Alex Lemon
EE263 (Introduction to Linear Dynamical Systems)
Homework 4
July 18, 2013

## 1 Time-delay localization

A beacon at an unknown location $x \in \mathbb{R}^2$ transmits a signal at an unknown time $\tau$. The signal is received at $n$ base stations at known locations $s_1, \ldots, s_n \in \mathbb{R}^2$. Each base station measures the arrival time of the transmitted signal. In particular, the measured arrival time at the $i$th base station is

$$t_i = \frac{1}{c}\|s_i - x\| + \tau + v_i, \qquad i = 1, \ldots, n,$$

where $c$ is the speed of light, and $v_i$ is a measurement error, which is assumed to be small.

(a) Explain how to estimate $x$ and $\tau$ using the arrival times $t_1, \ldots, t_n$.

(b) The file `time_delay_localization_data.m` defines the following variables.

- `S`, a $2 \times 9$ matrix whose columns are the locations of the base station
- `t`, the vector of arrival times
- `c`, the speed of light

Apply your method to this instance of the problem. Report your estimates for $x$ and $\tau$. You may assume that

$$|x_1| \leq 3000, \qquad |x_2| \leq 3000 \qquad \text{and} \qquad |\tau| \leq 5000.$$

## 2 Laplacian smoothing

Consider an undirected graph with $n$ nodes, labeled $1, \ldots, n$. The structure of the graph is described by a set $\mathcal{E} \subseteq \mathbb{N}_n^2$ of edges, where an ordered pair $(i, j) \in \mathbb{N}_n^2$ is contained in $\mathcal{E}$ if and only if $i < j$, and there is and edge between nodes $i$ and $j$. We are given noisy measurements $y_1, \ldots, y_n \in \mathbb{R}$ of some quantity of interest at each of the nodes.

We assume that the quantity of interest is likely to be similar at adjacent nodes. Thus, we choose smoothed estimates $\hat{y}_1, \ldots, \hat{y}_n \in \mathbb{R}$ in order to minimize the objective

$$J = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 + \lambda \sum_{(i,j) \in \mathcal{E}} (\hat{y}_i - \hat{y}_j)^2,$$

where $\lambda > 0$ is a parameter used to control the level of smoothing. The first term in $J$ penalizes deviations of the smoothed estimates from the noisy measurements, while the second term in $J$ penalizes differences in the smoothed estimates of adjacent nodes.

(a) For a fixed value of $\lambda > 0$, explain how to compute the smoothed estimates $\hat{y}_1, \ldots, \hat{y}_n$. What assumptions are needed for your method to work?

(b) The file `laplacian_smoothing_data.m` defines the following variables.

- `n`, the number of nodes
- `y`, the vector of noisy measurements
- `y_true`, the true values of the quantity of interest
- `Edges`, an $m \times 2$ array where each row represents an edge

We think of our graph as an $N \times N$ lattice of nodes. Most nodes are connected to their orthogonal neighbors; however, there is an obstruction in the middle of the lattice that separates certain pairs of nodes. In addition to defining the data, `laplacian_smoothing_data.m` draws a representation of the graph, showing the true and noisy values of the quantity of interest at each node.

For $\lambda \in \{0.2, 2.0, 20\}$, apply your method to the instance of the problem defined by `laplacian_smoothing_data.m`. Use the function `laplacian_smoothing_plot` to draw representations of the smoothed estimates.

## 3 Robust regression using the Huber penalty function

The Huber penalty function is

$$H_\delta(d) = \begin{cases} \frac{1}{2}d^2 & |d| \leq \delta, \\ \delta(|d| - \frac{1}{2}\delta) & \text{otherwise}, \end{cases}$$

where $\delta > 0$ is a parameter. Observe that the Huber penalty function is quadratic for small values of $d$, and linear for large values of $d$. Thus, the $H_\delta(d)$ attempts to combine the sensitivity of the squared-error loss function to small errors, and the robustness of the $\ell_1$ loss function to large errors.

(a) Suppose you want to fit a line to given data points $(t_1, x_1), \ldots, (t_N, x_N) \in \mathbb{R}^2$. Explain how to choose the parameters $a$ and $b$ in order to minimize the total Huber loss:

$$J = \sum_{i=1}^{N} H_\delta(at_i + b - x_i).$$

In particular, what is the weight function, and what is the update equation?

(b) Apply your method to the defined in the file `huber_penalty_function_data.m` using $\delta = 1$. Report your estimates of the parameters $a$ and $b$, and the corresponding value of the total Huber loss. Make a plot of the data, the line corresponding to your estimates of $a$ and $b$, and the line obtained using least-squares. Briefly comment on your results.

## 4 Least-squares deconvolution

A communications channel is modeled as a finite-impulse-response (FIR) filter:

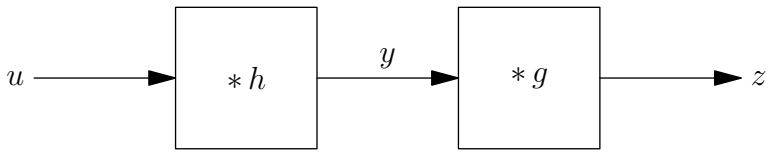
$$y(t) = \sum_{\tau=0}^{n-1} u(t-\tau)h(\tau),$$

where $u : \mathbb{Z} \to \mathbb{R}$ is the input to the channel, $y : \mathbb{Z} \to \mathbb{R}$ is the output of the channel, and $h(0), \ldots, h(n-1)$ is the impulse response of the channel. We say that $y$ is the discrete-time convolution of $h$ and $u$, and write $y = h * u$.

We want to design a deconvolution filter or equalizer, which is also an FIR filter:

$$z(t) = \sum_{\tau=0}^{m-1} y(t-\tau)g(\tau),$$

where $y : \mathbb{Z} \to \mathbb{R}$ is the input to the equalizer, $z : \mathbb{Z} \to \mathbb{R}$ is the output of the equalizer, and $g(0), \ldots, g(m-1)$ is the impulse response of the filter. We can design such a filter by specifying $g(0), \ldots, g(m-1)$. The system consisting of the communications channel and the equalizer is represented by the following block diagram.




The goal is to choose $g = (g(0), \ldots, g(m-1))$ so that the filter output is approximately equal to the channel input delayed by $D$ samples: that is, $z(t) \approx u(t-D)$. Since $z = g * h * u$, this means that we would like

$$(g * h)(t) \approx \begin{cases} 0 & t \neq D, \\ 1 & t = D. \end{cases}$$

We call $g * h$ the equalized impulse response; the goal is to make this as close as possible to a $D$-sample delay. Specifically, the least-squares equalizer $g$ minimizes the sum of squared errors,

$$\sum_{t \neq D} (g * h)(t)^2,$$

subject to the constraint that

$$(g * h)(D) = 1.$$

(a) Given $m$ and $D$, explain how to find the least-squares equalizer $g$.

(b) The file `deconv_data.m` defines the following variables.

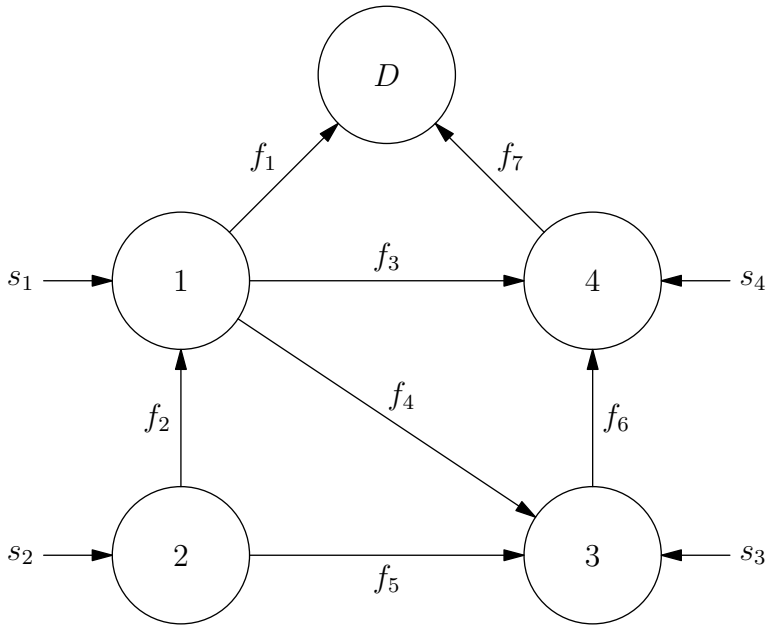
- `h`, the channel impulse response
- `y`, a sample realization of the channel output

Find the least-squares equalizer for this channel with $m = 20$ and $D = 12$. Plot the impulse responses $h$ of the channel, and $g$ of the equalizer; plot the equalized impulse response $g * h$.

(c) The sample realization of the channel output given in `deconv_data.m` was generated using a causal binary input: that is, $u(t) \in \{+1, -1\}$ for $t \geq 0$, and $u(t) = 0$ for $t < 0$. Compute the equalized output $z = g * y$. Plot $y$ and $z$ versus $t$. Give histograms of the distributions of $y$ and $z$ (remove the first and last $D$ samples of $z$ before making the histogram). Comment on your results.

## 5 Optimal flow on a data-collection network

We consider a communication network with $m$ regular nodes, a special destination destination node, and $n$ communication links. Each communication link connects two (distinct) nodes, and is bidirectional (that is, information can flow in either direction). We will assume that the network is connected (that is, there is a path, or sequence of links, between any two nodes in the network). We associate a directed arc with each communication link in order to define the "positive" direction of information flow on the link. (The communication link is symmetric, but we need some way of keeping track of the direction of information flow.) The flow or traffic on link $j$ is denoted $f_j$, and is positive if information is flowing in the direction of the corresponding arc, and negative if information is flowing in the opposite direction; the units of $f_j$ are bits per second. Thus, the traffic on the network is described by a vector $f \in \mathbb{R}^n$. An example of such a network is shown below.




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 that the flow on link 4 is 12 bits per second from node 1 to node 3. Similarly, $f_4 = -3$ means that 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. Intuitively, each of the nodes collects data, and the network routes all of the information to the special destination node. We let $s_i \geq 0$ denote the external information flow into node $i$. The vector $s \in \mathbb{R}^n$ of external flows is called the source vector. Information flow is conserved: 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 into that node is equal to the sum of all flows leaving the node by communications links connected to that node. (This assumption says that the regular nodes do not have long-term memory: they cannot store the data they receive, and must transmit it immediately.) Consider node 3 in the example network above. Links 4 and 5 enter this node, and link 6 leaves the node. Therefore, conservation of information at node 3 says that

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

Note that this equation is correct regardless of the signs of $f_4$, $f_5$ and $f_6$.

(a) You are given the vector of external flows, $s \in \mathbb{R}^m$, and the network topology, which is described by the node incidence matrix $A \in \mathbb{R}^{m \times n}$, where

$$A_{ij} = \begin{cases} +1 & \text{arc } j \text{ enters node } i, \\ -1 & \text{arc } j \text{ leaves node } i, \\ 0 & \text{otherwise.} \end{cases}$$

Note that each row of $A$ is associated with a node, and each column of $A$ is associated with a communication link. Explain how to find the vector of flows $f \in \mathbb{R}^n$ that minimizes the mean squared traffic on the network,

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

subject to the constraints imposed by conservation of information.

(b) Consider the example network given above. Suppose that the external flows are

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

A simple routing scheme is to route all of the external flow entering a node along a shortest path to the destination. For example, all of the external flow entering node 2 is sent to node 1, and then to the destination node. For node 3, there are two shortest paths to the destination; we arbitrarily choose the path through node 4. This simple routing scheme gives the flow

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

Compute the optimal flow for the example network, and compare its mean squared flow to that of $f_{\text{simple}}$.

## 6 Minimum-energy rendezvous

The dynamics of two vehicles at times $t = 0, 1, 2, \ldots$ are described by the equations

$$x(t+1) = Ax(t) + bu(t),$$
$$z(t+1) = Fz(t) + gv(t),$$

where

- $x(t) \in \mathbb{R}^n$ is the state of vehicle 1 at time $t$,
- $z(t) \in \mathbb{R}^n$ is the state of vehicle 2 at time $t$,
- $u(t) \in \mathbb{R}$ is the input to vehicle 1 at time $t$, and
- $v(t) \in \mathbb{R}$ is the input to vehicle 2 at time $t$.

You are given the initial states of the vehicles: $x(0) = x_0 \in \mathbb{R}^n$ and $z(0) = z_0 \in \mathbb{R}^n$. You want to find inputs for the two vehicles over the time interval $t = 0, \ldots, N-1$ such that they rendezvous at some state $w \in \mathbb{R}^n$ at time $t = N$: that is, such that $x(N) = z(N) = w$. The point $w \in \mathbb{R}^n$ is called the rendezvous point. You are allowed to choose the rendezvous point $w \in \mathbb{R}^n$, and the input sequences

$$u(0), \ldots, u(N-1), \qquad \text{and} \qquad v(0), \ldots, v(N-1).$$

The total input energy is

$$E = \sum_{t=0}^{N-1} u(t)^2 + \sum_{t=0}^{N-1} v(t)^2.$$

Explain how to find a rendezvous point, $w \in \mathbb{R}^n$, and corresponding input sequences, $u(t)$ and $v(t)$, that minimize the total input energy. State any assumptions that are needed for your method to work.