1. *True or false.* For each part, specify whether the statement is true or false. Do not give any explanation, proofs, or counterexamples; we will grade any question as incorrect if it contains any explanation.

   (a) If Q has orthonormal columns then $\|Q^T w\| \leq \|w\|$ for all w.

   (b) Suppose $A \in \mathbf{R}^{m\times p}$ and $B \in \mathbf{R}^{m\times q}$. If $\mathcal{N}(A) = \{0\}$ and $\mathcal{R}(A) \subseteq \mathcal{R}(B)$ then $p \leq q$.

   (c) If $V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}$ is invertible and $\mathcal{R}(V_1) = \mathcal{N}(A)$, then $\mathcal{N}(AV_2) = \{0\}$.

   (d) $\text{rank}([A \quad B]) = \text{rank}(A) = \text{rank}(B) \Longrightarrow \mathcal{R}(A) = \mathcal{R}(B)$.

   (e) $x \in \mathcal{N}(A^T) \iff x \notin \mathcal{R}(A)$.

   (f) If A is invertible, then AB is not full rank if and only if B is not full rank.

   (g) A is not full rank $\Rightarrow$ there is an $x \neq 0$, such that $Ax = 0$.

   Solution.

   (a) If Q has orthonormal columns then $\|Q^T w\| \leq \|w\|$ for all w.
   True.

   $$Q = [q_1, \ldots, q_r],$$

   suppose $q_i \in \mathbb{R}^n$, we can construct a orthonormal basis of $\mathbb{R}^n$ by adding $n - r$ orthonormal columns to Q. Then

   $$\begin{aligned}
   \|Q^T w\|^2 &= w^T Q Q^T w \\
   &= w^T \left(q_1 q_1^T + \ldots + q_r q_r^T\right) w \\
   &= \sum_{i=1}^{r} \left(w^T q_i\right)^2 \\
   &\leq \sum_{i=1}^{n} \left(w^T q_i\right)^2 \\
   &= w^T \left(q_1 q_1^T + \ldots + q_n q_n^T\right) w \\
   &= w^T I w \\
   &= w^T w \\
   &= \|w\|.
   \end{aligned}$$

   (b) Suppose $A \in \mathbf{R}^{m\times p}$ and $B \in \mathbf{R}^{m\times q}$. If $\mathcal{N}(A) = \{0\}$ and $\mathcal{R}(A) \subseteq \mathcal{R}(B)$ then $p \leq q$.
   True.

   $$\mathcal{N}(A) = \{0\} \Leftrightarrow \dim\left(\mathcal{R}(A)\right) = p,$$
   $$\mathcal{R}(A) \subseteq \mathcal{R}(B) \Leftrightarrow p = \dim\left(\mathcal{R}(A)\right) \leq \dim\left(\mathcal{R}(B)\right),$$

   and since

   $$q \geq \text{rank}(B) = \dim\left(\mathcal{R}(B)\right),$$

   we have

   $$q \geq p.$$

(c) If $V = \begin{bmatrix} V_1 & V_2 \end{bmatrix}$ is invertible and $\mathcal{R}(V_1) = \mathcal{N}(A)$, then $\mathcal{N}(AV_2) = \{0\}$.

True. $\forall x$ if $x \in \mathcal{N}(AV_2)$, then $V_2x \in \mathcal{N}(A) = \mathcal{R}(V_1)$, and by definition $V_2x \in \mathcal{R}(V_2)$ so $V_2x \in \mathcal{R}(V_1) \cap \mathcal{R}(V_2)$. But V is invertible therefore $\mathcal{R}(V_1) \cap \mathcal{R}(V_2) = 0$, then $V_2x = 0$. Since V_2 is full rank and skinny therefore $\mathcal{N}(V_2) = \{0\}$, so $x = 0$.

(d) $\text{rank}([A \quad B]) = \text{rank}(A) = \text{rank}(B) \Longrightarrow \mathcal{R}(A) = \mathcal{R}(B)$

True.

$\text{rank}([A \quad B]) = \text{rank}(A) \Longrightarrow \mathcal{R}(B) \subseteq \mathcal{R}(A)$.

$\text{rank}([A \quad B]) = \text{rank}(B) \Longrightarrow \mathcal{R}(A) \subseteq \mathcal{R}(B)$.

(e) $x \in \mathcal{N}(A^T) \iff x \notin \mathcal{R}(A)$

False.

$A = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

(f) If A is invertible, then AB is not full rank if and only if B is not full rank

True.

B is not full rank, A invertible $\Longrightarrow AB$ is not full rank obvious.

Define $C = AB$, then $B = A^{-1}C$. Use the first conclusion, $AB = C$ is not full rank, A^{-1} invertible $\Longrightarrow A^{-1}C = B$ is not full rank.

(g) A is not full rank $\Rightarrow$ there is an $x \neq 0$, such that $Ax = 0$

True. If A fat there is always an x s.t. $Ax = 0$.

If $A \in \mathbb{R}^{m \times n}$ is square or skinny, $m \geq n$. Then if there is an $x \neq 0$, such that $Ax = 0$ then $\dim\mathcal{N}(A) \geq 1$ therefore $\text{rank}(A) \leq n - 1$ which is not full rank.

4. *Power generation for a city.* A city has a series of generators that provide power to several key sites. Each site submits an estimate of the amount of power it plans to use for the year. Certain generators are more efficient for providing power to some of the sites than others. The following table provides a list of the generators and the amount of power it can generate per unit cost for each site.

	Generator 1	Generator 2	Generator 3	Generator 4
Site A	5	2	1	0
Site B	1	1	2	3
Site C	0	1	2	4
Site D	0	3	3	1
Site E	3	2	1	1
Site F	2	0	3	1
Site G	1	3	0	2

Here is the amount of power each site estimates it will use for the year:

Site A	Site B	Site C	Site D	Site E	Site F	Site G
20	5	3	4	10	5	5

The goal is to estimate how much money will be spent at each generator while trying to meet the sites' estimates. There is one requirement, however. Site A, a hospital, has been given top priority, and therefore must receive the **exact** amount of power it has requested. The other sites are allowed to receive a little more or less than the power it requested, though the goal is still to get as close as possible to meeting every site's request.

To avoid typos, the generator table and site requirements can be found at `powerGen.m`.

Here is what we want you to answer. Be sure to explain your process and how you arrived at your answer.

(a) Find the vector of money spent for each generator that minimizes the sum of the squared deviation between the sites' power requested and power generated, provided that Site A receives its exact estimate.

(b) Generator 4 has stopped working. Find the tradeoff curve showing how close the other sites can get to their estimates as the requirement for Site A is relaxed. Plot the sum of the squared deviation of what Sites B through G request and receive versus the difference squared between what Site A requests and what it receives.

Solution.

(a) The set of solutions which can satisfy the constraint is the set of x for which the following is true:

$$20 = [5\ 2\ 1\ 0] * x.$$

We can write this set as

$$x = \{x_0 + z : z \in \mathcal{N}([5\ 2\ 1\ 0])\},$$

where x_0 is an arbitrary solution, (i.e. $x_0 = [4\ 0\ 0\ 0]^T$). Let N be a basis for $\mathcal{N}([5\ 2\ 1\ 0])$. Then $x = x_0 + Nw$, where w is the vector we have control over. Thus, we have the following least squares problem.

$$\text{Minimize} \qquad ||\widetilde{A}(x_0 + Nw) - \widetilde{y}||^2,$$

where $\widetilde{A}$ is the matrix A with the top row removed and $\widetilde{y}$ is the vector y with the first entry removed. Rearranging some terms around, we have

$$\text{Minimize} \qquad ||\widetilde{A}Nw - (\widetilde{y} - \widetilde{A}x_0)||^2,$$

and so

$$w = ((\widetilde{A}N)^T(\widetilde{A}N))^{-1}(\widetilde{A}N)^T(\widetilde{y} - \widetilde{A}x_0),$$

and

$$x = x_0 + Nw.$$

This gives us the following cost vector:

$$x = \begin{bmatrix} 3.6084 \\ 0.9771 \\ 0.0039 \\ 0.0702 \end{bmatrix}$$

(b) This is a straightforward least squares problem with a weighted sum objective. We first remove the last column of the A matrix since Generator 4 no longer works. Define

$$\hat{A} = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 2 \\ 0 & 3 & 3 \\ 3 & 2 & 1 \\ 2 & 0 & 3 \\ 1 & 3 & 0 \end{bmatrix}, \qquad \hat{a} = [5\ 2\ 1], \qquad \hat{y} = \begin{bmatrix} 5 \\ 3 \\ 4 \\ 10 \\ 5 \\ 5 \end{bmatrix}.$$

Our least squares objective function becomes

$$||\hat{a}x - 20||^2 + \mu||\hat{A}x - \hat{y}||^2 = \left|\left|\begin{bmatrix} \hat{a} \\ \sqrt{\mu}\hat{A} \end{bmatrix} x - \begin{bmatrix} 20 \\ \sqrt{\mu}\hat{y} \end{bmatrix}\right|\right|^2$$

.

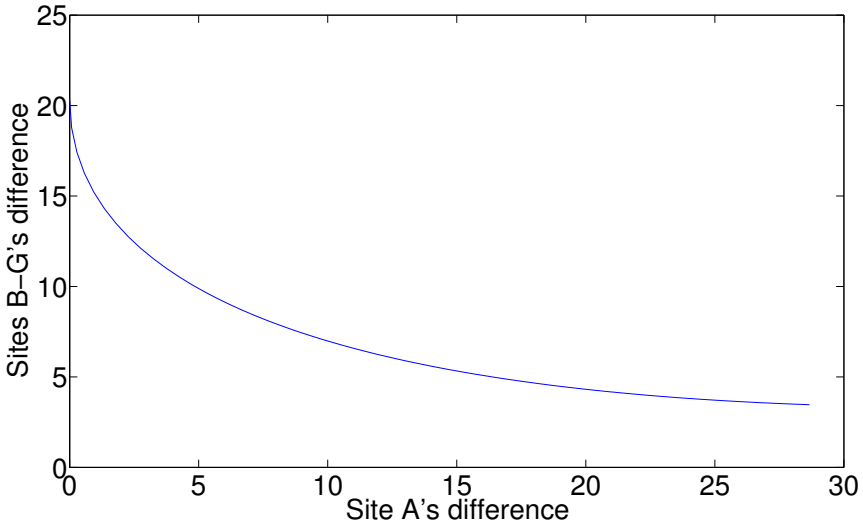
We then loop over values of μ to generate the tradeoff curve.

The following is Matlab code which calculates the solution for parts (a) and (b).

```
powerGen

% Part a
```

```
x0 = [4;0;0;0];
N  = null([5 2 1 0]);

A_tilde = A(2:end, :);
y_tilde = y(2:end);

w = (A_tilde*N) \ (y_tilde - A_tilde*x0);
x_constrained = x0 + N*w

% Part b
A_hat = A(2:end,1:3);
a_hat = A(1,1:3);
y_hat = y(2:end);

mu = [0:0.1:20];
for ind = 1:length(mu)
  % Solve constrained least squares when mu = 0
  if (ind == 1)
    x0 = [4;0;0];
    N  = null([5 2 1]);
    A_tilde = A(2:end, 1:3);
    y_tilde = y(2:end);

    w = (A_tilde*N) \ (y_tilde - A_tilde*x0);
    x = x0 + N*w;
  % Multi-object least squares when mu > 0
  else
    newA = [a_hat; sqrt(mu(ind))*A_hat];
    newy = [20; sqrt(mu(ind))*y_hat];
    x = (newA)\newy;
  end

  obj1(ind) = (a_hat*x - 20)^2;
  obj2(ind) = sum((A_hat*x - y_hat).^2);
end

plot(obj1, obj2);
xlabel('Site A''s difference');
ylabel('Sites B-G''s difference');

% Output:
% x_constrained =
%
%     3.6084
%     0.9771
```

```
%       0.0039
%       0.0702
```



5. *Estimating the channel.* In a communications channel, the input $x_1, \ldots, x_N$ and output $y_1, \ldots, y_N$ are related by the following equation:

$$y_n = \sum_{m=1}^{M} h_m x_{n-m+1} + v_n \qquad \text{for } n = 1, \ldots, N,$$

where $h_1, \ldots, h_M$ is the impulse response of the channel and $v_1, \ldots, v_N$ is some noise term. In order to estimate the impulse response of the channel, it is typical to send a known input sequence through the channel, and from the output deduce the channel coefficients.

Our task is to estimate the channel which minimizes the following objective function:

$$J = \sum_{n=1}^{N} \left(y_n - \sum_{m=1}^{M} h_m x_{n-m+1} \right)^2.$$

Assume that $x_n = 0$ for $n \leq 0$.

One thing that we have left out is the length of the impulse response. We can get around this problem by iterating over the lengths of the channel and observing how the residual behaves.

There are three parts to this question:

(a) Using the data provided in `channel.m`, find the impulse response of the channel assuming the length of the impulse response is 4. The known input and output signals are stored in the variables `x_known` and `y_known`, respectively. Explain how you arrived at those coefficients.

(b) Iterate your solution from part (a) assuming the length of the impulse response is $3, \ldots, 10$ and give the corresponding residuals. Plot the residual J as a function of the impulse response length. Is 4 a reasonable estimate? What would you recommend as a good length for the impulse response?

(c) You receive more output from the channel (stored in the variable `y_unknown`), except now you don't know what the input signal is. y_n runs from $n = 1, \ldots, N + 10$, although $x_n = 0$ for $n > N$. Using the channel you found in part (b), find and make a stem plot (using Matlab's `stem` command) of your estimated signal x_n for $n = 1, \ldots, N$. Also report the residual.

Solution.

(a) Using the equation for y_n given to us, we have

$$\begin{aligned}
y_1 &= h_1 x_1 \\
y_2 &= h_1 x_2 + h_2 x_1 \\
y_3 &= h_1 x_3 + h_2 x_2 + h_3 x_1 \\
y_4 &= h_1 x_4 + h_2 x_3 + h_3 x_2 + h_4 x_1 \\
y_5 &= h_1 x_5 + h_2 x_4 + h_3 x_3 + h_4 x_2 \\
&\vdots \\
y_{40} &= h_1 x_{40} + h_2 x_{39} + h_3 x_{38} + h_4 x_{37}
\end{aligned}$$

Thus, we can express our input/output relation as

$$y = \underbrace{\begin{bmatrix} x_1 & 0 & 0 & 0 \\ x_2 & x_1 & 0 & 0 \\ x_3 & x_2 & x_1 & 0 \\ x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ x_6 & x_5 & x_4 & x_3 \\ x_7 & x_6 & x_5 & x_4 \\ & & \vdots & \\ x_{40} & x_{39} & x_{38} & x_{37} \end{bmatrix}}_{A} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix}.$$

The channel is given by $h = (A^T A)^{-1} A^T y$, which gives us

$$h = \begin{bmatrix} 0.1721 \\ -0.0958 \\ 0.0080 \\ 1.1205 \end{bmatrix}$$

(b) As we increase the length of the channel, we simply add another column to our matrix A. We see in the figure that while a length 4 channel is a big improvement over a length 3 channel, a length 7 channel seems to be a good length. (Increasing the channel beyond that gives minimal improvement). The length 7 filter is:

$$h = \begin{bmatrix} 0.3141 \\ -0.1672 \\ -0.0114 \\ 1.1192 \\ -0.0166 \\ -0.2711 \\ 0.2955 \end{bmatrix}$$

(c) The roles of h and x are reversed. Our matrix will now contain the values of the filter we recovered from part (b), and will be skinny (since the output y is longer than the input x). Here is what the setup looks like

$$y = \begin{bmatrix} h_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ h_2 & h_1 & 0 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ h_3 & h_2 & h_1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 \\ h_4 & h_3 & h_2 & h_1 & 0 & 0 & 0 & 0 & \dots & 0 \\ h_5 & h_4 & h_3 & h_2 & h_1 & 0 & 0 & 0 & \dots & 0 \\ h_6 & h_5 & h_4 & h_3 & h_2 & h_1 & 0 & 0 & \dots & 0 \\ h_7 & h_6 & h_5 & h_4 & h_3 & h_2 & h_1 & 0 & \dots & 0 \\ 0 & h_7 & h_6 & h_5 & h_4 & h_3 & h_2 & h_1 & \dots & 0 \\ & & \vdots & & & & & & & \\ 0 & & & & & & & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ \vdots \\ x_{39} \end{bmatrix}.$$

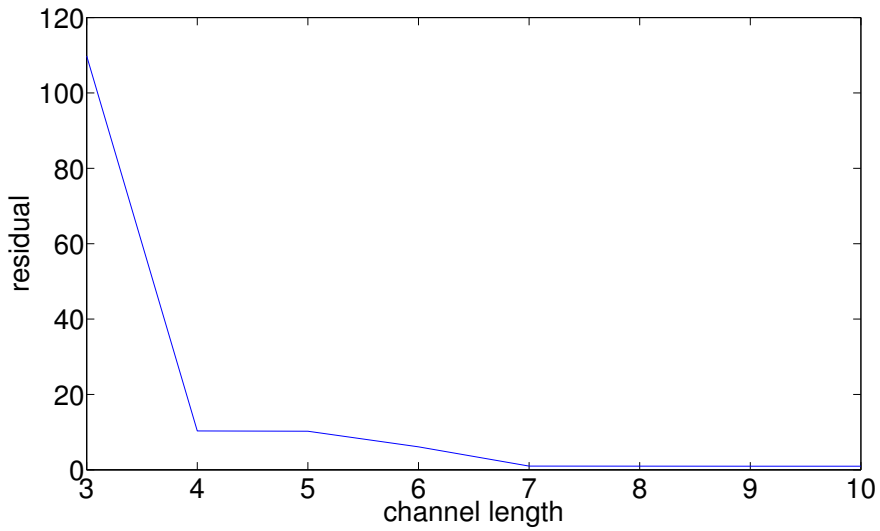
Note that the last few rows of the matrix are zero because the filter length is only 7. The residual we get is 0.4962.

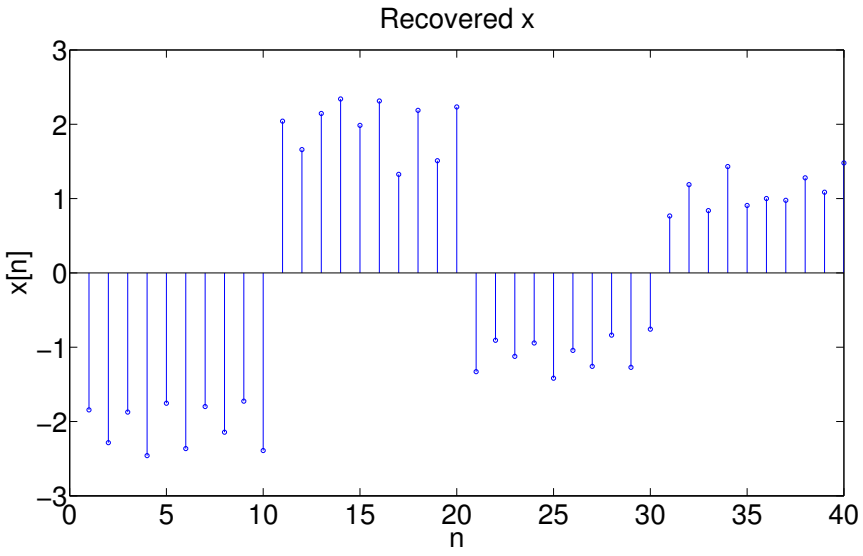
The following is Matlab code which calculates the solution.

```
channel;

% Part a
M = 4;
A = zeros(N,M);
for ind = 1:M
  A(:,ind) = [zeros(ind-1,1);x_known(1:end-(ind-1))];
end
h4 = A\y_known

% Part b
residuals = zeros(10,1);
for M = 3:10;
  A = zeros(N,M);
  for ind = 1:M
    A(:,ind) = [zeros(ind-1,1);x_known(1:end-(ind-1))];
  end

  hGuess = A\y_known;

  residuals(ind)= sum((y_known - A*hGuess).^2);
end
residuals(3:10)

% Part c
```

```
% 7 seems to be reasonable, so we recover that filter
M = 7;
A = zeros(N,M);
for ind = 1:M
  A(:,ind) = [zeros(ind-1,1);x_known(1:end-(ind-1))];
end
hBest = A\y_known

% Now recover x from y_unknown
H = zeros(N+10,N);
column = zeros(N+10,1);
column(1:M) = hBest;
for ind = 1:N
  H(:,ind) = [zeros(ind-1,1);column(1:end-(ind-1))];
end
x_recovered = H\y_unknown;
residual = sum((y_unknown - H*x_recovered).^2)

fSize = 40;
figure(1)
plot([3:10], residuals(3:10));
xlabel('channel length', 'fontSize', fSize);
ylabel('residual', 'fontSize', fSize);
set(gca,'fontSize', fSize)

figure(2)
stem(x_recovered);
title('Recovered x', 'fontSize', fSize);
xlabel('n', 'fontSize', fSize);
ylabel('x[n]', 'fontSize', fSize);
set(gca,'fontSize', fSize)

% Output
% h4 =
%
%     0.1721
%    -0.0958
%     0.0080
%     1.1205
%
%
% ans =
%
%   109.8208
%    10.3245
```

```
%    10.2338
%     6.0975
%     0.9878
%     0.9672
%     0.9635
%     0.9592
%
%
% hBest =
%
%     0.3141
%    -0.1672
%    -0.0114
%     1.1192
%    -0.0166
%    -0.2711
%     0.2955
%
%
% residual =
%
%     0.4962
```




Recovered x
x[n]
n
3
2
1
0
−1
−2
−3
0
5
10
15
20
25
30
35
40