EE263 Homework 5

Fall 2025

due Thursday 10/30, at 11:59 PM

6.990. Fleet modeling. In this problem, we will consider model estimation for vehicles in a fleet. We collect input and output data at multiple time instances, for each vehicle in a fleet of vehicles:

$$x^{(k)}(t) \in \mathbb{R}^n, \quad y^{(k)}(t) \in \mathbb{R}, \quad t = 1, \dots, T, \quad k = 1, \dots, K.$$

Here k denotes the vehicle number, t denotes the time, $x^{(k)}(t) \in \mathbb{R}^n$ the input, and $y^{(k)}(t) \in \mathbb{R}$ the output. (In the general case the output would also be a vector; but for simplicity here we consider the scalar output case.)

While it does not affect the problem, we describe a more specific application, where the vehicles are airplanes. The components of the inputs might be, for example, the deflections of various control surfaces and the thrust of the engines; the output might be vertical acceleration. Airlines are required to collect this data, called FOQA data, for every commercial flight. (This description is not needed to solve the problem.)

We will fit a model of the form

$$y^{(k)}(t) \approx a^{\mathsf{T}} x^{(k)}(t) + b^{(k)},$$

where $a \in \mathbb{R}^n$ is the (common) linear model parameter, and $b^{(k)} \in \mathbb{R}$ is the (individual) offset for the kth vehicle.

We will choose these to minimize the mean square error

$$E = \frac{1}{TK} \sum_{t=1}^{\mathsf{T}} \sum_{k=1}^{K} \left(y^{(k)}(t) - a^{\mathsf{T}} x^{(k)}(t) - b^{(k)} \right)^2.$$

a) Explain how to find the model parameters a and $b^{(1)}, \dots, b^{(K)}$.

b) Carry out your method on the data given in `fleet_mod_data.json`. The data is given as $k = 10$ matrices `X1`...`X10` and vectors `y1`...`y10` where each `Xi` is an $n \times T$ matrix X_k with columns $x^{(k)}(1), \dots, x^{(k)}(T)$. Each `yi` is a $1 \times T$ row vector y_k containing $y^{(k)}(1), \dots, y^{(k)}(T)$.

   Give the model parameters a and $b^{(1)}, \dots, b^{(K)}$, and report the associated mean square error E. Compare E to the (minimum) mean square error E^{com} obtained using a common offset $b = b^{(1)} = \cdots = b^{(K)}$ for all vehicles.

   By examining the offsets for the different vehicles, suggest a vehicle you might want to have a maintenance crew check out. (This is a simple, straightforward question; we don't want to hear a long explanation or discussion.)

   Hint: To collect the `Xi` matrices and `yi` vectors into a list, you may use the code

```
data = readclassjson("fleet_mod_data.json")
X = [data["X$i"] for i=1:data["K"]]
y = [data["y$i"] for i=1:data["K"]]
```

6.2300. Recursive estimation. A piecewise constant signal is filtered by convolving it with a smooth function. We start with $x \in \mathbb{R}^n$, and upsample (repeat values) to create $u \in \mathbb{R}^m$. The upsampling repeats each value k times, so that $m = kn$. The constant signal $u \in \mathbb{R}^m$ is given by

$$u_i = x_j \text{ for } k(j-1) < i \le kj$$

The signal u is convolved with a smooth function r, given by

$$r_j = \exp(-j^2/\sigma^2)$$

which is defined for $-q \le j \le q$. The convolution operation generates output $y \in \mathbb{R}^m$, given by

$$y_i = w_i + \sum_{j=\max(1,i-q)}^{\min(m,i+q)} r_{i-j} u_j \tag{1}$$

where w is random measurement noise. We have $n = 10$, $k = 5$, $\sigma = 2$, $q = 10$. We will use regularization parameter $\mu = 0.1$. The file `recursive.json` contains x, y and w, which satisfy equation (1).

a) Find matrix C such that $u = Cx$.

b) Find matrix B such that $y = Bu + w$.

c) Let $A = BC$. Find x^{reg}, the regularized least-squares estimate of x given y. That is, x^{reg} is the x that minimizes

$$\|Ax - y\|^2 + \mu\|x\|^2$$

Plot x^{reg} and x on the same plot. (*i.e.*, plot x_i versus i)

d) We would like to use a recursive method to compute the regularized least-squares estimate. Recall the usual recursive-least-squares algorithm:

$$\begin{aligned}
&P(0) = 0 \in \mathbb{R}^{n\times n}\\
&q(0) = 0 \in \mathbb{R}^n\\
&\textbf{for } i = 0, 1, \ldots,\\
&\qquad P(i+1) = P(i) + a_{i+1}a_{i+1}^\mathsf{T}\\
&\qquad q(i+1) = q(i) + y_{i+1}a_{i+1}
\end{aligned}$$

where a_i^T is the i'th row of A, and y_i is the corresponding ith measurement. Then the estimate based on $y_1, \ldots, y_i$ is $x_{\text{ls}}(i) = P(i)^{-1}q(i)$.

Explain how to modify this algorithm to recursively compute the regularized least-squares estimate.

e) Apply your algorithm to the given data. Plot your estimate when $i = 18$ and when $i = 30$.

f) After applying your algorithm, when $i = m$, you will have computed the same regularized least-squares estimate you did in part (c), but in a different way. At this point, you realize that the data $y_1, \ldots, y_{20}$ was incorrect, and you would like to remove it from your estimate. However, you have already thrown away $y_{21}, \ldots, y_m$. Give an algorithm to adjust your estimate to remove the effect of measurments $y_1, \ldots, y_{20}$. Plot the resulting estimate of x. Note that you only have access to $y_1, \cdots, y_{20}$, the final P and q from part (d), and $a_1, \cdots, a_{20}$.

8.1110. Simultaneous left inverse of two matrices. Consider a system where

$$y = Gx, \quad \tilde{y} = \tilde{G}x$$

where $G \in \mathbb{R}^{m \times n}$, $\tilde{G} \in \mathbb{R}^{m \times n}$. Here x is some variable we wish to estimate or find, y gives the measurements with some set of (linear) sensors, and $\tilde{y}$ gives the measurements with some *alternate* set of (linear) sensors. We want to find a *reconstruction matrix* $H \in \mathbb{R}^{n \times m}$ such that $HG = H\tilde{G} = I$. Such a reconstruction matrix has the nice property that it recovers x perfectly from *either* set of measurements (y or $\tilde{y}$), *i.e.*, $x = Hy = H\tilde{y}$. Consider the specific case

$$G = \begin{bmatrix} 2 & 3 \\ 1 & 0 \\ 0 & 4 \\ 1 & 1 \\ -1 & 2 \end{bmatrix}, \quad \tilde{G} = \begin{bmatrix} -3 & -1 \\ -1 & 0 \\ 2 & -3 \\ -1 & -3 \\ 1 & 2 \end{bmatrix}.$$

Either find an explicit reconstruction matrix H, or explain why there is no such H.

8.1150. Optimal flow on a data collection network. We consider a communications network with m nodes, plus a special destination node, and n communication links. Each communication link connects two (distinct) nodes and is bidirectional, *i.e.*, information can flow in either direction.

We will assume that the network is connected, *i.e.*, there is a path, or sequence of links, from every node (including the special destination node) to every other node. With each communication link we associate a directed arc, which defines the direction of information flow that we will call positive. Using these reference directions, the flow or traffic on link j is denoted f_j. (The units are bits per second, but that won't matter to us.)

The traffic on the network (*i.e.*, the flow in each communication link) is given by a vector $f \in \mathbb{R}^n$. A small example is shown in part 2 of this problem. In this example, nodes 1 and 3 are connected by communication link 4, and the associated arc points from node 1 to node 3. Thus $f_4 = 12$ means the flow on that link is 12 (bits per second), from node 1 to node 3. Similarly, $f_4 = -3$ means the flow on link 4 is 3 (bits per second), from node 3 to node 1.

External information enters each of the m regular nodes and flows across links to the special destination node. In other words, the network is used to collect information from the nodes and route it through the links to the special destination node. (That explains why we call it a data collection network.) At node i, an external information flow s_i (which is nonnegative) enters. The vector $s \in \mathbb{R}^m$ of external flows is sometimes called the *source vector*.

Information flow is conserved. This means that at each node (except the special destination node) the sum of all flows entering the node from communication links connected to that node, plus the external flow, equals the sum of the flows leaving that node on communication links. As an example, consider node 3 in the network of part 2. Links 4 and 5 enter this node, and link 6 leaves the node. Therefore, flow conservation at node 3 is given by

$$f_4 + f_5 + s_3 = f_6.$$

The first two terms on the left give the flow entering node 3 on links 4 and 5; the last term on the left gives the external flow entering node 3. The term on the righthand side gives the flow leaving over link 6. Note that this equation correctly expresses flow conservation regardless of the signs of f_4, f_5, and f_6. Finally, here is the problem.

a) The vector of external flows, $s \in \mathbb{R}^m$, and the network topology, are given, and you must find the flow f that satisfies the conservation equations, and minimizes the mean-square traffic on the network, *i.e.*,

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

Your answer should be in terms of the external flow s, and the *node incidence matrix* $A \in \mathbb{R}^{m\times n}$ that describes the network topology. The node incidence matrix is defined as

$$A_{ij} = \begin{cases} 1 & \text{arc } j \text{ enters (or points into) node } i \\ -1 & \text{arc } j \text{ leaves (or points out of) node } i \\ 0 & \text{otherwise.} \end{cases}$$

Note that each row of A is associated with a node on the network (not including the destination node), and each column is associated with an arc or link.

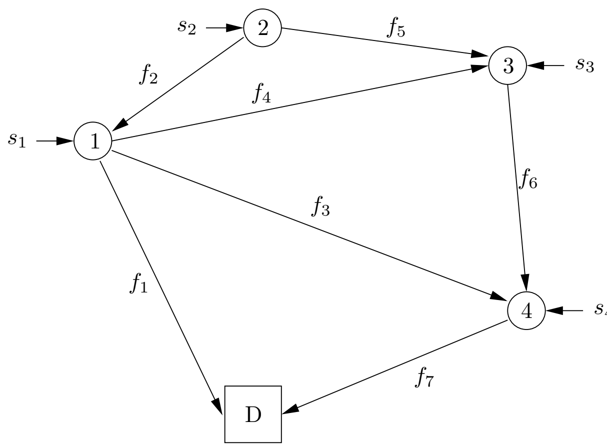
b) Now consider the specific (and very small) network shown below. The nodes are shown as circles, and the special destination node is shown as a square. The external flows are

$$s = \begin{bmatrix} 1 \\ 4 \\ 10 \\ 10 \end{bmatrix}.$$

One simple feasible flow is obtained by routing all the external flow entering each node along a shortest path to the destination. For example, all the external flow entering node 2 goes to node 1, then to the destination node. For node 3, which has two shortest paths to the destination, we arbitrarily choose the path through node 4. This simple routing scheme results in the feasible flow

$$f_{\text{simple}} = \begin{bmatrix} 5 \\ 4 \\ 0 \\ 0 \\ 0 \\ 10 \\ 20 \end{bmatrix}.$$

Find the mean square optimal flow for this problem (as in part 1). Compare the mean square flow of the optimal flow with the mean square flow of f_{simple}.



8.1230. Singularity of the KKT matrix. This problem concerns the general norm minimization with equality constraints problem (described in the lectures notes on pages 8-13),

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

where the variable is $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{m \times n}$, and $C \in \mathbb{R}^{k \times n}$. We asume that C is fat ($k \leq n$), *i.e.*, the number of equality constraints is no more than the number of variables.

Using Lagrange multipliers, we found that the solution can be obtained by solving the linear equations

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

for x and λ. (The vector x gives the solution of the norm minimization problem above.) The matrix above, which we will call $K \in \mathbb{R}^{(n+k)\times(n+k)}$, is called the *KKT matrix* for the problem. (KKT are the initials of some of the people who came up with the optimality conditions for a more general type of problem.)

One question that arises is, when is the KKT matrix K nonsingular? The answer is: K is nonsingular if and only if C is full rank and $\text{null}(A) \cap \text{null}(C) = \{0\}$.

You will fill in all details of the argument below.

a) Suppose C is not full rank. Show that K is singular.

b) Suppose that there is a nonzero $u \in \text{null}(A) \cap \text{null}(C)$. Use this u to show that K is singular.

c) Suppose that K is singular, so there exists a nonzero vector $[u^\mathsf{T}\ v^\mathsf{T}]^\mathsf{T}$ for which

$$\begin{bmatrix} A^\mathsf{T} A & C^\mathsf{T} \\ C & 0 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = 0.$$

Write this out as two block equations, $A^\mathsf{T} Au + C^\mathsf{T} v = 0$ and $Cu = 0$. Conclude that $u \in \text{null}(C)$. Multiply $A^\mathsf{T} Au + C^\mathsf{T} v = 0$ on the left by u^T, and use $Cu = 0$ to conclude that $\|Au\| = 0$, which implies $u \in \text{null}(A)$. Finish the argument that leads to the conclusion that either C is not full rank, or $\text{null}(A) \cap \text{null}(C) \neq \{0\}$.