CVX101 Homework 8 solutions

1. *Smoothed fit to given data.* Consider the problem

$$\text{minimize} \quad f(x) = \textstyle\sum_{i=1}^n \psi(x_i - y_i) + \lambda \sum_{i=1}^{n-1} (x_{i+1} - x_i)^2$$

where $\lambda > 0$ is smoothing parameter, ψ is a convex penalty function, and $x \in \mathbf{R}^n$ is the variable. We can interpret x as a smoothed fit to the vector y.

(a) What is the structure in the Hessian of f?

(b) Extend to the problem of making a smooth fit to two-dimensional data, *i.e.*, minimizing the function

$$\sum_{i,j=1}^n \psi(x_{ij} - y_{ij}) + \lambda \left(\sum_{i=1}^{n-1} \sum_{j=1}^{n} (x_{i+1,j} - x_{ij})^2 + \sum_{i=1}^{n} \sum_{j=1}^{n-1} (x_{i,j+1} - x_{ij})^2 \right),$$

with variable $X \in \mathbf{R}^{n \times n}$, where $Y \in \mathbf{R}^{n \times n}$ and $\lambda > 0$ are given.

Solution.

(a) Tridiagonal.

(b) Block-tridiagonal if we store the elements of X columnwise. The blocks have size $n \times n$. The diagonal blocks are tridiagonal. The blocks on the first sub-diagonal are diagonal.

2. *Maximum likelihood prediction of team ability.* A set of n teams compete in a tournament. We model each team's ability by a number $a_j \in [0, 1]$, $j = 1, \ldots, n$. When teams j and k play each other, the probability that team j wins is equal to $\mathbf{prob}(a_j - a_k + v > 0)$, where $v \sim \mathcal{N}(0, \sigma^2)$.

You are given the outcome of m past games. These are organized as

$$(j^{(i)}, k^{(i)}, y^{(i)}), \quad i = 1, \ldots, m,$$

meaning that game i was played between teams $j^{(i)}$ and $k^{(i)}$; $y^{(i)} = 1$ means that team $j^{(i)}$ won, while $y^{(i)} = -1$ means that team $k^{(i)}$ won. (We assume there are no ties.)

(a) Formulate the problem of finding the maximum likelihood estimate of team abilities, $\hat{a} \in \mathbf{R}^n$, given the outcomes, as a convex optimization problem. You will find the *game incidence matrix* $A \in \mathbf{R}^{m \times n}$, defined as

$$A_{il} = \begin{cases} y^{(i)} & l = j^{(i)} \\ -y^{(i)} & l = k^{(i)} \\ 0 & \text{otherwise,} \end{cases}$$

useful.

The prior constraints $\hat{a}_i \in [0,1]$ should be included in the problem formulation. Also, we note that if a constant is added to all team abilities, there is no change in the probabilities of game outcomes. This means that $\hat{a}$ is determined only up to a constant, like a potential. But this doesn't affect the ML estimation problem, or any subsequent predictions made using the estimated parameters.

(b) Find $\hat{a}$ for the team data given in `team_data.m`, in the matrix `train`. (This matrix gives the outcomes for a tournament in which each team plays each other team once.) You may find the CVX function `log_normcdf` helpful for this problem.

You can form A using the commands

```
A = sparse(1:m,train(:,1),train(:,3),m,n) + ...
    sparse(1:m,train(:,2),-train(:,3),m,n);
```

(c) Use the maximum likelihood estimate $\hat{a}$ found in part (b) to predict the outcomes of next year's tournament games, given in the matrix `test`, using $\hat{y}^{(i)} = \mathbf{sign}(\hat{a}_{j^{(i)}} - \hat{a}_{k^{(i)}})$. Compare these predictions with the actual outcomes, given in the third column of `test`. Give the fraction of correctly predicted outcomes.

The games played in `train` and `test` are the same, so another, simpler method for predicting the outcomes in `test` it to just assume the team that won last year's match will also win this year's match. Give the percentage of correctly predicted outcomes using this simple method.

Solution.

(a) The likelihood of the outcomes y given a is

$$p(y|a) = \prod_{i=1,\dots,n} \Phi\left(\frac{1}{\sigma} y^{(i)}(a_{j^{(i)}} - a_{k^{(i)}})\right),$$

where Φ is the cumulative distribution of the standard normal. The log-likelihood function is therefore

$$l(a) = \log p(y|a) = \sum_i \log \Phi\left((1/\sigma)(Aa)_i\right).$$

This is a concave function.

The maximum likelihood estimate $\hat{a}$ is any solution of

$$\begin{array}{ll} \text{maximize} & l(a) \\ \text{subject to} & 0 \preceq a \preceq 1. \end{array}$$

This is a convex optimization problem since the objective, which is maximized, is concave, and the constraints are $2n$ linear inequalities.

(b) The following code solves the problem

Using this code we get that $\hat{a} = (1.0, 0.0, 0.68, 0.37, 0.79, 0.58, 0.38, 0.09, 0.67, 0.58)$.

(c) The following code is used to predict the outcomes in the test set

The maximum likelihood estimate gives a correct prediction of 86.7% of the games in `test`. On the other hand, 75.6% of the games in `test` have the same outcome as the games in `train`.

3. *Allocation of interdiction effort.* A smuggler moves along a directed acyclic graph with m edges and n nodes, from a source node (which we take as node 1) to a destination node (which we take as node n), along some (directed) path. Each edge k has a detection failure probability p_k, which is the probability that the smuggler passes over that edge undetected. The detection events on the edges are independent, so the probability that the smuggler makes it to the destination node undetected is $\prod_{j\in\mathcal{P}} p_j$, where $\mathcal{P} \subset \{1,\dots,m\}$ is (the set of edges on) the smuggler's path. We assume that the smuggler knows the detection failure probabilities and will take a path that maximizes the probability of making it to the destination node undetected. We let $P^{\max}$ denote this maximum probability (over paths). (Note that this is a function of the edge detection failure probabilities.)

   The edge detection failure probability on an edge depends on how much interdiction resources are allocated to the edge. Here we will use a very simple model, with $x_j \in \mathbf{R}_+$ denoting the effort (say, yearly budget) allocated to edge j, with associated detection failure probability $p_j = e^{-a_j x_j}$, where $a_j \in \mathbf{R}_{++}$ are given. The constraints on x are a maximum for each edge, $x \preceq x^{\max}$, and a total budget constraint, $\mathbf{1}^T x \le B$.

   (a) Explain how to solve the problem of choosing the interdiction effort vector $x \in \mathbf{R}^m$, subject to the constraints, so as to minimize $P^{\max}$. Partial credit will be given for a method that involves an enumeration over all possible paths (in the objective or constraints). *Hint.* For each node i, let P_i denote the maximum of $\prod_{k\in\mathcal{P}} p_k$ over all paths $\mathcal{P}$ from the source node 1 to node i (so $P^{\max} = P_n$).

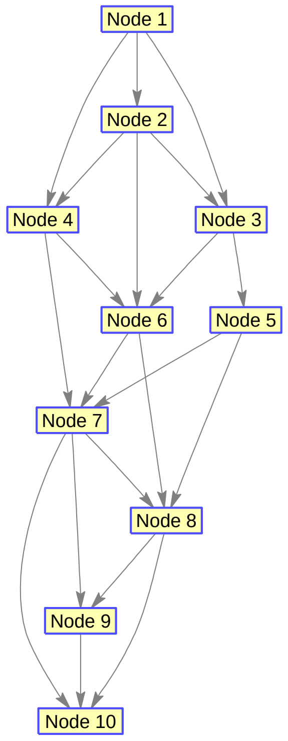
   (b) Carry out your method on the problem instance given in `interdict_alloc_data.m`. The data file contains the data a, $x^{\max}$, B, and the graph incidence matrix $A \in \mathbf{R}^{n\times m}$, where

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

   Give $P^{\max\star}$, the optimal value of $P^{\max}$, and compare it to the value of $P^{\max}$ obtained with uniform allocation of resources, *i.e.*, with $x = (B/m)\mathbf{1}$.

   Hint. Given a vector $z \in \mathbf{R}^n$, $A^T z$ is the vector of edge differences: $(A^T z)_j = z_k - z_l$ if edge j goes from node l to node k.

The following figure shows the topology of the graph in question. (The data file contains A; this figure, which is not needed to solve the problem, is shown here so you can visualize the graph.)



Solution. We will work with the logs of the detection failure probabilities, $y_j = \log p_j = -a_j x_j$. Consider these as edge weights on the graph. The smuggler chooses a path from source to destination that maximizes the total path weight (*i.e.*, the sum of weights along its edges). Thus $\log P^{\max}$ is the maximum, over the set of paths from source to destination, of a sum of linear functions of x; so it is a piecewise-linear convex function of x. It follows that minimizing $\log P^{\max}$ subject to the constraints $x^{\max} \succeq x \succeq 0$, $\mathbf{1}^T x \leq B$, is a convex optimization problem. Unfortunately, this formulation requires an enumeration of all paths. So let's find one that does not.

Following the given hint, we can write a recursion for P_i as follows:

$$P_i = \max_{k \in \mathrm{pred}(i)} \left(p_{ik} P_k \right),$$

where $\mathrm{pred}(i)$ denotes the predecessor nodes of i, and p_{ik} is the detection failure probability on the edge from k to i. We also have $P_1 = 1$, $P_n = P^{\max}$.

Letting $z_i = \log P_i$ we have the problem

$$\begin{array}{ll} \text{minimize} & z_n \\ \text{subject to} & z_i = \max_{k \in \text{pred}(i)} \left(-a_{ik} x_{ik} + z_k\right), \quad i = 2, \ldots, n \\ & z_1 = 0 \\ & x^{\max} \succeq x \succeq 0, \quad \mathbf{1}^T x \leq B, \end{array}$$

with variables $x \in \mathbf{R}^m$ and $z \in \mathbf{R}^n$. In the second line, we use x_{ik} to denote x_j, where j corresponds to the edge from node k to node i.

The objective and constraints in this problem are all monotone nondecreasing in z_i, so it follows that we can relax it to the convex problem

$$\begin{array}{ll} \text{minimize} & z_n \\ \text{subject to} & z_i \geq \max_{k \in \text{pred}(i)} \left(-a_{ik} x_{ik} + z_k\right), \quad i = 2, \ldots, n \\ & z_1 = 0 \\ & x^{\max} \succeq x \succeq 0, \quad \mathbf{1}^T x \leq B. \end{array}$$

Using the incidence matrix this can be written as the following LP:

$$\begin{array}{ll} \text{minimize} & z_n \\ \text{subject to} & A^T z \succeq -\,\mathbf{diag}(a) x \\ & z_1 = 0 \\ & x^{\max} \succeq x \succeq 0, \quad \mathbf{1}^T x \leq B. \end{array}$$

GP formulation. Equivalently, we can formulate the problem as a GP:

$$\begin{array}{ll} \text{minimize} & P_n \\ \text{subject to} & P_i \geq \max_{k \in \text{pred(i)}} \left(P_k y_{ik}^{-a_{ik}}\right), \quad i = 2, \ldots, n \\ & P_1 = 1 \\ & y \succeq 1 \\ & \prod_{i=1}^m y_i \leq \exp(B), \end{array}$$

over $P \in \mathbf{R}^n$ and $y \in \mathbf{R}^m$ where $y_i = \exp(x_i)$ (we use the notation y_{ik} as defined above).

The optimal probability of detection failure is $P^{\max} = 0.043$, so the smuggler will get through with probability 4.3%. Using uniform allocation of resources, we obtain $P^{\max} = 0.247$, so the smuggler gets through with probability 24.7%.

The following code was used to solve the problem.

```
interdict_alloc_data

% dynamic programming solution
cvx_begin
```

```
variables x(m) z(n)
minimize(z(n))
z(1) == 0
A'*z >= -diag(a)*x
x >= 0
x <= x_max
sum(x) <= B
cvx_end

% uniform allocation
x_unif=B/m*ones(m,1);
cvx_begin
variables z_unif(n)
minimize(z_unif(n))
z_unif(1) == 0
A'*z_unif >= -diag(a)*x_unif
cvx_end
```