if $x$ and $y$ are some set of coordinates satisfying the constraints, and $Q \in \mathbb{R}^{2 \times 2}$ is orthogonal, then
\[
\begin{bmatrix}
\tilde{x}_i \\
\tilde{y}_i
\end{bmatrix} = Q \begin{bmatrix}
x_i \\
y_i
\end{bmatrix}, \quad i = 1, \ldots, n
\]
is another set of coordinates that also satisfies the constraints, and achieves the same value of $J$. Intuitively, we can rotate or reflect any set of coordinates to obtain another set of coordinates that is just as good. We will live with this ambiguity.

a) Explain how to find coordinates $x$ and $y$ that minimize $J$ subject to the centering and spreading constraints.

b) The file `draw_graph_representation_data.m` defines the following variables.

- $A$, the adjacency matrix of the graph
- $x_{\text{circ}}$, a vector of $x$-coordinates obtained using a simple method that places the nodes at equally spaced points on a circle (the radius of the circle is chosen to satisfy the spreading constraints)
- $y_{\text{circ}}$, a vector of $y$-coordinates corresponding to $x_{\text{circ}}$

Apply your method to the graph described by $A$. Report the optimal value of $J$, and draw a corresponding representation of the graph. For comparison, compute the value of $J$ associated with $x_{\text{circ}}$ and $y_{\text{circ}}$, and draw the corresponding representation of the graph.

**Solution.**

a) We can write the centering constraints as
\[
1^T x = 0, \quad \text{and} \quad 1^T y = 0,
\]
and we can express the spreading constraints as
\[
\sum_{i=1}^{n} x_i^2 = x^T x = 1, \quad \sum_{i=1}^{n} y_i^2 = y^T y = 1, \quad \text{and} \quad \sum_{i=1}^{n} x_i y_i = x^T y = y^T x = 0.
\]
We can combine the centering and spreading constraints into a single matrix constraint:
\[
\begin{bmatrix}
x \\
y \\
1/\sqrt{n}
\end{bmatrix}^T \begin{bmatrix}
x \\
y \\
1/\sqrt{n}
\end{bmatrix} = \begin{bmatrix}
x^T x & x^T y & 1/\sqrt{n} x^T 1 \\
y^T x & y^T y & 1/\sqrt{n} y^T 1 \\
1/\sqrt{n} x^T 1 & 1/\sqrt{n} y^T 1 & 1/\sqrt{n} 1^T 1
\end{bmatrix} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}.
Define the node incidence matrix $B \in \mathbb{R}^{m \times n}$ such that

$$B_{kq} = \begin{cases} +1 & \text{edge } q \text{ is } (q,j) \text{ for some node } j > q, \\ -1 & \text{edge } q \text{ is } (i,q) \text{ for some node } i < q, \\ 0 & \text{otherwise}. \end{cases}$$

Then, we can write the objective function as

$$J = \sum_{(i,j) \in E} ((x_i - x_j)^2 + (y_i - y_j)^2) = \|Bx\|^2 + \|By\|^2.$$

Since each row of $B$ contains exactly one $+1$, and exactly one $-1$, we have that

$$B1 = 0.$$  

Thus, we can write the objective function as

$$J = \|Bx\|^2 + \|By\|^2 + \|B1\|^2 = \|B \begin{bmatrix} x & y & 1 \end{bmatrix}\|^2.$$

Consider the following optimization problem.

$$\begin{align*} \text{minimize} & \quad \|B \begin{bmatrix} x & y & z \end{bmatrix}\|^2_F = \text{trace} \left( \begin{bmatrix} x & y & z \end{bmatrix}^T B^T B \begin{bmatrix} x & y & z \end{bmatrix} \right) \\ \text{subject to} & \quad \begin{bmatrix} x & y & z \end{bmatrix}^T \begin{bmatrix} x & y & z \end{bmatrix} = I \end{align*}$$

If we were to require that $z = 1$, then optimal values of $x$ and $y$ would minimize $J$ subject to the centering and spreading constraints. Thus, this optimization problem is a relaxation of the problem that we want to solve. We know that a solution of this problem is to take $x$, $y$ and $z$ to be right singular vectors of $B$ corresponding to the three smallest singular values. Because $B1 = 0$, we know that $1$ is a right singular vector of $B$ corresponding to $0$, which is the smallest singular value. Thus, we can find a solution of our relaxation satisfying $z = 1$. The corresponding optimal values of $x$ and $y$ minimize $J$ subject to the centering and spreading constraints. In summary, we take $x$ and $y$ to be right singular vectors of $B$ corresponding to the second and third smallest singular values.

b) The optimal value of $J$ is $0.1073$; the corresponding graph is given in ???. The value of $J$ associated with the circle graph is $5.3279$; the corresponding graph is given in ???. The structure of the graph is much more apparent in the optimal graph than in the circle graph.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% clean up the workspace, and load the data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all; close all; clc
draw_graph_representation_data;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% construct the node incidence matrix

4
B = zeros(m,n);
[i,j] = find(triu(A));
for k = 1:m
    B(k,i(k)) = +1;
    B(k,j(k)) = -1;
end

% find a representation for the graph
[~,s,v] = svd(B);
x = v(:,n-1);
x = x - mean(x);
y = v(:,n-2);
y = y - mean(y);

figure();
plot(x, y, 'o');
hold on;
for k = 1:m
    plot(x([i(k) j(k)]), y([i(k) j(k)]));
end
hold off;
axis equal;
xlabel('x');
ylabel('y');

Jopt = norm(B*x)^2 + norm(B*y)^2

% draw the circular representation for comparison
figure();
plot(x_circ, y_circ, 'o');
hold on;
for k = 1:m
    plot(x_circ([i(k) j(k)]), y_circ([i(k) j(k)]), '-');
end
hold off;
axis equal;
xlabel('x');
ylabel('y');
Jcirc = norm(B*x_circ)^2 + norm(B*y_circ)^2

3. Blind signal detection. A binary signal \( s_1, \ldots, s_T \), with \( s_t \in \{-1, 1\} \) is transmitted to a receiver, which receives the (vector) signal \( y_t = as_t + v_t \in \mathbb{R}^n \), \( t = 1, \ldots, T \), where \( a \in \mathbb{R}^n \) and \( v_t \in \mathbb{R}^n \) is a noise signal. We’ll assume that \( a \neq 0 \), and that the noise signal is centered around zero, but is otherwise unknown. (This last statement is vague, but it will not matter.)

The receiver will form an approximation of the transmitted signal as

\[
s_t \approx \hat{s}_t = w^T y_t, \quad t = 1, \ldots, T,
\]

where \( w \in \mathbb{R}^n \) is a weight vector. Your job is to choose the weight vector \( w \) so that \( \hat{s}_t \approx s_t \). If you knew the vector \( a \), then a reasonable choice for \( w \) would be \( w^\dagger = a / \|a\|^2 \). This choice is the smallest (in norm) vector \( w \) for which \( w^T a = 1 \).

Here’s the catch: You don’t know the vector \( a \). Estimating the transmitted signal, given the received signal, when you don’t know the mapping from transmitted to received signal (in this case, the vector \( a \)) is called blind signal estimation or blind signal detection.

Here is one approach. Ignoring the noise signal, and assuming that we have chosen \( w \) so that \( w^T y_t \approx s_t \), we would have

\[
\frac{1}{T} \sum_{t=1}^{T} (w^T y_t)^2 \approx 1.
\]

Since \( w^T v_t \) gives the noise contribution to \( \hat{s}_t \), we want \( w \) to be as small as possible. This leads us to choose \( w \) to minimize \( \|w\| \) subject to \( \frac{1}{T} \sum_{t=1}^{T} (w^T y_t)^2 = 1 \). This doesn’t determine \( w \) uniquely; we can multiply it by \(-1\) and it still minimizes \( \|w\| \) subject to \( \frac{1}{T} \sum_{t=1}^{T} (w^T y_t)^2 = 1 \). So we can only hope to recover either an approximation of \( s_t \) or of \(-s_t\); if we don’t know \( a \) we really can’t do any better. (In practice we’d use other methods to determine whether we have recovered \( s_t \) or \(-s_t\).)

a) Explain how to find \( w \), given the received vector signal \( y_1, \ldots, y_T \), using concepts from the class.

b) Apply the method to the signal in the file \( \text{bs} \_\text{det} \_\text{data} \_\text{m} \), which contains a matrix \( Y \), whose columns are \( y_t \). Give the weight vector \( w \) that you find. Plot a histogram of the values of \( w^T y_t \) using \( \text{hist}(w^T Y, 50) \). You’ll know you’re doing well if the result has two peaks, one negative and one positive. Once you’ve chosen \( w \), a reasonable guess of \( s_t \) (or, possibly, its negative \(-s_t\)) is given by

\[
\tilde{s}_t = \text{sign}(w^T y_t), \quad t = 1, \ldots, T,
\]

where \( \text{sign}(u) \) is +1 for \( u \geq 0 \) and −1 for \( u < 0 \). The file \( \text{bs} \_\text{det} \_\text{data} \_\text{m} \) contains the original signal, as a row vector \( s \). Give your error rate, i.e., the fraction of times for which \( \tilde{s}_t \neq s_t \). (If this is more than 50%, you are welcome to flip the sign on \( w \).)
Figure 1: graph representation minimizing $J$
Figure 2: graph representation using circle heuristic
Solution. We can write
\[
\frac{1}{T} \sum_{t=1}^{T} (w^T y_t)^2 = \frac{1}{T} \|Y^T w\|^2,
\]
where \( Y = [y_1 \cdots y_T] \). We must minimize \( \|w\| \) subject to \( \|Y^T w\| = \sqrt{T} \). Both of these are homogeneous in \( w \), so we could just as well maximize \( \|Y^T w\| \), subject to \( \|w\| = 1 \), and then scale the solution so that \( \|Y^T w\| = \sqrt{T} \). To maximize \( \|Y^T w\| \) subject to \( \|w\| = 1 \) is easy: we take \( w \) to be \( v_1 \), the right singular vector associated with the largest singular value of \( Y^T \). (This is also, by the way, the left singular vector associated with the largest singular value of \( Y \).) We then scale \( v_1 \) by \( \alpha \), so that \( \|Y^T(\alpha v_1)\| = \alpha \sigma_1 = \sqrt{T} \), where \( \sigma_1 \) is the largest singular value (i.e., the norm) of \( Y \) (or \( Y^T \)). This yields
\[
w = \left( \frac{\sqrt{T}}{\sigma_1} \right) v_1.
\]
Note that we could just as well take the negative of this vector, which would also minimize \( \|w\| \) subject to \( \|Y^T w\| = \sqrt{T} \).

Here is the code to do this:

```matlab
% blind signal detection exercise
% solution

bs_det_data;
[U,S,V]=svd(Y');
w = (sqrt(T)/S(1,1))*V(:,1)

% now form estimate of original binary signal
shat = (Y'*w)';
hist(shat,50);
print -deps bs_det_hist

% error rate
stilde = sign(shat);
error_rate = sum(s~=stilde)/T
```
The error rate is 2.9%. The resulting histogram is shown below.

4. Simultaneously estimating student ability and exercise difficulty. Each of $n$ students takes an exam that contains $m$ questions. Student $j$ receives a grade $G_{ij} \geq 0$ on question $i$. One simple model for predicting grades is to estimate

$$G_{ij} \approx \hat{G}_{ij} = \frac{a_j}{d_i},$$

where $a_j \geq 0$ is the ability of student $j$, and $d_i > 0$ is the difficulty of question $i$. To ensure a unique model, we normalize the exam question difficulties so that the mean exam question difficulty across the $m$ questions is 1; otherwise, $a_j$ and $d_i$ would not be uniquely determined because we can scale them both by any positive constant without changing our estimates.

In this problem you are given the matrix $G \in \mathbb{R}^{m \times n}$ of grades. Your task is to find a set of nonnegative student abilities, and a set of positive, normalized question difficulties such that $G_{ij} \approx \hat{G}_{ij}$. In particular, choose your model in order to minimize the root mean squared error:

$$J = \sqrt{\frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (G_{ij} - \hat{G}_{ij})^2}.$$ 

a) Explain how to solve this problem. You can ignore the constraints that the $a_j$ be nonnegative, and the $d_i$ be positive.

b) Carry out your method on the data in ability_difficulty_data.m. Report the optimal value of $J$, the ratio of the optimal value of $J$ and the root mean squared value of $G_{ij}$; give the difficulties of the exam questions.
Solution.

a) Minimizing $J$ is equivalent to minimizing

$$mnJ^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (G_{ij} - \hat{G}_{ij})^2 = \|G - \hat{G}\|_F^2.$$ 

Define the vector $\tilde{d} \in \mathbb{R}^m$ such that

$$\tilde{d} = \begin{bmatrix} \frac{1}{d_1} \\ \vdots \\ \frac{1}{d_m} \end{bmatrix}.$$ 

Then, we can express $\hat{G}$ as

$$\hat{G} = \begin{bmatrix} a_1 \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} \frac{1}{d_1} \\ \vdots \\ \frac{1}{d_m} \end{bmatrix}^T \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = \tilde{d}a^T.$$ 

Thus, $\hat{G}$ is a rank-one matrix. In order to minimize $mnJ^2 = \|G - \hat{G}\|_F^2$, we choose $\hat{G}$ to be the best rank-one approximation of $G$ in the Frobenius norm:

$$\hat{G} = \sigma_1 u_1 v_1^T,$$

where $\sigma_1$, $u_1$ and $v_1$ are the first singular value, first left singular vector and first right singular of $G$, respectively. Then, we take $d = \alpha u_1$, where the normalization constant $\alpha$ is chosen so that the mean difficulty across the $m$ questions is 1. In particular, we have that

$$d_i = \frac{1}{u_1(i)} \left( \frac{1}{m} \sum_{k=1}^{m} \frac{1}{u_1(k)} \right)^{-1}, \quad i = 1, \ldots, m.$$ 

In order to satisfy

$$\hat{G} = \sigma_1 u_1 v_1^T = \tilde{d}a^T = \left( \frac{1}{m} \sum_{k=1}^{m} \frac{1}{u_1(k)} \right) u_1 a^T,$$

we choose $a$ to be

$$a = \sigma_1 v_1 \left( \frac{1}{m} \sum_{k=1}^{m} \frac{1}{u_1(k)} \right)^{-1}.$$

b) We obtain the following estimates for the difficulties of the questions:

$$d = \begin{bmatrix} 0.9429 \\ 1.2780 \\ 0.9015 \\ 0.9197 \\ 0.7729 \\ 1.0418 \\ 1.1433 \end{bmatrix}.$$
The RMS error is $J = 5.6759$, and the ratio of the RMS error to the RMS value of the grades is 0.3574. In other words, our simple model explains about 64% of the variation in the grades.

\[
\begin{align*}
\text{clear all} & \text{; close all} & \text{; clc} \\
\text{ability_difficulty_data;} \\
\end{align*}
\]

\[
\begin{align*}
\text{U} & \text{; S} \text{;} \text{V} = \text{svd}(G) \; \\
d & = 1 ./ \text{(U(:,1))} * \text{mean}(1 ./ \text{(U(:,1))}) \\
a & = S(1,1) * V(:,1) / \text{mean}(1 ./ \text{(U(:,1))}) \\
J & = \text{norm}(G - (1 ./ d) * a', 'fro') / \text{sqrt(m*n)} \\
Jrel & = J / (\text{norm}(G, 'fro') / \text{sqrt(m*n)}) \\
\end{align*}
\]

5. Square matrices and the SVD. Let $A$ be an $n \times n$ real matrix. State whether each of the following statements is true or false. Do not give any explanation or show any work.

a) If $x$ is an eigenvector of $A$, then $x$ is either a left or right singular vector of $A$

b) If $\lambda$ is an eigenvalue of $A$, then $|\lambda|$ is a singular value

c) If $A$ is symmetric, then every singular value of $A$ is also an eigenvalue of $A$

d) If $A$ is symmetric, then every singular vector of $A$ is also an eigenvector of $A$

e) If $A$ is symmetric with the following singular value decomposition

\[
A = U \Sigma V^T
\]

then $U = V$

f) If $A$ is invertible, then

$$\sigma_i \neq 0 \quad \text{for all } i = 1, \ldots, n$$

Solution.

a) False. Consider the matrix

\[
A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}
\]

The SVD of $A$ is

\[
U = \begin{bmatrix} 0.8507 & -0.5257 \\ 0.5257 & 0.8507 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 1.618 & 0 \\ 0 & 0.618 \end{bmatrix}, \quad V = \begin{bmatrix} 0.5257 & -0.8507 \\ 0.8507 & 0.5257 \end{bmatrix}
\]
\[ \begin{bmatrix} 1 \\ 0 \end{bmatrix} \] is an eigenvector of A but is neither a right nor a left singular vector of A.

b) **False.** For the matrix mentioned in (a), the eigenvalues are 1,1 while the singular values are 1.618,0.618

c) **False.** Singular values are non-negative for any matrix but the eigenvalues, even if the matrix is symmetric, can be negative.

d) **True.** If A is symmetric, it has the following eigenvalue decomposition

\[ A = Q\Lambda Q^T \]

where Q is orthogonal.
For all the negative eigenvalues in \( \Lambda \), if we retain their magnitude in \( \Lambda \) and multiply the corresponding eigenvectors in \( Q \) (or \( Q^T \)) by -1, we get the SVD of A (since Q was orthogonal). Therefore every singular vector of A is also an eigenvector of A.

e) **False.** Consider the matrix

\[ A = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \]

Its SVD is given by

\[
U = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad V = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}
\]

Clearly, \( U \neq V \)

f) **True.** Let A be an invertible matrix. This implies that \( \text{null}(A) = \{0\} \). If A has a singular value \( \sigma_j = 0 \) then \( Av_j = 0 \) where \( v_j \) is the right singular vector corresponding to \( \sigma_j \). This contradicts the fact that \( \text{null}(A) = \{0\} \). Therefore, if A is invertible then all the singular values of A are non-zero.

6. **Principal-components analysis of decathlon data.** In the decathlon athletes compete in \( n = 10 \) different track-and-field events: 100 m run, long jump, shot put, high jump, 400 m run, 110 m hurdle, discus, pole vault, javelin, and 1500 m run. The file `decathlon_pca_data.m` contains the results for the \( m = 24 \) athletes who completed all of the events in the decathlon in the 2008 Olympics; in particular, it defines the following variables.

- \( m \) and \( n \), the numbers of athletes and events, respectively
- \( \text{names} \), a cell array containing the last names and countries of the athletes
- \( \text{events} \), a cell array containing the names of the events
- \( \text{scores} \), a matrix containing the score of each athlete on each event; each row corresponds to an athlete, and each column corresponds to an event; the orderings of athletes and events are the same as in \( \text{names} \) and \( \text{events} \), respectively
First, we standardize the data by forming the matrix $X \in \mathbb{R}^{m \times n}$ such that

$$X_{ij} = \frac{\text{scores}(i, j) - m_j}{s_j},$$

where

$$m_j = \frac{1}{m} \sum_{i=1}^{m} \text{scores}(i, j) \quad \text{and} \quad s_j = \sqrt{\frac{1}{m-1} \sum_{i=1}^{m} (\text{scores}(i, j) - m_j)^2}$$

are, respectively, the mean and standard deviation of the scores for the $j$th event.

a) Let $X = \sum_{i=1}^{m} \sigma_j u_i v_i^T$ be the singular-value decomposition of $X$. Submit a plot showing $\sigma_j$ versus $j$. The fraction of the total power that is contained in the first $j$ dyads of the singular-value decomposition is defined to be

$$p_j = \frac{\sum_{j=1}^{j} \sigma_j^2}{\sum_{j=1}^{n} \sigma_j^2}.$$

Submit a plot of $p_j$ versus $j$, and report $p_2$, the fraction of the total power that is contained in the first two dyads.

b) Use the command `text` to create a plot in which the point $((v_1)_j, (v_2)_j)$ is labeled with the name of the $j$th event. Do similar events appear to be close to each other on your plot?

c) Let $t_i = \sum_{j=1}^{n} X_{ij}$ be the total standardized score of the $i$th athlete. Then, we have that

$$r_j = \sum_{i=1}^{m} t_i X_{ij}$$

is proportional to the correlation between an athlete’s total standardized score and his standardized score on the $j$th event. Download the function `spatial_plot` from the course website. Let $v_1$ and $v_2$ be the first two right singular vectors of $X$. Submit a copy of the plot generated using the following command.

`spatial_plot(v1, v2, r, 3)`

Which right singular vector seems to represent $r$?

d) Let $u_1$ and $u_2$ be the first two left singular vectors of $X$. Define the vector $\delta \in \mathbb{R}^m$

$$\delta_i = (X_{i1} + X_{i5} + X_{i6}) - (X_{i3} + X_{i7} + X_{i9}),$$

and let $t = (t_1, \ldots, t_m)$ be the vector of the total standardized scores of the athletes. Submit copies of the plots generated using the following commands.

`spatial_plot(u1, u2, t, length(names));`
`spatial_plot(u1, u2, delta, length(names));`

Give intuitive interpretations of the first two left singular vectors of $X$. 

14
Solution.

a) Plots of $\sigma_j$ and $p_j$ versus $j$ are given in ???, respectively. The fraction of the signal power contained in the first two dyads is 0.4902.

b) A scatter plot of the loadings of the events in the first two singular vectors is given in ???. We see that similar events tend to be close together. In particular, the throwing events (javelin, shot put, and discuss) are all close to each other, while events such as the 100 m dash and 110 m hurdles are also close together.

c) A scatter plot of $r$ is given in ???. The first right singular vector seems to represent the correlation of each event score with the total score.

d) Scatter plots of $t$ and $\delta$ are given in ???. Note that $\delta$ is a measure of the difference between an athlete’s scores on the running and throwing events. Thus, we can interpret the first left singular vector as a measure of an athlete’s total score, while the second left singular vector is a measure of the difference between an athlete’s scores on the running and throwing events.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% clean up the workspace, and load the data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all ; close all ; clc
decathlon_pca_data;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% standardize the scores
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
standardize = @(x) (x - mean(x)) / std(x);
X = nan(size(scores));
for i = 1:length(events)
    X(:,i) = standardize(scores(:,i));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% compute the SVD of X, and the vector of total scores
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[u,s,v] = svd(X);
total_X = sum(X,2);
figure();
stem(1:n , diag(s));
xlabel(‘i’);
ylabel(‘sigma(i)’);
figure();
stem(1:n , cumsum(diag(s).^2) / sum(diag(s).^2));
xlabel('i');
ylabel('fraction of power in first i modes');
p2 = sum(diag(s(1:2,1:2).^2)) / sum(diag(s).^2)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% plot the components of the events in the first two
% principal components
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
figure();
plot([-1 +1] , [0 0] , 'k' , [0 0] , [-1 +1] , 'k');
for i = 1:length(events)
    text(v(i,1), v(i,2), events{i});
end
axis equal;
axis([-1 +1 -1 +1]);
grid on;
xlabel('v1');
ylabel('v2');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% make a spatial plot of the correlation of an event score
% with the total score
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
spatial_plot(v(:,1), v(:,2), total_X' * X, 3);
xlabel('v1');
ylabel('v2');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% make a spatial plot of a competitor's total score
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
spatial_plot(u(:,1), u(:,2), total_X, length(names));
xlabel('u1');
ylabel('u2');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% make a spatial plot of the difference between a
% competitor's scores in the running and throwing events
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
running_scores = sum(X(:,[1 5 6]),2);
throwing_scores = sum(X(:,[3 7 9]),2);
delta = running_scores - throwing_scores;
spatial_plot(u(:,1), u(:,2), delta, length(names));
xlabel('u1');
ylabel('u2');
Figure 3: the singular values of $X$
Figure 4: the fraction of the signal power contained in the first $j$ dyads
Figure 5: loadings of the first two right singular vectors on the events
Figure 6: spatial plot of the first two right singular vectors and the correlation with the total score

Figure 7: spatial plot of the first two left singular vectors and the total score
7. **Minimum energy required to steer the state to zero.** Consider a controllable discrete-time system \( x(t + 1) = Ax(t) + Bu(t), \ x(0) = x_0 \). Let \( E(x_0) \) denote the minimum energy required to drive the state to zero, \( i.e. \)

\[
E(x_0) = \min \left\{ \sum_{\tau=0}^{t-1} \|u(\tau)\|^2 \mid x(t) = 0 \right\}.
\]

An engineer argues as follows:

This problem is like the minimum energy reachability problem, but ‘turned backwards in time’ since here we steer the state from a given state to zero, and in the reachability problem we steer the state from zero to a given state. The system \( z(t + 1) = A^{-1}z(t) - A^{-1}Bv(t) \) is the same as the given one, except time is running backwards. Therefore \( E(x_0) \) is the same as the minimum energy required for \( z \) to reach \( x_0 \) (a formula for which can be found in the lecture notes).

Either justify or refute the engineer’s statement. You can assume that \( A \) is invertible.

**Solution.** The backwards system is

\[
z(t) = A^{-1}z(t + 1) - A^{-1}Bv(t)
\]
The energy required for the state \( z \) to reach \( x_0 \) is given by
\[
E_z = x_0^T \left[ \sum_{\tau=0}^{t_f-1} A^{-\tau-1} BB^T (A^{-\tau-1})^T \right]^{-1} x_0 = x_0^T \left[ \sum_{\tau=1}^{t_f} A^{-\tau} BB^T (A^{-\tau})^T \right]^{-1} x_0
\]
and the energy required for the state \( x \) to go from \( x_0 \) to \( 0 \) is
\[
E_x = x_0^T (A_{t_f})^T \left[ \sum_{t=0}^{1} A^{-t} BB^T (A^{-t})^T \right]^{-1} A_{t_f} x_0
\]
Making the change of variables \( t = t_f - \tau \) we can rewrite
\[
E_x = x_0^T \left[ \sum_{t=t_f}^{1} A^{-t} BB^T (A^{-t})^T \right]^{-1} x_0 = E_z
\]
and we conclude that the engineer’s statement is correct.

8. **Sensor selection and observer design.** Consider the system
\[
\dot{x} = Ax, \quad y = Cx,
\]
with
\[
A = \begin{bmatrix}
1 & 0 & 0 & 0 \\
1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 \\
1 & 0 & 0 & 0
\end{bmatrix}, \quad C = \begin{bmatrix}
1 & 1 & 0 & 0 \\
0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}.
\]
(This problem concerns observer design so we’ve simplified things by not even including an input.) (The matrix \( A \) is the same as in problem ??, just to save you typing; there is no other connection between the problems.) We consider observers that (exactly and instantaneously) reconstruct the state from the output and its derivatives. Such observers have the form
\[
x(t) = F_0 y(t) + F_1 \frac{d}{dt} y(t) + \cdots + F_k \frac{d^k y(t)}{dt^k},
\]
where \( F_0, \ldots, F_k \) are matrices that specify the observer. (Of course we require this formula to hold for any trajectory of the system and any \( t \), i.e., the observer has to work!) Consider an observer defined by \( F_0, \ldots, F_k \). We say the degree of the observer is the largest \( j \) such that \( F_j \neq 0 \). The degree gives the highest derivative of \( y \) used to reconstruct the state. If the \( i \)th columns of \( F_0, \ldots, F_k \) are all zero, then the observer doesn’t use the \( i \)th sensor signal \( y_i(t) \) to reconstruct the state. We say the observer uses or requires the sensor \( i \) if at least one of the \( i \)th columns of \( F_0, \ldots, F_k \) is nonzero.

a) What is the minimum number of sensors required for such an observer? List all combinations (i.e., sets) of sensors, of this minimum number, for which there is an observer using only these sensors.

b) What is the minimum degree observer? List all combinations of sensors for which an observer of this minimum degree can be found.
Solution. For the observer to work, i.e., to always have

\[ x(t) = F_0 y(t) + \cdots + F_k \frac{d^k y(t)}{dt^k}, \]

we must have

\[
\begin{bmatrix}
F_0 & F_1 & \ldots & F_k \\
C & CA & \cdots & CA^{k-1}
\end{bmatrix} = I.
\]

In other words, \( F = [F_0 \; F_1 \; \ldots \; F_k], \) must be a left inverse of the observability matrix \( O_k. \)

a) Each row of the matrix \( C \) corresponds to a sensor. If we don’t use a sensor, we just delete the corresponding row of \( C \). So the problem is to analyse which sets of rows of \( C \) yield observability. We don’t care about the degree, so we may as well take degree 4 (since by the Cayley Hamilton theorem we don’t need to go to any higher degree). We’ll check every combination of sensors for observability. This can be done easily in matlab as follows:

\[
\begin{align*}
& c1=C(1,:); \% \text{first row, for first sensor} \\
& c2=C(2,:); \; c3=C(3,:); \\
& \text{rank([c1; c1*A; c1*A^2; c1*A^3]) ans = 2} \\
& \text{rank([c2; c2*A; c2*A^2; c2*A^3]) ans = 3} \\
& \text{rank([c3; c3*A; c3*A^2; c3*A^3]) ans = 2} \\
& \text{rank([c1; c1*A; c1*A^2; c1*A^3; c2; c2*A; c2*A^2; c2*A^3]) ans = 3} \\
& \text{rank([c1; c1*A; c1*A^2; c1*A^3; c3; c3*A; c3*A^2; c3*A^3]) ans = 3} \\
& \text{rank([c2; c2*A; c2*A^2; c2*A^3; c3; c3*A; c3*A^2; c3*A^3]) ans = 4} \\
& \text{rank([C; C*A; C*A^2; C*A^3]) ans = 4}
\end{align*}
\]

The minimum number of sensors required is 2. Only one combination of two sensors works: sensors 2 and 3.

b) To find the minimum degree observer, we must find the smallest number \( k \) that yields \( O_k \) rank 4. Checking this in matlab (or by hand — it’s not too hard) shows this happens for \( k = 2, \) i.e., \( C \) is not rank 4 (obviously) but

\[
\begin{bmatrix}
C \\
CA
\end{bmatrix}
\]

does have rank 4. Of course the calculation above shows we can have a degree 2 observer using all the sensors. But can we do it with fewer? The only possibility is sensors 2 and 3. Let’s try it:

\[
\text{rank([c2; c2*A; c3; c3*A]) ans = 4}
\]

So we can have a degree 2 sensor (which is the minimum degree possible) using only sensors 2 and 3, or, of course, all three sensors.