Homework 5

August 5, 2017

1. **Linear dynamical system with constant input.** We consider the system $\dot{x} = Ax + b$, with $x(t) \in \mathbb{R}^n$. A vector x_e is an equilibrium point if $0 = Ax_e + b$. (This means that the constant trajectory $x(t) = x_e$ is a solution of $\dot{x} = Ax + b$.)

   a) When is there an equilibrium point?

   b) When are there multiple equilibrium points?

   c) When is there a unique equilibrium point?

   d) Now suppose that x_e is an equilibrium point. Define $z(t) = x(t) - x_e$. Show that $\dot{z} = Az$. From this, give a general formula for $x(t)$ (involving x_e, $\exp(tA)$, $x(0)$).

   e) Show that if all eigenvalues of A have negative real part, then there is exactly one equilibrium point x_e, and for any trajectory $x(t)$, we have $x(t) \to x_e$ as $t \to \infty$.

2. **Eigenvalues and singular values of a symmetric matrix.** Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues, and let $\sigma_1, \ldots, \sigma_n$ be the singular values of a matrix $A \in \mathbb{R}^{n \times n}$, which satisfies $A = A^\mathsf{T}$. (The singular values are based on the full SVD: If $\operatorname{rank}(A) < n$, then some of the singular values are zero.) You can assume the eigenvalues (and of course singular values) are sorted, *i.e.*, $\lambda_1 \geq \cdots \geq \lambda_n$ and $\sigma_1 \geq \cdots \geq \sigma_n$. How are the eigenvalues and singular values related?

3. **Determining the number of signal sources.** The signal transmitted by n sources is measured at m receivers. The signal transmitted by each of the sources at sampling period k, for $k = 1, \ldots, p$, is denoted by an n-vector $x(k) \in \mathbb{R}^n$. The gain from the j-th source to the i-th receiver is denoted by $a_{ij} \in \mathbb{R}$. The signal measured at the receivers is then

$$y(k) = A\,x(k) + v(k), \quad k = 1, \ldots, p,$$

   where $v(k) \in \mathbb{R}^m$ is a vector of sensor noises, and $A \in \mathbb{R}^{m \times n}$ is the matrix of source to receiver gains. However, we do not know the gains a_{ij}, nor the transmitted signal $x(k)$, nor even the number of sources present n. We only have the following additional *a priori* information:

   - We expect the number of sources to be less than the number of receivers (*i.e.*, $n < m$, so that A is skinny);
   - A is full-rank and well-conditioned;

- All sources have roughly the same average power, the signal $x(k)$ is unpredictable, and the source signals are unrelated to each other; Hence, given enough samples (*i.e.*, p large) the vectors $x(k)$ will 'point in all directions';
- The sensor noise $v(k)$ is small relative to the received signal $A\,x(k)$.

Here's the question:

a) You are given a large number of vectors of sensor measurements $y(k) \in \mathbb{R}^m,\ k = 1, \dots, p$. How would you estimate the number of sources, n? Be sure to clearly describe your proposed method for determining n, and to explain when and why it works.

b) Try your method on the signals given in the file `num_sources.json`. Running this script will define the variables:

   - `m`, the number of receivers;
   - `p`, the number of signal samples;
   - `Y`, the receiver sensor measurements, an array of size `m` by `p` (the k-th column of `Y` is $y(k)$.)

   What can you say about the number of signal sources present? Repeat your analysis for the signals given in the files `nsource2.m`, and `nsource3.m`. Each of the three sets of signal samples was obtained under different conditions, *i.e.*, a different number of sources, and different source to sensor gains (but the conditions are the same for all the samples in each set). For each of the three sets of sensor signals, what can you say about the number of signal sources present?

Note: Our problem description and assumptions are not precise. An important part of this problem is to explain your method, and clarify the assumptions.

4. A heuristic for MAXCUT. Consider a graph with n nodes and m edges, with the nodes labeled $1, \dots, n$ and the edges labeled $1, \dots, m$. We partition the nodes into two groups, B and C, *i.e.*, $B \cap C = \emptyset$, $B \cup C = \{1, \dots, n\}$. We define the number of *cuts* associated with this partition as the number of edges between pairs of nodes when one of the nodes is in B and the other is in C. A famous problem, called the MAXCUT problem, involves choosing a partition (*i.e.*, B and C) that maximizes the number of cuts for a given graph. For any partition, the number of cuts can be no more than m. If the number of cuts is m, nodes in group B connect only to nodes in group C and the graph is bipartite.

The MAXCUT problem has many applications. We describe one here, although you do not need it to solve this problem. Suppose we have a communication system that operates with a two-phase clock. During periods $t = 0, 2, 4, \dots$, each node in group B transmits data to nodes in group C that it is connected to; during periods $t = 1, 3, 5, \dots$, each node in group C transmits to the nodes in group B that it is connected to. The number of cuts, then, is exactly the number of successful transmissions that can occur in a two-period cycle. The MAXCUT problem is to assign nodes to the two groups so as to maximize the overall efficiency of communication.

It turns out that the MAXCUT problem is hard to solve exactly, at least if we don't want to resort to an exhaustive search over all, or most of, the 2^{n-1} possible partitions. In this problem we explore a sophisticated heuristic method for finding a good (if not the best) partition in a way that scales to large graphs.

We will encode the partition as a vector $x \in \mathbb{R}^n$, with $x_i \in \{-1, 1\}$. The associated partition has $x_i = 1$ for $i \in B$ and $x_i = -1$ for $i \in C$. We describe the graph by its node adjacency matrix $A \in \mathbb{R}^{n \times n}$ C), with

$$A_{ij} = \begin{cases} 1 & \text{there is an edge between node } i \text{ and node } j \\ 0 & \text{otherwise} \end{cases}$$

Note that A is symmetric and $A_{ii} = 0$ (since we do not have self-loops in our graph).

a) Find a symmetric matrix P, with $P_{ii} = 0$ for $i = 1, \ldots, n$, and a constant d, for which $x^\mathsf{T} P x + d$ is the number of cuts encoded by any partitioning vector x. Explain how to calculate P and d from A. Of course, P and d cannot depend on x.

   The MAXCUT problem can now be stated as the optimization problem

$$\begin{array}{ll} \text{maximize} & x^\mathsf{T} P x + d \\ \text{subject to} & x_i^2 = 1, \quad i = 1, \ldots, n, \end{array}$$

   with variable $x \in \mathbb{R}^n$.

b) A famous heuristic for approximately solving MAXCUT is to replace the n constraints $x_i^2 = 1$, $i = 1, \ldots, n$, with a single constraint $\sum_{i=1}^n x_i^2 = n$, creating the so-called *relaxed* problem

$$\begin{array}{ll} \text{maximize} & x^\mathsf{T} P x + d \\ \text{subject to} & \displaystyle\sum_{i=1}^n x_i^2 = n. \end{array}$$

   Explain how to solve this relaxed problem (even if you could not solve part (a)).

   Let $x^\star$ be a solution to the relaxed problem. We generate our candidate partition with $x_i = \text{sign}(x_i^\star)$. (This means that $x_i = 1$ if $x_i^\star \geq 0$, and $x_i = -1$ if $x_i^\star < 0$.) really

   Remark: We can give a geometric interpretation of the relaxed problem, which will also explain why it's called relaxed. The constraints in the problem in part (a), that $x_i^2 = 1$, require x to lie on the vertices of the unit hypercube. In the relaxed problem, the constraint set is the unit ball of unit radius. Because this constraint set is larger than the original constraint set (*i.e.*, it includes it), we say the constraints have been relaxed.

c) Run the MAXCUT heuristic described in part (b) on the data given in `max_cut_data.json`. How many cuts does your partition yield?

   A simple alternative to MAXCUT is to generate a large number of random partitions, using the random partition that maximizes the number of cuts as an approximate solution. Carry out this method with 1000 random partitions generated by `x = sign(rand(n,1)-0.5)`. What is the largest number of cuts obtained by these random partitions?

Note: There are many other heuristics for approximately solving the MAXCUT problem. However, we are not interested in them. In particular, please do not submit any other method for approximately solving MAXCUT.

5. **Simultaneously estimating student ability and exercise difficulty.** Each of n students takes an exam that contains m questions. Student j receives (nonnegative) grade G_{ij} on question i. One simple model for predicting the grades is to estimate $G_{ij} \approx \hat{G}_{ij} = a_j/d_i$, where a_j is a (nonnegative) number that gives the *ability* of student j, and d_i is a (positive) number that gives the *difficulty* of exam question i. Given a particular model, we could simultaneously scale the student abilities and the exam difficulties by any positive number, without affecting $\hat{G}_{ij}$. Thus, to ensure a unique model, we will *normalize* the exam question difficulties d_i, so that the mean exam question difficulty across the m questions is 1.

   In this problem, you are given a complete set of grades (*i.e.*, the matrix $G \in \mathbb{R}^{m\times n}$). Your task is to find a set of nonnegative student abilities, and a set of positive, normalized question difficulties, so that $G_{ij} \approx \hat{G}_{ij}$. In particular, choose your model to minimize the RMS error, J,

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

   This can be compared to the RMS value of the grades,

$$\left(\frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} G_{ij}^2 \right)^{1/2}.$$

   a) Explain how to solve this problem, using any concepts from EE263. If your method is approximate, or not guaranteed to find the global minimum value of J, say so. If carrying out your method requires some rank or other conditions to hold, say so.

      Note: You do not have to concern yourself with the requirement that a_j are nonnegative and d_i are positive. You can just assume this works out, or is easily corrected.

   b) Carry out your method on the data found in `grade_data.json`. Give the optimal value of J, and also express it as a fraction of the RMS value of the grades. Give the difficulties of the 7 problems on the exam.

6. **Identification of gene regulatory networks.** Consider a gene regulatory network (GRN) with n genes labeled $1, \ldots, n$. The expression level of each gene is associated with the concentration of a specific mRNA sequence. Since we can measure the concentrations of mRNA sequences using oligonucleotide microarrays, we will use the mRNA concentrations as proxies for the expression levels of the corresponding genes.

   Let $x_i(t)$ denote the concentration of the mRNA sequence associated with the ith gene. A simple model for the dynamics of the GRN is

$$\dot{x}_i(t) = -\alpha_i x_i(t) + \sum_{j \neq i} W_{ij} x_j(t) + \beta_i(t) + \epsilon_i(t), \qquad i = 1, \ldots, n,$$

where $x_i(t)$ is the concentration of the mRNA sequence associated with the expression level of the ith gene at time t, W_{ij} is the coupling constant from the jth gene to the ith gene, α_i is the self-regulation rate of the ith gene, $\beta_i(t)$ is the external stimulus applied to the ith gene at time t, and $\epsilon_i(t)$ is an error term that includes measurement errors and unmodeled effects. (We assume that $\epsilon_i(t)$ is small.)

a) Suppose we apply a stimulus sequence to the GRN, and measure $\beta(1), \ldots, \beta(T)$, $x(1), \ldots, x(T)$ and $\dot{x}(1), \ldots, \dot{x}(T)$. Explain how to use the measurements to estimate the model parameters α_i and W_{ij}.

b) Apply your method to the data given in `grn_data.json`. Report the gene with the highest self-regulation rate, as well as the corresponding self-regulation rate. We say that the genes i and j are strongly coupled if $|W_{ij}|$ is large. Which two genes are most strongly coupled? What is the corresponding coupling constant?

c) Suppose the external stimulus applied to the system is constant over time: that is, there exists a vector $\bar{\beta} \in \mathbb{R}^n$ such that $\beta(t) = \bar{\beta}$ for all t. Derive a closed-form expression for $x(t)$ in terms of $x(0)$, α, W, and $\bar{\beta}$.

d) Simulate the system using $\bar{\beta} = \beta(1)$ (the value of $\beta(t)$ at time $t = 1$ in the experimental data), the value of $x(0)$ given in the data file, and the model parameters that you estimated above. Plot $x_j(t)$ versus time for $0 \leq t \leq T_{\text{sim}} = 10$ and $j = 1, 2, 3$. (Put all of your plots on the same set of axes.)