EE364a Homework 6 solutions

6.2 *ℓ_1-, ℓ_2-, and ℓ_∞-norm approximation by a constant vector.* What is the solution of the norm approximation problem with one scalar variable $x \in \mathbf{R}$,

$$\text{minimize} \quad \|x\mathbf{1} - b\|,$$

for the ℓ_1-, ℓ_2-, and ℓ_∞-norms?

Solution.

(a) ℓ_2-norm: the average $\mathbf{1}^T b/m$.

(b) ℓ_1-norm: the (or a) median of the coefficients of b.

(c) ℓ_∞-norm: the midrange point $(\max b_i - \min b_i)/2$.

6.9 *Minimax rational function fitting.* Show that the following problem is quasiconvex:

$$\text{minimize} \quad \max_{i=1,\ldots,k} \left| \frac{p(t_i)}{q(t_i)} - y_i \right|$$

where

$$p(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_m t^m, \qquad q(t) = 1 + b_1 t + \cdots + b_n t^n,$$

and the domain of the objective function is defined as

$$D = \{(a, b) \in \mathbf{R}^{m+1} \times \mathbf{R}^n \mid q(t) > 0,\ \alpha \le t \le \beta\}.$$

In this problem we fit a rational function $p(t)/q(t)$ to given data, while constraining the denominator polynomial to be positive on the interval $[\alpha, \beta]$. The optimization variables are the numerator and denominator coefficients a_i, b_i. The interpolation points $t_i \in [\alpha, \beta]$, and desired function values y_i, $i = 1, \ldots, k$, are given.

Solution. Let's show the objective is quasiconvex. Its domain is convex. Since $q(t_i) > 0$ for $i = 1, \ldots, k$, we have

$$\max_{i=1,\ldots,k} |p(t_i)/q(t_i) - y_i| \le \gamma$$

if and only if

$$-\gamma q(t_i) \le p(t_i) - y_i q(t_i) \le \gamma q(t_i), \quad i = 1, \ldots, k,$$

which is a pair of linear inequalities.

7.8 *Estimation using sign measurements.* We consider the measurement setup

$$y_i = \mathbf{sign}(a_i^T x + b_i + v_i), \quad i = 1, \ldots, m,$$

where $x \in \mathbf{R}^n$ is the vector to be estimated, and $y_i \in \{-1, 1\}$ are the measurements. The vectors $a_i \in \mathbf{R}^n$ and scalars $b_i \in \mathbf{R}$ are known, and v_i are IID noises with a log-concave probability density. (You can assume that $a_i^T x + b_i + v_i = 0$ does not occur.) Show that maximum likelihood estimation of x is a convex optimization problem.

Solution. We re-order the observations so that $y_i = 1$ for $i = 1, \ldots, k$ and $y_i = 0$ for $i = k+1, \ldots, m$. The probability of this event is

$$\begin{array}{l} \prod_{i=1}^k \mathbf{prob}(a_i^T x + b_i + v_i > 0) \cdot \prod_{i=k+1}^m \mathbf{prob}(a_i^T x + b_i + v_i < 0) \\ = \prod_{i=1}^k F(-a_i^T x - b_i) \cdot \prod_{i=k+1}^m (1 - F(-a_i^T x - b_i)), \end{array}$$

where F is the cumulative distribution of the noise density. The integral of a log-concave function is log-concave, so F is log-concave, and so is $1 - F$. The log-likelihood function is

$$l(x) = \sum_{i=1}^k \log F(-a_i^T x - b_i) + \sum_{i=k+1}^m \log(1 - F(-a_i^T x - b_i)),$$

which is concave. Therefore, maximizing it is a convex problem.

A5.13 *Fitting with censored data.* In some experiments there are two kinds of measurements or data available: The usual ones, in which you get a number (say), and *censored data*, in which you don't get the specific number, but are told something about it, such as a lower bound. A classic example is a study of lifetimes of a set of subjects (say, laboratory mice). For those who have died by the end of data collection, we get the lifetime. For those who have not died by the end of data collection, we do not have the lifetime, but we do have a lower bound, *i.e.*, the length of the study. These are the censored data values.

We wish to fit a set of data points,

$$(x^{(1)}, y^{(1)}), \ldots, (x^{(K)}, y^{(K)}),$$

with $x^{(k)} \in \mathbf{R}^n$ and $y^{(k)} \in \mathbf{R}$, with a linear model of the form $y \approx c^T x$. The vector $c \in \mathbf{R}^n$ is the model parameter, which we want to choose. We will use a least-squares criterion, *i.e.*, choose c to minimize

$$J = \sum_{k=1}^K \left(y^{(k)} - c^T x^{(k)} \right)^2.$$

Here is the tricky part: some of the values of $y^{(k)}$ are censored; for these entries, we have only a (given) lower bound. We will re-order the data so that $y^{(1)}, \ldots, y^{(M)}$ are given (*i.e.*, uncensored), while $y^{(M+1)}, \ldots, y^{(K)}$ are all censored, *i.e.*, unknown, but larger than D, a given number. All the values of $x^{(k)}$ are known.

(a) Explain how to find c (the model parameter) and $y^{(M+1)}, \ldots, y^{(K)}$ (the censored data values) that minimize J.

(b) Carry out the method of part (a) on the data values in `cens_fit_data.m`. Report $\hat{c}$, the value of c found using this method.

Also find $\hat{c}_{\rm ls}$, the least-squares estimate of c obtained by simply ignoring the censored data samples, *i.e.*, the least-squares estimate based on the data

$$(x^{(1)}, y^{(1)}), \ldots, (x^{(M)}, y^{(M)}).$$

The data file contains $c_{\rm true}$, the true value of c, in the vector `c_true`. Use this to give the two relative errors

$$\frac{\|c_{\rm true} - \hat{c}\|_2}{\|c_{\rm true}\|_2}, \qquad \frac{\|c_{\rm true} - \hat{c}_{\rm ls}\|_2}{\|c_{\rm true}\|_2}.$$

Solution.

(a) The trick is to introduce dummy variables to serve as placeholders for the measurements which are censored, *i.e.*, $y^{(k)}$, $k = M+1, \ldots, K$. By introducing the dummy variables $(z^{(1)}, \ldots, z^{(K-M)})$, we get the QP

$$\begin{array}{ll} \text{minimize} & \sum_{k=1}^{M} \left(y^{(k)} - c^T x^{(k)}\right)^2 + \sum_{k=M+1}^{K} \left(z^{(k-M)} - c^T x^{(k)}\right)^2 \\ \text{subject to} & z^{(k)} \geq D, \quad k = 1, \ldots, K-M, \end{array}$$

where the variables are c and $z^{(k)}$, $k = 1, \ldots, K-M$.

(b) The following code solves the problem

```
cens_fit_data;

% Using censored data method
cvx_begin
    variables c(n) z(K-M)
    minimize(sum_square(y-X(:,1:M)'*c)+sum_square(z-X(:,M+1:K)'*c))
    subject to
        z >= D
cvx_end
c_cens = c;

% Comparison to least squares method, ignoring all censored data
cvx_begin
    variable c(n)
    minimize(sum_square(y-X(:,1:M)'*c))
cvx_end
```

```
c_ls = c;

[c_true c_cens c_ls]
cens_relerr = norm(c_cens-c_true)/norm(c_true)
ls_relerr = norm(c_ls-c_true)/norm(c_true)
```

We get the following estimates of our parameter vector:

```
[c_true c_cens c_ls] =
   -0.4326   -0.2946   -0.3476
   -1.6656   -1.7541   -1.7955
    0.1253    0.2589    0.2000
    0.2877    0.2241    0.1672
   -1.1465   -0.9917   -0.8357
    1.1909    1.3018    1.3005
    1.1892    1.4262    1.8276
   -0.0376   -0.1554   -0.5612
    0.3273    0.3785    0.3686
    0.1746    0.2261   -0.0454
   -0.1867   -0.0826   -0.1096
    0.7258    1.0427    1.5265
   -0.5883   -0.4648   -0.4980
    2.1832    2.1942    2.4164
   -0.1364   -0.3586   -0.5563
    0.1139   -0.1973   -0.3701
    1.0668    1.0194    0.9900
    0.0593   -0.1186   -0.2539
   -0.0956   -0.1211   -0.1762
   -0.8323   -0.7523   -0.4349
```

This gives a relative error of 0.1784 for $\hat{c}$, and a relative error of 0.3907 for $\hat{c}_{\mathrm{ls}}$.

A5.17 *Fitting a generalized additive regression model.* A *generalized additive model* has the form

$$f(x) = \alpha + \sum_{j=1}^{n} f_j(x_j),$$

for $x \in \mathbf{R}^n$, where $\alpha \in \mathbf{R}$ is the offset, and $f_j : \mathbf{R} \to \mathbf{R}$, with $f_j(0) = 0$. The functions f_j are called the *regressor functions*. When each f_j is linear, *i.e.*, has the form $w_j x_j$, the generalized additive model is the same as the standard (linear) regression model. Roughly speaking, a generalized additive model takes into account nonlinearities in each regressor x_j, but not nonlinear interactions among the regressors. To visualize a generalized additive model, it is common to plot each regressor function (when n is not too large).

We will restrict the functions f_j to be piecewise-affine, with given knot points $p_1 < \cdots < p_K$. This means that f_j is affine on the intervals $(-\infty, p_1]$, $[p_1, p_2]$, $\ldots$, $[p_{K-1}, p_K]$, $[p_K, \infty)$, and continuous at $p_1, \ldots, p_K$. Let C denote the total (absolute value of) change in slope across all regressor functions and all knot points. The value C is a measure of nonlinearity of the regressor functions; when $C = 0$, the generalized additive model reduces to a linear regression model.

Now suppose we observe samples or data $(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)}) \in \mathbf{R}^n \times \mathbf{R}$, and wish to fit a generalized additive model to the data. We choose the offset and the regressor functions to minimize

$$\frac{1}{N}\sum_{i=1}^{N}(y^{(i)} - f(x^{(i)})^2 + \lambda C,$$

where $\lambda > 0$ is a regularization parameter. (The first term is the mean-square error.)

(a) Explain how to solve this problem using convex optimization.

(b) Carry out the method of part (a) using the data in the file `gen_add_reg_data.m`. This file contains the data, given as an $N \times n$ matrix `X` (whose rows are $(x^{(i)})^T$), a column vector `y` (which give $y^{(i)}$), a vector `p` that gives the knot points, and the scalar `lambda`.

Give the mean-square error achieved by your generalized additive regression model. Compare the estimated and true regressor functions in a 3×3 array of plots (using the plotting code in the data file as a template), over the range $-10 \leq x_i \leq 10$. The true regressor functions (to be used only for plotting, of course) are given in the cell array `f`.

Hints.

- You can represent each regressor function f_j as a linear combination of the basis functions $b_0(u) = u$ and $b_i(u) = (u - p_k)_+ - (-p_k)_+$ for $k = 1, 2, \ldots, K$, where $(a)_+ = \max\{a, 0\}$.
- You might find the matrix $\texttt{XX} = [b_0(\texttt{X})\ b_1(\texttt{X})\ \cdots\ b_K(\texttt{X})]$ useful.

Solution. There is not much more to say beyond showing the code and the plot.

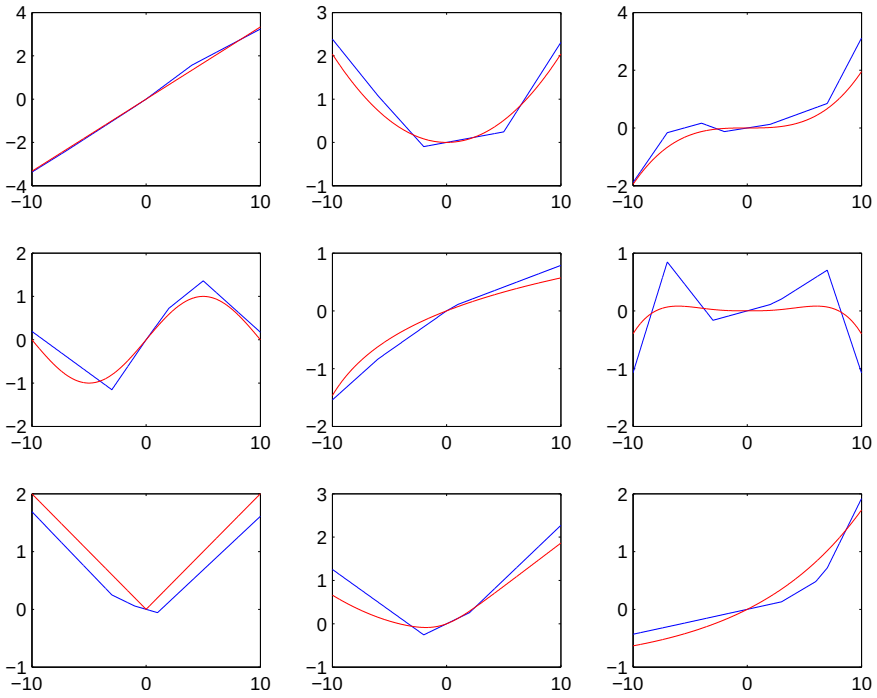
```
% Fitting a generalized additive regression model.
gen_add_reg_data;

%build an augmented data matrix XX
XX=X;
for ii=1:K
    XX=[XX,max(0,X-p(ii))+min(p(ii),0)];
end
```

```
%Perform regression
cvx_begin
    variables alpha c(9*(K+1))
    minimize(1/N*sum_square(y-alpha-XX*c)+lambda*norm(c,1))
cvx_end

%Plot functions.
xx=linspace(-10,10,1024);
yy=zeros(9,1024);
figure
for jj=1:9
    yy(jj,:)=c(jj)*xx;
    for ii=1:K
        yy(jj,:)=yy(jj,:)+c(ii*9+jj)*(pos(xx-p(ii))-pos(-p(ii)));
    end
    subplot(3,3,jj);
    plot(xx,yy(jj,:));
    hold on;
    plot(xx,f{jj}(xx),'r')
end


print -depsc gen_add_reg.eps
```



The blue and red lines correspond to the estimated and true regressors, respectively.

A11.4 *Blending existing circuit designs.* In circuit design, we must select the widths of a set of n components, given by the vector $w = (w_1, \ldots, w_n)$, which must satisfy width limits

$$W^{\min} \leq w_i \leq W^{\max}, \quad i = 1, \ldots, n,$$

where $W^{\min}$ and $W^{\max}$ are given (positive) values. (You can assume there are no other constraints on w.) The design is judged by three objectives, each of which we would like to be small: the circuit power $P(w)$, the circuit delay $D(w)$, and the total circuit area $A(w)$. These three objectives are (complicated) posynomial functions of w.

You *do not know* the functions P, D, or A. (That is, you do not know the coefficients or exponents in the posynomial expressions.) You *do know* a set of k designs, given by $w^{(1)}, \ldots, w^{(k)} \in \mathbf{R}^n$, and their associated objective values

$$P(w^{(j)}), \quad D(w^{(j)}), \quad A(w^{(j)}), \quad j = 1, \ldots, k.$$

You can assume that these designs satisfy the width limits. The goal is to find a design w that satisfies the width limits, and the design specifications

$$P(w) \leq P_{\text{spec}}, \quad D(w) \leq D_{\text{spec}}, \quad A(w) \leq A_{\text{spec}},$$

where P_{spec}, D_{spec}, and A_{spec} are given.

Now consider the specific data given in `blend_design_data.m`. Give the following.

- A feasible design (*i.e.*, w) that satisfies the specifications.
- A clear argument as to how you know that your design satisfies the specifications, even though you do not know the formulas for P, D, and A.
- Your method for finding w, including any code that you write.

Hints/comments.

- You do not need to know *anything* about circuit design to solve this problem.
- See the title of this problem.

Solution. We're given very little to go on in this problem! Clearly, the solution must rely on some basic properties of posynomials, since we are not even told what the posynomials P, D, and A are, other than their values at the designs $w^{(1)}, \ldots, w^{(k)}$.

First ideas and attempt. The title of this course, and the title of the problem, suggests that you should consider blended designs, which have the form

$$w = \theta_1 w^{(1)} + \cdots + \theta_k w^{(k)},$$

where $\theta \succeq 0$, $\mathbf{1}^T\theta = 1$. When the function f is convex, we can say that

$$f(w) \leq \theta_1 f(w^{(1)}) + \cdots + \theta_k f(w^{(k)}),$$

by Jensen's inequality. We're getting close, since we can now make a statement that our design has an f-value less than some other number that depends on the f-values at our given designs, and our blending parameters. But no cigar, since P, D, and A are posynomials, and therefore not known to be convex.

An attempt that works. A little reflection leads us in the right direction. When f is a posynomial function, the function g defined by

$$g(x) = \log f(\exp x)$$

is convex, where $\exp x$ is meant elementwise. Therefore the functions $\log P$, $\log D$, and $\log A$ are convex functions of the variables $x = \log w$. And this means, for example, that

$$\log P(\exp x) \leq \theta_1 \log P(\exp x^{(1)}) + \cdots + \theta_k \log P(\exp x^{(k)}),$$

whenever $\theta \succeq 0$, $\mathbf{1}^T\theta = 1$, and $x^{(j)} = \log w^{(j)}$. Letting $P^{(j)}$ denote $P(w^{(j)})$, and defining the blended design w as

$$\log w = \theta_1 \log w^{(1)} + \cdots + \theta_k \log w^{(k)},$$

we write the inequality above as

$$\log P(w) \leq \theta_1 \log P^{(1)} + \cdots + \theta_k \log P^{(k)}.$$

We've now got a method to form a blended design, for which we can predict an upper bound on how large the power can be. It's just linear interpolation on a log scale; equivalently, the geometric interpolation

$$w_i = (w_i^{(1)})^{\theta_1} \cdots (w_i^{(k)})^{\theta_k}.$$

Note that w will satisfy the width limits, since the original designs do.

Of course similar inequalities hold for the delay and area. We can try to find a blend that meets the specifications by solving the LP feasibility problem

$$\begin{array}{ll} \text{find} & \theta \\ \text{subject to} & \sum_{j=1}^k \theta_j \log P^{(j)} \leq \log P_{\text{spec}} \\ & \sum_{j=1}^k \theta_j \log D^{(j)} \leq \log D_{\text{spec}} \\ & \sum_{j=1}^k \theta_j \log A^{(j)} \leq \log A_{\text{spec}} \\ & \mathbf{1}^T\theta = 1, \quad \theta \succeq 0, \end{array}$$

with variable θ (the blending parameter).

It's important to understand exactly what we have done here. If the LP above is feasible, then the blend obtained from the feasible θ (and done geometrically, as described above) *must satisfy the design specifications*. We can make this statement *without knowing the specific posynomial expressions for P, D, and A*. It's nothing more than Jensen's inequality, but even so, it's pretty interesting.

On the other hand, if the LP above is infeasible, then we can say nothing. The design specifications could be infeasible, or feasible; we do not know.

The following code attempts to find a feasible blending parameter.

```
% solution for circuit blending problem
blend_design_data;
% find feasible blend factors
cvx_begin
    variable theta(k)
    log(P)*theta <= log(P_spec);
    log(D)*theta <= log(D_spec);
    log(A)*theta <= log(A_spec);
    sum(theta) == 1;
    theta >= 0;
cvx_end
% now create design
w = exp(log(W)*theta)
```

It turns out it's feasible, so we've solved the problem! So indeed there exists a design which satisfies the specifications. Our blending parameter is

$$\theta = (0.0128, 0.5096, 0.0050, 0.4626, 0.0075, 0.0026),$$

resulting in widths

$$w = (2.6203, 3.3033, 2.9562, 3.2743, 2.3037, 3.6820, 2.9177, 3.7012, 3.9140, 3.4115).$$

This solution is not unique, so you might be interested to know how we graded the numerical answers. (Of course, the main point was not to get the right answer, but to realize that you must do the blending in log space, and to clearly explain why you could be certain that your design meets the specifications.)

To grade the numerical answers, we found the bounding box (*i.e.*, smallest and largest possible value of each width) over the inequalities above. The bounds were:

$$\begin{aligned} \underline{w} &= (2.5064, 3.1177, 2.8817, 3.2293, 2.2109, 3.5823, 2.8526, 3.5293, 3.7406, 3.3164) \\ \overline{w} &= (2.8151, 3.4431, 3.0924, 3.3212, 2.4757, 3.7381, 3.0056, 3.8082, 4.0065, 3.4695). \end{aligned}$$

Any number outside these ranges is wrong. (However, numbers inside the ranges need not be correct.)

An approach that almost works. We mention another approach which is correct, but fails to solve this particular problem instance. The functions $P(\exp x)$, $D(\exp x)$, and $A(\exp x)$ are convex, where $x = \log w$. (This follows immediately from the fact that their logs are convex, and the exponential of a convex function is convex.) Using the notation above, we have that the inequality

$$P(w) \leq \theta_1 P^{(1)} + \cdots + \theta_k P^{(k)}$$

holds. It follows that if the feasibility LP

$$\begin{array}{ll} \text{find} & \theta \\ \text{subject to} & \sum_{j=1}^k \theta_j P^{(j)} \leq P_{\text{spec}} \\ & \sum_{j=1}^k \theta_j D^{(j)} \leq D_{\text{spec}} \\ & \sum_{j=1}^k \theta_j A^{(j)} \leq A_{\text{spec}} \\ & \mathbf{1}^T\theta = 1, \quad \theta \succeq 0, \end{array}$$

is feasible, then we've found a set of blending parameters that must achieve the given specs. The logic here is absolutely correct. Note that the blending parameters here are the same as above—they correspond to geometric or logarithmic blending, and not arithmetic blending. This LP above is more stringent than the first one given above; that is, it has a smaller feasible set, which means it will work in fewer cases than the one given above.

For the given example, this LP is not feasible, so for this particular problem instance, this approach won't work. (You might ask: Did the teaching staff intentionally arrange for the given problem instance to fail using this approach? Would they really do something like that?)

A15.4 *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 $\mathcal{P} \subset \{1,\ldots,m\}$ 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, $\mathbf{1}^T x \leq B$.

(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 $P^{\max} = P_n$).

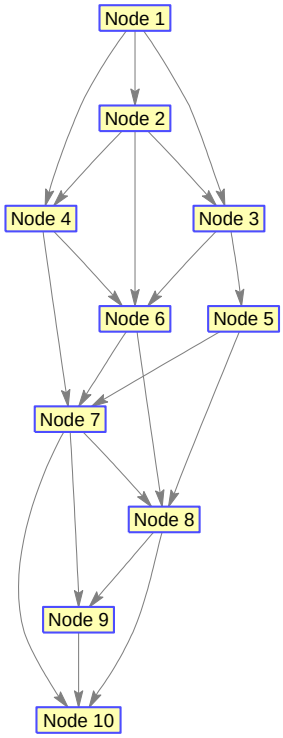
(b) Carry out your method on the problem instance given in `interdict_alloc_data.m`. The data file contains the data a, $x^{\max}$, B, and the graph incidence matrix $A \in \mathbf{R}^{n\times m}$, where

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

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:

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

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
```

A17.4 *Online advertising displays.* When a user goes to a website, one of a set of n ads, labeled $1, \ldots, n$, is displayed. This is called an *impression*. We divide some time interval (say, one day) into T periods, labeled $t = 1, \ldots, T$. Let $N_{it} \geq 0$ denote the number of impressions in period t for which we display ad i. In period t there will be a total of $I_t > 0$ impressions, so we must have $\sum_{i=1}^{n} N_{it} = I_t$, for $t = 1, \ldots, T$. (The numbers I_t might be known from past history.) You can treat all these numbers as real. (This is justified since they are typically very large.)

The revenue for displaying ad i in period t is $R_{it} \geq 0$ per impression. (This might come from click-through payments, for example.) The total revenue is $\sum_{t=1}^{T} \sum_{i=1}^{n} R_{it} N_{it}$. To maximize revenue, we would simply display the ad with the highest revenue per impression, and no other, in each display period.

We also have in place a set of m contracts that require us to display certain numbers of ads, or mixes of ads (say, associated with the products of one company), over certain periods, with a penalty for any shortfalls. Contract j is characterized by a set of ads $\mathcal{A}_j \subseteq \{1, \ldots, n\}$ (while it does not affect the math, these are often disjoint), a set of periods $\mathcal{T}_j \subseteq \{1, \ldots, T\}$, a target number of impressions $q_j \geq 0$, and a shortfall penalty rate $p_j > 0$. The *shortfall* s_j for contract j is

$$s_j = \left(q_j - \sum_{t \in \mathcal{T}_j} \sum_{i \in A_j} N_{it} \right)_+ ,$$

where $(u)_+$ means $\max\{u, 0\}$. (This is the number of impressions by which we fall short of the target value q_j.) Our contracts require a total penalty payment equal to $\sum_{j=1}^{m} p_j s_j$. Our net profit is the total revenue minus the total penalty payment.

(a) Explain how to find the display numbers N_{it} that maximize net profit. The data in this problem are $R \in \mathbf{R}^{n\times T}$, $I \in \mathbf{R}^T$ (here I is the vector of impressions, not the identity matrix), and the contract data $\mathcal{A}_j$, $\mathcal{T}_j$, q_j, and p_j, $j = 1, \ldots, m$.

(b) Carry out your method on the problem with data given in `ad_disp_data.m`. The data $\mathcal{A}_j$ and $\mathcal{T}_j$, for $j = 1, \ldots, m$ are given by matrices $A^{\text{contr}} \in \mathbf{R}^{n\times m}$ and $T^{\text{contr}} \in \mathbf{R}^{T\times m}$, with

$$A_{ij}^{\text{contr}} = \begin{cases} 1 & i \in \mathcal{A}_j \\ 0 & \text{otherwise,} \end{cases} \qquad T_{tj}^{\text{contr}} = \begin{cases} 1 & t \in \mathcal{T}_j \\ 0 & \text{otherwise.} \end{cases}$$

Report the optimal net profit, and the associated revenue and total penalty payment. Give the same three numbers for the strategy of simply displaying in each period only the ad with the largest revenue per impression.

Solution.

(a) The constraints $N_{it} \geq 0$, $\sum_{i=1}^n N_{it} = I_t$ are linear. The contract shortfalls s_j are convex functions of N, so the net profit

$$\sum_{t=1}^{T}\sum_{i=1}^{n} R_{it}N_{it} - \sum_{j=1}^{m} p_j s_j$$

is concave. So we have a convex optimization problem.

(b) We can express the vector of shortfalls as

$$s = \left(q - \mathbf{diag}\left(A^{\text{contr }T} N T^{\text{contr}}\right)\right)_+$$

The following code solves the problem:

```
clear all;
% ad display problem
ad_disp_data;

% optimal display
cvx_begin
    variable N(n,T)
    s = pos(q-diag(Acontr'*N*Tcontr));
    maximize(R(:)'*N(:)-p'*s)
    subject to
        N >= 0;
        sum(N)' == I;
cvx_end
opt_s=s;
```

```
opt_net_profit = cvx_optval
opt_penalty = p'*opt_s
opt_revenue = opt_net_profit+opt_penalty

% display, ignoring contracts
cvx_begin
    variable N(n,T)
    maximize(R(:)'*N(:))
    subject to
        N >= 0;
        sum(N)' == I;
cvx_end
greedy_s = pos(q-diag(Acontr'*N*Tcontr));
greedy_net_profit = cvx_optval-p'*greedy_s
greedy_penalty = p'*greedy_s
greedy_revenue = cvx_optval
```

The performance of the optimal and greedy strategies is given in the table below. The optimal strategy gives up a bit of (gross) revenue, in order to pay substantially less penalty, and ends up way ahead of the greedy strategy.

strategy	revenue	penalty	net profit
optimal	268.23	37.66	230.57
greedy	305.10	232.26	72.84