EE263 Homework 5

Summer 2023

4.830. True/false questions about linear algebra. Determine whether each of the following statements is true or false. In each case, give either a proof or a counterexample.

a) If Q has orthonormal columns, then $\|Q^\mathsf{T} w\| \leq \|w\|$ for all vectors w.

b) Suppose $A \in \mathbb{R}^{m\times p}$ and $B \in \mathbb{R}^{m\times q}$. If $\mathrm{null}(A) = \{0\}$ and $\mathrm{range}(A) \subset \mathrm{range}(B)$, then $p \leq q$.

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

d) If $\mathrm{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right) = \mathrm{rank}(A) = \mathrm{rank}(B)$, then $\mathrm{range}(A) = \mathrm{range}(B)$.

e) Suppose $A \in \mathbb{R}^{m\times n}$. Then, $x \in \mathrm{null}(A^\mathsf{T})$ if and only if $x \notin \mathrm{range}(A)$.

f) Suppose A is invertible. Then, AB is not full rank if and only if B is not full rank.

g) If A is not full rank, then there is a nonzero vector x such that $Ax = 0$.

Solution.

a) The statement is true. Suppose $Q \in \mathbb{R}^{m\times n}$. Because the columns of Q are orthonormal, and hence linearly independent, we know that $m \geq n$. If $m = n$, then Q is an orthogonal matrix, so QQ^T, and we have that

$$\|Q^\mathsf{T} w\|^2 = w^\mathsf{T}(QQ^\mathsf{T})w = w^\mathsf{T} w = \|w\|^2.$$

Now consider the case when $m > n$. Let $q_1, \ldots, q_n$ be the columns of Q:

$$Q = \begin{bmatrix} q_1 & \cdots & q_n \end{bmatrix}.$$

Then, we can extend $(q_1, \ldots, q_n)$ to an orthonormal basis $(q_1, \ldots, q_m)$ for $\mathbb{R}^m$. Define the matrix $\tilde{Q} \in \mathbb{R}^{m\times(m-n)}$ such that

$$\tilde{Q} = \begin{bmatrix} q_{m+1} & \cdots & q_n \end{bmatrix}.$$

Then, we have that $\begin{bmatrix} Q & \tilde{Q} \end{bmatrix}$ is an orthogonal matrix, so that

$$\|w\|^2 = \left\| \begin{bmatrix} Q & \tilde{Q} \end{bmatrix}^\mathsf{T} w \right\|^2 = \left\| \begin{bmatrix} Q^\mathsf{T} \\ \tilde{Q}^\mathsf{T} \end{bmatrix} w \right\|^2 = \|Q^\mathsf{T} w\|^2 + \|\tilde{Q}^\mathsf{T} w\|^2 \geq \|Q^\mathsf{T} w\|^2.$$

Combining these results, we see that if the columns of Q are orthonormal, then $\|Q^\mathsf{T} w\| \leq \|w\|$ for all vectors w. (This result is known as Bessel's inequality.)

b) The statement is true. Because $\mathrm{range}(A) \subset \mathrm{range}(B)$, we have that

$$\mathrm{rank}(A) = \dim(\mathrm{range}(A)) \leq \dim(\mathrm{range}(B)) = \mathrm{rank}(B).$$

Since the rank of a matrix is bounded by its number of columns, we have that

$$\operatorname{rank}(B) \leq q.$$

Conservation of dimension tells us that

$$\operatorname{rank}(A) = \dim(\mathbb{R}^p) - \dim(\operatorname{null}(A)) = \dim(\mathbb{R}^p) - \dim(\{0\}) = p - 0 = p.$$

Combining these results, we have that

$$p = \operatorname{rank}(A) \leq \operatorname{rank}(B) \leq q.$$

c) The statement is true. Suppose $y \in \operatorname{null}(AV_2)$. Then, $V_2 y \in \operatorname{null}(A) = \operatorname{range}(V_1)$, so there exists a vector x such that $V_1 x = V_2 y$. Therefore, we have that

$$V_1 x - V_2 y = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} x \\ -y \end{bmatrix} = 0.$$

Because $\begin{bmatrix} V_1 & V_2 \end{bmatrix}$ is invertible, this implies that

$$\begin{bmatrix} x \\ -y \end{bmatrix} = 0,$$

and hence that $y = 0$. Thus, we see that $\operatorname{null}(AV_2) = \{0\}$.

d) The statement is true. It is sufficient to show that if $\operatorname{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right) = \operatorname{rank}(A)$, then $\operatorname{range}(A) = \operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$. This implies that if $\operatorname{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right) = \operatorname{rank}(A) = \operatorname{rank}(B)$, then

$$\operatorname{range}(A) = \operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right) = \operatorname{range}(B).$$

Consider any $y \in \operatorname{range}(A)$. There exists a vector x such that $Ax = y$. Then, we have that

$$\begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} x \\ 0 \end{bmatrix} = Ax = y,$$

so that $y \in \operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$. Thus, we have that $\operatorname{range}(A) \subset \operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$. (Note that this result holds for any matrices A and B.) Now suppose that $\operatorname{rank}(A) = \operatorname{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$. Let $(q_1, \ldots, q_m)$ be a basis for $\operatorname{range}(A)$. Since $\operatorname{range}(A)$ is a subspace of $\operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$, we can extend this basis to a basis $(q_1, \ldots, q_n)$ for $\operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$. However, it must be the case that

$$n = \operatorname{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right) = \operatorname{rank}(A) = m.$$

Thus, $(q_1, \ldots, q_m)$ is a basis for both $\operatorname{range}(A)$ and $\operatorname{range}(\begin{bmatrix} A & B \end{bmatrix})$. Therefore, for any $y \in \operatorname{range}(\begin{bmatrix} A & B \end{bmatrix})$, there exist scalars $c_1, \ldots, c_m$ such that

$$y = c_1 q_1 + \cdots + c_m q_m.$$

Because $(q_1, \ldots, q_m)$ is also a basis for $\operatorname{range}(A)$, this implies that $y \in \operatorname{range}(A)$. This shows that if $\operatorname{rank}(A) = \operatorname{rank}\left(\begin{bmatrix} A & B \end{bmatrix}\right)$, then $\operatorname{range}\left(\begin{bmatrix} A & B \end{bmatrix}\right) \subset \operatorname{range}(A)$, and thereby completes the proof.

e) The statement is false. In fact, neither direction of the equivalence is true. Consider the matrix

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

For $x = 0$, we have that $x \in \text{null}(A^\mathsf{T})$, and $x \in \text{range}(A)$; for $x = (1, 1)$, we have that $x \notin \text{range}(A)$, and $x \notin \text{null}(A^\mathsf{T})$.

f) The statement is true. We claim that if A is invertible, then $\text{rank}(AB) = \text{rank}(B)$. This implies that AB is full rank if and only if B is full rank. Let $(q_1, \ldots, q_k)$ be a basis for $\text{range}(B)$. We claim that $(Aq_1, \ldots, Aq_k)$ is a basis for AB. Consider any $y \in \text{range}(AB)$. There exists a vector x such that $(AB)x = y$. Note that Bx is a vector in $\text{range}(B)$, so there exist scalars $c_1, \ldots, c_k$ such that

$$Bx = c_1 q_1 + \cdots + c_k q_k.$$

Then, we have that

$$y = (AB)x = A(Bx) = A(c_1 q_1 + \cdots + c_k q_k) = c_1(Aq_1) + \cdots + c_k(Aq_k).$$

This shows that $(Aq_1, \ldots, Aq_k)$ spans $\text{range}(AB)$. (This is true for any matrices A and B.) Suppose there exist scalars $d_1, \ldots, d_k$ such that

$$d_1(Aq_1) + \cdots + d_k(Aq_k) = A(d_1 q_1 + \cdots + d_k q_k) = 0.$$

Since A is invertible, this implies that

$$d_1 q_1 + \cdots + d_k q_k = A^{-1} 0 = 0.$$

Then, because $(q_1, \ldots, q_k)$ is a basis, and hence linearly independent, we have that

$$d_1 = \cdots = d_k = 0.$$

This shows that $(Aq_1, \ldots, Aq_k)$ is linearly independent, and completes the proof that $(Aq_1, \ldots, Aq_k)$ is a basis for $\text{range}(AB)$. Since the dimension of a subspace is the number of vectors in a basis for the subspace, we have that

$$\text{rank}(B) = \dim(\text{range}(B)) = k = \dim(\text{range}(AB)) = \text{rank}(AB).$$

g) The statement is true. If A is strictly fat, then there exists a nonzero vector x such that $Ax = 0$ irrespective of whether of A is full rank. Suppose A is skinny (or square). If A is not full rank, then the columns of A must be linearly dependent: that is, there exist scalars $x_1, \ldots, x_n$, at least one of which is nonzero, such that

$$x_1 A_{*1} + \cdots + x_n A_{*n} = 0.$$

In matrix form, this equation says that

$$\begin{bmatrix} A_{*1} & \cdots & A_{*n} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = Ax = 0,$$

where $x \neq 0$. ■

6.1240. Iteratively reweighted least squares for 1-norm approximation. In an ordinary least squares problem, we are given $A \in \mathbb{R}^{m \times n}$ (skinny and full rank) and $y \in \mathbb{R}^m$, and we choose $x \in \mathbb{R}^n$ in order to minimize

$$\|Ax - y\|_2^2 = \sum_{i=1}^{m} (\tilde{a}_i^T x - y_i)^2.$$

Note that the penalty that we assign to a measurement error does not depend on the sensor from which the measurement was taken. However, this is not always the right thing to do: if we believe that one sensor is more accurate than another, we might want to assign a larger penalty to an error in the measurement from the more accurate sensor. We can account for differences in the accuracies of our sensors by assigning sensor i a weight $w_i > 0$, and then minimizing

$$\sum_{i=1}^{m} w_i (\tilde{a}_i^T x - y_i)^2.$$

By giving larger weights to more accurate sensors, we can account for differences in the precision of our sensors.

a) *Weighted least squares.* Explain how to choose x in order to minimize

$$\sum_{i=1}^{m} w_i (\tilde{a}_i^T x - y_i)^2,$$

where the weights $w_1, \ldots, w_m > 0$ are given.

b) *Iteratively reweighted least squares for ℓ_1-norm approximation.* Consider a cost function of the form

$$\sum_{i=1}^{m} w_i(x) (\tilde{a}_i^T x - y_i)^2. \tag{1}$$

One heuristic for minimizing a cost function of the form given in (1) is *iteratively reweighted least squares*, which works as follows. First, we choose an initial point $x^{(0)} \in \mathbb{R}^n$. Then, we generate a sequence of points $x^{(1)}, x^{(2)}, \ldots \in \mathbb{R}^n$ by choosing $x^{(k+1)}$ in order to minimize

$$\sum_{i=1}^{m} w_i(x^{(k)}) (\tilde{a}_i^T x^{(k+1)} - y_i)^2.$$

Each step of this algorithm involves updating our weights, and solving a weighted least squares problem. Suppose we want to use this method to solve minimize the ℓ_1-norm approximation error, which is defined to be

$$\|Ax - y\|_1 = \sum_{i=1}^{m} |\tilde{a}_i^T x - y_i|,$$

where the matrix $A \in \mathbb{R}^{m \times n}$ and the vector $y \in \mathbb{R}^m$ are given. How should we choose the weights $w_i(x)$ to make the cost function in (1) equal to the ℓ_1-norm approximation error?

c) *Numerical example.* The file `l1_irwls_data.json` contains data $(t_1, y_1), \ldots, (t_m, y_m)$. We want to fit an affine model to this data:

$$y_i = x_1 + x_2 t_i, \qquad i = 1, \ldots, m.$$

Choose $x^{(0)}$ to be the vector of least-squares parameter estimates: that is, choose $x^{(0)}$ in order to minimize

$$\sum_{i=1}^{m} ((x_1^{(0)} + x_2^{(0)} t_i) - y_i)^2.$$

Generate $x^{(1)}, x^{(2)}, \ldots$ using iteratively reweighted least squares for ℓ_1-norm approximation. You can stop generating iterates when $\|x^{(k+1)} - x^{(k)}\| < 10^{-6}$. Report your values of $x^{(0)}$ and the final $x^{(k)}$ in your sequence of points. Draw a scatterplot of the data points (t_i, y_i). Add the fitted lines corresponding to $x^{(0)}$ and the final $x^{(k)}$ to your scatterplot. What do you observe?

Remark. Suppose we fit the least-squares line to some data. Then, a point that is very far from the least-squares line may be an *outlier*: that is, a point that does not seem to follow the same model as the rest of the data. Because such points may not follow the same model as the rest of data, it may make sense to give such points less weight. This idea is the intuition behind iteratively reweighted least squares for ℓ_1-norm approximation.

Solution.

a) We can express the objective function as

$$\begin{aligned}
\sum_{i=1}^{m} w_i(\tilde{a}_i^T x - y_i)^2 &= \sum_{i=1}^{m} (\sqrt{w_i}\tilde{a}_i^T x - \sqrt{w_i} y_i)^2 \\
&= \left\| \begin{bmatrix} \sqrt{w_1}\tilde{a}_1^T \\ \vdots \\ \sqrt{w_1}\tilde{a}_m^T \end{bmatrix} x - \begin{bmatrix} \sqrt{w_1} y_1 \\ \vdots \\ \sqrt{w_m} y_m \end{bmatrix} \right\|^2 \\
&= \|W^{\frac{1}{2}} A x - W^{\frac{1}{2}} y\|^2,
\end{aligned}$$

where $W^{\frac{1}{2}} = \mathrm{diag}(\sqrt{w_1}, \ldots, \sqrt{w_m})$. Note that $(W^{\frac{1}{2}})^2 = W = \mathrm{diag}(w_1, \ldots, w_m)$. Thus, minimizing this objective function is a least-squares problem; the solution is

$$x = ((W^{\frac{1}{2}} A)^T (W^{\frac{1}{2}} A))^{-1} (W^{\frac{1}{2}} A)^T (W^{\frac{1}{2}} y) = (A^T W A)^{-1} A^T W y.$$

b) If we choose $w_i(x) = 1/|\tilde{a}_i^T x - y_i|$, then we have that

$$\sum_{i=1}^{m} w_i(x)(\tilde{a}_i^T x - y_i)^2 = \sum_{i=1}^{m} \frac{1}{|\tilde{a}_i^T x - y_i|} (\tilde{a}_i^T x - y_i)^2 = \sum_{i=1}^{m} |\tilde{a}_i^T x - y_i| = \|Ax - y\|_1.$$

Note that $w_i(x)$ is undefined if $w_i(x) = |\tilde{a}_i^T x - y_i| = 0$. In this case, we can just take $w_i(x)$ to be some large value $w_{\max}$. However, note that in practice it is extremely unlikely that one of the residuals will be exactly equal to zero.

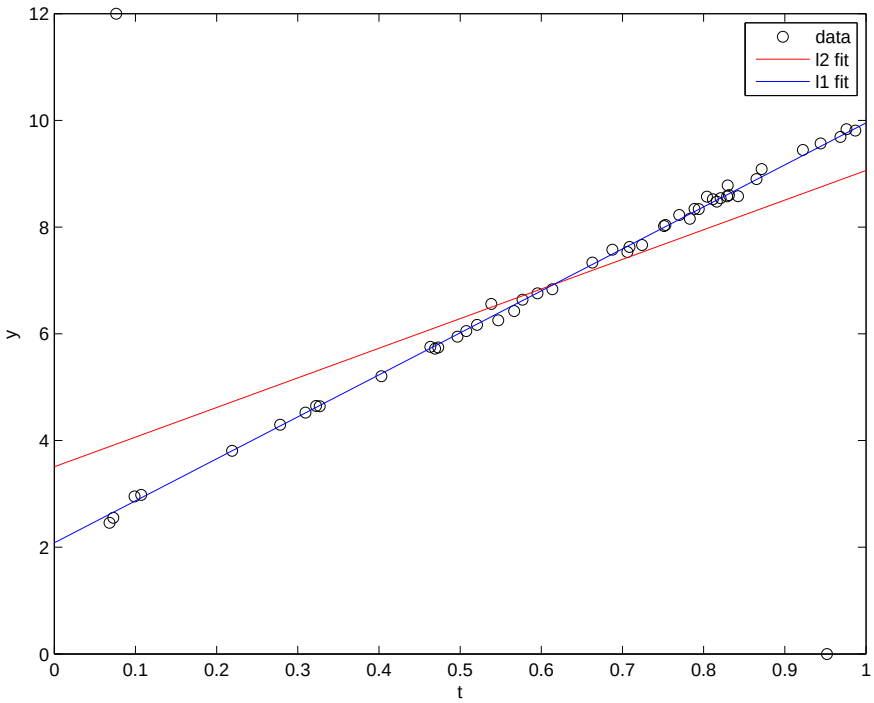
c) For fitting a line in two-dimensions, we have that $\tilde{a}_i = (1, t_i)$. The following code applies iteratively reweighted least squares for ℓ_1-norm approximation to the data defined in `l1_irwls_data.json`.

```
A = [ones(m) zeros(m)]
A[:,2] = t

x = A \ y
x_prev = x
x0 = copy(x)

while true
    w = 1.0 ./ abs.(A*x - y)
    x_prev = copy(x)
    W = diagm(vec(w))
    x = (A'*W*A) \ A'*W*y
    println(norm(x-x_prev))

    if norm(x-x_prev) < 10e-6
        print("Finished!")
        break
    end
end

scatter(t, y, label="(t,y)")
plot!(t, A*x0, label="(t, Ax0)")
plot!(t, A*x, label="(t, Axf)")
```

The initial and final parameter estimates are

$$x^{(0)} = \begin{bmatrix} 3.5084 \\ 5.5540 \end{bmatrix} \qquad \text{and} \qquad x^{(20)} = \begin{bmatrix} 2.0839 \\ 7.8694 \end{bmatrix}.$$

The ℓ_2- and ℓ_1-norm approximations of the data are given in the plot below. We see that the ℓ_2-norm approximation is much more sensitive to the outliers than the ℓ_1-norm approximation.



8.160. Designing an equalizer for backwards-compatible wireless transceivers. You want to design the equalizer for a new line of wireless handheld transceivers (more commonly called walkie-talkies). The transmitter for the new line of transceivers has already been designed (and cannot be changed) – if the input signal is $x \in \mathbb{R}^n$, then the transmitted signal is $y = A_{\text{new}}x \in \mathbb{R}^m$, where $A_{\text{new}} \in \mathbb{R}^{m\times n}$ is known. An equalizer for A_{new} is a matrix $B \in \mathbb{R}^{n\times m}$ such that $By = x$ for every $x \in \mathbb{R}^n$.

The new line of transceivers will replace an older model. Given an input signal $x \in \mathbb{R}^n$, the old line of transceivers transmit a signal $y_{\text{old}} = A_{\text{old}}x \in \mathbb{R}^m$, where $A_{\text{old}} \in \mathbb{R}^{m\times n}$ is known. In addition to providing exact equalization for the new line of transceivers, you want your equalizer to be able to at least partially equalize signals transmitted using the old line of transceivers. In other words, to the extent that it is possible, you want the new line of transceivers to be backwards compatible with the old line of transceivers.

a) Explain how to find an equalizer B that minimizes

$$J = \|BA_{\text{old}} - I\|_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}(BA_{\text{old}} - I)_{ij}^2$$

among all B that exactly equalize A_{new}. Such a B is an exact equalizer for A_{new}, and an approximate equalizer for A_{old}. State any assumptions that are needed for your method to work.

b) The file `backwards_compatible_transceiver_data.json` defines the following variables.

- `Anew`, the $m \times n$ matrix that describes the transmitter used in the new line of transceivers

- `Aold`, the $m \times n$ matrix that describes the transmitter used in the old line of transceivers
- `x`, a vector of length n that serves as an example input signal

Apply your method to this example data. Report the optimal value of J. The pseudoinverse $A_{\text{new}}^{\dagger}$ is another exact equalizer for A_{new}. Compare the optimal value of J, and the value of J achieved by $A_{\text{new}}^{\dagger}$.

c) The example signal x defined in the data file is a binary signal. Form the signal $y_{\text{old}} = A_{\text{old}}x$ transmitted by the old line of transceivers, and construct an estimate of x by equalizing y_{old} using B, and then rounding the result to a binary signal. More concretely, compute the estimate $\hat{x} \in \mathbb{R}^n$, where

$$\hat{x}_i = \begin{cases} 1 & (By_{\text{old}})_i > \frac{1}{2}, \\ 0 & \text{otherwise.} \end{cases}$$

Report the bit error rate of your estimate, which is defined as

$$\frac{1}{n}\sum_{i=1}^{n} \mathbf{I}(x_i \neq \hat{x}_i),$$

where $\mathbf{I}(x_i \neq \hat{x}_i)$ is an indicator function:

$$\mathbf{I}(x_i \neq \hat{x}_i) = \begin{cases} 1 & x_i \neq \hat{x}_i, \\ 0 & \text{otherwise.} \end{cases}$$

Similarly, report the bit error rate if $A_{\text{new}}^{\dagger}$ is used as the equalizer.

Solution.

a) Write the equalizer $B \in \mathbb{R}^{n \times m}$ in terms of its rows:

$$B = \begin{bmatrix} b_1^{\mathsf{T}} \\ \vdots \\ b_n^{\mathsf{T}} \end{bmatrix}.$$

We require that B be an exact equalizer for A_{new}: that is, $BA_{\text{new}} = I$. We can express this condition in terms of the rows of B as

$$A_{\text{new}}^{\mathsf{T}} b_i = e_i, \qquad i = 1, \ldots, n,$$

where e_i denotes the ith standard basis vector in $\mathbb{R}^n$. Similarly, we can write our objective in terms of the rows of B:

$$J = \sum_{i=1}^{n}\sum_{j=1}^{n}(BA_{\text{old}} - I)_{ij}^2 = \sum_{i=1}^{n}\|(BA_{\text{old}} - I)_{i*}\|^2 = \sum_{i=1}^{n}\|A_{\text{old}}^{\mathsf{T}} b_i - e_i\|^2.$$

Thus, we want to solve the following optimization problem.

$$\begin{array}{ll} \underset{B \in \mathbb{R}^{n \times m}}{\text{minimize}} & \sum_{i=1}^{n} \|A_{\text{old}}^{\mathsf{T}} b_i - e_i\|^2 \\ \text{subject to} & A_{\text{new}}^{\mathsf{T}} b_i = e_i \qquad i = 1, \ldots, n \end{array}$$

This problem is separable in the rows of B, allowing us to decompose it into n vector optimization problems:

$$\begin{array}{ll} \underset{b_i \in \mathbb{R}^m}{\text{minimize}} & \|A_{\text{old}}^{\mathsf{T}} b_i - e_i\|^2 \\ \text{subject to} & A_{\text{new}}^{\mathsf{T}} b_i = e_i \end{array}$$

for $i = 1, \ldots, n$. Each of these problems is a linearly constrained minimum-norm problem; the solution of such a problem can be obtained by solving the following system of equations:

$$\begin{bmatrix} A_{\text{old}} A_{\text{old}}^{\mathsf{T}} & A_{\text{new}} \\ A_{\text{new}}^{\mathsf{T}} & 0 \end{bmatrix} \begin{bmatrix} b_i \\ \lambda_i \end{bmatrix} = \begin{bmatrix} A_{\text{old}} e_i \\ e_i \end{bmatrix}, \qquad i = 1, \ldots, n.$$

This method works as long as each of these optimization problems is feasible – that is, as long as we can find a matrix $B \in \mathbb{R}^{n \times m}$ such that

$$BA_{\text{new}} = I.$$

In other words, we require that A_{new} be skinny and full rank (or, equivalently, left invertible). In order for the KKT system to have a unique solution, we require that $A_{\text{new}}^{\mathsf{T}}$ be fat and full rank, and

$$\begin{bmatrix} A_{\text{old}}^{\mathsf{T}} \\ A_{\text{new}}^{\mathsf{T}} \end{bmatrix}$$

be skinny and full rank. Equivalently, we require that A_{new} be skinny and full rank, and $\begin{bmatrix} A_{\text{old}} & A_{\text{new}} \end{bmatrix}$ be fat and full rank.

b) The optimal value of J is 3.2361; in comparison, the value of J achieved by $A_{\text{new}}^{\dagger}$ is 8.0901, which is significantly higher.

c) The bit error rate using the equalizer B is 0.0333, while the bit error rate using $A_{\text{new}}^{\dagger}$ is 0.1000. Thus, we see that B has a much lower bit error rate than $A_{\text{new}}^{\dagger}$.

```
using LinearAlgebra

include("readclassjson.jl")

data = readclassjson("backwards_compatible_transceiver_data.json")

A_new = data["Anew"]

A_old = data["Aold"]

x = data["x"]

block_matrix = [2*A_old*A_old' A_new; A_new' zeros(30, 30)]
```

```
B = zeros(30, 50)

E = I(30)

for i in 1:30
    B[i,:] = (block_matrix\[(2*A_old*E[i,:])' E[i,:]']')[1:50]
end

B

B*A_new

norm(B*A_old - E)^2

A_pinv = inv(A_new'*A_new)*A_new'

norm(A_pinv*A_old - E)^2

y_old = A_old*x

x_est = B*y_old

x_est_pinv = A_pinv*y_old

for i in 1:30
    if x_est[i]>0.5
        x_est[i] = 1
    else
        x_est[i] = 0
    end
end

x_est

for i in 1:30
    if x_est_pinv[i]>0.5
        x_est_pinv[i] = 1
    else
        x_est_pinv[i] = 0
    end
end

x_est_pinv
```

```
function get_bit_rate(x_est, x, n)
    wrong = 0
    for i in 1:n
        if x_est[i] != x[i]
            wrong+=1
        end
    end
    return wrong/n
end

get_bit_rate(x_est, x, 30)

get_bit_rate(x_est_pinv, x, 30)
```

■

8.1460. Filling-in missing data. In this problem we have a signal, $y_i \in \mathbb{R}$ for $i = 1, \ldots, n$, which we view as $y \in \mathbb{R}^n$. We will have $n = 100$. The signal y comes from measurements of a physical system, and so y_{i+1} is measured a short time interval after y_i. Unfortunately, during the data acquisition process some of the data was lost and so the signal we have has gaps in it. Specifically, we have a known set $K \subset \mathbb{Z}$ and we know y_i only for values $i \in K$.

The data for this problem is in the file `missing.json`. The supplied vector `known` contains the list of known points K, and the vector `yknown` is the list of values of y at the points in K. The length of `yknown` is therefore $|K|$.

a) For a signal $z \in \mathbb{R}^n$, we define the discrete derivative $z^{\text{der}} \in \mathbb{R}^{n-1}$ by

$$z_i^{\text{der}} = z_{i+1} - z_i \quad \text{for } i = 1, \ldots, n-1$$

Find the matrix G such that $z^{\text{der}} = Gz$

b) Our first approach will be to find the signal z which minimizes $\|z^{\text{der}}\|$ and satisfies

$$z_i = y_i \quad \text{if } i \in K$$

Give a method finding the optimal z.

c) Find the optimal z in the previous part and plot z_i against i. Be sure to plot the points (i, z_i), not just a line joining them.

d) One way to do a better job at filling in the missing data is to put additional criteria on our estimate. Here we will do this by additionally penalizing the second derivative of z. Define the discrete second derivative $z^{\text{hes}} \in \mathbb{R}^{n-2}$ by

$$z_i^{\text{hes}} = z_{i+2} - 2z_{i+1} + z_i \quad \text{for } i = 1, \ldots, n-2$$

Find the matrix H such that $z^{\text{hes}} = Hz$

e) Define the two objective functions

$$J_1 = \|Gz\|^2 \qquad J_2 = \|Hz\|^2$$

We would like to find the signal z that minimizes

$$J_1 + \mu J_2$$

and satisfies

$$z_i = y_i \quad \text{if } i \in K$$

Give a method for finding the optimal z.

f) Plot the trade-off curve of J_2 (on the vertical axis) versus J_1 (on the horizontal axis). Give the interpretation of the endpoints of this curve.

g) Find the optimal z for the three different cases $\mu = 5, 20, 100$.

Solution.

a) The matrix G is

$$G_{ij} = \begin{cases} -1 & \text{if } i = j \\ 1 & \text{if } j = i + 1 \\ 0 & \text{otherwise} \end{cases}$$

b) The problem is in the form

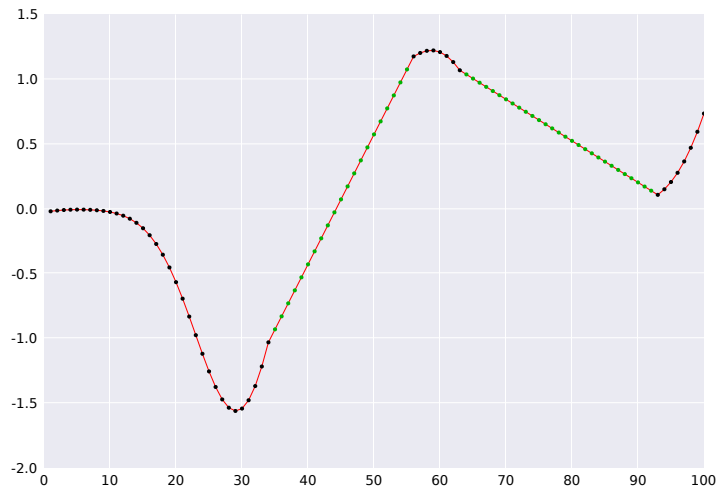
$$\begin{array}{ll} \text{minimize} & \|Az - b\| \\ \text{subject to} & Cz = d \end{array}$$

which (from the lecture notes) has solution

$$\begin{bmatrix} x \\ \lambda \end{bmatrix} = \begin{bmatrix} A^\mathsf{T} A & C^\mathsf{T} \\ C & 0 \end{bmatrix}^{-1} \begin{bmatrix} A^\mathsf{T} b \\ d \end{bmatrix}$$

Here we have $A = G$, $b = 0$. The matrix C consists of the rows i of the identity matrix for which $i \in K$, and $d = y$.

c) The optimal z is below.



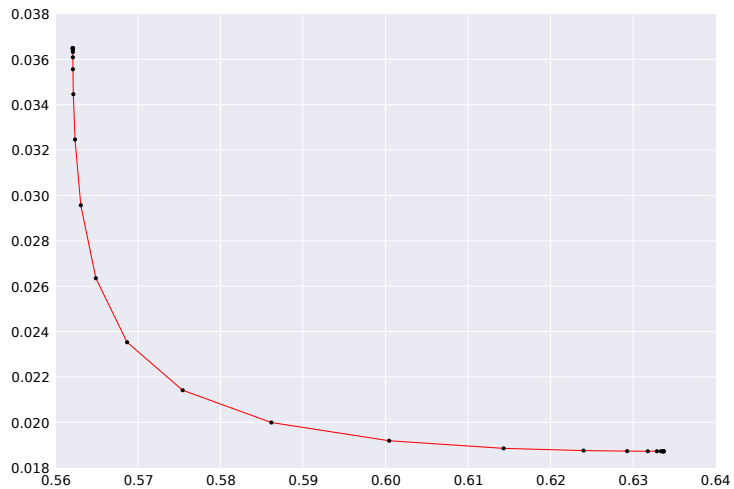
d) The matrix H is

$$H_{ij} = \begin{cases} 1 & \text{if } i = j \\ -2 & \text{if } j = i + 1 \\ 1 & \text{if } j = i + 2 \\ 0 & \text{otherwise} \end{cases}$$

e) This also has the same form as part b). In this case

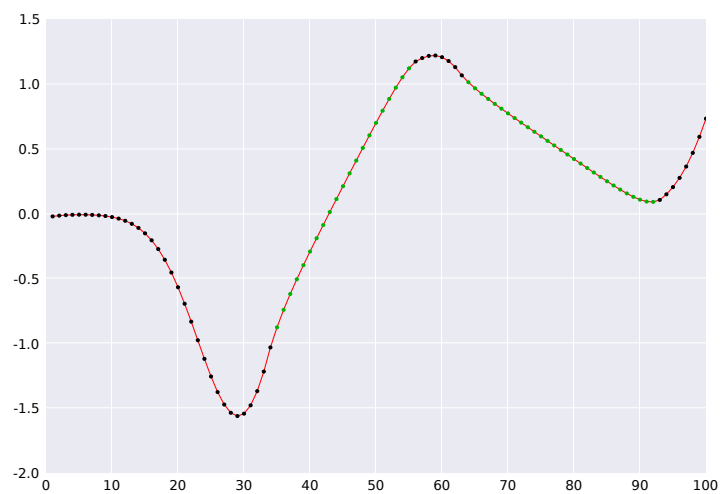
$$A = \begin{bmatrix} D \\ \sqrt{\mu} H \end{bmatrix}$$

f) The trade-off curve is

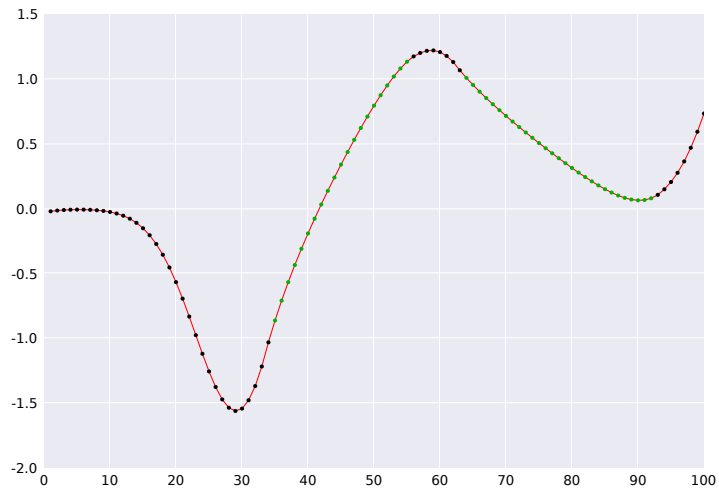


The bottom right corner is the solution when we minimize J_2, with J_1 unconstrained. It is the solution with smallest second derivative that interpolates the data. Similarly the upper left corner minimizes J_1, and is the solution with smallest first derivative that interpolates the data.

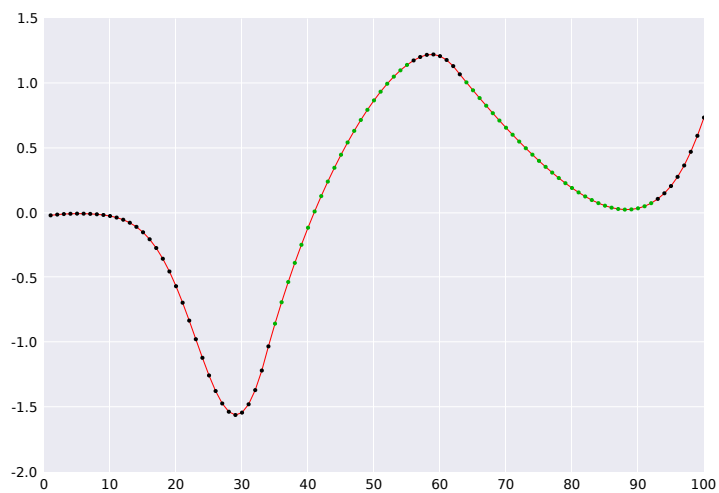
g) The optimal z with $\mu = 5$ is below.



The optimal z with $\mu = 20$ is below.



The optimal z with $\mu = 100$ is below.



9.1360. A simple population model. We consider a certain population of fish (say) each (yearly) season. $x(t) \in \mathbb{R}^3$ will describe the population of fish at year $t \in \mathbb{Z}$, as follows:

- $x_1(t)$ denotes the number of fish less than one year old
- $x_2(t)$ denotes the number of fish between one and two years old
- $x_3(t)$ denotes the number of fish between two and three years

(We will ignore the fact that these numbers are integers.) The population evolves from year t to year $t+1$ as follows.

- The number of fish less than one year old in the next year $(t+1)$ is equal to the total number of offspring born during the current year. Fish that are less than one year old in the current year (t) bear no offspring. Fish that are between one and two years old in the current year (t) bear an average of 2 offspring each. Fish that are between two and three years old in the current year (t) bear an average of 1 offspring each.
- 40% of the fish less than one year old in the current year (t) die; the remaining 60% live on to be between one and two years old in the next year $(t+1)$.
- 30% of the one-to-two year old fish in the current year die, and 70% live on to be two-to-three year old fish in the next year.
- All of the two-to-three year old fish in the current year die.

Express the population dynamics as an autonomous linear system with state $x(t)$, *i.e.*, in the form $x(t+1) = Ax(t)$. **Remark:** this example is silly, but more sophisticated population dynamics models are very useful and widely used.

Solution. We have that

$$\begin{aligned} x_1(t+1) &= 2x_2(t) + x_3(t), \\ x_2(t+1) &= 0.6x_1(t), \\ x_3(t+1) &= 0.7x_2(t). \end{aligned}$$

Thus, we have that $x(t+1) = Ax(t)$, where

$$A = \begin{bmatrix} 0.0 & 2.0 & 1.0 \\ 0.6 & 0.0 & 0.0 \\ 0.0 & 0.7 & 0.0 \end{bmatrix}.$$