Additional Exercises for *Convex Optimization*

Stephen Boyd Lieven Vandenberghe

May 7, 2017

This is a collection of additional exercises, meant to supplement those found in the book *Convex Optimization*, by Stephen Boyd and Lieven Vandenberghe. These exercises were used in several courses on convex optimization, EE364a (Stanford), EE236b (UCLA), or 6.975 (MIT), usually for homework, but sometimes as exam questions. Some of the exercises were originally written for the book, but were removed at some point. Many of them include a computational component using CVX, a Matlab package for convex optimization; files required for these exercises can be found at the book web site `www.stanford.edu/~boyd/cvxbook/`. We are in the process of adapting many of these problems to be compatible with two other packages for convex optimization: CVXPY (Python) and Convex.jl (Julia). Some of the exercises require a knowledge of elementary analysis.

You are free to use these exercises any way you like (for example in a course you teach), provided you acknowledge the source. In turn, we gratefully acknowledge the teaching assistants (and in some cases, students) who have helped us develop and debug these exercises. Pablo Parrilo helped develop some of the exercises that were originally used in 6.975, and the instructors of EE364a during summer quarters helped developed others.

Course instructors can obtain solutions by email to us. Please specify the course you are teaching and give its URL.

We'll update this document as new exercises become available, so the exercise numbers and sections will occasionally change. We have categorized the exercises into sections that follow the book chapters, as well as various additional application areas. Some exercises fit into more than one section, or don't fit well into any section, so we have just arbitrarily assigned these.

Stephen Boyd and Lieven Vandenberghe

Contents

1 Convex sets

1.1 Is the set $\{a \in \mathbf{R}^k \mid p(0) = 1,\ |p(t)| \leq 1 \text{ for } \alpha \leq t \leq \beta\}$, where

$$p(t) = a_1 + a_2 t + \cdots + a_k t^{k-1},$$

convex?

1.2 *Set distributive characterization of convexity* [Rockafellar]. Show that $C \subseteq \mathbf{R}^n$ is convex if and only if $(\alpha + \beta)C = \alpha C + \beta C$ for all nonnegative α, β.

1.3 *Composition of linear-fractional functions.* Suppose $\phi : \mathbf{R}^n \to \mathbf{R}^m$ and $\psi : \mathbf{R}^m \to \mathbf{R}^p$ are the linear-fractional functions

$$\phi(x) = \frac{Ax + b}{c^T x + d}, \qquad \psi(y) = \frac{Ey + f}{g^T y + h},$$

with domains $\mathbf{dom}\,\phi = \{x \mid c^T x + d > 0\}$, $\mathbf{dom}\,\psi = \{y \mid g^T x + h > 0\}$. We associate with ϕ and ψ the matrices

$$\begin{bmatrix} A & b \\ c^T & d \end{bmatrix}, \qquad \begin{bmatrix} E & f \\ g^T & h \end{bmatrix},$$

respectively.

Now consider the composition Γ of ψ and ϕ, *i.e.*, $\Gamma(x) = \psi(\phi(x))$, with domain

$$\mathbf{dom}\,\Gamma = \{x \in \mathbf{dom}\,\phi \mid \phi(x) \in \mathbf{dom}\,\psi\}.$$

Show that Γ is linear-fractional, and that the matrix associated with it is the product

$$\begin{bmatrix} E & f \\ g^T & h \end{bmatrix} \begin{bmatrix} A & b \\ c^T & d \end{bmatrix}.$$

1.4 *Dual of exponential cone.* The exponential cone $K_{\text{exp}} \subseteq \mathbf{R}^3$ is defined as

$$K_{\text{exp}} = \{(x, y, z) \mid y > 0,\ ye^{x/y} \leq z\}.$$

Find the dual cone K_{exp}^*.

We are not worried here about the fine details of what happens on the boundaries of these cones, so you really needn't worry about it. But we make some comments here for those who do care about such things.

The cone K_{exp} as defined above is not closed. To obtain its closure, we need to add the points

$$\{(x, y, z) \mid x \leq 0,\ y = 0,\ z \geq 0\}.$$

(This makes no difference, since the dual of a cone is equal to the dual of its closure.)

1.5 *Dual of intersection of cones.* Let C and D be closed convex cones in $\mathbf{R}^n$. In this problem we will show that

$$(C \cap D)^* = C^* + D^*.$$

Here, $+$ denotes set addition: $C^* + D^*$ is the set $\{u + v \mid u \in C^*,\ v \in D^*\}$. In other words, the dual of the intersection of two closed convex cones is the sum of the dual cones.

(a) Show that $C \cap D$ and $C^* + D^*$ are convex cones. (In fact, $C \cap D$ and $C^* + D^*$ are closed, but we won't ask you to show this.)

(b) Show that $(C \cap D)^* \supseteq C^* + D^*$.

(c) Now let's show $(C \cap D)^* \subseteq C^* + D^*$. You can do this by first showing

$$(C \cap D)^* \subseteq C^* + D^* \iff C \cap D \supseteq (C^* + D^*)^*.$$

You can use the following result:

If K is a closed convex cone, then $K^{**} = K$.

Next, show that $C \cap D \supseteq (C^* + D^*)^*$ and conclude $(C \cap D)^* = C^* + D^*$.

(d) Show that the dual of the polyhedral cone $V = \{x \mid Ax \succeq 0\}$ can be expressed as

$$V^* = \{A^T v \mid v \succeq 0\}.$$

1.6 *Polar of a set.* The *polar* of $C \subseteq \mathbf{R}^n$ is defined as the set

$$C^\circ = \{y \in \mathbf{R}^n \mid y^T x \leq 1 \text{ for all } x \in C\}.$$

(a) Show that C° is convex (even if C is not).

(b) What is the polar of a cone?

(c) What is the polar of the unit ball for a norm $\|\cdot\|$?

(d) What is the polar of the set $C = \{x \mid \mathbf{1}^T x = 1, x \succeq 0\}$?

(e) Show that if C is closed and convex, with $0 \in C$, then $(C^\circ)^\circ = C$.

1.7 *Dual cones in* $\mathbf{R}^2$. Describe the dual cone for each of the following cones.

(a) $K = \{0\}$.

(b) $K = \mathbf{R}^2$.

(c) $K = \{(x_1, x_2) \mid |x_1| \leq x_2\}$.

(d) $K = \{(x_1, x_2) \mid x_1 + x_2 = 0\}$.

2 Convex functions

2.1 *Maximum of a convex function over a polyhedron.* Show that the maximum of a convex function f over the polyhedron $\mathcal{P} = \mathbf{conv}\{v_1, \ldots, v_k\}$ is achieved at one of its vertices, *i.e.*,

$$\sup_{x \in \mathcal{P}} f(x) = \max_{i=1,\ldots,k} f(v_i).$$

(A stronger statement is: the maximum of a convex function over a closed bounded convex set is achieved at an extreme point, *i.e.*, a point in the set that is not a convex combination of any other points in the set.) *Hint.* Assume the statement is false, and use Jensen's inequality.

2.2 *A general vector composition rule.* Suppose

$$f(x) = h(g_1(x), g_2(x), \ldots, g_k(x))$$

where $h : \mathbf{R}^k \to \mathbf{R}$ is convex, and $g_i : \mathbf{R}^n \to \mathbf{R}$. Suppose that for each i, one of the following holds:

- h is nondecreasing in the ith argument, and g_i is convex
- h is nonincreasing in the ith argument, and g_i is concave
- g_i is affine.

Show that f is convex. (This composition rule subsumes all the ones given in the book, and is the one used in software systems such as CVX.) You can assume that $\mathbf{dom}\, h = \mathbf{R}^k$; the result also holds in the general case when the monotonicity conditions listed above are imposed on $\tilde{h}$, the extended-valued extension of h.

2.3 *Logarithmic barrier for the second-order cone.* The function $f(x,t) = -\log(t^2 - x^T x)$, with $\mathbf{dom}\, f = \{(x,t) \in \mathbf{R}^n \times \mathbf{R} \mid t > \|x\|_2\}$ (*i.e.*, the second-order cone), is convex. (The function f is called the logarithmic barrier function for the second-order cone.) This can be shown many ways, for example by evaluating the Hessian and demonstrating that it is positive semidefinite. In this exercise you establish convexity of f using a relatively painless method, leveraging some composition rules and known convexity of a few other functions.

(a) Explain why $t - (1/t)u^T u$ is a concave function on $\mathbf{dom}\, f$. *Hint.* Use convexity of the quadratic over linear function.

(b) From this, show that $-\log(t - (1/t)u^T u)$ is a convex function on $\mathbf{dom}\, f$.

(c) From this, show that f is convex.

2.4 *A quadratic-over-linear composition theorem.* Suppose that $f : \mathbf{R}^n \to \mathbf{R}$ is nonnegative and convex, and $g : \mathbf{R}^n \to \mathbf{R}$ is positive and concave. Show that the function f^2/g, with domain $\mathbf{dom}\, f \cap \mathbf{dom}\, g$, is convex.

2.5 *A perspective composition rule* [Maréchal]. Let $f : \mathbf{R}^n \to \mathbf{R}$ be a convex function with $f(0) \leq 0$.

(a) Show that the perspective $tf(x/t)$, with domain $\{(x,t) \mid t > 0,\ x/t \in \mathbf{dom}\, f\}$, is nonincreasing as a function of t.

(b) Let g be concave and positive on its domain. Show that the function

$$h(x) = g(x)f(x/g(x)), \qquad \mathbf{dom}\, h = \{x \in \mathbf{dom}\, g \mid x/g(x) \in \mathbf{dom}\, f\}$$

is convex.

(c) As an example, show that

$$h(x) = \frac{x^T x}{(\prod_{k=1}^n x_k)^{1/n}}, \qquad \mathbf{dom}\, h = \mathbf{R}^n_{++}$$

is convex.

2.6 *Perspective of log determinant.* Show that $f(X,t) = nt\log t - t\log\det X$, with $\mathbf{dom}\, f = \mathbf{S}^n_{++} \times \mathbf{R}_{++}$, is convex in (X,t). Use this to show that

$$\begin{aligned} g(X) &= n(\mathbf{tr}\, X)\log(\mathbf{tr}\, X) - (\mathbf{tr}\, X)(\log\det X) \\ &= n\left(\sum_{i=1}^n \lambda_i\right)\left(\log\sum_{i=1}^n \lambda_i - \sum_{i=1}^n \log\lambda_i\right), \end{aligned}$$

where λ_i are the eigenvalues of X, is convex on $\mathbf{S}^n_{++}$.

2.7 *Pre-composition with a linear fractional mapping.* Suppose $f : \mathbf{R}^m \to \mathbf{R}$ is convex, and $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, $c \in \mathbf{R}^n$, and $d \in \mathbf{R}$. Show that $g : \mathbf{R}^n \to \mathbf{R}$, defined by

$$g(x) = (c^T x + d)f((Ax+b)/(c^T x + d)), \qquad \mathbf{dom}\, g = \{x \mid c^T x + d > 0\},$$

is convex.

2.8 *Scalar valued linear fractional functions.* A function $f : \mathbf{R}^n \to \mathbf{R}$ is called *linear fractional* if it has the form $f(x) = (a^T x + b)/(c^T x + d)$, with $\mathbf{dom}\, f = \{x \mid c^T x + d > 0\}$. When is a linear fractional function convex? When is a linear fractional function quasiconvex?

2.9 Show that the function

$$f(x) = \frac{\|Ax - b\|_2^2}{1 - x^T x}$$

is convex on $\{x \mid \|x\|_2 < 1\}$.

2.10 *Weighted geometric mean.* The geometric mean $f(x) = (\prod_k x_k)^{1/n}$ with $\mathbf{dom}\, f = \mathbf{R}^n_{++}$ is concave, as shown on page 74. Extend the proof to show that

$$f(x) = \prod_{k=1}^n x_k^{\alpha_k}, \qquad \mathbf{dom}\, f = \mathbf{R}^n_{++}$$

is concave, where α_k are nonnegative numbers with $\sum_k \alpha_k \le 1$.

2.11 Suppose that $f : \mathbf{R}^n \to \mathbf{R}$ is convex, and define

$$g(x,t) = f(x/t), \quad \mathbf{dom}\, g = \{(x,t) \mid x/t \in \mathbf{dom}\, f,\ t > 0\}.$$

Show that g is quasiconvex.

2.12 *Continued fraction function.* Show that the function

$$f(x) = \cfrac{1}{x_1 - \cfrac{1}{x_2 - \cfrac{1}{x_3 - \cfrac{1}{x_4}}}}$$

defined where every denominator is positive, is convex and decreasing. (There is nothing special about $n = 4$ here; the same holds for any number of variables.)

2.13 *Circularly symmetric Huber function.* The scalar Huber function is defined as

$$f_{\rm hub}(x) = \begin{cases} (1/2)x^2 & |x| \leq 1 \\ |x| - 1/2 & |x| > 1. \end{cases}$$

This convex function comes up in several applications, including robust estimation. This problem concerns generalizations of the Huber function to $\mathbf{R}^n$. One generalization to $\mathbf{R}^n$ is given by $f_{\rm hub}(x_1) + \cdots + f_{\rm hub}(x_n)$, but this function is not circularly symmetric, *i.e.*, invariant under transformation of x by an orthogonal matrix. A generalization to $\mathbf{R}^n$ that *is* circularly symmetric is

$$f_{\rm cshub}(x) = f_{\rm hub}(\|x\|) = \begin{cases} (1/2)\|x\|_2^2 & \|x\|_2 \leq 1 \\ \|x\|_2 - 1/2 & \|x\|_2 > 1. \end{cases}$$

(The subscript stands for 'circularly symmetric Huber function'.) Show that $f_{\rm cshub}$ is convex. Find the conjugate function $f^*_{\rm cshub}$.

2.14 *Reverse Jensen inequality.* Suppose f is convex, $\lambda_1 > 0$, $\lambda_i \leq 0$, $i = 2, \ldots, k$, and $\lambda_1 + \cdots + \lambda_n = 1$, and let $x_1, \ldots, x_n \in \mathbf{dom}\, f$. Show that the inequality

$$f(\lambda_1 x_1 + \cdots + \lambda_n x_n) \geq \lambda_1 f(x_1) + \cdots + \lambda_n f(x_n)$$

always holds. *Hints.* Draw a picture for the $n = 2$ case first. For the general case, express x_1 as a convex combination of $\lambda_1 x_1 + \cdots + \lambda_n x_n$ and $x_2, \ldots, x_n$, and use Jensen's inequality.

2.15 *Monotone extension of a convex function.* Suppose $f : \mathbf{R}^n \to \mathbf{R}$ is convex. Recall that a function $h : \mathbf{R}^n \to \mathbf{R}$ is monotone nondecreasing if $h(x) \geq h(y)$ whenever $x \succeq y$. The *monotone extension* of f is defined as

$$g(x) = \inf_{z \succeq 0} f(x + z).$$

(We will assume that $g(x) > -\infty$.) Show that g is convex and monotone nondecreasing, and satisfies $g(x) \leq f(x)$ for all x. Show that if h is any other convex function that satisfies these properties, then $h(x) \leq g(x)$ for all x. Thus, g is the maximum convex monotone underestimator of f.

Remark. For simple functions (say, on $\mathbf{R}$) it is easy to work out what g is, given f. On $\mathbf{R}^n$, it can be very difficult to work out an explicit expression for g. However, systems such as CVX can immediately handle functions such as g, defined by partial minimization.

2.16 *Circularly symmetric convex functions.* Suppose $f : \mathbf{R}^n \to \mathbf{R}$ is convex and symmetric with respect to rotations, *i.e.*, $f(x)$ depends only on $\|x\|_2$. Show that f must have the form $f(x) = \phi(\|x\|_2)$, where $\phi : \mathbf{R} \to \mathbf{R}$ is nondecreasing and convex, with $\mathbf{dom}\, f = \mathbf{R}$. (Conversely, any function of this form is symmetric and convex, so this form characterizes such functions.)

2.17 *Infimal convolution.* Let $f_1, \ldots, f_m$ be convex functions on $\mathbf{R}^n$. Their *infimal convolution*, denoted $g = f_1 \diamond \cdots \diamond f_m$ (several other notations are also used), is defined as

$$g(x) = \inf\{f_1(x_1) + \cdots + f_m(x_m) \mid x_1 + \cdots + x_m = x\},$$

with the natural domain (*i.e.*, defined by $g(x) < \infty$). In one simple interpretation, $f_i(x_i)$ is the cost for the ith firm to produce a mix of products given by x_i; $g(x)$ is then the optimal cost obtained if the firms can freely exchange products to produce, all together, the mix given by x. (The name 'convolution' presumably comes from the observation that if we replace the sum above with the product, and the infimum above with integration, then we obtain the normal convolution.)

(a) Show that g is convex.

(b) Show that $g^* = f_1^* + \cdots + f_m^*$. In other words, the conjugate of the infimal convolution is the sum of the conjugates.

2.18 *Conjugate of composition of convex and linear function.* Suppose $A \in \mathbf{R}^{m\times n}$ with $\mathbf{rank}\, A = m$, and g is defined as $g(x) = f(Ax)$, where $f : \mathbf{R}^m \to \mathbf{R}$ is convex. Show that

$$g^*(y) = f^*((A^\dagger)^T y), \qquad \mathbf{dom}(g^*) = A^T \mathbf{dom}(f^*),$$

where $A^\dagger = (AA^T)^{-1}A$ is the pseudo-inverse of A. (This generalizes the formula given on page 95 for the case when A is square and invertible.)

2.19 [Roberts and Varberg] Suppose $\lambda_1, \ldots, \lambda_n$ are positive. Show that the function $f : \mathbf{R}^n \to \mathbf{R}$, given by

$$f(x) = \prod_{i=1}^n (1 - e^{-x_i})^{\lambda_i},$$

is concave on

$$\mathbf{dom}\, f = \left\{ x \in \mathbf{R}^n_{++} \;\middle|\; \sum_{i=1}^n \lambda_i e^{-x_i} \leq 1 \right\}.$$

Hint. The Hessian is given by

$$\nabla^2 f(x) = f(x)(yy^T - \mathbf{diag}(z))$$

where $y_i = \lambda_i e^{-x_i}/(1 - e^{-x_i})$ and $z_i = y_i/(1 - e^{-x_i})$.

2.20 Show that the following functions $f : \mathbf{R}^n \to \mathbf{R}$ are convex.

(a) The difference between the maximum and minimum value of a polynomial on a given interval, as a function of its coefficients:

$$f(x) = \sup_{t\in[a,b]} p(t) - \inf_{t\in[a,b]} p(t) \quad \text{where} \quad p(t) = x_1 + x_2 t + x_3 t^2 + \cdots + x_n t^{n-1}.$$

a, b are real constants with $a < b$.

(b) The 'exponential barrier' of a set of inequalities:

$$f(x) = \sum_{i=1}^{m} e^{-1/f_i(x)}, \qquad \mathbf{dom}\, f = \{x \mid f_i(x) < 0, i = 1, \ldots, m\}.$$

The functions f_i are convex.

(c) The function

$$f(x) = \inf_{\alpha > 0} \frac{g(y + \alpha x) - g(y)}{\alpha}$$

if g is convex and $y \in \mathbf{dom}\, g$. (It can be shown that this is the directional derivative of g at y in the direction x.)

2.21 *Symmetric convex functions of eigenvalues.* A function $f : \mathbf{R}^n \to \mathbf{R}$ is said to be *symmetric* if it is invariant with respect to a permutation of its arguments, *i.e.*, $f(x) = f(Px)$ for any permutation matrix P. An example of a symmetric function is $f(x) = \log(\sum_{k=1}^{n} \exp x_k)$.

In this problem we show that if $f : \mathbf{R}^n \to \mathbf{R}$ is *convex* and *symmetric*, then the function $g : \mathbf{S}^n \to \mathbf{R}$ defined as $g(X) = f(\lambda(X))$ is convex, where $\lambda(X) = (\lambda_1(X),\ \lambda_2(x),\ \ldots,\ \lambda_n(X))$ is the vector of eigenvalues of X. This implies, for example, that the function

$$g(X) = \log \mathbf{tr}\, e^X = \log \sum_{k=1}^{n} e^{\lambda_k(X)}$$

is convex on $\mathbf{S}^n$.

(a) A square matrix S is *doubly stochastic* if its elements are nonnegative and all row sums and column sums are equal to one. It can be shown that every doubly stochastic matrix is a convex combination of permutation matrices.

Show that if f is convex and symmetric and S is doubly stochastic, then

$$f(Sx) \leq f(x).$$

(b) Let $Y = Q\, \mathbf{diag}(\lambda) Q^T$ be an eigenvalue decomposition of $Y \in \mathbf{S}^n$ with Q orthogonal. Show that the $n \times n$ matrix S with elements $S_{ij} = Q_{ij}^2$ is doubly stochastic and that $\mathbf{diag}(Y) = S\lambda$.

(c) Use the results in parts (a) and (b) to show that if f is convex and symmetric and $X \in \mathbf{S}^n$, then

$$f(\lambda(X)) = \sup_{V \in \mathcal{V}} f(\mathbf{diag}(V^T X V))$$

where $\mathcal{V}$ is the set of $n \times n$ orthogonal matrices. Show that this implies that $f(\lambda(X))$ is convex in X.

2.22 *Convexity of nonsymmetric matrix fractional function.* Consider the function $f : \mathbf{R}^{n \times n} \times \mathbf{R}^n \to \mathbf{R}$, defined by

$$f(X, y) = y^T X^{-1} y, \qquad \mathbf{dom}\, f = \{(X, y) \mid X + X^T \succ 0\}.$$

When this function is restricted to $X \in \mathbf{S}^n$, it is convex.

Is f convex? If so, prove it. If not, give a (simple) counterexample.

2.23 Show that the following functions $f : \mathbf{R}^n \to \mathbf{R}$ are convex.

(a) $f(x) = -\exp(-g(x))$ where $g : \mathbf{R}^n \to \mathbf{R}$ has a convex domain and satisfies

$$\begin{bmatrix} \nabla^2 g(x) & \nabla g(x) \\ \nabla g(x)^T & 1 \end{bmatrix} \succeq 0$$

for $x \in \mathbf{dom}\, g$.

(b) The function

$$f(x) = \max\{\|APx - b\| \mid P \text{ is a permutation matrix}\}$$

with $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$.

2.24 *Convex hull of functions.* Suppose g and h are convex functions, bounded below, with $\mathbf{dom}\, g = \mathbf{dom}\, h = \mathbf{R}^n$. The convex hull function of g and h is defined as

$$f(x) = \inf\{\theta g(y) + (1-\theta)h(z) \mid \theta y + (1-\theta)z = x,\ 0 \le \theta \le 1\},$$

where the infimum is over θ, y, z. Show that the convex hull of h and g is convex. Describe $\mathbf{epi}\, f$ in terms of $\mathbf{epi}\, g$ and $\mathbf{epi}\, h$.

2.25 Show that a function $f : \mathbf{R} \to \mathbf{R}$ is convex if and only if $\mathbf{dom}\, f$ is convex and

$$\det \begin{bmatrix} 1 & 1 & 1 \\ x & y & z \\ f(x) & f(y) & f(z) \end{bmatrix} \ge 0$$

for all x, y, $z \in \mathbf{dom}\, f$ with $x < y < z$.

2.26 *Generalization of the convexity of* $\log\det X^{-1}$. Let $P \in \mathbf{R}^{n\times m}$ have rank m. In this problem we show that the function $f : \mathbf{S}^n \to \mathbf{R}$, with $\mathbf{dom}\, f = \mathbf{S}^n_{++}$, and

$$f(X) = \log\det(P^T X^{-1} P)$$

is convex. To prove this, we assume (without loss of generality) that P has the form

$$P = \begin{bmatrix} I \\ 0 \end{bmatrix},$$

where I. The matrix $P^T X^{-1} P$ is then the leading $m \times m$ principal submatrix of X^{-1}.

(a) Let Y and Z be symmetric matrices with $0 \prec Y \preceq Z$. Show that $\det Y \le \det Z$.

(b) Let $X \in \mathbf{S}^n_{++}$, partitioned as

$$X = \begin{bmatrix} X_{11} & X_{12} \\ X_{12}^T & X_{22} \end{bmatrix},$$

with $X_{11} \in \mathbf{S}^m$. Show that the optimization problem

$$\begin{array}{ll} \text{minimize} & \log\det Y^{-1} \\ \text{subject to} & \begin{bmatrix} Y & 0 \\ 0 & 0 \end{bmatrix} \preceq \begin{bmatrix} X_{11} & X_{12} \\ X_{12}^T & X_{22} \end{bmatrix}, \end{array}$$

with variable $Y \in \mathbf{S}^m$, has the solution

$$Y = X_{11} - X_{12}X_{22}^{-1}X_{12}^T.$$

(As usual, we take $\mathbf{S}^m_{++}$ as the domain of $\log\det Y^{-1}$.)

Hint. Use the Schur complement characterization of positive definite block matrices (page 651 of the book): if $C \succ 0$ then

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \succeq 0$$

if and only if $A - BC^{-1}B^T \succeq 0$.

(c) Combine the result in part (b) and the minimization property (page 3-19, lecture notes) to show that the function

$$f(X) = \log\det(X_{11} - X_{12}X_{22}^{-1}X_{12}^T)^{-1},$$

with $\mathbf{dom}\, f = \mathbf{S}^n_{++}$, is convex.

(d) Show that $(X_{11} - X_{12}X_{22}^{-1}X_{12}^T)^{-1}$ is the leading $m \times m$ principal submatrix of X^{-1}, *i.e.*,

$$(X_{11} - X_{12}X_{22}^{-1}X_{12}^T)^{-1} = P^T X^{-1} P.$$

Hence, the convex function f defined in part (c) can also be expressed as $f(X) = \log\det(P^T X^{-1} P)$.

Hint. Use the formula for the inverse of a symmetric block matrix:

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 0 \\ 0 & C^{-1} \end{bmatrix} + \begin{bmatrix} -I \\ C^{-1}B^T \end{bmatrix} (A - BC^{-1}B^T)^{-1} \begin{bmatrix} -I \\ C^{-1}B^T \end{bmatrix}^T$$

if C and $A - BC^{-1}B^T$ are invertible.

2.27 *Functions of a random variable with log-concave density.* Suppose the random variable X on $\mathbf{R}^n$ has log-concave density, and let $Y = g(X)$, where $g : \mathbf{R}^n \to \mathbf{R}$. For each of the following statements, either give a counterexample, or show that the statement is true.

(a) If g is affine and not constant, then Y has log-concave density.

(b) If g is convex, then $\mathbf{prob}(Y \leq a)$ is a log-concave function of a.

(c) If g is concave, then $\mathbf{E}\,((Y-a)_+)$ is a convex and log-concave function of a. (This quantity is called the tail expectation of Y; you can assume it exists. We define $(s)_+$ as $(s)_+ = \max\{s, 0\}$.)

2.28 *Majorization.* Define C as the set of all permutations of a given n-vector a, *i.e.*, the set of vectors $(a_{\pi_1}, a_{\pi_2}, \ldots, a_{\pi_n})$ where $(\pi_1, \pi_2, \ldots, \pi_n)$ is one of the $n!$ permutations of $(1, 2, \ldots, n)$.

(a) The support function of C is defined as $S_C(y) = \max_{x \in C} y^T x$. Show that

$$S_C(y) = a_{[1]}y_{[1]} + a_{[2]}y_{[2]} + \cdots + a_{[n]}y_{[n]}.$$

($u_{[1]}$, $u_{[2]}$, …, $u_{[n]}$ denote the components of an n-vector u in nonincreasing order.)

Hint. To find the maximum of $y^T x$ over $x \in C$, write the inner product as

$$\begin{aligned} y^T x &= (y_1 - y_2)x_1 + (y_2 - y_3)(x_1 + x_2) + (y_3 - y_4)(x_1 + x_2 + x_3) + \cdots \\ &\quad + (y_{n-1} - y_n)(x_1 + x_2 + \cdots + x_{n-1}) + y_n(x_1 + x_2 + \cdots + x_n) \end{aligned}$$

and assume that the components of y are sorted in nonincreasing order.

(b) Show that x satisfies $x^T y \leq S_C(y)$ for all y if and only if

$$s_k(x) \leq s_k(a), \quad k = 1, \ldots, n-1, \qquad s_n(x) = s_n(a),$$

where s_k denotes the function $s_k(x) = x_{[1]} + x_{[2]} + \cdots + x_{[k]}$. When these inequalities hold, we say the vector a *majorizes* the vector x.

(c) Conclude from this that the conjugate of S_C is given by

$$S_C^*(x) = \begin{cases} 0 & \text{if } x \text{ is majorized by } a \\ +\infty & \text{otherwise.} \end{cases}$$

Since S_C^* is the indicator function of the convex hull of C, this establishes the following result: x is a convex combination of the permutations of a if and only if a majorizes x.

2.29 *Convexity of products of powers.* This problem concerns the product of powers function $f : \mathbf{R}^n_{++} \to \mathbf{R}$ given by $f(x) = x_1^{\theta_1} \cdots x_n^{\theta_n}$, where $\theta \in \mathbf{R}^n$ is a vector of powers. We are interested in finding values of θ for which f is convex or concave. You already know a few, for example when $n = 2$ and $\theta = (2, -1)$, f is convex (the quadratic-over-linear function), and when $\theta = (1/n)\mathbf{1}$, f is concave (geometric mean). Of course, if $n = 1$, f is convex when $\theta \geq 1$ or $\theta \leq 0$, and concave when $0 \leq \theta \leq 1$.

Show each of the statements below. We will not read long or complicated proofs, or ones that involve Hessians. We are looking for short, snappy ones, that (where possible) use composition rules, perspective, partial minimization, or other operations, together with known convex or concave functions, such as the ones listed in the previous paragraph. Feel free to use the results of earlier statements in later ones.

(a) When $n = 2$, $\theta \succeq 0$, and $\mathbf{1}^T\theta = 1$, f is concave.

(b) When $\theta \succeq 0$ and $\mathbf{1}^T\theta = 1$, f is concave. (This is the same as part (a), but here it is for general n.)

(c) When $\theta \succeq 0$ and $\mathbf{1}^T\theta \leq 1$, f is concave.

(d) When $\theta \preceq 0$, f is convex.

(e) When $\mathbf{1}^T\theta = 1$ and exactly *one* of the elements of θ is positive, f is convex.

(f) When $\mathbf{1}^T\theta \geq 1$ and exactly *one* of the elements of θ is positive, f is convex.

Remark. Parts (c), (d), and (f) exactly characterize the cases when f is either convex or concave. That is, if none of these conditions on θ hold, f is neither convex nor concave. Your teaching staff has, however, kindly refrained from asking you to show this.

2.30 *Huber penalty.* The infimal convolution of two functions f and g on $\mathbf{R}^n$ is defined as

$$h(x) = \inf_y \left(f(y) + g(x - y)\right)$$

(see exercise 2.17). Show that the infimal convolution of $f(x) = \|x\|_1$ and $g(x) = (1/2)\|x\|_2^2$, *i.e.*, the function

$$h(x) = \inf_y \left(f(y) + g(x - y)\right) = \inf_y \left(\|y\|_1 + \frac{1}{2}\|x - y\|_2^2\right),$$

is the *Huber penalty*

$$h(x) = \sum_{i=1}^{n} \phi(x_i), \qquad \phi(u) = \begin{cases} u^2/2 & |u| \leq 1 \\ |u| - 1/2 & |u| > 1. \end{cases}$$

2.31 Suppose the function $h : \mathbf{R} \to \mathbf{R}$ is convex, nondecreasing, with $\mathbf{dom}\, h = \mathbf{R}$, and $h(t) = h(0)$ for $t \leq 0$.

(a) Show that the function $f(x) = h(\|x\|_2)$ is convex on $\mathbf{R}^n$.

(b) Show that the conjugate of f is $f^*(y) = h^*(\|y\|_2)$.

(c) As an example, derive the conjugate of $f(x) = (1/p)\|x\|_2^p$ for $p > 1$, by applying the result of part (b) with the function

$$h(t) = \frac{1}{p} \max\{0, t\}^p = \begin{cases} \frac{1}{p} t^p & t \geq 0 \\ 0 & t < 0. \end{cases}$$

3 Convex optimization problems

3.1 *Minimizing a function over the probability simplex.* Find simple necessary and sufficient conditions for $x \in \mathbf{R}^n$ to minimize a differentiable convex function f over the probability simplex, $\{x \mid \mathbf{1}^T x = 1,\ x \succeq 0\}$.

3.2 *'Hello World' in CVX*.* Use CVX, CVXPY, or Convex.jl to verify the optimal values you obtained (analytically) for exercise 4.1 in *Convex Optimization*.

3.3 *Reformulating constraints in CVX*.* Each of the following CVX code fragments describes a convex constraint on the scalar variables `x`, `y`, and `z`, but violates the CVX rule set, and so is invalid. Briefly explain why each fragment is invalid. Then, rewrite each one in an equivalent form that conforms to the CVX rule set. In your reformulations, you can use linear equality and inequality constraints, and inequalities constructed using CVX functions. You can also introduce additional variables, or use LMIs. Be sure to explain (briefly) why your reformulation is equivalent to the original constraint, if it is not obvious.

Check your reformulations by creating a small problem that includes these constraints, and solving it using CVX. Your test problem doesn't have to be feasible; it's enough to verify that CVX processes your constraints without error.

Remark. This *looks* like a problem about 'how to use CVX software', or 'tricks for using CVX'. But it really checks whether you understand the various composition rules, convex analysis, and constraint reformulation rules.

(a) `norm([x + 2*y, x - y]) == 0`

(b) `square(square(x + y)) <= x - y`

(c) `1/x + 1/y <= 1; x >= 0; y >= 0`

(d) `norm([max(x,1), max(y,2)]) <= 3*x + y`

(e) `x*y >= 1; x >= 0; y >= 0`

(f) `(x + y)^2/sqrt(y) <= x - y + 5`

(g) `x^3 + y^3 <= 1; x >= 0; y >= 0`

(h) `x + z <= 1 + sqrt(x*y - z^2); x >= 0; y >= 0`

3.4 *Optimal activity levels.* Solve the optimal activity level problem described in exercise 4.17 in *Convex Optimization*, for the instance with problem data

$$A = \begin{bmatrix} 1 & 2 & 0 & 1 \\ 0 & 0 & 3 & 1 \\ 0 & 3 & 1 & 1 \\ 2 & 1 & 2 & 5 \\ 1 & 0 & 3 & 2 \end{bmatrix}, \qquad c^{\max} = \begin{bmatrix} 100 \\ 100 \\ 100 \\ 100 \\ 100 \end{bmatrix}, \qquad p = \begin{bmatrix} 3 \\ 2 \\ 7 \\ 6 \end{bmatrix}, \qquad p^{\text{disc}} = \begin{bmatrix} 2 \\ 1 \\ 4 \\ 2 \end{bmatrix}, \qquad q = \begin{bmatrix} 4 \\ 10 \\ 5 \\ 10 \end{bmatrix}.$$

You can do this by forming the LP you found in your solution of exercise 4.17, or more directly, using CVX*. Give the optimal activity levels, the revenue generated by each one, and the total revenue generated by the optimal solution. Also, give the average price per unit for each activity level, *i.e.*, the ratio of the revenue associated with an activity, to the activity level. (These numbers

should be between the basic and discounted prices for each activity.) Give a *very brief* story explaining, or at least commenting on, the solution you find.

3.5 *Minimizing the ratio of convex and concave piecewise-linear functions.* We consider the problem

$$\begin{array}{ll} \text{minimize} & \dfrac{\max_{i=1,\ldots,m}(a_i^T x + b_i)}{\min_{i=1,\ldots,p}(c_i^T x + d_i)} \\ \text{subject to} & Fx \preceq g, \end{array}$$

with variable $x \in \mathbf{R}^n$. We assume that $c_i^T x + d_i > 0$ and $\max_{i=1,\ldots,m}(a_i^T x + b_i) \geq 0$ for all x satisfying $Fx \preceq g$, and that the feasible set is nonempty and bounded. This problem is quasiconvex, and can be solved using bisection, with each iteration involving a feasibility LP. Show how the problem can be solved by solving *one* LP, using a trick similar to one described in §4.3.2.

3.6 *Two problems involving two norms.* We consider the problem

$$\text{minimize} \quad \frac{\|Ax - b\|_1}{1 - \|x\|_\infty}, \tag{1}$$

and the very closely related problem

$$\text{minimize} \quad \frac{\|Ax - b\|_1^2}{1 - \|x\|_\infty}. \tag{2}$$

In both problems, the variable is $x \in \mathbf{R}^n$, and the data are $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^m$. Note that the only difference between problem (1) and (2) is the square in the numerator. In both problems, the constraint $\|x\|_\infty < 1$ is implicit. You can assume that $b \notin \mathcal{R}(A)$, in which case the constraint $\|x\|_\infty < 1$ can be replaced with $\|x\|_\infty \leq 1$.

Answer the following two questions, for each of the two problems. (So you will answer four questions all together.)

(a) Is the problem, exactly as stated (and for all problem data), convex? If not, is it quasiconvex? Justify your answer.

(b) Explain how to solve the problem. Your method can involve an SDP solver, an SOCP solver, an LP solver, or any combination. You can include a one-parameter bisection, if necessary. (For example, you can solve the problem by bisection on a parameter, where each iteration consists of solving an SOCP feasibility problem.)

Give the best method you can. In judging best, we use the following rules:

- *Bisection methods are worse than 'one-shot' methods.* Any method that solves the problem above by solving *one* LP, SOCP, or SDP problem is better than any method that uses a one-parameter bisection. In other words, use a bisection method only if you cannot find a 'one-shot' method.
- *Use the simplest solver needed to solve the problem.* We consider an LP solver to be simpler than an SOCP solver, which is considered simpler than an SDP solver. Thus, a method that uses an LP solver is better than a method that uses an SOCP solver, which in turn is better than a method that uses an SDP solver.

3.7 *The illumination problem.* In lecture 1 we encountered the function

$$f(p) = \max_{i=1,\ldots,n} |\log a_i^T p - \log I_{\rm des}|$$

where $a_i \in \mathbf{R}^m$, and $I_{\rm des} > 0$ are given, and $p \in \mathbf{R}^m_+$.

(a) Show that $\exp f$ is convex on $\{p \mid a_i^T p > 0,\ i = 1, \ldots, n\ \}$.

(b) Show that the constraint 'no more than half of the total power is in any 10 lamps' is convex (*i.e.*, the set of vectors p that satisfy the constraint is convex).

(c) Show that the constraint 'no more than half of the lamps are on' is (in general) *not* convex.

3.8 *Schur complements and LMI representation.* Recognizing Schur complements (see §A5.5) often helps to represent nonlinear convex constraints as linear matrix inequalities (LMIs). Consider the function

$$f(x) = (Ax + b)^T (P_0 + x_1 P_1 + \cdots + x_n P_n)^{-1} (Ax + b)$$

where $A \in \mathbf{R}^{m\times n}$, $b \in \mathbf{R}^m$, and $P_i = P_i^T \in \mathbf{R}^{m\times m}$, with domain

$$\mathbf{dom}\, f = \{x \in \mathbf{R}^n \mid P_0 + x_1 P_1 + \cdots + x_n P_n \succ 0\}.$$

This is the composition of the matrix fractional function and an affine mapping, and so is convex. Give an LMI representation of $\mathbf{epi}\, f$. That is, find a symmetric matrix $F(x,t)$, affine in (x,t), for which

$$x \in \mathbf{dom}\, f, \quad f(x) \le t \qquad \Longleftrightarrow \qquad F(x,t) \succeq 0.$$

Remark. LMI representations, such as the one you found in this exercise, can be directly used in software systems such as CVX.

3.9 *Complex least-norm problem.* We consider the complex least ℓ_p-norm problem

$$\begin{array}{ll} \text{minimize} & \|x\|_p \\ \text{subject to} & Ax = b, \end{array}$$

where $A \in \mathbf{C}^{m\times n}$, $b \in \mathbf{C}^m$, and the variable is $x \in \mathbf{C}^n$. Here $\|\cdot\|_p$ denotes the ℓ_p-norm on $\mathbf{C}^n$, defined as

$$\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$$

for $p \ge 1$, and $\|x\|_\infty = \max_{i=1,\ldots,n} |x_i|$. We assume A is full rank, and $m < n$.

(a) Formulate the complex least ℓ_2-norm problem as a least ℓ_2-norm problem with real problem data and variable. *Hint.* Use $z = (\Re x, \Im x) \in \mathbf{R}^{2n}$ as the variable.

(b) Formulate the complex least ℓ_∞-norm problem as an SOCP.

(c) Solve a random instance of both problems with $m = 30$ and $n = 100$. To generate the matrix A, you can use the Matlab command `A = randn(m,n) + i*randn(m,n)`. Similarly, use `b = randn(m,1) + i*randn(m,1)` to generate the vector b. Use the Matlab command `scatter` to plot the optimal solutions of the two problems on the complex plane, and comment (briefly) on what you observe. You can solve the problems using the CVX functions `norm(x,2)` and `norm(x,inf)`, which are overloaded to handle complex arguments. To utilize this feature, you will need to declare variables to be `complex` in the `variable` statement. (In particular, you do not have to manually form or solve the SOCP from part (b).)

3.10 *Linear programming with random cost vector.* We consider the linear program

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b. \end{array}$$

Here, however, the cost vector c is random, normally distributed with mean $\mathbf{E}\, c = c_0$ and covariance $\mathbf{E}(c - c_0)(c - c_0)^T = \Sigma$. ($A$, b, and x are deterministic.) Thus, for a given $x \in \mathbf{R}^n$, the cost $c^T x$ is a (scalar) Gaussian variable.

We can attach several different meanings to the goal 'minimize $c^T x$'; we explore some of these below.

(a) How would you minimize the expected cost $\mathbf{E}\, c^T x$ subject to $Ax \preceq b$?

(b) In general there is a tradeoff between small expected cost and small cost variance. One way to take variance into account is to minimize a linear combination

$$\mathbf{E}\, c^T x + \gamma\, \mathbf{var}(c^T x) \tag{3}$$

of the expected value $\mathbf{E}\, c^T x$ and the variance $\mathbf{var}(c^T x) = \mathbf{E}(c^T x)^2 - (\mathbf{E}\, c^T x)^2$. This is called the 'risk-sensitive cost', and the parameter $\gamma \geq 0$ is called the *risk-aversion parameter*, since it sets the relative values of cost variance and expected value. (For $\gamma > 0$, we are willing to tradeoff an increase in expected cost for a decrease in cost variance). How would you minimize the risk-sensitive cost? Is this problem a convex optimization problem? Be as specific as you can.

(c) We can also minimize the risk-sensitive cost, but with $\gamma < 0$. This is called 'risk-seeking'. Is this problem a convex optimization problem?

(d) Another way to deal with the randomness in the cost $c^T x$ is to formulate the problem as

$$\begin{array}{ll} \text{minimize} & \beta \\ \text{subject to} & \mathbf{prob}(c^T x \geq \beta) \leq \alpha \\ & Ax \preceq b. \end{array}$$

Here, α is a fixed parameter, which corresponds roughly to the reliability we require, and might typically have a value of 0.01. Is this problem a convex optimization problem? Be as specific as you can. Can you obtain risk-seeking by choice of α? Explain.

3.11 Formulate the following optimization problems as semidefinite programs. The variable is $x \in \mathbf{R}^n$; $F(x)$ is defined as

$$F(x) = F_0 + x_1 F_1 + x_2 F_2 + \cdots + x_n F_n$$

with $F_i \in \mathbf{S}^m$. The domain of f in each subproblem is $\mathbf{dom}\, f = \{x \in \mathbf{R}^n \mid F(x) \succ 0\}$.

(a) Minimize $f(x) = c^T F(x)^{-1} c$ where $c \in \mathbf{R}^m$.

(b) Minimize $f(x) = \max_{i=1,\ldots,K} c_i^T F(x)^{-1} c_i$ where $c_i \in \mathbf{R}^m$, $i = 1, \ldots, K$.

(c) Minimize $f(x) = \sup_{\|c\|_2 \leq 1} c^T F(x)^{-1} c$.

(d) Minimize $f(x) = \mathbf{E}(c^T F(x)^{-1} c)$ where c is a random vector with mean $\mathbf{E}\, c = \bar{c}$ and covariance $\mathbf{E}(c - \bar{c})(c - \bar{c})^T = S$.

3.12 *A matrix fractional function* [Ando] Show that $X = B^T A^{-1} B$ solves the SDP

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}\, X \\ \text{subject to} & \left[\begin{array}{cc} A & B \\ B^T & X \end{array} \right] \succeq 0, \end{array}$$

with variable $X \in \mathbf{S}^n$, where $A \in \mathbf{S}^m_{++}$ and $B \in \mathbf{R}^{m \times n}$ are given.

Conclude that $\mathbf{tr}(B^T A^{-1} B)$ is a convex function of (A, B), for A positive definite.

3.13 *Trace of harmonic mean of matrices* [Ando]. The matrix $H(A, B) = 2(A^{-1} + B^{-1})^{-1}$ is known as the *harmonic mean* of positive definite matrices A and B. Show that $X = (1/2)H(A, B)$ solves the SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}\, X \\ \text{subject to} & \left[\begin{array}{cc} X & X \\ X & X \end{array} \right] \preceq \left[\begin{array}{cc} A & 0 \\ 0 & B \end{array} \right], \end{array}$$

with variable $X \in \mathbf{S}^n$. The matrices $A \in \mathbf{S}^n_{++}$ and $B \in \mathbf{S}^n_{++}$ are given. Conclude that the function $\mathbf{tr}\left((A^{-1} + B^{-1})^{-1}\right)$, with domain $\mathbf{S}^n_{++} \times \mathbf{S}^n_{++}$, is concave.

Hint. Verify that the matrix

$$R = \left[\begin{array}{cc} A^{-1} & I \\ B^{-1} & -I \end{array} \right]$$

is nonsingular. Then apply the congruence transformation defined by R to the two sides of matrix inequality in the SDP, to obtain an equivalent inequality

$$R^T \left[\begin{array}{cc} X & X \\ X & X \end{array} \right] R \preceq R^T \left[\begin{array}{cc} A & 0 \\ 0 & B \end{array} \right] R.$$

3.14 *Trace of geometric mean of matrices* [Ando].

$$G(A, B) = A^{1/2} \left(A^{-1/2} B A^{-1/2} \right)^{1/2} A^{1/2}$$

is known as the *geometric mean* of positive definite matrices A and B. Show that $X = G(A, B)$ solves the SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}\, X \\ \text{subject to} & \left[\begin{array}{cc} A & X \\ X & B \end{array} \right] \succeq 0. \end{array}$$

The variable is $X \in \mathbf{S}^n$. The matrices $A \in \mathbf{S}^n_{++}$ and $B \in \mathbf{S}^n_{++}$ are given.

Conclude that the function $\mathbf{tr}\, G(A, B)$ is concave, for $A,\ B$ positive definite.

Hint. The symmetric matrix square root is monotone: if U and V are positive semidefinite with $U \preceq V$ then $U^{1/2} \preceq V^{1/2}$.

3.15 *Transforming a standard form convex problem to conic form.* In this problem we show that any convex problem can be cast in conic form, provided some technical conditions hold. We start with a standard form convex problem with linear objective (without loss of generality):

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m, \\ & Ax = b, \end{array}$$

where $f_i : \mathbf{R}^n \to \mathbf{R}$ are convex, and $x \in \mathbf{R}^n$ is the variable. For simplicity, we will assume that $\mathbf{dom}\, f_i = \mathbf{R}^n$ for each i.

Now introduce a new scalar variable $t \in \mathbf{R}$ and form the convex problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & tf_i(x/t) \leq 0, \quad i = 1, \ldots, m, \\ & Ax = b, \quad t = 1. \end{array}$$

Define

$$K = \mathbf{cl}\{(x,t) \in \mathbf{R}^{n+1} \mid tf_i(x/t) \leq 0,\ i = 1, \ldots, m,\ t > 0\}.$$

Then our original problem can be expressed as

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & (x,t) \in K, \\ & Ax = b, \quad t = 1. \end{array}$$

This is a conic problem when K is proper.

You will relate some properties of the original problem to K.

(a) Show that K is a convex cone. (It is closed by definition, since we take the closure.)

(b) Suppose the original problem is strictly feasible, *i.e.*, there exists a point $\bar{x}$ with $f_i(x) < 0$, $i = 1, \ldots, m$. (This is called Slater's condition.) Show that K has nonempty interior.

(c) Suppose that the inequalities define a bounded set, *i.e.*, $\{x \mid f_i(x) \leq 0,\ i = 1, \ldots, m\}$ is bounded. Show that K is pointed.

3.16 *Exploring nearly optimal points.* An optimization algorithm will find *an* optimal point for a problem, provided the problem is feasible. It is often useful to explore the set of nearly optimal points. When a problem has a 'strong minimum', the set of nearly optimal points is small; all such points are close to the original optimal point found. At the other extreme, a problem can have a 'soft minimum', which means that there are many points, some quite far from the original optimal point found, that are feasible and have nearly optimal objective value. In this problem you will use a typical method to explore the set of nearly optimal points.

We start by finding the optimal value $p^\star$ of the given problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p, \end{array}$$

as well as an optimal point $x^\star \in \mathbf{R}^n$. We then pick a small positive number ϵ, and a vector $c \in \mathbf{R}^n$, and solve the problem

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & h_i(x) = 0, \quad i = 1, \ldots, p \\ & f_0(x) \leq p^\star + \epsilon. \end{array}$$

Note that any feasible point for this problem is ϵ-suboptimal for the original problem. Solving this problem multiple times, with different c's, will generate (perhaps different) ϵ-suboptimal points. If

the problem has a strong minimum, these points will all be close to each other; if the problem has a weak minimum, they can be quite different.

There are different strategies for choosing c in these experiments. The simplest is to choose the c's randomly; another method is to choose c to have the form $\pm e_i$, for $i = 1, \ldots, n$. (This method gives the 'range' of each component of x, over the ϵ-suboptimal set.)

You will carry out this method for the following problem, to determine whether it has a strong minimum or a weak minimum. You can generate the vectors c randomly, with enough samples for you to come to your conclusion. You can pick $\epsilon = 0.01p^\star$, which means that we are considering the set of 1% suboptimal points.

The problem is a minimum fuel optimal control problem for a vehicle moving in $\mathbf{R}^2$. The position at time kh is given by $p(k) \in \mathbf{R}^2$, and the velocity by $v(k) \in \mathbf{R}^2$, for $k = 1, \ldots, K$. Here $h > 0$ is the sampling period. These are related by the equations

$$p(k+1) = p(k) + hv(k), \quad v(k+1) = (1-\alpha)v(k) + (h/m)f(k), \quad k = 1, \ldots, K-1,$$

where $f(k) \in \mathbf{R}^2$ is the force applied to the vehicle at time kh, $m > 0$ is the vehicle mass, and $\alpha \in (0, 1)$ models drag on the vehicle; in the absense of any other force, the vehicle velocity decreases by the factor $1 - \alpha$ in each discretized time interval. (These formulas are approximations of more accurate formulas that involve matrix exponentials.)

The force comes from two thrusters, and from gravity:

$$f(k) = \begin{bmatrix} \cos\theta_1 \\ \sin\theta_1 \end{bmatrix} u_1(k) + \begin{bmatrix} \cos\theta_2 \\ \sin\theta_2 \end{bmatrix} u_2(k) + \begin{bmatrix} 0 \\ -mg \end{bmatrix}, \quad k = 1, \ldots, K-1.$$

Here $u_1(k) \in \mathbf{R}$ and $u_2(k) \in \mathbf{R}$ are the (nonnegative) thruster force magnitudes, θ_1 and θ_2 are the directions of the thrust forces, and $g = 10$ is the constant acceleration due to gravity.

The total fuel use is

$$F = \sum_{k=1}^{K-1} \left(u_1(k) + u_2(k) \right).$$

(Recall that $u_1(k) \geq 0$, $u_2(k) \geq 0$.)

The problem is to minimize fuel use subject to the initial condition $p(1) = 0$, $v(1) = 0$, and the way-point constraints

$$p(k_i) = w_i, \quad i = 1, \ldots, M.$$

(These state that at the time hk_i, the vehicle must pass through the location $w_i \in \mathbf{R}^2$.) In addition, we require that the vehicle should remain in a square operating region,

$$\|p(k)\|_\infty \leq P^{\max}, \quad k = 1, \ldots, K.$$

Both parts of this problem concern the specific problem instance with data given in `thrusters_data.*`.

(a) Find an optimal trajectory, and the associated minimum fuel use $p^\star$. Plot the trajectory $p(k)$ in $\mathbf{R}^2$ (*i.e.*, in the p_1, p_2 plane). Verify that it passes through the way-points.

(b) Generate several 1% suboptimal trajectories using the general method described above, and plot the associated trajectories in $\mathbf{R}^2$. Would you say this problem has a strong minimum, or a weak minimum?

3.17 *Minimum fuel optimal control.* Solve the minimum fuel optimal control problem described in exercise 4.16 of *Convex Optimization*, for the instance with problem data

$$A = \begin{bmatrix} -1 & 0.4 & 0.8 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 0 \\ 0.3 \end{bmatrix}, \qquad x_{\mathrm{des}} = \begin{bmatrix} 7 \\ 2 \\ -6 \end{bmatrix}, \qquad N = 30.$$

You can do this by forming the LP you found in your solution of exercise 4.16, or more directly using CVX. Plot the actuator signal $u(t)$ as a function of time t.

3.18 *Heuristic suboptimal solution for Boolean LP.* This exercise builds on exercises 4.15 and 5.13 in *Convex Optimization*, which involve the Boolean LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b \\ & x_i \in \{0,1\}, \quad i = 1, \ldots, n, \end{array}$$

with optimal value $p^\star$. Let x^{rlx} be a solution of the LP relaxation

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b \\ & 0 \preceq x \preceq \mathbf{1}, \end{array}$$

so $L = c^T x^{\mathrm{rlx}}$ is a lower bound on $p^\star$. The relaxed solution x^{rlx} can also be used to guess a Boolean point $\hat{x}$, by rounding its entries, based on a threshold $t \in [0,1]$:

$$\hat{x}_i = \begin{cases} 1 & x_i^{\mathrm{rlx}} \geq t \\ 0 & \text{otherwise,} \end{cases}$$

for $i = 1, \ldots, n$. Evidently $\hat{x}$ is Boolean (*i.e.*, has entries in $\{0,1\}$). If it is feasible for the Boolean LP, *i.e.*, if $A\hat{x} \preceq b$, then it can be considered a guess at a good, if not optimal, point for the Boolean LP. Its objective value, $U = c^T \hat{x}$, is an upper bound on $p^\star$. If U and L are close, then $\hat{x}$ is nearly optimal; specifically, $\hat{x}$ cannot be more than $(U - L)$-suboptimal for the Boolean LP.

This rounding need not work; indeed, it can happen that for all threshold values, $\hat{x}$ is infeasible. But for some problem instances, it can work well.

Of course, there are many variations on this simple scheme for (possibly) constructing a feasible, good point from x^{rlx}.

Finally, we get to the problem. Generate problem data using one of the following.

Matlab code:

```
rand('state',0);
n=100;
m=300;
A=rand(m,n);
b=A*ones(n,1)/2;
c=-rand(n,1);
```

Python code:

```
import numpy as np
np.random.seed(0)
(m, n) = (300, 100)
A = np.random.rand(m, n); A = np.asmatrix(A)
b = A.dot(np.ones((n, 1)))/2; b = np.asmatrix(b)
c = -np.random.rand(n, 1); c = np.asmatrix(c)
```

Julia code:

```
srand(0);
n=100;
m=300;
A=rand(m,n);
b=A*ones(n,1)/2;
c=-rand(n,1);
```

You can think of x_i as a job we either accept or decline, and $-c_i$ as the (positive) revenue we generate if we accept job i. We can think of $Ax \preceq b$ as a set of limits on m resources. A_{ij}, which is positive, is the amount of resource i consumed if we accept job j; b_i, which is positive, is the amount of resource i available.

Find a solution of the relaxed LP and examine its entries. Note the associated lower bound L. Carry out threshold rounding for (say) 100 values of t, uniformly spaced over $[0, 1]$. For each value of t, note the objective value $c^T\hat{x}$ and the maximum constraint violation $\max_i(A\hat{x} - b)_i$. Plot the objective value and the maximum violation versus t. Be sure to indicate on the plot the values of t for which $\hat{x}$ is feasible, and those for which it is not.

Find a value of t for which $\hat{x}$ is feasible, and gives minimum objective value, and note the associated upper bound U. Give the gap $U - L$ between the upper bound on $p^\star$ and the lower bound on $p^\star$.

In Matlab, if you define vectors `obj` and `maxviol`, you can find the upper bound as `U=min(obj(find(maxviol<=0)))`

3.19 *Optimal operation of a hybrid vehicle.* Solve the instance of the hybrid vehicle operation problem described in exercise 4.65 in *Convex Optimization*, with problem data given in the file `hybrid_veh_data.*`, and fuel use function $F(p) = p + \gamma p^2$ (for $p \geq 0$).

Hint. You will actually formulate and solve a *relaxation* of the original problem. You may find that some of the equality constraints you relaxed to inequality constraints do not hold for the solution found. This is not an error: it just means that there is no incentive (in terms of the objective) for the inequality to be tight. You can fix this in (at least) two ways. One is to go back and adjust certain variables, without affecting the objective and maintaining feasibility, so that the relaxed constraints hold with equality. Another simple method is to add to the objective a term of the form

$$\epsilon \sum_{t=1}^{T} \max\{0, -P_{\mathrm{mg}}(t)\},$$

where ϵ is small and positive. This makes it more attractive to use the brakes to extract power from the wheels, even when the battery is (or will be) full (which removes any fuel incentive).

Find the optimal fuel consumption, and compare to the fuel consumption with a non-hybrid version of the same vehicle (*i.e.*, one without a battery). Plot the braking power, engine power, motor/generator power, and battery energy versus time.

How would you use optimal dual variables for this problem to find $\partial F_{\text{total}}/\partial E_{\text{batt}}^{\max}$, *i.e.*, the partial derivative of optimal fuel consumption with respect to battery capacity? (You can just assume that this partial derivative exists.) You do not have to give a long derivation or proof; you can just state how you would find this derivative from optimal dual variables for the problem. Verify your method numerically, by changing the battery capacity a small amount and re-running the optimization, and comparing this to the prediction made using dual variables.

3.20 *Optimal vehicle speed scheduling.* A vehicle (say, an airplane) travels along a fixed path of n segments, between $n+1$ waypoints labeled $0, \ldots, n$. Segment i starts at waypoint $i-1$ and terminates at waypoint i. The vehicle starts at time $t = 0$ at waypoint 0. It travels over each segment at a constant (nonnegative) speed; s_i is the speed on segment i. We have lower and upper limits on the speeds: $s^{\min} \preceq s \preceq s^{\max}$. The vehicle does not stop at the waypoints; it simply proceeds to the next segment. The travel distance of segment i is d_i (which is positive), so the travel time over segment i is d_i/s_i. We let τ_i, $i = 1, \ldots, n$, denote the time at which the vehicle arrives at waypoint i. The vehicle is required to arrive at waypoint i, for $i = 1, \ldots, n$, between times $\tau_i^{\min}$ and $\tau_i^{\max}$, which are given. The vehicle consumes fuel over segment i at a rate that depends on its speed, $\Phi(s_i)$, where Φ is positive, increasing, and convex, and has units of kg/s.

You are given the data d (segment travel distances), $s^{\min}$ and $s^{\max}$ (speed bounds), $\tau^{\min}$ and $\tau^{\max}$ (waypoint arrival time bounds), and the fuel use function $\Phi : \mathbf{R} \to \mathbf{R}$. You are to choose the speeds $s_1, \ldots, s_n$ so as to minimize the total fuel consumed in kg.

(a) Show how to pose this as a convex optimization problem. If you introduce new variables, or change variables, you must explain how to recover the optimal speeds from the solution of your problem. If convexity of the objective or any constraint function in your formulation is not obvious, explain why it is convex.

(b) Carry out the method of part (a) on the problem instance with data in `veh_speed_sched_data.m`. Use the fuel use function $\Phi(s_i) = as_i^2 + bs_i + c$ (the parameters a, b, and c are defined in the data file). What is the optimal fuel consumption? Plot the optimal speed versus segment, using the matlab command `stairs` or the function `step` from matplotlib in Python and Julia to better show constant speed over the segments.

3.21 *Norm approximation via SOCP, for ℓ_p-norms with rational p.*

(a) Use the observation at the beginning of exercise 4.26 in *Convex Optimization* to express the constraint

$$y \leq \sqrt{z_1 z_2}, \qquad y, z_1, z_2 \geq 0,$$

with variables y, z_1, z_2, as a second-order cone constraint. Then extend your result to the constraint

$$y \leq (z_1 z_2 \cdots z_n)^{1/n}, \qquad y \geq 0, \qquad z \succeq 0,$$

where n is a positive integer, and the variables are $y \in \mathbf{R}$ and $z \in \mathbf{R}^n$. First assume that n is a power of two, and then generalize your formulation to arbitrary positive integers.

(b) Express the constraint

$$f(x) \leq t$$

as a second-order cone constraint, for the following two convex functions f:

$$f(x) = \begin{cases} x^\alpha & x \geq 0 \\ 0 & x < 0, \end{cases}$$

where α is rational and greater than or equal to one, and

$$f(x) = x^\alpha, \qquad \mathbf{dom}\, f = \mathbf{R}_{++},$$

where α is rational and negative.

(c) Formulate the norm approximation problem

$$\text{minimize} \quad \|Ax - b\|_p$$

as a second-order cone program, where p is a rational number greater than or equal to one. The variable in the optimization problem is $x \in \mathbf{R}^n$. The matrix $A \in \mathbf{R}^{m\times n}$ and the vector $b \in \mathbf{R}^m$ are given. For an m-vector y, the norm $\|y\|_p$ is defined as

$$\|y\|_p = \left(\sum_{k=1}^m |y_k|^p\right)^{1/p}$$

when $p \geq 1$.

3.22 *Linear optimization over the complement of a convex set.* Suppose $\mathcal{C} \subseteq \mathbf{R}^n_+$ is a closed bounded convex set with $0 \in \mathcal{C}$, and $c \in \mathbf{R}^n_+$. We define

$$\tilde{\mathcal{C}} = \mathbf{cl}(\mathbf{R}^n_+ \setminus \mathcal{C}) = \mathbf{cl}\{x \in \mathbf{R}^n_+ \mid x \notin \mathcal{C}\},$$

which is the closure of the complement of $\mathcal{C}$ in $\mathbf{R}^n_+$.

Show that $c^T x$ has a minimizer over $\tilde{\mathcal{C}}$ of the form αe_k, where $\alpha \geq 0$ and e_k is the kth standard unit vector. (If you have not had a course on analysis, you can give an intuitive argument.)

If follows that we can minimize $c^T x$ over $\tilde{\mathcal{C}}$ by solving n one-dimensional optimization problems (which, indeed, can each be solved by bisection, provided we can check whether a point is in $\mathcal{C}$ or not).

3.23 *Jensen's inequality for posynomials.* Suppose $f : \mathbf{R}^n \to \mathbf{R}$ is a posynomial function, $x,\ y \in \mathbf{R}^n_{++}$, and $\theta \in [0, 1]$. Define $z \in \mathbf{R}^n_{++}$ by $z_i = x_i^\theta y_i^{1-\theta}$, $i = 1, \ldots, n$. Show that $f(z) \leq f(x)^\theta f(y)^{1-\theta}$.

Interpretation. We can think of z as a θ-weighted geometric mean between x and y. So the statement above is that a posynomial, evaluated at a weighted geometric mean of two points, is no more than the weighted geometric mean of the posynomial evaluated at the two points.

3.24 *CVX implementation of a concave function.* Consider the concave function $f : \mathbf{R} \to \mathbf{R}$ defined by

$$f(x) = \begin{cases} (x+1)/2 & x > 1 \\ \sqrt{x} & 0 \leq x \leq 1, \end{cases}$$

with $\mathbf{dom}\, f = \mathbf{R}_+$. Give a CVX implementation of f, via a partially specified optimization problem. Check your implementation by maximizing $f(x) + f(a - x)$ for several interesting values of a (say, $a = -1$, $a = 1$, and $a = 3$).

3.25 The following optimization problem arises in portfolio optimization:

$$\begin{array}{ll} \text{maximize} & \dfrac{r^T x + d}{\|Rx + q\|_2} \\ \text{subject to} & \sum_{i=1}^{n} f_i(x_i) \leq b \\ & x \succeq c. \end{array}$$

The variable is $x \in \mathbf{R}^n$. The functions f_i are defined as

$$f_i(x) = \alpha_i x_i + \beta_i |x_i| + \gamma_i |x_i|^{3/2},$$

with $\beta_i > |\alpha_i|$, $\gamma_i > 0$. We assume there exists a feasible x with $r^T x + d > 0$.

Show that this problem can be solved by solving an SOCP (if possible) or a sequence of SOCP feasibility problems (otherwise).

3.26 *Positive nonconvex QCQP.* We consider a (possibly nonconvex) QCQP, with nonnegative variable $x \in \mathbf{R}^n$,

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & x \succeq 0, \end{array}$$

where $f_i(x) = (1/2)x^T P_i x + q_i^T x + r_i$, with $P_i \in \mathbf{S}^n$, $q_i \in \mathbf{R}^n$, and $r_i \in \mathbf{R}$, for $i = 0, \ldots, m$. We do *not* assume that $P_i \succeq 0$, so this need not be a convex problem.

Suppose that $q_i \preceq 0$, and P_i have nonpositive off-diagonal entries, *i.e.*, they satisfy

$$(P_i)_{jk} \leq 0, \quad j \neq k, \quad j, k = 1, \ldots, n,$$

for $i = 0, \ldots, m$. (A matrix with nonpositive off-diagonal entries is called a *Z-matrix*.) Explain how to reformulate this problem as a convex problem.

Hint. Change variables using $y_j = \phi(x_j)$, for some suitable function ϕ.

3.27 *Affine policy.* We consider a family of LPs, parametrized by the random variable u, which is uniformly distributed on $\mathcal{U} = [-1, 1]^p$,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b(u), \end{array}$$

where $x \in \mathbf{R}^n$, $A \in \mathbf{R}^{m \times n}$, and $b(u) = b_0 + Bu \in \mathbf{R}^m$ is an affine function of u. You can think of u_i as representing a deviation of the ith parameter from its nominal value. The parameters might represent (deviations in) levels of resources available, or other varying limits.

The problem is to be solved many times; in each time, the value of u (*i.e.*, a sample) is given, and then the decision variable x is chosen. The mapping from u into the decision variable $x(u)$ is called the *policy*, since it gives the decision variable value for each value of u. When enough time and computing hardware is available, we can simply solve the LP for each new value of u; this is an optimal policy, which we denote $x^\star(u)$.

In some applications, however, the decision $x(u)$ must be made very quickly, so solving the LP is not an option. Instead we seek a suboptimal policy, which is affine: $x^{\text{aff}}(u) = x_0 + Ku$, where x_0 is

called the *nominal decision* and $K \in \mathbf{R}^{n\times p}$ is called the *feedback gain matrix*. (Roughly speaking, x_0 is our guess of x before the value of u has been revealed; Ku is our modification of this guess, once we know u.) We determine the policy (*i.e.*, suitable values for x_0 and K) ahead of time; we can then evaluate the policy (that is, find $x^{\mathrm{aff}}(u)$ given u) very quickly, by matrix multiplication and addition.

We will choose x_0 and K in order to minimize the expected value of the objective, while insisting that for any value of u, feasibility is maintained:

$$\begin{array}{ll} \text{minimize} & \mathbf{E}\, c^T x^{\mathrm{aff}}(u) \\ \text{subject to} & Ax^{\mathrm{aff}}(u) \preceq b(u) \quad \forall u \in \mathcal{U}. \end{array}$$

The variables here are x_0 and K. The expectation in the objective is over u, and the constraint requires that $Ax^{\mathrm{aff}}(u) \preceq b(u)$ hold almost surely.

(a) Explain how to find optimal values of x_0 and K by solving a standard explicit convex optimization problem (*i.e.*, one that does not involve an expectation or an infinite number of constraints, as the one above does.) The numbers of variables or constraints in your formulation should not grow exponentially with the problem dimensions n, p, or m.

(b) Carry out your method on the data given in `affine_pol_data.m`. To evaluate your affine policy, generate 100 independent samples of u, and for each value, compute the objective value of the affine policy, $c^T x^{\mathrm{aff}}(u)$, and of the optimal policy, $c^T x^\star(u)$. Scatter plot the objective value of the affine policy (y-axis) versus the objective value of the optimal policy (x-axis), and include the line $y = x$ on the plot. Report the average values of $c^T x^{\mathrm{aff}}(u)$ and $c^T x^\star(u)$ over your samples. (These are estimates of $\mathbf{E}\, c^T x^{\mathrm{aff}}(u)$ and $\mathbf{E}\, c^T x^\star(u)$. The first number, by the way, can be found exactly.)

3.28 *Probability bounds.* Consider random variables X_1, X_2, X_3, X_4 that take values in $\{0, 1\}$. We are given the following marginal and conditional probabilities:

$$\begin{aligned} \mathbf{prob}(X_1 = 1) &= 0.9, \\ \mathbf{prob}(X_2 = 1) &= 0.9, \\ \mathbf{prob}(X_3 = 1) &= 0.1, \\ \mathbf{prob}(X_1 = 1, X_4 = 0 \mid X_3 = 1) &= 0.7, \\ \mathbf{prob}(X_4 = 1 \mid X_2 = 1, X_3 = 0) &= 0.6. \end{aligned}$$

Explain how to find the minimum and maximum possible values of $\mathbf{prob}(X_4 = 1)$, over all (joint) probability distributions consistent with the given data. Find these values and report them.

Hints. (You should feel free to ignore these hints.)

- Matlab:
  - CVX supports multidimensional arrays; for example, `variable p(2,2,2,2)` declares a 4-dimensional array of variables, with each of the four indices taking the values 1 or 2.
  - The function `sum(p,i)` sums a multidimensional array `p` along the ith index.
  - The expression `sum(a(:))` gives the sum of all entries of a multidimensional array `a`. You might want to use the function definition `sum_all = @(A) sum( A(:));`, so `sum_all(a)` gives the sum of all entries in the multidimensional array `a`.

- Python:
  - Create a 1-d Variable and manually index the entries. You should come up with a reasonable scheme to avoid confusion.
- Julia:
  - You can create a multidimensional array of variables in Convex.jl. For example, the following creates a 4-dimensional array of variables, with each of the four indices taking the values 1 or 2.

    ```
    p = [Variable() for i in 1:16];
    p = reshape(p, 2, 2, 2, 2)
    ```
  - You can use the function `sum` to sum over various indices in the multidimesional array.

    ```
    sum(p[:,:,:,:]) # sum all entries
    sum(p[1,:,2,:]) # fix first and third indices
    ```
  - To create constraints with the variables in the array, you need to access each variable independently. Something like `p >= 0` will not work.

3.29 *Robust quadratic programming.* In this problem, we consider a robust variation of the (convex) *quadratic* program

$$\begin{array}{ll} \text{minimize} & (1/2)x^TPx + q^Tx + r \\ \text{subject to} & Ax \preceq b. \end{array}$$

For simplicity we assume that only the matrix P is subject to errors, and the other parameters (q, r, A, b) are exactly known. The robust quadratic program is defined as

$$\begin{array}{ll} \text{minimize} & \sup_{P\in\mathcal{E}}((1/2)x^TPx + q^Tx + r) \\ \text{subject to} & Ax \preceq b \end{array}$$

where $\mathcal{E}$ is the set of possible matrices P.

For each of the following sets $\mathcal{E}$, express the robust QP as a *tractable* convex problem. Be as specific as you can. (Here, tractable means that the problem can be reduced to an LP, QP, QCQP, SOCP, or SDP. But you do not have to work out the reduction, if it is complicated; it is enough to argue that it can be reduced to one of these.)

(a) A finite set of matrices: $\mathcal{E} = \{P_1, \ldots, P_K\}$, where $P_i \in \mathbf{S}^n_+$, $i = 1, \ldots, K$.

(b) A set specified by a nominal value $P_0 \in \mathbf{S}^n_+$ plus a bound on the eigenvalues of the deviation $P - P_0$:

$$\mathcal{E} = \{P \in \mathbf{S}^n \mid -\gamma I \preceq P - P_0 \preceq \gamma I\}$$

where $\gamma \in \mathbf{R}$ and $P_0 \in \mathbf{S}^n_+$.

(c) An ellipsoid of matrices:

$$\mathcal{E} = \left\{ P_0 + \sum_{i=1}^K P_i u_i \;\middle|\; \|u\|_2 \leq 1 \right\}.$$

You can assume $P_i \in \mathbf{S}^n_+$, $i = 0, \ldots, K$.

3.30 *Smallest confidence ellipsoid.* Suppose the random variable X on $\mathbf{R}^n$ has log-concave density p. Formulate the following problem as a convex optimization problem: Find an ellipsoid $\mathcal{E}$ that satisfies $\mathbf{prob}(X \in \mathcal{E}) \geq 0.95$ and is smallest, in the sense of minimizing the sum of the squares of its semi-axis lengths. You do not need to worry about how to solve the resulting convex optimization problem; it is enough to formulate the smallest confidence ellipsoid problem as the problem of minimizing a convex function over a convex set involving the parameters that define $\mathcal{E}$.

3.31 *Stochastic optimization via Monte Carlo sampling.* In (convex) stochastic optimization, the goal is to minimize a cost function of the form $F(x) = \mathbf{E}\, f(x, \omega)$, where ω is a random variable on Ω, and $f : \mathbf{R}^n \times \Omega \to \mathbf{R}$ is convex in its first argument for each $\omega \in \Omega$. (For simplicity we consider the unconstrained problem; it is not hard to include constraints.) Evidently F is convex. Let $p^\star$ denote the optimal value, *i.e.*, $p^\star = \inf_x F(x)$ (which we assume is finite).

In a few very simple cases we can work out what F is analytically, but in general this is not possible. Moreover in many applications, we do not know the distribution of ω; we only have access to an oracle that can generate independent samples from the distribution.

A standard method for approximately solving the stochastic optimization problem is based on Monte Carlo sampling. We first generate N independent samples, $\omega_1, \ldots, \omega_N$, and form the empirical expectation

$$\hat{F}(x) = \frac{1}{N} \sum_{i=1}^{N} f(x, \omega_i).$$

This is a random function, since it depends on the particular samples drawn. For each x, we have $\mathbf{E}\, \hat{F}(x) = F(x)$, and also $\mathbf{E}(\hat{F}(x) - F(x))^2 \propto 1/N$. Roughly speaking, for N large enough, $\hat{F}(x) \approx F(x)$.

To (approximately) minimize F, we instead minimize $\hat{F}(x)$. The minimizer, $\hat{x}^\star$, and the optimal value $\hat{p}^\star = \hat{F}(\hat{x}^\star)$, are also random variables. The hope is that for N large enough, we have $\hat{p}^\star \approx p^\star$. (In practice, stochastic optimization via Monte Carlo sampling works very well, even when N is not that big.)

One way to check the result of Monte Carlo sampling is to carry it out multiple times. We repeatedly generate different batches of samples, and for each batch, we find $\hat{x}^\star$ and $\hat{p}^\star$. If the values of $\hat{p}^\star$ are near each other, it's reasonable to believe that we have (approximately) minimized F. If they are not, it means our value of N is too small.

Show that $\mathbf{E}\, \hat{p}^\star \leq p^\star$.

This inequality implies that if we repeatedly use Monte Carlo sampling and the values of $\hat{p}^\star$ that we get are all very close, then they are (likely) close to $p^\star$.

Hint. Show that for any function $G : \mathbf{R}^n \times \Omega \to \mathbf{R}$ (convex or not in its first argument), and any random variable ω on Ω, we have

$$\inf_x \mathbf{E}\, G(x, \omega) \geq \mathbf{E} \inf_x G(x, \omega).$$

3.32 *Satisfying a minimum number of constraints.* Consider the problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0 \text{ holds for at least } k \text{ values of } i, \end{array}$$

with variable $x \in \mathbf{R}^n$, where the objective f_0 and the constraint functions f_i, $i = 1, \ldots, m$ (with $m \geq k$), are convex. Here we require that only k of the constraints hold, instead of all m of them. In general this is a hard combinatorial problem; the brute force solution is to solve all $\binom{m}{k}$ convex problems obtained by choosing subsets of k constraints to impose, and selecting one with smallest objective value.

In this problem we explore a convex restriction that can be an effective heuristic for the problem.

(a) Suppose $\lambda > 0$. Show that the constraint

$$\sum_{i=1}^{m} (1 + \lambda f_i(x))_+ \leq m - k$$

guarantees that $f_i(x) \leq 0$ holds for at least k values of i. ($(u)_+$ means $\max\{u, 0\}$.)

Hint. For each $u \in \mathbf{R}$, $(1 + \lambda u)_+ \geq 1(u > 0)$, where $1(u > 0) = 1$ for $u > 0$, and $1(u > 0) = 0$ for $u \leq 0$.

(b) Consider the problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & \sum_{i=1}^{m} (1 + \lambda f_i(x))_+ \leq m - k \\ & \lambda > 0, \end{array}$$

with variables x and λ. This is a restriction of the original problem: If (x, λ) are feasible for it, then x is feasible for the original problem. Show how to solve this problem using convex optimization. (This may involve a change of variables.)

(c) Apply the method of part (b) to the problem instance

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & a_i^T x \leq b_i \text{ holds for at least } k \text{ values of } i, \end{array}$$

with $m = 70$, $k = 58$, and $n = 12$. The vectors b, c and the matrix A with rows a_i^T are given in the file `satisfy_some_constraints_data.*`.

Report the optimal value of λ, the objective value, and the actual number of constraints that are satisfied (which should be larger than or equal to k). To determine if a constraint is satisfied, you can use the tolerance $a_i^T x - b_i \leq \epsilon^{\text{feas}}$, with $\epsilon^{\text{feas}} = 10^{-5}$.

A standard trick is to take this tentative solution, choose the k constraints with the smallest values of $f_i(x)$, and then minimize $f_0(x)$ subject to these k constraints (*i.e.*, ignoring the other $m - k$ constraints). This improves the objective value over the one found using the restriction. Carry this out for the problem instance, and report the objective value obtained.

3.33 [Barvinok, Pataki] A standard form semidefinite program is defined as

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) \\ \text{subject to} & \mathbf{tr}(A_i X) = b_i, \quad i = 1, \ldots, m \\ & X \succeq 0. \end{array} \tag{4}$$

The variable X and the coefficients $A_1, \ldots, A_m$ are symmetric $n \times n$ matrices.

A matrix $\hat{X}$ is called an *extreme point* of the feasible set of (4) if $\hat{X}$ is feasible and if the only matrix $V \in \mathbf{S}^n$ that satisfies the conditions

$$\mathbf{tr}\,(A_i V) = 0, \quad i = 1, \ldots, m, \qquad \hat{X} + V \succeq 0, \qquad \hat{X} - V \succeq 0 \tag{5}$$

is $V = 0$. In this problem we work out a bound on the rank of extreme points.

(a) Suppose $\hat{X}$ is feasible for (4) and has rank r. Define an eigenvalue decomposition

$$\hat{X} = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} \Lambda_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} Q_1 & Q_2 \end{bmatrix}^T,$$

where $\begin{bmatrix} Q_1 & Q_2 \end{bmatrix}$ is orthogonal, Q_1 has r columns, and Λ_1 is a diagonal $r \times r$ matrix with positive diagonal elements. Show that V satisfies (5) if and only if it can be expressed as $V = Q_1 Y Q_1^T$ where $Y \in \mathbf{S}^r$ satisfies

$$\mathbf{tr}\,(Q_1^T A_i Q_1 Y) = 0, \quad i = 1, \ldots, m, \qquad \Lambda_1 + Y \succeq 0, \qquad \Lambda_1 - Y \succeq 0.$$

(b) Show that if $r(r+1)/2 > m$, then $\hat{X}$ is not an extreme point.

(c) Interpret the SDP (4) as a relaxation of the non-convex QCQP

$$\begin{array}{ll} \text{minimize} & x^T C x \\ \text{subject to} & x^T A_i x = b_i, \quad i = 1, \ldots, m. \end{array}$$

What are the implications of part b for the exactness of the relaxation? (You can assume that the feasible set of (4) is non-empty and bounded. Under this assumption, the SDP is guaranteed to have optimal solutions that are extreme points of the feasible set.)

4 Duality

4.1 *Numerical perturbation analysis example.* Consider the quadratic program

$$\begin{array}{ll} \text{minimize} & x_1^2 + 2x_2^2 - x_1x_2 - x_1 \\ \text{subject to} & x_1 + 2x_2 \leq u_1 \\ & x_1 - 4x_2 \leq u_2, \\ & 5x_1 + 76x_2 \leq 1, \end{array}$$

with variables x_1, x_2, and parameters u_1, u_2.

(a) Solve this QP, for parameter values $u_1 = -2$, $u_2 = -3$, to find optimal primal variable values $x_1^\star$ and $x_2^\star$, and optimal dual variable values $\lambda_1^\star$, $\lambda_2^\star$ and $\lambda_3^\star$. Let $p^\star$ denote the optimal objective value. Verify that the KKT conditions hold for the optimal primal and dual variables you found (within reasonable numerical accuracy).

Matlab hint: See §3.7 of the CVX users' guide to find out how to retrieve optimal dual variables. To specify the quadratic objective, use `quad_form()`.

(b) We will now solve some perturbed versions of the QP, with

$$u_1 = -2 + \delta_1, \qquad u_2 = -3 + \delta_2,$$

where δ_1 and δ_2 each take values from $\{-0.1, 0, 0.1\}$. (There are a total of nine such combinations, including the original problem with $\delta_1 = \delta_2 = 0$.) For each combination of δ_1 and δ_2, make a prediction $p^\star_{\text{pred}}$ of the optimal value of the perturbed QP, and compare it to $p^\star_{\text{exact}}$, the exact optimal value of the perturbed QP (obtained by solving the perturbed QP). Put your results in the two righthand columns in a table with the form shown below. Check that the inequality $p^\star_{\text{pred}} \leq p^\star_{\text{exact}}$ holds.

δ_1	δ_2	$p^\star_{\text{pred}}$	$p^\star_{\text{exact}}$
0	0		
0	−0.1		
0	0.1		
−0.1	0		
−0.1	−0.1		
−0.1	0.1		
0.1	0		
0.1	−0.1		
0.1	0.1		

4.2 *A determinant maximization problem.* We consider the problem

$$\begin{array}{ll} \text{minimize} & \log\det X^{-1} \\ \text{subject to} & A_i^T X A_i \preceq B_i, \quad i = 1, \ldots, m, \end{array}$$

with variable $X \in \mathbf{S}^n$, and problem data $A_i \in \mathbf{R}^{n \times k_i}$, $B_i \in \mathbf{S}^{k_i}_{++}$, $i = 1, \ldots, m$. The constraint $X \succ 0$ is implicit.

We can give several interpretations of this problem. Here is one, from statistics. Let z be a random variable in $\mathbf{R}^n$, with covariance matrix X, which is unknown. However, we do have (matrix) upper

bounds on the covariance of the random variables $y_i = A_i^T z \in \mathbf{R}^{k_i}$, which is $A_i^T X A_i$. The problem is to find the covariance matrix for z, that is consistent with the known upper bounds on the covariance of y_i, that has the largest volume confidence ellipsoid.

Derive the Lagrange dual of this problem. Be sure to state what the dual variables are (*e.g.*, vectors, scalars, matrices), any constraints they must satisfy, and what the dual function is. If the dual function has any implicit equality constraints, make them explicit. You can assume that $\sum_{i=1}^m A_i A_i^T \succ 0$, which implies the feasible set of the original problem is bounded.

What can you say about the optimal duality gap for this problem?

4.3 The relative entropy between two vectors x, $y \in \mathbf{R}^n_{++}$ is defined as

$$\sum_{k=1}^n x_k \log(x_k/y_k).$$

This is a convex function, jointly in x and y. In the following problem we calculate the vector x that minimizes the relative entropy with a given vector y, subject to equality constraints on x:

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{k=1}^n x_k \log(x_k/y_k) \\ \text{subject to} & Ax = b \\ & \mathbf{1}^T x = 1 \end{array}$$

The optimization variable is $x \in \mathbf{R}^n$. The domain of the objective function is $\mathbf{R}^n_{++}$. The parameters $y \in \mathbf{R}^n_{++}$, $A \in \mathbf{R}^{m\times n}$, and $b \in \mathbf{R}^m$ are given.

Derive the Lagrange dual of this problem and simplify it to get

$$\text{maximize} \quad b^T z - \log \textstyle\sum_{k=1}^n y_k e^{a_k^T z}$$

(a_k is the kth column of A).

4.4 *Source localization from range measurements* [Beck, Stoica, and Li]. A signal emitted by a source at an unknown position $x \in \mathbf{R}^n$ ($n = 2$ or $n = 3$) is received by m sensors at known positions y_1, $\ldots$, $y_m \in \mathbf{R}^n$. From the strength of the received signals, we can obtain noisy estimates d_k of the distances $\|x - y_k\|_2$. We are interested in estimating the source position x based on the measured distances d_k.

In the following problem the error between the squares of the actual and observed distances is minimized:

$$\text{minimize} \quad f_0(x) = \sum_{k=1}^m \left(\|x - y_k\|_2^2 - d_k^2\right)^2.$$

Introducing a new variable $t = x^T x$, we can express this as

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{k=1}^m \left(t - 2y_k^T x + \|y_k\|_2^2 - d_k^2\right)^2 \\ \text{subject to} & x^T x - t = 0. \end{array} \tag{6}$$

The variables are $x \in \mathbf{R}^n$, $t \in \mathbf{R}$. Although this problem is not convex, it can be shown that strong duality holds. (It is a variation on the problem discussed on page 229 and in exercise 5.29 of *Convex Optimization*.)

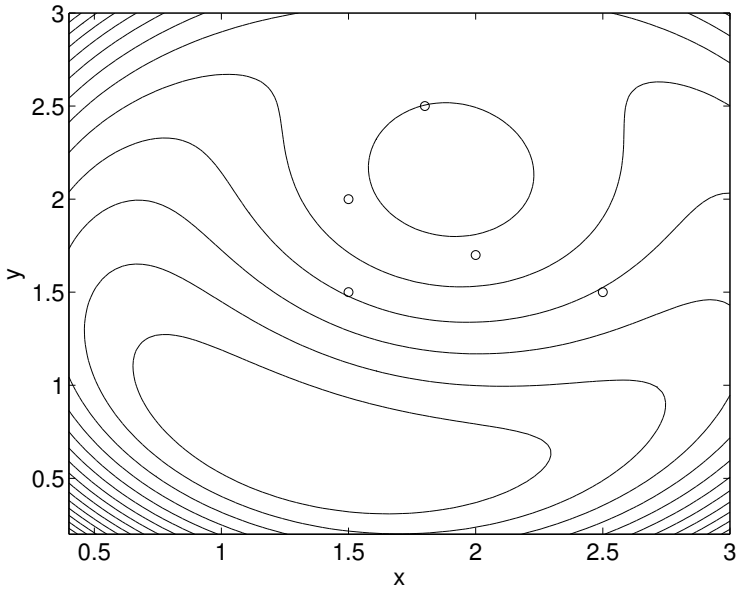
Solve (6) for an example with $m = 5$,

$$y_1 = \begin{bmatrix} 1.8 \\ 2.5 \end{bmatrix}, \qquad y_2 = \begin{bmatrix} 2.0 \\ 1.7 \end{bmatrix}, \qquad y_3 = \begin{bmatrix} 1.5 \\ 1.5 \end{bmatrix}, \qquad y_4 = \begin{bmatrix} 1.5 \\ 2.0 \end{bmatrix}, \qquad y_5 = \begin{bmatrix} 2.5 \\ 1.5 \end{bmatrix},$$

and

$$d = (2.00,\ 1.24,\ 0.59,\ 1.31,\ 1.44).$$

The figure shows some contour lines of the cost function f_0, with the positions y_k indicated by circles.



To solve the problem, you can note that $x^\star$ is easily obtained from the KKT conditions for (6) if the optimal multiplier $\nu^\star$ for the equality constraint is known. You can use one of the following two methods to find $\nu^\star$.

- Derive the dual problem, express it as an SDP, and solve it using CVX.
- Reduce the KKT conditions to a nonlinear equation in ν, and pick the correct solution (similarly as in exercise 5.29 of *Convex Optimization*).

4.5 *Projection on the ℓ_1 ball.* Consider the problem of projecting a point $a \in \mathbf{R}^n$ on the unit ball in ℓ_1-norm:

$$\begin{array}{ll} \text{minimize} & (1/2)\|x - a\|_2^2 \\ \text{subject to} & \|x\|_1 \leq 1. \end{array}$$

Derive the dual problem and describe an efficient method for solving it. Explain how you can obtain the optimal x from the solution of the dual problem.

4.6 *A nonconvex problem with strong duality.* On page 229 of *Convex Optimization*, we consider the problem

$$\begin{array}{ll} \text{minimize} & f(x) = x^T A x + 2b^T x \\ \text{subject to} & x^T x \leq 1 \end{array} \tag{7}$$

with variable $x \in \mathbf{R}^n$, and data $A \in \mathbf{S}^n$, $b \in \mathbf{R}^n$. We do not assume that A is positive semidefinite, and therefore the problem is not necessarily convex. In this exercise we show that x is (globally) optimal if and only if there exists a λ such that

$$\|x\|_2 \leq 1, \qquad \lambda \geq 0, \qquad A + \lambda I \succeq 0, \qquad (A + \lambda I)x = -b, \qquad \lambda(1 - \|x\|_2^2) = 0. \tag{8}$$

From this we will develop an efficient method for finding the global solution. The conditions (8) are the KKT conditions for (7) with the inequality $A + \lambda I \succeq 0$ added.

(a) Show that if x and λ satisfy (8), then $f(x) = \inf_{\tilde{x}} L(\tilde{x}, \lambda) = g(\lambda)$, where L is the Lagrangian of the problem and g is the dual function. Therefore strong duality holds, and x is globally optimal.

(b) Next we show that the conditions (8) are also necessary. Assume that x is globally optimal for (7). We distinguish two cases.

(i) $\|x\|_2 < 1$. Show that (8) holds with $\lambda = 0$.

(ii) $\|x\|_2 = 1$. First prove that $(A + \lambda I)x = -b$ for some $\lambda \geq 0$. (In other words, the negative gradient $-(Ax + b)$ of the objective function is normal to the unit sphere at x, and point away from the origin.) You can show this by contradiction: if the condition does not hold, then there exists a direction v with $v^T x < 0$ and $v^T(Ax + b) < 0$. Show that $f(x + tv) < f(x)$ for small positive t.
It remains to show that $A + \lambda I \succeq 0$. If not, there exists a w with $w^T(A + \lambda I)w < 0$, and without loss of generality we can assume that $w^T x \neq 0$. Show that the point $y = x + tw$ with $t = -2w^T x / w^T w$ satisfies $\|y\|_2 = 1$ and $f(y) < f(x)$.

(c) The optimality conditions (8) can be used to derive a simple algorithm for (7). Using the eigenvalue decomposition $A = \sum_{i=1}^n \alpha_i q_i q_i^T$, of A, we make a change of variables $y_i = q_i^T x$, and write (7) as

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n \alpha_i y_i^2 + 2\sum_{i=1}^n \beta_i y_i \\ \text{subject to} & y^T y \leq 1 \end{array}$$

where $\beta_i = q_i^T b$. The transformed optimality conditions (8) are

$$\|y\|_2 \leq 1, \qquad \lambda \geq -\alpha_n, \qquad (\alpha_i + \lambda)y_i = -\beta_i, \quad i = 1, \ldots, n, \qquad \lambda(1 - \|y\|_2) = 0,$$

if we assume that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_n$. Give an algorithm for computing the solution y and λ.

4.7 *Connection between perturbed optimal cost and Lagrange dual functions.* In this exercise we explore the connection between the optimal cost, as a function of perturbations to the righthand sides of the constraints,

$$p^\star(u) = \inf\{f_0(x) \mid \exists x \in \mathcal{D},\ f_i(x) \leq u_i,\ i = 1, \ldots, m\},$$

(as in §5.6), and the Lagrange dual function

$$g(\lambda) = \inf_x \left(f_0(x) + \lambda_1 f_1(x) + \cdots + \lambda_m f_m(x)\right),$$

with domain restricted to $\lambda \succeq 0$. We assume the problem is convex. We consider a problem with inequality constraints only, for simplicity.

We have seen several connections between $p^\star$ and g:

- *Slater's condition and strong duality.* Slater's condition is: there exists $u \prec 0$ for which $p^\star(u) < \infty$. Strong duality (which follows) is: $p^\star(0) = \sup_\lambda g(\lambda)$. (Note that we include the condition $\lambda \succeq 0$ in the domain of g.)
- *A global inequality.* We have $p^\star(u) \geq p^\star(0) - \lambda^{\star T} u$, for any u, where $\lambda^\star$ maximizes g.
- *Local sensitivity analysis.* If $p^\star$ is differentiable at 0, then we have $\nabla p^\star(0) = -\lambda^\star$, where $\lambda^\star$ maximizes g.

In fact the two functions are closely related by conjugation. Show that

$$p^\star(u) = (-g)^*(-u).$$

Here $(-g)^*$ is the conjugate of the function $-g$. You can show this for $u \in \mathbf{int}\,\mathbf{dom}\, p^\star$.

Hint. Consider the problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & \tilde{f}_i(x) = f_i(x) - u_i \leq 0, \quad i = 1, \ldots, m. \end{array}$$

Verify that Slater's condition holds for this problem, for $u \in \mathbf{int}\,\mathbf{dom}\, p^\star$.

4.8 *Exact penalty method for SDP.* Consider the pair of primal and dual SDPs

$$\begin{array}{llll} \text{(P)} & \text{minimize} & c^T x & \\ & \text{subject to} & F(x) \preceq 0 & \end{array} \qquad \begin{array}{llll} \text{(D)} & \text{maximize} & \mathbf{tr}(F_0 Z) & \\ & \text{subject to} & \mathbf{tr}(F_i Z) + c_i = 0, & i = 1, \ldots, m \\ & & Z \succeq 0, & \end{array}$$

where $F(x) = F_0 + x_1 F_1 + \cdots + x_n F_n$ and $F_i \in \mathbf{S}^p$ for $i = 0, \ldots, n$. Let $Z^\star$ be a solution of (D). Show that every solution $x^\star$ of the unconstrained problem

$$\text{minimize} \quad c^T x + M \max\{0, \lambda_{\max}(F(x))\},$$

where $M > \mathbf{tr}\, Z^\star$, is a solution of (P).

4.9 *Quadratic penalty.* Consider the problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m, \end{array} \tag{9}$$

where the functions $f_i : \mathbf{R}^n \to \mathbf{R}$ are differentiable and convex.

Show that

$$\phi(x) = f_0(x) + \alpha \sum_{i=1}^m \max\{0, f_i(x)\}^2,$$

where $\alpha > 0$, is convex. Suppose $\tilde{x}$ minimizes ϕ. Show how to find from $\tilde{x}$ a feasible point for the dual of (9). Find the corresponding lower bound on the optimal value of (9).

4.10 *Binary least-squares.* We consider the non-convex least-squares approximation problem with binary constraints

$$\begin{array}{ll} \text{minimize} & \|Ax-b\|_2^2 \\ \text{subject to} & x_k^2 = 1, \quad k=1,\ldots,n, \end{array} \tag{10}$$

where $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$. We assume that $\mathbf{rank}(A) = n$, *i.e.*, A^TA is nonsingular.

One possible application of this problem is as follows. A signal $\hat{x} \in \{-1,1\}^n$ is sent over a noisy channel, and received as $b = A\hat{x} + v$ where $v \sim \mathcal{N}(0,\sigma^2 I)$ is Gaussian noise. The solution of (10) is the maximum likelihood estimate of the input signal $\hat{x}$, based on the received signal b.

(a) Derive the Lagrange dual of (10) and express it as an SDP.

(b) Derive the dual of the SDP in part (a) and show that it is equivalent to

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(A^TAZ) - 2b^TAz + b^Tb \\ \text{subject to} & \mathbf{diag}(Z) = \mathbf{1} \\ & \left[\begin{array}{cc} Z & z \\ z^T & 1 \end{array}\right] \succeq 0. \end{array} \tag{11}$$

Interpret this problem as a relaxation of (10). Show that if

$$\mathbf{rank}\left(\left[\begin{array}{cc} Z & z \\ z^T & 1 \end{array}\right]\right) = 1 \tag{12}$$

at the optimum of (11), then the relaxation is exact, *i.e.*, the optimal values of problems (10) and (11) are equal, and the optimal solution z of (11) is optimal for (10). This suggests a heuristic for rounding the solution of the SDP (11) to a feasible solution of (10), if (12) does not hold. We compute the eigenvalue decomposition

$$\left[\begin{array}{cc} Z & z \\ z^T & 1 \end{array}\right] = \sum_{i=1}^{n+1} \lambda_i \left[\begin{array}{c} v_i \\ t_i \end{array}\right]\left[\begin{array}{c} v_i \\ t_i \end{array}\right]^T,$$

where $v_i \in \mathbf{R}^n$ and $t_i \in \mathbf{R}$, and approximate the matrix by a rank-one matrix

$$\left[\begin{array}{cc} Z & z \\ z^T & 1 \end{array}\right] \approx \lambda_1 \left[\begin{array}{c} v_1 \\ t_1 \end{array}\right]\left[\begin{array}{c} v_1 \\ t_1 \end{array}\right]^T.$$

(Here we assume the eigenvalues are sorted in decreasing order). Then we take $x = \mathbf{sign}(v_1)$ as our guess of good solution of (10).

(c) We can also give a probabilistic interpretation of the relaxation (11). Suppose we interpret z and Z as the first and second moments of a random vector $v \in \mathbf{R}^n$ (*i.e.*, $z = \mathbf{E}\, v$, $Z = \mathbf{E}\, vv^T$). Show that (11) is equivalent to the problem

$$\begin{array}{ll} \text{minimize} & \mathbf{E}\,\|Av-b\|_2^2 \\ \text{subject to} & \mathbf{E}\, v_k^2 = 1, \quad k=1,\ldots,n, \end{array}$$

where we minimize over all possible probability distributions of v.

This interpretation suggests another heuristic method for computing suboptimal solutions of (10) based on the result of (11). We choose a distribution with first and second moments $\mathbf{E}\, v = z$, $\mathbf{E}\, vv^T = Z$ (for example, the Gaussian distribution $\mathcal{N}(z, Z - zz^T)$). We generate a number of samples $\tilde{v}$ from the distribution and round them to feasible solutions $x = \mathbf{sign}(\tilde{v})$. We keep the solution with the lowest objective value as our guess of the optimal solution of (10).

(d) Solve the dual problem (11) using CVX. Generate problem instances using the Matlab code

```
randn('state',0)
m = 50;
n = 40;
A = randn(m,n);
xhat = sign(randn(n,1));
b = A*xhat + s*randn(m,1);
```

for four values of the noise level s: $s = 0.5$, $s = 1$, $s = 2$, $s = 3$. For each problem instance, compute suboptimal feasible solutions x using the the following heuristics and compare the results.

(i) $x^{(a)} = \mathbf{sign}(x_{\mathrm{ls}})$ where x_{ls} is the solution of the least-squares problem

$$\text{minimize} \quad \|Ax - b\|_2^2.$$

(ii) $x^{(b)} = \mathbf{sign}(z)$ where z is the optimal value of the variable z in the SDP (11).

(iii) $x^{(c)}$ is computed from a rank-one approximation of the optimal solution of (11), as explained in part (b) above.

(iv) $x^{(d)}$ is computed by rounding 100 samples of $\mathcal{N}(z, Z - zz^T)$, as explained in part (c) above.

4.11 *Monotone transformation of the objective.* Consider the optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m. \end{array} \tag{13}$$

where $f_i : \mathbf{R}^n \to \mathbf{R}$ for $i = 0, 1, \ldots, m$ are convex. Suppose $\phi : \mathbf{R} \to \mathbf{R}$ is increasing and convex. Then the problem

$$\begin{array}{ll} \text{minimize} & \tilde{f}_0(x) = \phi(f_0(x)) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \end{array} \tag{14}$$

is convex and equivalent to it; in fact, it has the same optimal set as (13).

In this problem we explore the connections between the duals of the two problems (13) and (14). We assume f_i are differentiable, and to make things specific, we take $\phi(a) = \exp a$.

(a) Suppose λ is feasible for the dual of (13), and $\bar{x}$ minimizes

$$f_0(x) + \sum_{i=1}^m \lambda_i f_i(x).$$

Show that $\bar{x}$ also minimizes

$$\exp f_0(x) + \sum_{i=1}^m \tilde{\lambda}_i f_i(x)$$

for appropriate choice of $\tilde{\lambda}$. Thus, $\tilde{\lambda}$ is dual feasible for (14).

(b) Let $p^\star$ denote the optimal value of (13) (so the optimal value of (14) is $\exp p^\star$). From λ we obtain the bound

$$p^\star \geq g(\lambda),$$

where g is the dual function for (13). From $\tilde{\lambda}$ we obtain the bound $\exp p^\star \geq \tilde{g}(\tilde{\lambda})$, where $\tilde{g}$ is the dual function for (14). This can be expressed as

$$p^\star \geq \log \tilde{g}(\tilde{\lambda}).$$

How do these bounds compare? Are they the same, or is one better than the other?

4.12 *Variable bounds and dual feasibility.* In many problems the constraints include *variable bounds*, as in

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m \\ & l_i \leq x_i \leq u_i, \quad i = 1, \ldots, n. \end{array} \tag{15}$$

Let $\mu \in \mathbf{R}^n_+$ be the Lagrange multipliers associated with the constraints $x_i \leq u_i$, and let $\nu \in \mathbf{R}^n_+$ be the Lagrange multipliers associated with the constraints $l_i \geq x_i$. Thus the Lagrangian is

$$L(x, \lambda, \mu, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \mu^T(x - u) + \nu^T(l - x).$$

(a) Show that for any $x \in \mathbf{R}^n$ and any λ, we can choose $\mu \succeq 0$ and $\nu \succeq 0$ so that x minimizes $L(x, \lambda, \mu, \nu)$. In particular, it is very easy to find dual feasible points.

(b) Construct a dual feasible point (λ, μ, ν) by applying the method you found in part (a) with $x = (l + u)/2$ and $\lambda = 0$. From this dual feasible point you get a lower bound on $f^\star$. Show that this lower bound can be expressed as

$$f^\star \geq f_0((l + u)/2) - ((u - l)/2)^T \, |\nabla f_0((l + u)/2)|$$

where $|\cdot|$ means componentwise. Can you prove this bound directly?

4.13 *Deducing costs from samples of optimal decision.* A system (such as a firm or an organism) chooses a vector of values x as a solution of the LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \succeq b, \end{array}$$

with variable $x \in \mathbf{R}^n$. You can think of $x \in \mathbf{R}^n$ as a vector of activity levels, $b \in \mathbf{R}^m$ as a vector of requirements, and $c \in \mathbf{R}^n$ as a vector of costs or prices for the activities. With this interpretation, the LP above finds the cheapest set of activity levels that meet all requirements. (This interpretation is not needed to solve the problem.)

We suppose that A is known, along with a set of data

$$(b^{(1)}, x^{(1)}), \quad \ldots, \quad (b^{(r)}, x^{(r)}),$$

where $x^{(j)}$ is an optimal point for the LP, with $b = b^{(j)}$. (The solution of an LP need not be unique; all we say here is that $x^{(j)}$ is *an* optimal solution.) Roughly speaking, we have samples of optimal decisions, for different values of requirements.

You *do not* know the cost vector c. Your job is to compute the tightest possible bounds on the costs c_i from the given data. More specifically, you are to find $c_i^{\max}$ and $c_i^{\min}$, the maximum and minimum possible values for c_i, consistent with the given data.

Note that if x is optimal for the LP for a given c, then it is also optimal if c is scaled by any positive factor. To normalize c, then, we will assume that $c_1 = 1$. Thus, we can interpret c_i as the relative cost of activity i, compared to activity 1.

(a) Explain how to find $c_i^{\max}$ and $c_i^{\min}$. Your method can involve the solution of a reasonable number (not exponential in n, m or r) of convex or quasiconvex optimization problems.

(b) Carry out your method using the data found in `deducing_costs_data.m`. You may need to determine whether individual inequality constraints are tight; to do so, use a tolerance threshold of $\epsilon = 10^{-3}$. (In other words: if $a_k^T x - b_k \leq 10^{-3}$, you can consider this inequality as tight.)

Give the values of $c_i^{\max}$ and $c_i^{\min}$, and make a very brief comment on the results.

4.14 *Kantorovich inequality.*

(a) Suppose $a \in \mathbf{R}^n$ with $a_1 \geq a_2 \geq \cdots \geq a_n > 0$, and $b \in \mathbf{R}^n$ with $b_k = 1/a_k$.
Derive the KKT conditions for the convex optimization problem

$$\begin{array}{ll} \text{minimize} & -\log(a^T x) - \log(b^T x) \\ \text{subject to} & x \succeq 0, \quad \mathbf{1}^T x = 1. \end{array}$$

Show that $x = (1/2, 0, \ldots, 0, 1/2)$ is optimal.

(b) Suppose $A \in \mathbf{S}^n_{++}$ with eigenvalues λ_k sorted in decreasing order. Apply the result of part (a), with $a_k = \lambda_k$, to prove the *Kantorovich inequality:*

$$2\left(u^T A u\right)^{1/2}\left(u^T A^{-1} u\right)^{1/2} \leq \sqrt{\frac{\lambda_1}{\lambda_n}} + \sqrt{\frac{\lambda_n}{\lambda_1}}$$

for all u with $\|u\|_2 = 1$.

4.15 State and solve the optimality conditions for the problem

$$\begin{array}{ll} \text{minimize} & \log\det\left(\begin{bmatrix} X_1 & X_2 \\ X_2^T & X_3 \end{bmatrix}^{-1}\right) \\ \text{subject to} & \mathbf{tr}\, X_1 = \alpha \\ & \mathbf{tr}\, X_2 = \beta \\ & \mathbf{tr}\, X_3 = \gamma. \end{array}$$

The optimization variable is

$$X = \begin{bmatrix} X_1 & X_2 \\ X_2^T & X_3 \end{bmatrix},$$

with $X_1 \in \mathbf{S}^n$, $X_2 \in \mathbf{R}^{n\times n}$, $X_3 \in \mathbf{S}^n$. The domain of the objective function is $\mathbf{S}^{2n}_{++}$. We assume $\alpha > 0$, and $\alpha\gamma > \beta^2$.

4.16 Consider the optimization problem

$$\begin{array}{ll} \text{minimize} & -\log\det X + \mathbf{tr}(SX) \\ \text{subject to} & X \text{ is tridiagonal} \end{array}$$

with domain $\mathbf{S}^n_{++}$ and variable $X \in \mathbf{S}^n$. The matrix $S \in \mathbf{S}^n$ is given. Show that the optimal $X_{\rm opt}$ satisfies

$$(X^{-1}_{\rm opt})_{ij} = S_{ij}, \quad |i-j| \leq 1.$$

4.17 We denote by $f(A)$ the sum of the largest r eigenvalues of a symmetric matrix $A \in \mathbf{S}^n$ (with $1 \leq r \leq n$), *i.e.*,

$$f(A) = \sum_{k=1}^{r} \lambda_k(A),$$

where $\lambda_1(A), \ldots, \lambda_n(A)$ are the eigenvalues of A sorted in decreasing order.

(a) Show that the optimal value of the SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(AX) \\ \text{subject to} & \mathbf{tr}\, X = r \\ & 0 \preceq X \preceq I, \end{array}$$

with variable $X \in \mathbf{S}^n$, is equal to $f(A)$.

(b) Show that f is a convex function.

(c) Assume $A(x) = A_0 + x_1A_1 + \cdots + x_mA_m$, with $A_k \in \mathbf{S}^n$. Use the observation in part (a) to formulate the optimization problem

$$\text{minimize} \quad f(A(x)),$$

with variable $x \in \mathbf{R}^m$, as an SDP.

4.18 *An exact penalty function.* Suppose we are given a convex problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \ i = 1, \ldots, m \end{array} \tag{16}$$

with dual

$$\begin{array}{ll} \text{maximize} & g(\lambda) \\ \text{subject to} & \lambda \succeq 0. \end{array} \tag{17}$$

We assume that Slater's condition holds, so we have strong duality and the dual optimum is attained. For simplicity we will assume that there is a unique dual optimal solution $\lambda^\star$.

For fixed $t > 0$, consider the unconstrained minimization problem

$$\text{minimize } f_0(x) + t \max_{i=1,\ldots,m} f_i(x)^+, \tag{18}$$

where $f_i(x)^+ = \max\{f_i(x), 0\}$.

(a) Show that the objective function in (18) is convex.

(b) We can express (18) as

$$\begin{array}{ll} \text{minimize} & f_0(x) + ty \\ \text{subject to} & f_i(x) \leq y, \quad i = 1.\ldots, m \\ & 0 \leq y \end{array} \tag{19}$$

where the variables are x and $y \in \mathbf{R}$.

Find the Lagrange dual problem of (19) and express it in terms of the Lagrange dual function g for problem (16).

(c) Use the result in (b) to prove the following property. If $t > \mathbf{1}^T\lambda^\star$, then any minimizer of (18) is also an optimal solution of (16).

(The second term in (18) is called a *penalty function* for the constraints in (16). It is zero if x is feasible, and adds a penalty to the cost function when x is infeasible. The penalty function is called *exact* because for t large enough, the solution of the unconstrained problem (18) is also a solution of (16).)

4.19 *Infimal convolution.* Let $f_1, \ldots, f_m$ be convex functions on $\mathbf{R}^n$. Their *infimal convolution*, denoted $g = f_1 \diamond \cdots \diamond f_m$ (several other notations are also used), is defined as

$$g(x) = \inf\{f_1(x_1) + \cdots + f_m(x_m) \mid x_1 + \cdots + x_m = x\},$$

with the natural domain (*i.e.*, defined by $g(x) < \infty$). In one simple interpretation, $f_i(x_i)$ is the cost for the ith firm to produce a mix of products given by x_i; $g(x)$ is then the optimal cost obtained if the firms can freely exchange products to produce, all together, the mix given by x. (The name 'convolution' presumably comes from the observation that if we replace the sum above with the product, and the infimum above with integration, then we obtain the normal convolution.)

(a) Show that g is convex.

(b) Show that $g^* = f_1^* + \cdots + f_m^*$. In other words, the conjugate of the infimal convolution is the sum of the conjugates.

(c) Verify the identity in part (b) for the specific case of two strictly convex quadratic functions, $f_i(x) = (1/2)x^T P_i x$, with $P_i \in \mathbf{S}^n_{++}$, $i = 1, 2$.

Hint: Depending on how you work out the conjugates, you might find the matrix identity $(X + Y)^{-1}Y = X^{-1}(X^{-1} + Y^{-1})^{-1}$ useful.

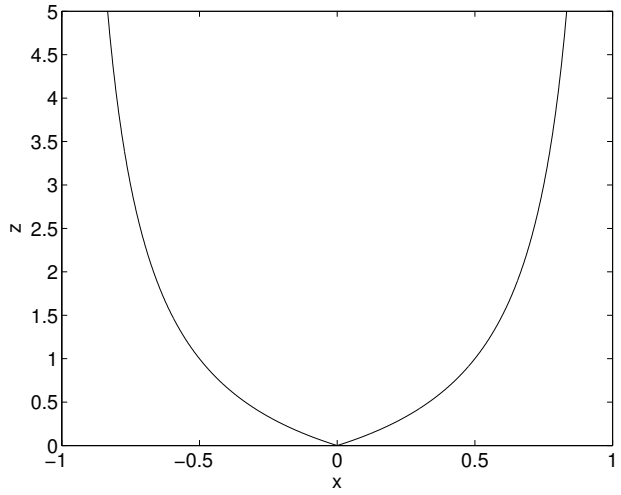
4.20 Derive the Lagrange dual of the optimization problem

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^n \phi(x_i) \\ \text{subject to} & Ax = b \end{array}$$

with variable $x \in \mathbf{R}^n$, where

$$\phi(u) = \frac{|u|}{c - |u|} = -1 + \frac{c}{c - |u|}, \qquad \mathbf{dom}\,\phi = (-c, c).$$

c is a positive parameter. The figure shows ϕ for $c = 1$.



4.21 *Robust LP with polyhedral cost uncertainty.* We consider a robust linear programming problem, with polyhedral uncertainty in the cost:

$$\begin{array}{ll} \text{minimize} & \sup_{c\in\mathcal{C}} c^T x \\ \text{subject to} & Ax \succeq b, \end{array}$$

with variable $x \in \mathbf{R}^n$, where $\mathcal{C} = \{c \mid Fc \preceq g\}$. You can think of x as the quantities of n products to buy (or sell, when $x_i < 0$), $Ax \succeq b$ as constraints, requirements, or limits on the available quantities, and $\mathcal{C}$ as giving our knowledge or assumptions about the product prices at the time we place the order. The objective is then the worst possible (*i.e.*, largest) possible cost, given the quantities x, consistent with our knowledge of the prices.

In this exercise, you will work out a tractable method for solving this problem. You can assume that $\mathcal{C} \neq \emptyset$, and the inequalities $Ax \succeq b$ are feasible.

(a) Let $f(x) = \sup_{c\in\mathcal{C}} c^T x$ be the objective in the problem above. Explain why f is convex.

(b) Find the dual of the problem

$$\begin{array}{ll} \text{maximize} & c^T x \\ \text{subject to} & Fc \preceq g, \end{array}$$

with variable c. (The problem data are x, F, and g.) Explain why the optimal value of the dual is $f(x)$.

(c) Use the expression for $f(x)$ found in part (b) in the original problem, to obtain a single LP equivalent to the original robust LP.

(d) Carry out the method found in part (c) to solve a robust LP with the data below. In MATLAB:

```
rand('seed',0);
A = rand(30,10);
b = rand(30,1);
c_nom = 1+rand(10,1);  % nominal c values
```

In Python:

```
import numpy as np
np.random.seed(10)
(m, n) = (30, 10)
A = np.random.rand(m, n); A = np.asmatrix(A)
b = np.random.rand(m, 1); b = np.asmatrix(b)
c_nom = np.ones((n, 1)) + np.random.rand(n, 1); c_nom = np.asmatrix(c_nom)
```

In Julia:

```
srand(10);
n = 10;
m = 30;
A = rand(m, n);
b = rand(m, 1);
c_nom = 1 + rand(n, 1);
```

Then, use $\mathcal{C}$ described as follows. Each c_i deviates no more than 25% from its nominal value, *i.e.*, $0.75c_{\rm nom} \preceq c \preceq 1.25c_{\rm nom}$, and the average of c does not deviate more than 10% from the average of the nominal values, *i.e.*, $0.9(\mathbf{1}^T c_{\rm nom})/n \leq \mathbf{1}^T c/n \leq 1.1(\mathbf{1}^T c_{\rm nom})/n$.
Compare the worst-case cost $f(x)$ and the nominal cost $c_{\rm nom}^T x$ for x optimal for the robust problem, and for x optimal for the nominal problem (*i.e.*, the case where $\mathcal{C} = \{c_{\rm nom}\}$). Compare the values and make a brief comment.

4.22 *Diagonal scaling with prescribed column and row sums* [Marshall and Olkin]. Let A be an $n \times n$ matrix with positive entries, and let c and d be positive n-vectors that satisfy $\mathbf{1}^T c = \mathbf{1}^T d = 1$. Consider the geometric program

$$\begin{array}{ll} \text{minimize} & x^T A y \\ \text{subject to} & \prod_{i=1}^n x_i^{c_i} = 1 \\ & \prod_{j=1}^n y_j^{d_j} = 1, \end{array}$$

with variables $x, y \in \mathbf{R}^n$ (and implicit constraints $x \succ 0$, $y \succ 0$). Write this geometric program in convex form and derive the optimality conditions. Show that if x and y are optimal, then the matrix

$$B = \frac{1}{x^T A y}\, \mathbf{diag}(x) A\, \mathbf{diag}(y)$$

satisfies $B\mathbf{1} = c$ and $B^T\mathbf{1} = d$.

4.23 [Schoenberg] Suppose m balls in $\mathbf{R}^n$, with centers a_i and radii r_i, have a nonempty intersection. We define y to be a point in the intersection, so

$$\|y - a_i\|_2 \leq r_i, \quad i = 1, \ldots, m. \tag{20}$$

Suppose we move the centers to new positions b_i in such a way that the distances between the centers do not increase:

$$\|b_i - b_j\|_2 \leq \|a_i - a_j\|_2, \quad i, j = 1, \ldots, m. \tag{21}$$

We will prove that the intersection of the translated balls is nonempty, *i.e.*, there exists a point x with $\|x - b_i\|_2 \leq r_i$, $i = 1, \ldots, m$. To show this we prove that the optimal value of

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & \|x - b_i\|_2^2 \leq r_i^2 + t, \quad i = 1, \ldots, m, \end{array} \tag{22}$$

with variables $x \in \mathbf{R}^n$ and $t \in \mathbf{R}$, is less than or equal to zero.

(a) Show that (21) implies that

$$t - (x - b_i)^T(x - b_j) \leq -(y - a_i)^T(y - a_j) \quad \text{for } i, j \in I,$$

if (x, t) is feasible in (22), and $I \subseteq \{1, \ldots, m\}$ is the set of active constraints at x, t.

(b) Suppose x, t are optimal in (22) and that $\lambda_1, \ldots, \lambda_m$ are optimal dual variables. Use the optimality conditions for (22) and the inequality in part a to show that

$$t = t - \|\sum_{i=1}^m \lambda_i(x - b_i)\|_2^2 \leq -\|\sum_{i=1}^m \lambda_i(y - a_i)\|_2^2.$$

4.24 *Controlling a switched linear system via duality.* We consider a discrete-time dynamical system with state $x_t \in \mathbf{R}^n$. The state propagates according to the recursion

$$x_{t+1} = A_t x_t, \quad t = 0, 1, \ldots, T - 1,$$

where the matrices A_t are to be chosen from a finite set $\mathcal{A} = \{A^{(1)}, \ldots, A^{(K)}\}$ in order to control the state x_t over a finite time horizon of length T. More formally, the switched-linear control problem is

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^T f(x_t) \\ \text{subject to} & x_{t+1} = A^{(u_t)} x_t, \quad \text{for } t = 0, \ldots, T - 1 \end{array}$$

The problem variables are $x_t \in \mathbf{R}^n$, for $t = 1, \ldots, T$, and $u_t \in \{1, \ldots, K\}$, for $t = 0, \ldots, T - 1$. We assume the initial state, $x_0 \in \mathbf{R}^n$ is a problem parameter (*i.e.*, is known and fixed). You may assume the function f is convex, though it isn't necessary for this problem.

Note that, to find a feasible point, we take any sequence $u_0, \ldots, u_{T-1} \in \{1, \ldots, K\}$; we then generate a feasible point according to the recursion

$$x_{t+1} = A^{(u_t)} x_t, \quad t = 0, 1, \ldots, T - 1.$$

The switched-linear control problem is *not* convex, and is hard to solve globally. Instead, we consider a heuristic based on Lagrange duality.

(a) Find the dual of the switched-linear control problem explicitly in terms of x_0, $A^{(1)}, \ldots, A^{(K)}$, the function f, and its conjugate f^*. Your formulation cannot involve a number of constraints or objective terms that is exponential in K or T. (This includes minimization or maximization with an exponential number of terms.)

(b) Given optimal dual variables $\nu_1^\star, \ldots, \nu_T^\star$ corresponding to the T constraints of the switched-linear control problem, a heuristic to choose u_t is to minimize the Langrangian using these optimal dual variables:

$$(\tilde{u}_0, \ldots, \tilde{u}_{T-1}) \in \operatorname*{argmin}_{u_0, \ldots, u_{T-1} \in \{1, \ldots, K\}} \inf_{x_1, \ldots, x_T} L(x_1, \ldots, x_T, u_0, \ldots, u_{T-1}, \nu_1^\star, \ldots, \nu_T^\star),$$

Given the optimal dual variables, show (explicitly) how to find $\tilde{u}_0, \ldots, \tilde{u}_{T-1}$.

(c) Consider the case $f(x) = (1/2)x^T Qx$. with $Q \in \mathbf{S}^n_{++}$. For the data given in `sw_lin_ctrl_data.*`, solve the dual problem and report its optimal value $d^\star$, which is a lower bound on $p^\star$. (As a courtesy, we also included $p^\star$ in the data file, so you can check your bound.)

Note: Julia users might want to use the ECOS solver, by including `using ECOS`, and solving by using `solve!(prob, ECOSSolver())`.

(d) Using the same data as is part (c), carry out the heuristic method of part (b) to compute $\tilde{u}_0, \ldots \tilde{u}_{T-1}$. Use these values to generate a feasible point. Report the value of the objective at this feasible point, which is an upper bound on $p^\star$.

4.25 [Friedland and Karlin] Let A be an $n \times n$ matrix with positive entries, and let u and v be two positive n-vectors. Show that one can compute positive diagonal matrices D_1 and D_2 that satisfy

$$(D_1 A D_2)u = u, \qquad (D_1 A D_2)^T v = v \tag{23}$$

by the following method. Define $\alpha_i = u_i v_i$ for $i = 1, \ldots, n$, and solve the optimization problem

$$\begin{array}{ll} \text{minimize} & \prod_{i=1}^n (\sum_{j=1}^n A_{ij} x_j)^{\alpha_i} \\ \text{subject to} & \prod_{i=1}^n x_i^{\alpha_i} = 1 \end{array} \tag{24}$$

with domain $\{x \in \mathbf{R}^n \mid x \succ 0\}$. Then use the solution x to define

$$D_1 = \mathbf{diag}(u)\,\mathbf{diag}(Ax)^{-1}, \qquad D_2 = \mathbf{diag}(u)^{-1}\,\mathbf{diag}(x).$$

The first equality in (23) follows immediately from the expressions of D_1 and D_2. To show the second equality in (23), express (24) as a convex optimization problem and derive the optimality conditions.

4.26 Consider the optimization problem

$$\text{minimize } \|Ax - b\|_2 + \gamma \|x\|_1$$

with $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$, and $\gamma > 0$. The variable is an n-vector x.

(a) Derive the Lagrange dual of the equivalent problem

$$\begin{array}{ll} \text{minimize} & \|y\|_2 + \gamma \|x\|_1 \\ \text{subject to} & Ax - b = y \end{array}$$

with variables $x \in \mathbf{R}^n$ and $y \in \mathbf{R}^m$.

(b) Suppose $Ax^\star - b \neq 0$ where $x^\star$ is an optimal point. Define $r = (Ax^\star - b)/\|Ax^\star - b\|_2$. Show that

$$\|A^T r\|_\infty \leq \gamma, \qquad r^T A x^\star + \gamma \|x^\star\|_1 = 0.$$

(c) Show that if the Euclidean norm of the ith column of A is less than γ, then $x_i^\star = 0$.

4.27 Consider the optimization problem

$$\text{minimize} \quad \sum_{i=1}^{m} h(\|A_i x + b_i\|_2) - c^T x$$

with variable $x \in \mathbf{R}^n$, where $c \in \mathbf{R}^n$, $A_i \in \mathbf{R}^{3\times n}$, $b_i \in \mathbf{R}^3$, and

$$h(u) = \begin{cases} (u-1)^2/2 & u \geq 1 \\ 0 & \text{otherwise.} \end{cases}$$

Derive the Lagrange dual of the equivalent problem

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{m} h(\|y_i\|_2) - c^T x \\ \text{subject to} & A_i x + b_i - y_i = 0, \quad i = 1, \ldots, m, \end{array}$$

with variables $x \in \mathbf{R}^n$ and $y_i \in \mathbf{R}^3$ for $i = 1, \ldots, m$.

This optimization problem describes the equilibrium of a structure consisting of m elastic cables suspended between different points or nodes. Some of the nodes are anchored, other nodes are free. The variable x contains the displacements of the free nodes. The vector c specifies the external forces applied to the nodes. The norm $\|A_i x + b_i\|_2$ is the distance between the endpoints of the ith cable as a function of the node displacements. The ith term in the sum in the cost function is the potential energy stored in the ith cable, assuming its undeformed length is one.

4.28 *Robust least squares with polyhedral uncertainty.* We consider a robust least-squares problem

$$\text{minimize} \quad \sum_{i=1}^{m} \sup_{a_i \in P_i} (a_i^T x - b_i)^2$$

with variable $x \in \mathbf{R}^n$. Each set P_i is a nonempty and bounded polyhedron, defined as

$$P_i = \{a_i \in \mathbf{R}^n \mid C_i a_i \preceq d_i\}$$

with $C_i \in \mathbf{R}^{p_i \times n}$, $d_i \in \mathbf{R}^{p_i}$. If we introduce variables $t_i \geq \sup_{a_i \in P_i} |a_i^T x - b_i|$ we can write the problem as

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{m} t_i^2 \\ \text{subject to} & \sup_{a_i \in P_i} \max\{a_i^T x - b_i, -a_i^T x + b_i\} \leq t_i, \quad i = 1, \ldots, m. \end{array}$$

Formulate this problem as a QP.

4.29 For an $m \times n$-matrix A (with $m \geq n$) and an integer k between 1 and n, we define $f(A)$ as the sum of the largest k singular values of A:

$$f(A) = \sigma_1(A) + \cdots + \sigma_k(A),$$

where $\sigma_1(A)$, $\sigma_2(A)$, $\ldots$, $\sigma_n(A)$ denote the singular values of A in nonincreasing order.

(a) Show that $f(A)$ is the optimal value of the SDP

$$\begin{array}{ll} \text{maximize} & \mathbf{tr}(A^T X) \\ \text{subject to} & \begin{bmatrix} U & X \\ X^T & V \end{bmatrix} \succeq 0 \\ & U \preceq I \\ & V \preceq I \\ & \mathbf{tr}\, U + \mathbf{tr}\, V = 2k, \end{array}$$

with variables $X \in \mathbf{R}^{m\times n}$, $U \in \mathbf{S}^m$, $V \in \mathbf{S}^n$.

Hint. The singular value decomposition of A can be written as $A = P\Sigma Q^T$, where $P \in \mathbf{R}^{m\times m}$ and $Q \in \mathbf{R}^{n\times n}$ are orthogonal matrices ($P^T P = I$, $Q^T Q = I$), and Σ is a diagonal $m\times n$ matrix with elements $\Sigma_{ii} = \sigma_i(A)$ for $i = 1, \ldots, n$, and $\Sigma_{ij} = 0$ for $i \neq j$. Use the decomposition to reformulate the SDP as an equivalent SDP in which A in the objective is replaced by Σ.

(b) What does the result of part (a) imply about the convexity properties of $f(A)$?

(c) Derive the Lagrange dual of the SDP in part (a). Use the dual problem to give an SDP formulation of the problem

$$\text{minimize} \quad f(A_0 + x_1 A_1 + \cdots + x_p A_p)$$

with variable $x \in \mathbf{R}^p$, where $A_0, \ldots, A_p$ are given $m \times n$ matrices.

4.30 Consider the convex optimization problem

$$\text{minimize} \quad c^T x + \frac{1}{\mu}\sum_{i=1}^{m} \log(1 + e^{\mu(a_i^T x - b_i)}) \tag{25}$$

with variable $x \in \mathbf{R}^n$, where μ is a positive constant.

(a) Derive the Lagrange dual of the equivalent problem

$$\begin{array}{ll} \text{minimize} & c^T x + \dfrac{1}{\mu}\displaystyle\sum_{i=1}^{m} \log(1 + \exp(\mu y_i)) \\ \text{subject to} & Ax - b \preceq y \end{array}$$

with variables $x \in \mathbf{R}^n$, $y \in \mathbf{R}^m$, where A is the $m \times n$-matrix with ith row a_i^T.

(b) Suppose the pair of primal and dual linear programs

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax \preceq b \end{array} \qquad\qquad \begin{array}{ll} \text{maximize} & -b^T z \\ \text{subject to} & A^T z + c = 0 \\ & z \succeq 0 \end{array}$$

has a finite optimal value $p^\star$ and a dual optimal solution $z^\star$ that satisfies $z^\star \preceq \mathbf{1}$. Let $q^\star$ be the optimal value of (25). Show that

$$p^\star \leq q^\star \leq p^\star + \frac{m \log 2}{\mu}.$$

5 Approximation and fitting

5.1 *Three measures of the spread of a group of numbers.* For $x \in \mathbf{R}^n$, we define three functions that measure the spread or width of the set of its elements (or coefficients). The first function is the *spread*, defined as

$$\phi_{\mathrm{sprd}}(x) = \max_{i=1,\ldots,n} x_i - \min_{i=1,\ldots,n} x_i.$$

This is the width of the smallest interval that contains all the elements of x.

The second function is the *standard deviation*, defined as

$$\phi_{\mathrm{stdev}}(x) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i^2 - \left(\frac{1}{n} \sum_{i=1}^{n} x_i \right)^2 \right)^{1/2}.$$

This is the statistical standard deviation of a random variable that takes the values $x_1, \ldots, x_n$, each with probability $1/n$.

The third function is the average absolute deviation from the median of the values:

$$\phi_{\mathrm{aamd}}(x) = (1/n) \sum_{i=1}^{n} |x_i - \mathrm{med}(x)|,$$

where $\mathrm{med}(x)$ denotes the median of the components of x, defined as follows. If $n = 2k-1$ is odd, then the median is defined as the value of middle entry when the components are sorted, *i.e.*, $\mathrm{med}(x) = x_{[k]}$, the kth largest element among the values $x_1, \ldots, x_n$. If $n = 2k$ is even, we define the median as the average of the two middle values, *i.e.*, $\mathrm{med}(x) = (x_{[k]} + x_{[k+1]})/2$.

Each of these functions measures the spread of the values of the entries of x; for example, each function is zero if and only if all components of x are equal, and each function is unaffected if a constant is added to each component of x.

Which of these three functions is convex? For each one, either show that it is convex, or give a counterexample showing it is not convex. By a counterexample, we mean a specific x and y such that Jensen's inequality fails, *i.e.*, $\phi((x+y)/2) > (\phi(x) + \phi(y))/2$.

5.2 *Minimax rational fit to the exponential.* (See exercise 6.9 of *Convex Optimization*.) We consider the specific problem instance with data

$$t_i = -3 + 6(i-1)/(k-1), \quad y_i = e^{t_i}, \quad i = 1, \ldots, k,$$

where $k = 201$. (In other words, the data are obtained by uniformly sampling the exponential function over the interval $[-3, 3]$.) Find a function of the form

$$f(t) = \frac{a_0 + a_1 t + a_2 t^2}{1 + b_1 t + b_2 t^2}$$

that minimizes $\max_{i=1,\ldots,k} |f(t_i) - y_i|$. (We require that $1 + b_1 t_i + b_2 t_i^2 > 0$ for $i = 1, \ldots, k$.)

Find optimal values of a_0, a_1, a_2, b_1, b_2, and give the optimal objective value, computed to an accuracy of 0.001. Plot the data and the optimal rational function fit on the same plot. On a different plot, give the fitting error, *i.e.*, $f(t_i) - y_i$.

Hint. To check if a feasibility problem is feasible, in Matlab, you can use `strcmp(cvx_status,'Solved')` after `cvx_end`. In Python, use `problem.status == 'optimal'`. In Julia, use `problem.status == :Optimal`. In Julia, make sure to use the ECOS solver.

5.3 *Approximation with trigonometric polynomials.* Suppose $y : \mathbf{R} \to \mathbf{R}$ is a 2π-periodic function. We will approximate y with the trigonometric polynomial

$$f(t) = \sum_{k=0}^{K} a_k \cos(kt) + \sum_{k=1}^{K} b_k \sin(kt).$$

We consider two approximations: one that minimizes the L_2-norm of the error, defined as

$$\|f - y\|_2 = \left(\int_{-\pi}^{\pi} (f(t) - y(t))^2 \, dt \right)^{1/2},$$

and one that minimizes the L_1-norm of the error, defined as

$$\|f - y\|_1 = \int_{-\pi}^{\pi} |f(t) - y(t)| \, dt.$$

The L_2 approximation is of course given by the (truncated) Fourier expansion of y.

To find an L_1 approximation, we discretize t at $2N$ points,

$$t_i = -\pi + i\pi/N, \quad i = 1, \ldots, 2N,$$

and approximate the L_1 norm as

$$\|f - y\|_1 \approx (\pi/N) \sum_{i=1}^{2N} |f(t_i) - y(t_i)|.$$

(A standard rule of thumb is to take N at least 10 times larger than K.) The L_1 approximation (or really, an approximation of the L_1 approximation) can now be found using linear programming.

We consider a specific case, where y is a 2π-periodic square-wave, defined for $-\pi \leq t \leq \pi$ as

$$y(t) = \begin{cases} 1 & |t| \leq \pi/2 \\ 0 & \text{otherwise.} \end{cases}$$

(The graph of y over a few cycles explains the name 'square-wave'.)

Find the optimal L_2 approximation and (discretized) L_1 optimal approximation for $K = 10$. You can find the L_2 optimal approximation analytically, or by solving a least-squares problem associated with the discretized version of the problem. Since y is even, you can take the sine coefficients in your approximations to be zero. Show y and the two approximations on a single plot.

In addition, plot a histogram of the residuals (*i.e.*, the numbers $f(t_i) - y(t_i)$) for the two approximations. Use the same horizontal axis range, so the two residual distributions can easily be compared. (Matlab command `hist` might be helpful here.) Make some brief comments about what you see.

5.4 *Penalty function approximation.* We consider the approximation problem

$$\text{minimize} \quad \phi(Ax - b)$$

where $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$, the variable is $x \in \mathbf{R}^n$, and $\phi : \mathbf{R}^m \to \mathbf{R}$ is a convex penalty function that measures the quality of the approximation $Ax \approx b$. We will consider the following choices of penalty function:

(a) *Euclidean norm.*

$$\phi(y) = \|y\|_2 = (\sum_{k=1}^{m} y_k^2)^{1/2}.$$

(b) *ℓ_1-norm.*

$$\phi(y) = \|y\|_1 = \sum_{k=1}^{m} |y_k|.$$

(c) *Sum of the largest $m/2$ absolute values.*

$$\phi(y) = \sum_{k=1}^{\lfloor m/2 \rfloor} |y|_{[k]}$$

where $|y|_{[1]}$, $|y|_{[2]}$, $|y|_{[3]}$, $\ldots$, denote the absolute values of the components of y sorted in decreasing order.

(d) *A piecewise-linear penalty.*

$$\phi(y) = \sum_{k=1}^{m} h(y_k), \qquad h(u) = \begin{cases} 0 & |u| \leq 0.2 \\ |u| - 0.2 & 0.2 \leq |u| \leq 0.3 \\ 2|u| - 0.5 & |u| \geq 0.3. \end{cases}$$

(e) *Huber penalty.*

$$\phi(y) = \sum_{k=1}^{m} h(y_k), \qquad h(u) = \begin{cases} u^2 & |u| \leq M \\ M(2|u| - M) & |u| \geq M \end{cases}$$

with $M = 0.2$.

(f) *Log-barrier penalty.*

$$\phi(y) = \sum_{k=1}^{m} h(y_k), \qquad h(u) = -\log(1 - u^2), \quad \mathbf{dom}\, h = \{u \mid |u| < 1\}.$$

Here is the problem. Generate data A and b as follows:

```
m = 200;
n = 100;
A = randn(m,n);
b = randn(m,1);
b = b/(1.01*max(abs(b)));
```

(The normalization of b ensures that the domain of $\phi(Ax-b)$ is nonempty if we use the log-barrier penalty.) To compare the results, plot a histogram of the vector of residuals $y = Ax - b$, for each of the solutions x, using the Matlab command

```
hist(A*x-b,m/2);
```

Some additional hints and remarks for the individual problems:

(a) This problem can be solved using least-squares (`x=A\b`).

(b) Use the CVX function `norm(y,1)`.

(c) Use the CVX function `norm_largest()`.

(d) Use CVX, with the overloaded `max()`, `abs()`, and `sum()` functions.

(e) Use the CVX function `huber()`.

(f) The current version of CVX handles the logarithm using an iterative procedure, which is slow and not entirely reliable. However, you can reformulate this problem as

$$\text{maximize} \quad \left(\prod_{k=1}^{m} \left((1-(Ax-b)_k)(1+(Ax-b)_k)\right)\right)^{1/2m},$$

and use the CVX function `geo_mean()`.

5.5 *$\ell_{1.5}$ optimization.* Optimization and approximation methods that use both an ℓ_2-norm (or its square) and an ℓ_1-norm are currently very popular in statistics, machine learning, and signal and image processing. Examples include Huber estimation, LASSO, basis pursuit, SVM, various ℓ_1-regularized classification methods, total variation de-noising, etc. Very roughly, an ℓ_2-norm corresponds to Euclidean distance (squared), or the negative log-likelihood function for a Gaussian; in contrast the ℓ_1-norm gives 'robust' approximation, *i.e.*, reduced sensitivity to outliers, and also tends to yield sparse solutions (of whatever the argument of the norm is). (All of this is just background; you don't need to know any of this to solve the problem.)

In this problem we study a natural method for blending the two norms, by using the $\ell_{1.5}$-norm, defined as

$$\|z\|_{1.5} = \left(\sum_{i=1}^{k} |z_i|^{3/2}\right)^{2/3}$$

for $z \in \mathbf{R}^k$. We will consider the simplest approximation or regression problem:

$$\text{minimize} \quad \|Ax-b\|_{1.5},$$

with variable $x \in \mathbf{R}^n$, and problem data $A \in \mathbf{R}^{m\times n}$ and $b \in \mathbf{R}^m$. We will assume that $m > n$ and the A is full rank (*i.e.*, rank n). The hope is that this $\ell_{1.5}$-optimal approximation problem should share some of the good features of ℓ_2 and ℓ_1 approximation.

(a) Give optimality conditions for this problem. Try to make these as simple as possible.

(b) Explain how to formulate the $\ell_{1.5}$-norm approximation problem as an SDP. (Your SDP can include linear equality and inequality constraints.)

(c) Solve the specific numerical instance generated by the following code:

```
randn('state',0);
A=randn(100,30);
b=randn(100,1);
```

Numerically verify the optimality conditions. Give a histogram of the residuals, and repeat for the ℓ_2-norm and ℓ_1-norm approximations. You can use any method you like to solve the problem (but of course you must explain how you did it); in particular, you do not need to use the SDP formulation found in part (b).

5.6 *Total variation image interpolation.* A grayscale image is represented as an $m \times n$ matrix of intensities U^{orig}. You are given the values U_{ij}^{orig}, for $(i,j) \in \mathcal{K}$, where $\mathcal{K} \subset \{1,\ldots,m\} \times \{1,\ldots,n\}$. Your job is to *interpolate* the image, by guessing the missing values. The reconstructed image will be represented by $U \in \mathbf{R}^{m \times n}$, where U satisfies the interpolation conditions $U_{ij} = U_{ij}^{\text{orig}}$ for $(i,j) \in \mathcal{K}$.

The reconstruction is found by minimizing a roughness measure subject to the interpolation conditions. One common roughness measure is the ℓ_2 variation (squared),

$$\sum_{i=2}^{m}\sum_{j=1}^{n}(U_{ij} - U_{i-1,j})^2 + \sum_{i=1}^{m}\sum_{j=2}^{n}(U_{ij} - U_{i,j-1})^2.$$

Another method minimizes instead the *total variation*,

$$\sum_{i=2}^{m}\sum_{j=1}^{n}|U_{ij} - U_{i-1,j}| + \sum_{i=1}^{m}\sum_{j=2}^{n}|U_{ij} - U_{i,j-1}|.$$

Evidently both methods lead to convex optimization problems.

Carry out ℓ_2 and total variation interpolation on the problem instance with data given in `tv_img_interp.m`. This will define `m`, `n`, and matrices `Uorig` and `Known`. The matrix `Known` is $m \times n$, with (i,j) entry one if $(i,j) \in \mathcal{K}$, and zero otherwise. The mfile also has skeleton plotting code. (We give you the entire original image so you can compare your reconstruction to the original; obviously your solution cannot access U_{ij}^{orig} for $(i,j) \notin \mathcal{K}$.)

5.7 *Piecewise-linear fitting.* In many applications some function in the model is not given by a formula, but instead as tabulated data. The tabulated data could come from empirical measurements, historical data, numerically evaluating some complex expression or solving some problem, for a set of values of the argument. For use in a convex optimization model, we then have to fit these data with a convex function that is compatible with the solver or other system that we use. In this problem we explore a very simple problem of this general type.

Suppose we are given the data (x_i, y_i), $i = 1,\ldots,m$, with $x_i,\ y_i \in \mathbf{R}$. We will assume that x_i are sorted, *i.e.*, $x_1 < x_2 < \cdots < x_m$. Let $a_0 < a_1 < a_2 < \cdots < a_K$ be a set of fixed knot points, with $a_0 \leq x_1$ and $a_K \geq x_m$. Explain how to find the convex piecewise linear function f, defined over $[a_0, a_K]$, with knot points a_i, that minimizes the least-squares fitting criterion

$$\sum_{i=1}^{m}(f(x_i) - y_i)^2.$$

You must explain what the variables are and how they parametrize f, and how you ensure convexity of f.

Hints. One method to solve this problem is based on the Lagrange basis, $f_0, \ldots, f_K$, which are the piecewise linear functions that satisfy

$$f_j(a_i) = \delta_{ij}, \quad i, j = 0, \ldots, K.$$

Another method is based on defining $f(x) = \alpha_i x + \beta_i$, for $x \in (a_{i-1}, a_i]$. You then have to add conditions on the parameters α_i and β_i to ensure that f is continuous and convex.

Apply your method to the data in the file `pwl_fit_data.m`, which contains data with $x_j \in [0, 1]$. Find the best affine fit (which corresponds to $a = (0, 1)$), and the best piecewise-linear convex function fit for 1, 2, and 3 internal knot points, evenly spaced in $[0, 1]$. (For example, for 3 internal knot points we have $a_0 = 0$, $a_1 = 0.25$, $a_2 = 0.50$, $a_3 = 0.75$, $a_4 = 1$.) Give the least-squares fitting cost for each one. Plot the data and the piecewise-linear fits found. Express each function in the form

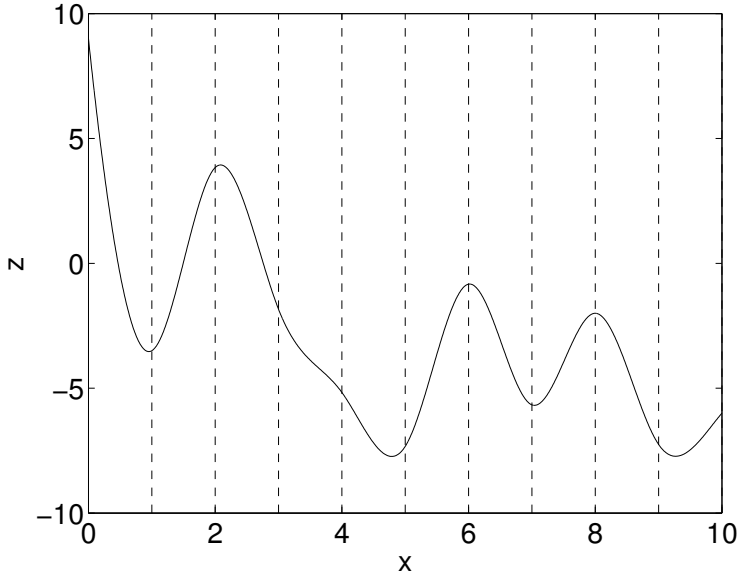
$$f(x) = \max_{i=1\ldots,K} (\alpha_i x + \beta_i).$$

(In this form the function is easily incorporated into an optimization problem.)

5.8 *Least-squares fitting with convex splines.* A *cubic spline* (or *fourth-order spline*) with breakpoints α_0, α_1, ..., α_M (that satisfy $\alpha_0 < \alpha_1 < \cdots < \alpha_M$) is a piecewise-polynomial function with the following properties:

- the function is a cubic polynomial on each interval $[\alpha_i, \alpha_{i+1}]$
- the function values, and the first and second derivatives are continuous on the interval (α_0, α_M).

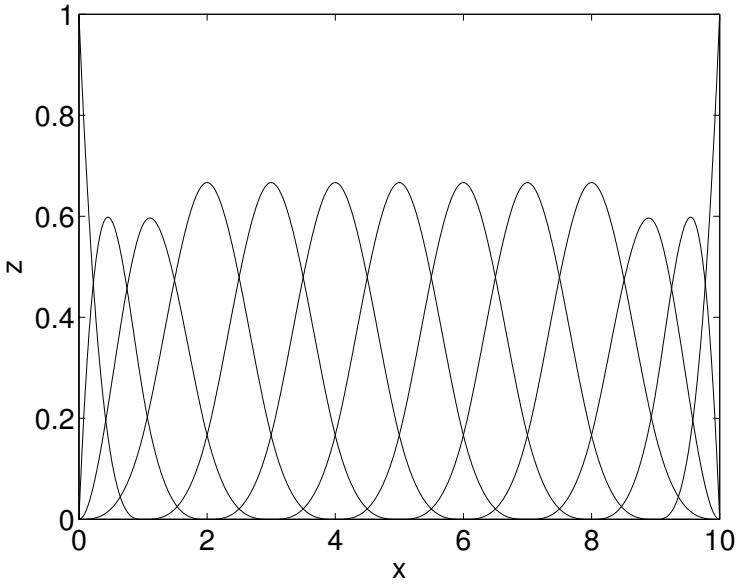
The figure shows an example of a cubic spline $f(t)$ with $M = 10$ segments and breakpoints $\alpha_0 = 0$, $\alpha_1 = 1$, ..., $\alpha_{10} = 10$.



In approximation problems with splines it is convenient to parametrize a spline as a linear combination of basis functions, called *B-splines*. The precise definition of B-splines is not important for our purposes; it is sufficient to know that every cubic spline can be written as a linear combination of $M + 3$ cubic B-splines $g_k(t)$, *i.e.*, in the form

$$f(t) = x_1 g_1(t) + \cdots + x_{M+3} g_{M+3}(t) = x^T g(t),$$

and that there exist efficient algorithms for computing $g(t) = (g_1(t), \ldots, g_{M+3}(t))$. The next figure shows the 13 B-splines for the breakpoints 0, 1, ..., 10.



In this exercise we study the problem of fitting a cubic spline to a set of data points, subject to the constraint that the spline is a convex function. Specifically, the breakpoints $\alpha_0, \ldots, \alpha_M$ are fixed, and we are given N data points (t_k, y_k) with $t_k \in [\alpha_0, \alpha_M]$. We are asked to find the convex cubic spline $f(t)$ that minimizes the least-squares criterion

$$\sum_{k=1}^{N} (f(t_k) - y_k)^2.$$

We will use B-splines to parametrize f, so the variables in the problem are the coefficients x in $f(t) = x^T g(t)$. The problem can then be written as

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{k=1}^{N} \left(x^T g(t_k) - y_k \right)^2 \\ \text{subject to} & x^T g(t) \text{ is convex in } t \text{ on } [\alpha_0, \alpha_M]. \end{array} \tag{26}$$

(a) Express problem (26) as a convex optimization problem of the form

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2^2 \\ \text{subject to} & Gx \preceq h. \end{array}$$

(b) Use CVX to solve a specific instance of the optimization problem in part (a). As in the figures above, we take $M = 10$ and $\alpha_0 = 0$, $\alpha_1 = 1$, ..., $\alpha_{10} = 10$.

Download the Matlab files `spline_data.m` and `bsplines.m`. The first m-file is used to generate the problem data. The command `[t, y] = spline_data` will generate two vectors t, y of length $N = 51$, with the data points t_k, y_k.

The second function can be used to compute the B-splines, and their first and second derivatives, at any given point $u \in [0, 10]$. The command `[g, gp, gpp] = bsplines(u)` returns three vectors of length 13 with elements $g_k(u)$, $g_k'(u)$, and $g_k''(u)$. (The right derivatives are returned for $u = 0$, and the left derivatives for $u = 10$.)

Solve the convex spline fitting problem (26) for this example, and plot the optimal spline.

5.9 *Robust least-squares with interval coefficient matrix.* An *interval matrix* in $\mathbf{R}^{m\times n}$ is a matrix whose entries are intervals:

$$\mathcal{A} = \{A \in \mathbf{R}^{m\times n} \mid |A_{ij} - \bar{A}_{ij}| \leq R_{ij},\ i = 1, \ldots, m,\ j = 1, \ldots, n\}.$$

The matrix $\bar{A} \in \mathbf{R}^{m\times n}$ is called the *nominal value* or *center value*, and $R \in \mathbf{R}^{m\times n}$, which is elementwise nonnegative, is called the *radius*.

The robust least-squares problem, with interval matrix, is

$$\text{minimize} \quad \sup_{A\in\mathcal{A}} \|Ax - b\|_2,$$

with optimization variable $x \in \mathbf{R}^n$. The problem data are $\mathcal{A}$ (*i.e.*, $\bar{A}$ and R) and $b \in \mathbf{R}^m$. The objective, as a function of x, is called the *worst-case residual norm*. The robust least-squares problem is evidently a convex optimization problem.

(a) Formulate the interval matrix robust least-squares problem as a standard optimization problem, *e.g.*, a QP, SOCP, or SDP. You can introduce new variables if needed. Your reformulation should have a number of variables and constraints that grows linearly with m and n, and not exponentially.

(b) Consider the specific problem instance with $m = 4$, $n = 3$,

$$\mathcal{A} = \begin{bmatrix} 60 \pm 0.05 & 45 \pm 0.05 & -8 \pm 0.05 \\ 90 \pm 0.05 & 30 \pm 0.05 & -30 \pm 0.05 \\ 0 \pm 0.05 & -8 \pm 0.05 & -4 \pm 0.05 \\ 30 \pm 0.05 & 10 \pm 0.05 & -10 \pm 0.05 \end{bmatrix}, \qquad b = \begin{bmatrix} -6 \\ -3 \\ 18 \\ -9 \end{bmatrix}.$$

(The first part of each entry in $\mathcal{A}$ gives $\bar{A}_{ij}$; the second gives R_{ij}, which are all 0.05 here.) Find the solution x_{ls} of the nominal problem (*i.e.*, minimize $\|\bar{A}x - b\|_2$), and robust least-squares solution x_{rls}. For each of these, find the nominal residual norm, and also the worst-case residual norm. Make sure the results make sense.

5.10 *Identifying a sparse linear dynamical system.* A linear dynamical system has the form

$$x(t + 1) = Ax(t) + Bu(t) + w(t), \quad t = 1, \ldots, T - 1,$$

where $x(t) \in \mathbf{R}^n$ is the state, $u(t) \in \mathbf{R}^m$ is the input signal, and $w(t) \in \mathbf{R}^n$ is the process noise, at time t. We assume the process noises are IID $\mathcal{N}(0, W)$, where $W \succ 0$ is the covariance matrix.

The matrix $A \in \mathbf{R}^{n\times n}$ is called the dynamics matrix or the state transition matrix, and the matrix $B \in \mathbf{R}^{n\times m}$ is called the input matrix.

You are given accurate measurements of the state and input signal, *i.e.*, $x(1), \ldots, x(T)$, $u(1), \ldots, u(T-1)$, and W is known. Your job is to find a state transition matrix $\hat{A}$ and input matrix $\hat{B}$ from these data, that are plausible, and in addition are sparse, *i.e.*, have many zero entries. (The sparser the better.)

By doing this, you are effectively estimating the structure of the dynamical system, *i.e.*, you are determining which components of $x(t)$ and $u(t)$ affect which components of $x(t+1)$. In some applications, this structure might be more interesting than the actual values of the (nonzero) coefficients in $\hat{A}$ and $\hat{B}$.

By plausible, we mean that

$$\sum_{t=1}^{T-1} \left\| W^{-1/2} \left(x(t+1) - \hat{A}x(t) - \hat{B}u(t) \right) \right\|_2^2 \leq n(T-1) + 2\sqrt{2n(T-1)}.$$

(You can just take this as our definition of plausible. But to explain this choice, we note that when $\hat{A} = A$ and $\hat{B} = B$, the left-hand side is χ^2, with $n(T-1)$ degrees of freedom, and so has mean $n(T-1)$ and standard deviation $\sqrt{2n(T-1)}$. Thus, the constraint above states that the LHS does not exceed the mean by more than 2 standard deviations.)

(a) Describe a method for finding $\hat{A}$ and $\hat{B}$, based on convex optimization.

We are looking for a *very simple* method, that involves solving *one* convex optimization problem. (There are many extensions of this basic method, that would improve the simple method, *i.e.*, yield sparser $\hat{A}$ and $\hat{B}$ that are still plausible. We're not asking you to describe or implement any of these.)

(b) Carry out your method on the data found in `sparse_lds_data.m`. Give the values of $\hat{A}$ and $\hat{B}$ that you find, and verify that they are plausible.

In the data file, we give you the true values of A and B, so you can evaluate the performance of your method. (Needless to say, you are not allowed to use these values when forming $\hat{A}$ and $\hat{B}$.) Using these true values, give the number of false positives and false negatives in both $\hat{A}$ and $\hat{B}$. A false positive in $\hat{A}$, for example, is an entry that is nonzero, while the corresponding entry in A is zero. A false negative is an entry of $\hat{A}$ that is zero, while the corresponding entry of A is nonzero. To judge whether an entry of $\hat{A}$ (or $\hat{B}$) is nonzero, you can use the test $|\hat{A}_{ij}| \geq 0.01$ (or $|\hat{B}_{ij}| \geq 0.01$).

5.11 *Measurement with bounded errors.* A series of K measurements $y_1, \ldots, y_K \in \mathbf{R}^p$, are taken in order to estimate an unknown vector $x \in \mathbf{R}^q$. The measurements are related to the unknown vector x by $y_i = Ax + v_i$, where v_i is a measurement noise that satisfies $\|v_i\|_\infty \leq \alpha$ but is otherwise unknown. (In other words, the entries of $v_1, \ldots, v_K$ are no larger than α.) The matrix A and the measurement noise norm bound α are known. Let X denote the set of vectors x that are consistent with the observations $y_1, \ldots, y_K$, *i.e.*, the set of x that could have resulted in the measurements made. Is X convex?

Now we will examine what happens when the measurements are occasionally in error, *i.e.*, for a few i we have no relation between x and y_i. More precisely suppose that I_{fault} is a subset of $\{1, \ldots, K\}$,

and that $y_i = Ax + v_i$ with $\|v_i\|_\infty \leq \alpha$ (as above) for $i \notin I_{\text{fault}}$, but for $i \in I_{\text{fault}}$, there is no relation between x and y_i. The set I_{fault} is the set of times of the faulty measurements.

Suppose you know that I_{fault} has at most J elements, *i.e.*, out of K measurements, at most J are faulty. You do not know I_{fault}; you know only a bound on its cardinality (size). For what values of J is X, the set of x consistent with the measurements, convex?

5.12 *Least-squares with some permuted measurements.* We want to estimate a vector $x \in \mathbf{R}^n$, given some linear measurements of x corrupted with Gaussian noise. Here's the catch: some of the measurements have been *permuted*.

More precisely, our measurement vector $y \in \mathbf{R}^m$ has the form

$$y = P(Ax + v),$$

where v_i are IID $\mathcal{N}(0,1)$ measurement noises, $x \in \mathbf{R}^n$ is the vector of parameters we wish to estimate, and $P \in \mathbf{R}^{m\times m}$ is a permutation matrix. (This means that each row and column of P has exactly one entry equal to one, and the remaining $m-1$ entries zero.) We assume that $m > n$ and that at most k of the measurements are permuted; *i.e.*, $Pe_i \neq e_i$ for no more than k indices i. We are interested in the case when $k < m$ (*e.g.* $k = 0.4m$); that is, only *some* of the measurements have been permuted. We want to estimate x and P.

Once we make a guess $\hat{P}$ for P, we can get the maximum likelihood estimate of x by minimizing $\|Ax - \hat{P}^T y\|_2$. The residual $A\hat{x} - \hat{P}^T y$ is then our guess of what v is, and should be consistent with being a sample of a $\mathcal{N}(0, I)$ vector.

In principle, we can find the maximum likelihood estimate of x and P by solving a set of $\binom{m}{k}(k!-1)$ least-squares problems, and choosing one that has minimum residual. But this is not practical unless m and k are both very small.

Describe a *heuristic* method for approximately solving this problem, using convex optimization. (There are many different approaches which work quite well.)

You might find the following fact useful. The solution to

$$\text{minimize} \quad \|Ax - P^T y\|$$

over $P \in \mathbf{R}^{m\times m}$ a permutation matrix, is the permutation that matches the smallest entry in y with the smallest entry in Ax, does the same for the second smallest entries and so forth.

Carry out your method on the data in `ls_perm_meas_data.*`. Give your estimate of the permuted indices. The data file includes the true permutation matrix and value of x (which of course you cannot use in forming your estimate). Compare the estimate of x you get after your guessed permutation with the estimate obtained assuming $P = I$.

Remark. This problem comes up in several applications. In target tracking, we get multiple noisy measurements of a set of targets, and then guess which targets are the same in the different sets of measurements. If some of our guesses are wrong (*i.e.*, our target association is wrong) we have the present problem. In vision systems the problem arises when we have multiple camera views of a scene, which give us noisy measurements of a set of features. A feature correspondence algorithm guesses which features in one view correspond to features in other views. If we make some feature correspondence errors, we have the present problem.

Note. If you are using Julia, you might have to set the solver to run more than the default number of iterations, using `solve!(problem, SCSSolver(max_iters=10000))`.

5.13 *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.*`. Report $\hat{c}$, the value of c found using this method.

Also find $\hat{c}_{\mathrm{ls}}$, 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_{true}, the true value of c, in the vector `c_true`. Use this to give the two relative errors

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

5.14 *Spectrum analysis with quantized measurements.* A sample is made up of n compounds, in quantities $q_i \geq 0$, for $i = 1, \ldots, n$. Each compound has a (nonnegative) spectrum, which we represent as a vector $s^{(i)} \in \mathbf{R}^m_+$, for $i = 1, \ldots, n$. (Precisely what $s^{(i)}$ means won't matter to us.) The spectrum of the sample is given by $s = \sum_{i=1}^n q_i s^{(i)}$. We can write this more compactly as $s = Sq$, where $S \in \mathbf{R}^{m \times n}$ is a matrix whose columns are $s^{(1)}, \ldots, s^{(n)}$.

Measurement of the spectrum of the sample gives us an interval for each spectrum value, *i.e.*, $l, u \in \mathbf{R}^m_+$ for which

$$l_i \leq s_i \leq u_i, \quad i = 1, \ldots, m.$$

(We don't directly get s.) This occurs, for example, if our measurements are quantized.

Given l and u (and S), we cannot in general deduce q exactly. Instead, we ask you to do the following. For each compound i, find the range of possible values for q_i consistent with the spectrum measurements. We will denote these ranges as $q_i \in [q_i^{\min}, q_i^{\max}]$. Your job is to find $q_i^{\min}$ and $q_i^{\max}$.

Note that if $q_i^{\min}$ is large, we can confidently conclude that there is a significant amount of compound i in the sample. If $q_i^{\max}$ is small, we can confidently conclude that there is not much of compound i in the sample.

(a) Explain how to find $q_i^{\min}$ and $q_i^{\max}$, given S, l, and u.

(b) Carry out the method of part (a) for the problem instance given in `spectrum_data.m`. (Executing this file defines the problem data, and plots the compound spectra and measurement bounds.) Plot the minimum and maximum values versus i, using the commented out code in the data file. Report your values for $q_4^{\min}$ and $q_4^{\max}$.

5.15 *Learning a quadratic pseudo-metric from distance measurements.* We are given a set of N pairs of points in $\mathbf{R}^n$, $x_1, \ldots, x_N$, and $y_1, \ldots, y_N$, together with a set of distances $d_1, \ldots, d_N > 0$.

The goal is to find (or estimate or learn) a quadratic pseudo-metric d,

$$d(x, y) = \Big((x - y)^T P (x - y)\Big)^{1/2},$$

with $P \in \mathbf{S}_+^n$, which approximates the given distances, *i.e.*, $d(x_i, y_i) \approx d_i$. (The pseudo-metric d is a metric only when $P \succ 0$; when $P \succeq 0$ is singular, it is a pseudo-metric.)

To do this, we will choose $P \in \mathbf{S}_+^n$ that minimizes the mean squared error objective

$$\frac{1}{N} \sum_{i=1}^{N} (d_i - d(x_i, y_i))^2.$$

(a) Explain how to find P using convex or quasiconvex optimization. If you cannot find an exact formulation (*i.e.*, one that is guaranteed to minimize the total squared error objective), give a formulation that approximately minimizes the given objective, subject to the constraints.

(b) Carry out the method of part (a) with the data given in `quad_metric_data.m`. The columns of the matrices `X` and `Y` are the points x_i and y_i; the row vector `d` gives the distances d_i. Give the optimal mean squared distance error.

We also provide a test set, with data `X_test`, `Y_test`, and `d_test`. Report the mean squared distance error on the test set (using the metric found using the data set above).

5.16 *Polynomial approximation of inverse using eigenvalue information.* We seek a polynomial of degree k, $p(a) = c_0 + c_1 a + c_2 a^2 + \cdots + c_k a^k$, for which

$$p(A) = c_0 I + c_1 A + c_2 A^2 \cdots + c_k A^k$$

is an approximate inverse of the nonsingular matrix A, for all $A \in \mathcal{A} \subset \mathbf{R}^{n \times n}$. When $\hat{x} = p(A)b$ is used as an approximate solution of the linear equation $Ax = b$, the associated residual norm is $\|A(p(A)b) - b\|_2$. We will judge our polynomial (*i.e.*, the coefficients $c_0, \ldots, c_k$) by the worst case residual over $A \in \mathcal{A}$ and b in the unit ball:

$$R^{\mathrm{wc}} = \sup_{A \in \mathcal{A},\ \|b\|_2 \leq 1} \|A(p(A)b) - b\|_2.$$

The set of matrices we take is $\mathcal{A} = \{A \in \mathbf{S}^n \mid \sigma(A) \subseteq \Omega\}$, where $\sigma(A)$ is the set of eigenvalues of A (*i.e.*, its spectrum), and $\Omega \subset \mathbf{R}$ is a union of a set of intervals (that do not contain 0).

(a) Explain how to find coefficients $c_0^\star, \ldots, c_k^\star$ that minimize R^{wc}. Your solution can involve expressions that involve the supremum of a polynomial (with scalar argument) over an interval.

(b) Carry out your method for $k = 4$ and $\Omega = [-0.6, -0.3] \cup [0.7, 1.8]$. You can replace the supremum of a polynomial over Ω by a maximum over uniformly spaced (within each interval) points in Ω, with spacing 0.01. Give the optimal value $R^{\mathrm{wc}\star}$ and the optimal coefficients $c^\star = (c_0^\star, \ldots, c_k^\star)$.

Remarks. (Not needed to solve the problem.)

- The approximate inverse $p(A)b$ would be computed by recursively, requiring the multiplication of A with a vector k times.
- This approximate inverse could be used as a preconditioner for an iterative method.
- The Cayley-Hamilton theorem tells us that the inverse of any (invertible) matrix is a polynomial of degree $n-1$ of the matrix. Our hope here, however, is to get a single polynomial, of relatively low degree, that serves as an approximate inverse for many different matrices.

5.17 *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.

5.18 *Multi-label support vector machine.* The basic SVM described in the book is used for classification of data with two labels. In this problem we explore an extension of SVM that can be used to carry out classification of data with more than two labels. Our data consists of pairs $(x_i, y_i) \in \mathbf{R}^n \times \{1, \ldots, K\}$, $i = 1, \ldots, m$, where x_i is the feature vector and y_i is the label of the ith data point. (So the labels can take the values $1, \ldots, K$.) Our classifier will use K affine functions, $f_k(x) = a_k^T x + b_k$, $k = 1, \ldots, K$, which we also collect into affine function from $\mathbf{R}^n$ into $\mathbf{R}^K$ as $f(x) = Ax + b$. (The rows of A are a_k^T.) Given feature vector x, we guess the label $\hat{y} = \operatorname{argmax}_k f_k(x)$. We assume that exact ties never occur, or if they do, an arbitrary choice can be made. Note that if a multiple of $\mathbf{1}$ is added to b, the classifier does not change. Thus, without loss of generality, we can assume that $\mathbf{1}^T b = 0$.

To correctly classify the data examples, we need $f_{y_i}(x_i) > \max_{k \neq y_i} f_k(x_i)$ for all i. This is a set of homogeneous strict inequalities in a_k and b_k, which are feasible if and only if the set of nonstrict inequalities $f_{y_i}(x_i) \geq 1 + \max_{k \neq y_i} f_k(x_i)$ are feasible. This motivates the loss function

$$L(A, b) = \sum_{i=1}^{m} \left(1 + \max_{k \neq y_i} f_k(x_i) - f_{y_i}(x_i)\right)_+,$$

where $(u)_+ = \max\{u, 0\}$. The multi-label SVM chooses A and b to minimize

$$L(A, b) + \mu \|A\|_F^2,$$

subject to $\mathbf{1}^T b = 0$, where $\mu > 0$ is a regularization parameter. (Several variations on this are possible, such as regularizing b as well, or replacing the Frobenius norm squared with the sum of norms of the columns of A.)

(a) Show how to find A and b using convex optimization. Be sure to justify any changes of variables or reformulation (if needed), and convexity of the objective and constraints in your formulation.

(b) Carry out multi-label SVM on the data given in `multi_label_svm_data.m`. Use the data given in `X` and `y` to fit the SVM model, for a range of values of μ. This data set includes an additional set of data, `Xtest` and `ytest`, that you can use to test the SVM models. Plot the test set classification error rate (*i.e.*, the fraction of data examples in the test set for which $\hat{y} \neq y$) versus μ.

You don't need to try more than 10 or 20 values of μ, and we suggest choosing them uniformly on a log scale, from (say) 10^{-2} to 10^2.

5.19 *Colorization with total variation regularization.* A $m \times n$ color image is represented as three matrices of intensities $R, G, B \in \mathbf{R}^{m \times n}$, with entries in $[0, 1]$, representing the red, green, and blue pixel intensities, respectively. A color image is converted to a monochrome image, represented as one matrix $M \in \mathbf{R}^{m \times n}$, using

$$M = 0.299R + 0.587G + 0.114B.$$

(These weights come from different perceived brightness of the three primary colors.)

In *colorization*, we are given M, the monochrome version of an image, and the color values of *some* of the pixels; we are to guess its color version, *i.e.*, the matrices R, G, B. Of course that's a very underdetermined problem. A very simple technique is to minimize the total variation of (R, G, B), defined as

$$\mathbf{tv}(R, G, B) = \sum_{i=1}^{m-1} \sum_{j=1}^{n-1} \left\| \begin{bmatrix} R_{ij} - R_{i,j+1} \\ G_{ij} - G_{i,j+1} \\ B_{ij} - B_{i,j+1} \\ R_{ij} - R_{i+1,j} \\ G_{ij} - G_{i+1,j} \\ B_{ij} - B_{i+1,j} \end{bmatrix} \right\|_2,$$

subject to consistency with the given monochrome image, the known ranges of the entries of (R, G, B) (*i.e.*, in $[0, 1]$), and the given color entries. Note that the sum above is of the norm of 6-vectors, and not the norm-squared. (The 6-vector is an approximation of the spatial gradient of (R, G, B).)

Carry out this method on the data given in `image_colorization_data.*`. The file loads `flower.png` and provides the monochrome version of the image, `M`, along with vectors of known color intensities, `R_known`, `G_known`, and `B_known`, and `known_ind`, the indices of the pixels with known values. If `R` denotes the red channel of an image, then `R(known_ind)` returns the known red color intensities in Matlab, and `R[known_ind]` returns the same in Python and Julia. The file also creates an image, `flower_given.png`, that is monochrome, with the known pixels colored.

The `tv` function, invoked as `tv(R,G,B)`, gives the total variation. CVXPY has the `tv` function built-in, but CVX and CVX.jl do not, so we have provided the files `tv.m` and `tv.jl` which contain implementations for you to use.

In Python and Julia we have also provided the function `save_img(filename,R,G,B)` which writes the image defined by the matrices `R`, `G`, `B`, to the file `filename`. To view an image in Matlab use the `imshow` function.

The problem instance is a small image, 75×75, so the solve time is reasonable, say, under ten seconds or so in CVX or CVXPY, and around 60 seconds in Julia.

Report your optimal objective value and, if you have access to a color printer, attach your reconstructed image. If you don't have access to a color printer, it's OK to just give the optimal objective value.

5.20 *Recovering latent periodic signals.* First, a definition: a signal $x \in \mathbf{R}^n$ is *p-periodic* with $p < n$ if $x_{i+p} = x_i$ for $i = 1, \ldots, n - p$.

In this problem, we consider a noisy, measured signal $y \in \mathbf{R}^n$ which is (approximately) the sum of a several periodic signals, with unknown periods. Given only the noisy signal y, our task is to recover these latent periodic signals. In particular, y is given as

$$y = v + \sum_{p \in \mathcal{P}} x^{(p)},$$

where $v \in \mathbf{R}^n$ is a (small) random noise term, and $x^{(p)}$ is a p-periodic signal. The set $\mathcal{P} \subset \{1, \ldots, p_{\max}\}$ contains the periods of the latent periodic signals that compose y.

If $\mathcal{P}$ were known, we could approximately recover the latent periodic signals $x^{(p)}$ using, say, least squares. Because $\mathcal{P}$ is *not* known, we instead propose to recover the latent periodic signals $x^{(p)}$ by solving the following optimization problem:

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \sum_{p=1}^{p_{\max}} w_p \|\hat{x}^{(p)}\|_2 \\ \text{subject to} & \hat{y} = \sum_{p=1}^{p_{\max}} \hat{x}^{(p)} \\ & \hat{x}^{(p)} \text{ is } p\text{-periodic, for } p = 1, \ldots, p_{\max}. \end{array}$$

The variables are $\hat{y}$ and $\hat{x}^{(p)}$, for $p = 1, \ldots, p_{\max}$. The first sum in the objective penalizes the squared deviation of the measured signal y from our estimate $\hat{y}$, and the second sum is a heuristic for producing vectors $\hat{x}^{(p)}$ that contain only zeros. The weight vector $w \succeq 0$ is increasing in its indices, which encodes our desire that the latent periodic signals have small period.

(a) Explain how to solve the given optimization problem using convex optimization, and how to use it to (approximately) recover the set $\mathcal{P}$ and the latent periodic signals $x^{(p)}$, for $p \in \mathcal{P}$.

(b) The file `periodic_signals_data.*` contains a signal y, as well as a weight vector w. Return your best guess of the set $\mathcal{P}$. plot the measured signal y, as well as the different periodic components that (approximately) compose it. (Use separate graphs for each signal, so you should have $|\mathcal{P}| + 1$ graphs.)

6 Statistical estimation

6.1 *Maximum likelihood estimation of x and noise mean and covariance.* Consider the maximum likelihood estimation problem with the linear measurement model

$$y_i = a_i^T x + v_i, \quad i = 1, \ldots, m.$$

The vector $x \in \mathbf{R}^n$ is a vector of unknown parameters, y_i are the measurement values, and v_i are independent and identically distributed measurement errors.

In this problem we make the assumption that the *normalized* probability density function of the errors is given (normalized to have zero mean and unit variance), but not their mean and variance. In other words, the density of the measurement errors v_i is

$$p(z) = \frac{1}{\sigma} f(\frac{z-\mu}{\sigma}),$$

where f is a given, normalized density. The parameters μ and σ are the mean and standard deviation of the distribution p, and are not known.

The maximum likelihood estimates of x, μ, σ are the maximizers of the log-likelihood function

$$\sum_{i=1}^m \log p(y_i - a_i^T x) = -m \log \sigma + \sum_{i=1}^m \log f(\frac{y_i - a_i^T x - \mu}{\sigma}),$$

where y is the observed value. Show that if f is log-concave, then the maximum likelihood estimates of x, μ, σ can be determined by solving a convex optimization problem.

6.2 *Mean and covariance estimation with conditional independence constraints.* Let $X \in \mathbf{R}^n$ be a Gaussian random variable with density

$$p(x) = \frac{1}{(2\pi)^{n/2}(\det S)^{1/2}} \exp(-(x-a)^T S^{-1}(x-a)/2).$$

The conditional density of a subvector $(X_i, X_j) \in \mathbf{R}^2$ of X, given the remaining variables, is also Gaussian, and its covariance matrix R_{ij} is equal to the Schur complement of the 2×2 submatrix

$$\begin{bmatrix} S_{ii} & S_{ij} \\ S_{ij} & S_{jj} \end{bmatrix}$$

in the covariance matrix S. The variables X_i, X_j are called *conditionally independent* if the covariance matrix R_{ij} of their conditional distribution is diagonal.

Formulate the following problem as a convex optimization problem. We are given N independent samples $y_1, \ldots, y_N \in \mathbf{R}^n$ of X. We are also given a list $\mathcal{N} \in \{1, \ldots, n\} \times \{1, \ldots, n\}$ of pairs of conditionally independent variables: $(i, j) \in \mathcal{N}$ means X_i and X_j are conditionally independent. The problem is to compute the maximum likelihood estimate of the mean a and the covariance matrix S, subject to the constraint that X_i and X_j are conditionally independent for $(i, j) \in \mathcal{N}$.

6.3 *Maximum likelihood estimation for exponential family.* A probability distribution or density on a set $\mathcal{D}$, parametrized by $\theta \in \mathbf{R}^n$, is called an *exponential family* if it has the form

$$p_\theta(x) = a(\theta) \exp(\theta^T c(x)),$$

for $x \in \mathcal{D}$, where $c : \mathcal{D} \to \mathbf{R}^n$, and $a(\theta)$ is a normalizing function. Here we intepret $p_\theta(x)$ as a density function when $\mathcal{D}$ is a continuous set, and a probability distribution when $\mathcal{D}$ is discrete. Thus we have

$$a(\theta) = \left(\int_{\mathcal{D}} \exp(\theta^T c(x))\, dx \right)^{-1}$$

when p_θ is a density, and

$$a(\theta) = \left(\sum_{x \in \mathcal{D}} \exp(\theta^T c(x)) \right)^{-1}$$

when p_θ represents a distribution. We consider only values of θ for which the integral or sum above is finite. Many families of distributions have this form, for appropriate choice of the parameter θ and function c.

(a) When $c(x) = x$ and $\mathcal{D} = \mathbf{R}^n_+$, what is the associated family of densities? What is the set of valid values of θ?

(b) Consider the case with $\mathcal{D} = \{0, 1\}$, with $c(0) = 0$, $c(1) = 1$. What is the associated exponential family of distributions? What are the valid values of the parameter $\theta \in \mathbf{R}$?

(c) Explain how to represent the normal family $\mathcal{N}(\mu, \Sigma)$ as an exponential family. *Hint.* Use parameter $(z, Y) = (\Sigma^{-1}\mu, \Sigma^{-1})$. With this parameter, $\theta^T c(x)$ has the form $z^T c_1(x) + \mathbf{tr}\, Y C_2(x)$, where $C_2(x) \in \mathbf{S}^n$.

(d) *Log-likelihood function.* Show that for any $x \in \mathcal{D}$, the log-likelihood function $\log p_\theta(x)$ is concave in θ. This means that maximum-likelihood estimation for an exponential family leads to a convex optimization problem. You don't have to give a formal proof of concavity of $\log p_\theta(x)$ in the general case: You can just consider the case when $\mathcal{D}$ is finite, and state that the other cases (discrete but infinite $\mathcal{D}$, continuous $\mathcal{D}$) can be handled by taking limits of finite sums.

(e) *Optimality condition for ML estimation.* Let $\ell_\theta(x_1, \ldots, x_K)$ be the log-likelihood function for K IID samples, $x_1, \ldots, x_k$, from the distribution or density p_θ. Assuming $\log p_\theta$ is differentiable in θ, show that

$$(1/K)\nabla_\theta \ell_\theta(x_1, \ldots, x_K) = \frac{1}{K} \sum_{i=1}^{K} c(x_i) - \mathop{\mathbf{E}}_{\theta} c(x).$$

(The subscript under $\mathbf{E}$ means the expectation under the distribution or density p_θ.)

Interpretation. The ML estimate of θ is characterized by the empirical mean of $c(x)$ being equal to the expected value of $c(x)$, under the density or distribution p_θ. (We assume here that the maximizer of ℓ is characterized by the gradient vanishing.)

6.4 *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.

6.5 *Estimating a vector with unknown measurement nonlinearity.* (A specific instance of exercise 7.9 in *Convex Optimization*.) We want to estimate a vector $x \in \mathbf{R}^n$, given some measurements

$$y_i = \phi(a_i^T x + v_i), \quad i = 1, \ldots, m.$$

Here $a_i \in \mathbf{R}^n$ are known, v_i are IID $\mathcal{N}(0, \sigma^2)$ random noises, and $\phi : \mathbf{R} \to \mathbf{R}$ is an unknown monotonic increasing function, known to satisfy

$$\alpha \leq \phi'(u) \leq \beta,$$

for all u. (Here α and β are known positive constants, with $\alpha < \beta$.) We want to find a maximum likelihood estimate of x and ϕ, given y_i. (We also know a_i, σ, α, and β.)

This sounds like an infinite-dimensional problem, since one of the parameters we are estimating is a function. In fact, we only need to know the m numbers $z_i = \phi^{-1}(y_i)$, $i = 1, \ldots, m$. So by estimating ϕ we really mean estimating the m numbers $z_1, \ldots, z_m$. (These numbers are not arbitrary; they must be consistent with the prior information $\alpha \leq \phi'(u) \leq \beta$ for all u.)

(a) Explain how to find a maximum likelihood estimate of x and ϕ (*i.e.*, $z_1, \ldots, z_m$) using convex optimization.

(b) Carry out your method on the data given in `nonlin_meas_data.m`, which includes a matrix $A \in \mathbf{R}^{m \times n}$, with rows $a_1^T, \ldots, a_m^T$. Give $\hat{x}_{\mathrm{ml}}$, the maximum likelihood estimate of x. Plot your estimated function $\hat{\phi}_{\mathrm{ml}}$. (You can do this by plotting $(\hat{z}_{\mathrm{ml}})_i$ versus y_i, with y_i on the vertical axis and $(\hat{z}_{\mathrm{ml}})_i$ on the horizontal axis.)

Hint. You can assume the measurements are numbered so that y_i are sorted in nondecreasing order, *i.e.*, $y_1 \leq y_2 \leq \cdots \leq y_m$. (The data given in the problem instance for part (b) is given in this order.)

6.6 *Maximum likelihood estimation of an increasing nonnegative signal.* We wish to estimate a scalar signal $x(t)$, for $t = 1, 2, \ldots, N$, which is known to be nonnegative and monotonically nondecreasing:

$$0 \leq x(1) \leq x(2) \leq \cdots \leq x(N).$$

This occurs in many practical problems. For example, $x(t)$ might be a measure of wear or deterioration, that can only get worse, or stay the same, as time t increases. We are also given that $x(t) = 0$ for $t \leq 0$.

We are given a noise-corrupted moving average of x, given by

$$y(t) = \sum_{\tau=1}^{k} h(\tau) x(t-\tau) + v(t), \quad t = 2, \ldots, N+1,$$

where $v(t)$ are independent $\mathcal{N}(0,1)$ random variables.

(a) Show how to formulate the problem of finding the maximum likelihood estimate of x, given y, taking into account the prior assumption that x is nonnegative and monotonically nondecreasing, as a convex optimization problem. Be sure to indicate what the problem variables are, and what the problem data are.

(b) We now consider a specific instance of the problem, with problem data (*i.e.*, N, k, h, and y) given in the file `ml_estim_incr_signal_data.*`. (This file contains the true signal `xtrue`, which of course you cannot use in creating your estimate.) Find the maximum likelihood estimate $\hat{x}_{\mathrm{ml}}$, and plot it, along with the true signal. Also find and plot the maximum likelihood estimate $\hat{x}_{\mathrm{ml,free}}$ *not taking into account the signal nonnegativity and monotonicity.*
Hints.

- Matlab: The function `conv` (convolution) is overloaded to work with CVX.
- Python: Numpy has a function `convolve` which performs convolution. CVXPY has `conv` which does the same thing for variables.
- Julia: The function `conv` is overloaded to work with Convex.jl.

6.7 *Relaxed and discrete A-optimal experiment design.* This problem concerns the A-optimal experiment design problem, described on page 387, with data generated as follows.

```
n = 5; % dimension of parameters to be estimated
p = 20; % number of available types of measurements
m = 30; % total number of measurements to be carried out
randn('state', 0);
V=randn(n,p); % columns are vi, the possible measurement vectors
```

Solve the relaxed A-optimal experiment design problem,

$$\begin{array}{ll} \text{minimize} & (1/m)\,\mathbf{tr}\left(\sum_{i=1}^p \lambda_i v_i v_i^T\right)^{-1} \\ \text{subject to} & \mathbf{1}^T\lambda = 1, \quad \lambda \succeq 0, \end{array}$$

with variable $\lambda \in \mathbf{R}^p$. Find the optimal point $\lambda^\star$ and the associated optimal value of the relaxed problem. This optimal value is a lower bound on the optimal value of the discrete A-optimal experiment design problem,

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}\left(\sum_{i=1}^p m_i v_i v_i^T\right)^{-1} \\ \text{subject to} & m_1 + \cdots + m_p = m, \quad m_i \in \{0, \ldots, m\}, \quad i = 1, \ldots, p, \end{array}$$

with variables $m_1, \ldots, m_p$. To get a suboptimal point for this discrete problem, round the entries in $m\lambda^\star$ to obtain integers $\hat{m}_i$. If needed, adjust these by hand or some other method to ensure that they sum to m, and compute the objective value obtained. This is, of course, an upper bound on the optimal value of the discrete problem. Give the gap between this upper bound and the lower bound obtained from the relaxed problem. Note that the two objective values can be interpreted as mean-square estimation error $\mathbf{E}\,\|\hat{x} - x\|_2^2$.

6.8 *Optimal detector design.* We adopt here the notation of §7.3 of the book. Explain how to design a (possibly randomized) detector that minimizes the worst-case probability of our estimate being off by more than one,

$$P_{\text{wc}} = \max_\theta \mathbf{prob}(|\hat{\theta} - \theta| \geq 2).$$

(The probability above is under the distribution associated with θ.)

Carry out your method for the problem instance with data in `off_by_one_det_data.m`. Give the optimal detection probability matrix D. Compare the optimal worst-case probability $P_{\text{wc}}^\star$ with the worst-case probability $P_{\text{wc}}^{\text{ml}}$ obtained using a maximum-likelihood detector.

6.9 *Experiment design with condition number objective.* Explain how to solve the experiment design problem (§7.5) with the condition number $\mathbf{cond}(E)$ of E (the error covariance matrix) as the objective to be minimized.

6.10 *Worst-case probability of loss.* Two investments are made, with random returns R_1 and R_2. The total return for the two investments is $R_1 + R_2$, and the probability of a loss (including breaking even, *i.e.*, $R_1 + R_2 = 0$) is $p^{\text{loss}} = \mathbf{prob}(R_1 + R_2 \leq 0)$. The goal is to find the worst-case (*i.e.*, maximum possible) value of p^{loss}, consistent with the following information. Both R_1 and R_2 have Gaussian marginal distributions, with known means μ_1 and μ_2 and known standard deviations σ_1 and σ_2. In addition, it is known that R_1 and R_2 are correlated with correlation coefficient ρ, *i.e.*,

$$\mathbf{E}(R_1 - \mu_1)(R_2 - \mu_2) = \rho\sigma_1\sigma_2.$$

Your job is to find the worst-case p^{loss} over any joint distribution of R_1 and R_2 consistent with the given marginals and correlation coefficient.

We will consider the specific case with data

$$\mu_1 = 8, \quad \mu_2 = 20, \quad \sigma_1 = 6, \quad \sigma_2 = 17.5, \quad \rho = -0.25.$$

We can compare the results to the case when R_1 and R_2 are jointly Gaussian. In this case we have

$$R_1 + R_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2 + 2\rho\sigma_1\sigma_2),$$

which for the data given above gives $p^{\text{loss}} = 0.050$. Your job is to see how much larger p^{loss} can possibly be.

This is an infinite-dimensional optimization problem, since you must maximize p^{loss} over an infinite-dimensional set of joint distributions. To (approximately) solve it, we discretize the values that R_1 and R_2 can take on, to $n = 100$ values $r_1, \ldots, r_n$, uniformly spaced from $r_1 = -30$ to $r_n = +70$. We use the discretized marginals $p^{(1)}$ and $p^{(2)}$ for R_1 and R_2, given by

$$p_i^{(k)} = \mathbf{prob}(R_k = r_i) = \frac{\exp\left(-(r_i - \mu_k)^2/(2\sigma_k^2)\right)}{\sum_{j=1}^n \exp\left(-(r_j - \mu_k)^2/(2\sigma_k^2)\right)},$$

for $k = 1, 2$, $i = 1, \ldots, n$.

Formulate the (discretized) problem as a convex optimization problem, and solve it. Report the maximum value of p^{loss} you find. Plot the joint distribution that yields the maximum value of p^{loss} using the Matlab commands `mesh` and `contour`.

Remark. You might be surprised at both the maximum value of p^{loss}, and the joint distribution that achieves it.

6.11 *Minimax linear fitting.* Consider a linear measurement model $y = Ax + v$, where $x \in \mathbf{R}^n$ is a vector of parameters to be estimated, $y \in \mathbf{R}^m$ is a vector of measurements, $v \in \mathbf{R}^m$ is a set of measurement errors, and $A \in \mathbf{R}^{m \times n}$ with rank n, with $m \geq n$. We know y and A, but we don't know v; our goal is to estimate x. We make only one assumption about the measurement error v: $\|v\|_\infty \leq \epsilon$.

We will estimate x using a linear estimator $\hat{x} = By$; we must choose the estimation matrix $B \in \mathbf{R}^{n \times m}$. The estimation error is $e = \hat{x} - x$. We will choose B to minimize the maximum possible value of $\|e\|_\infty$, where the maximum is over all values of x and all values of v satisfying $\|v\|_\infty \leq \epsilon$.

(a) Show how to find B via convex optimization.

(b) *Numerical example.* Solve the problem instance given in `minimax_fit_data.m`. Display the $\hat{x}$ you obtain and report $\|\hat{x} - x^{\text{true}}\|_\infty$. Here x^{true} is the value of x used to generate the measurement y; it is given in the data file.

6.12 *Cox proportional hazards model.* Let T be a continuous random variable taking on values in $\mathbf{R}_+$. We can think of T as modeling an event that takes place at some unknown future time, such as the death of a living person or a machine failure.

The *survival function* is $S(t) = \mathbf{prob}(T \geq t)$, which satisfies $S(0) = 1$, $S'(t) \leq 0$, and $\lim_{t \to \infty} S(t) = 0$. The *hazard rate* is given by $\lambda(t) = -S'(t)/S(t) \in \mathbf{R}_+$, and has the following interpretation: For

small $\delta > 0$, $\lambda(t)\delta$ is approximately the probability of the event occurring in $[t, t+\delta]$, given that it has not occurred up to time t. The survival function can be expressed in terms of the hazard rate:

$$S(t) = \exp\left(-\int_0^t \lambda(\tau)\, d\tau\right).$$

(The hazard rate must have infinite integral over $[0, \infty)$.)

The *Cox proportional hazards model* gives the hazard rate as a function of some features or explanatory variables (assumed constant in time) $x \in \mathbf{R}^n$. In particular, λ is given by

$$\lambda(t) = \lambda_0(t)\exp(w^T x),$$

where λ_0 (which is nonnegative, with infinite integral) is called the *baseline hazard rate*, and $w \in \mathbf{R}^n$ is a vector of model parameters. (The name derives from the fact that $\lambda(t)$ is proportional to $\exp(w_i x_i)$, for each i.)

Now suppose that we have observed a set of independent samples, with event times t^j and feature values x^j, for $j = 1, \ldots, N$. In other words, we observe that the event with features x^j occurred at time t^j. You can assume that the baseline hazard rate λ_0 is known. Show that maximum likelihood estimation of the parameter w is a convex optimization problem.

Remarks. Regularization is typically included in Cox proportional hazards fitting; for example, adding ℓ_1 regularization yields a sparse model, which selects the features to be used. The basic Cox proportional hazards model described here is readily extended to include discrete times of the event, censored measurements (which means that we only observe T to be in an interval), and the effects of features that can vary with time.

6.13 *Maximum likelihood estimation for an affinely transformed distribution.* Let z be a random variable on $\mathbf{R}^n$ with density $p_z(u) = \exp -\phi(\|u\|_2)$, where $\phi : \mathbf{R} \to \mathbf{R}$ is convex and increasing. Examples of such distributions include the standard normal $\mathcal{N}(0, \sigma^2 I)$, with $\phi(u) = (u)_+^2 + \alpha$, and the multivariable Laplacian distribution, with $\phi(u) = (u)_+ + \beta$, where α and β are normalizing constants, and $(a)_+ = \max\{a, 0\}$. Now let x be the random variable $x = Az + b$, where $A \in \mathbf{R}^{n\times n}$ is nonsingular. The distribution of x is parametrized by A and b.

Suppose $x_1, \ldots, x_N$ are independent samples from the distribution of x. Explain how to find a maximum likelihood estimate of A and b using convex optimization. If you make any further assumptions about A and b (beyond invertiblility of A), you must justify it.

Hint. The density of $x = Az + b$ is given by

$$p_x(v) = \frac{1}{|\det A|} p_z(A^{-1}(v - b)).$$

6.14 *A simple MAP problem.* We seek to estimate a point $x \in \mathbf{R}^2_+$, with exponential prior density $p(x) = \exp -(x_1 + x_2)$, based on the measurements

$$y_1 = x_1 + v_1, \quad y_2 = x_2 + v_2, \quad y_3 = x_1 - x_2 + v_3,$$

where v_1, v_2, v_3 are IID $\mathcal{N}(0, 1)$ random variables (also independent of x). A naïve estimate of x is given by $\hat{x}_{\text{naive}} = (y_1, y_2)$.

(a) Explain how to find the MAP estimate of x, given the observations y_1, y_2, y_3.

(b) Generate 100 random instances of x and y, from the given distributions. For each instance, find the MAP estimate $\hat{x}_{\text{map}}$ and the naïve estimate x_{naive}. Give a scatter plot of the MAP estimation error, *i.e.*, $\hat{x}_{\text{map}} - x$, and another scatter plot of the naïve estimation error, $\hat{x}_{\text{naive}} - x$.

6.15 *Minimum possible maximum correlation.* Let Z be a random variable taking values in $\mathbf{R}^n$, and let $\Sigma \in \mathbf{S}^n_{++}$ be its covariance matrix. We do not know Σ, but we do know the variance of m linear functions of Z. Specifically, we are given nonzero vectors $a_1, \ldots, a_m \in \mathbf{R}^n$ and $\sigma_1, \ldots, \sigma_m > 0$ for which

$$\mathbf{var}(a_i^T Z) = \sigma_i^2, \quad i = 1, \ldots, m.$$

For $i \neq j$ the correlation of Z_i and Z_j is defined to be

$$\rho_{ij} = \frac{\Sigma_{ij}}{\sqrt{\Sigma_{ii}\Sigma_{jj}}}.$$

Let $\rho^{\max} = \max_{i \neq j} |\rho_{ij}|$ be the maximum (absolute value) of the correlation among entries of Z. If $\rho^{\max}$ is large, then at least two components of Z are highly correlated (or anticorrelated).

(a) Explain how to find the smallest value of $\rho^{\max}$ that is consistent with the given information, using convex or quasiconvex optimization. If your formulation involves a change of variables or other transformation, justify it.

(b) The file `correlation_bounds_data.*` contains $\sigma_1, \ldots, \sigma_m$ and the matrix A with columns $a_1, \ldots, a_m$. Find the minimum value of $\rho^{\max}$ that is consistent with this data. Report your minimum value of $\rho^{\max}$, and give a corresponding covariance matrix Σ that achieves this value. You can report the minimum value of $\rho^{\max}$ to an accuracy of 0.01.

6.16 *Direct standardization.* Consider a random variable $(x, y) \in \mathbf{R}^n \times \mathbf{R}$, and N samples $(x_1, y_1), \ldots, (x_N, y_N) \in \mathbf{R}^n \times \mathbf{R}$, which we will use to estimate the (marginal) distribution of y. If the given samples were chosen according to the joint distribution of (x, y), a reasonable estimate for the distribution of y would be the uniform empirical distribution, which takes on values $y_1, \ldots, y_N$ each with probability $1/N$. (If y is Boolean, *i.e.*, $y \in \{0, 1\}$, we are using the fraction of samples with $y = 1$ as our estimate of $\mathbf{prob}(y = 1)$.)

The bad news is that the samples $(x_1, y_1), \ldots, (x_N, y_N) \in \mathbf{R}^n \times \mathbf{R}$ were *not* chosen from the distribution of (x, y), but instead from another (unknown, but presumably similar) distribution. The good news is that we know $\mathbf{E}\, x$, the expected value of x. We will use our knowledge of $\mathbf{E}\, x$, together with the samples, to estimate the distribution of y. *Direct standardization* replaces the uniform empirical distribution with a weighted one, which takes on values y_i with probability π_i, where $\pi \succeq 0$, $\mathbf{1}^T \pi = 1$. The weights or sample probabilities π are found by maximizing the entropy $-\sum_{i=1}^N \pi_i \log \pi_i$, subject to the requirement that the weighted sample expected value of x matches the known probabilities of x in the distribution, $\mathbf{E}\, x$. This can be expressed as $\sum_{i=1}^N \pi_i x_i = \mathbf{E}\, x$. (Both x_i and $\mathbf{E}\, x$ are known.)

(a) Explain why choosing π is a convex optimization problem.

(b) Consider the simple case with $n = 1$, and $x \in \{0, 1\}$, so $\mathbf{E}\, x = \mathbf{prob}(x = 1)$. Find the optimal sample weights $\pi_i^\star$ (analytically). Explain your solution in the following case. The samples are people, with $x = 0$ meaning the person is male, and $x = 1$ meaning the person is female. The

overall population is known to have equal numbers of females and males, but in the sample population the male : female proportions are $0.7 : 0.3$.

(c) The data in `direct_std_data.*` contain the samples $x^{(i)}$ and $y^{(i)}$, as well as $\mathbf{E}\,x$. Find the weights $\pi^\star$, and report the weighted empirical distribution. On the same plot, compare the cumulative distributions of

- the uniform empirical distribution,
- the weighted empirical distribution using $\pi^\star$, and
- the true distribution of y.

The true and empirical distributions are provided in the data file. (For example, the 20 elements of `p_true` give $\mathbf{prob}(y = 1)$ up to $\mathbf{prob}(y = 20)$, in order).

Note: Julia users might want to use the ECOS solver, by including `using ECOS`, and solving by using `solve!(prob, ECOSSolver())`.

Note: You don't need to know this to solve the problem, but the data for part (c) are real. The random variable x is a vector of a student's gender, age, and mother's and father's educational attainment, and y is the student's score on a standardized test.

7 Geometry

7.1 *Efficiency of maximum volume inscribed ellipsoid.* In this problem we prove the following geometrical result. Suppose C is a polyhedron in $\mathbf{R}^n$, symmetric about the origin, and described as

$$C = \{x \mid -1 \leq a_i^T x \leq 1,\ i = 1, \ldots, p\}.$$

Let

$$\mathcal{E} = \{x \mid x^T Q^{-1} x \leq 1\},$$

with $Q \in \mathbf{S}^n_{++}$, be the maximum volume ellipsoid with center at the origin, inscribed in C. Then the ellipsoid

$$\sqrt{n}\mathcal{E} = \{x \mid x^T Q^{-1} x \leq n\}$$

(*i.e.*, the ellipsoid $\mathcal{E}$, scaled by a factor $\sqrt{n}$ about the origin) *contains* C.

(a) Show that the condition $\mathcal{E} \subseteq C$ is equivalent to $a_i^T Q a_i \leq 1$ for $i = 1, \ldots, p$.

(b) The volume of $\mathcal{E}$ is proportional to $(\det Q)^{1/2}$, so we can find the maximum volume ellipsoid $\mathcal{E}$ inside C by solving the convex problem

$$\begin{array}{ll} \text{minimize} & \log\det Q^{-1} \\ \text{subject to} & a_i^T Q a_i \leq 1, \quad i = 1. \ldots, p. \end{array} \tag{27}$$

The variable is the matrix $Q \in \mathbf{S}^n$ and the domain of the objective function is $\mathbf{S}^n_{++}$.
Derive the Lagrange dual of problem (27).

(c) Note that Slater's condition for (27) holds ($a_i^T Q a_i < 1$ for $Q = \epsilon I$ and $\epsilon > 0$ small enough), so we have strong duality, and the KKT conditions are necessary and sufficient for optimality. What are the KKT conditions for (27)?
Suppose Q is optimal. Use the KKT conditions to show that

$$x \in C \quad \Longrightarrow \quad x^T Q^{-1} x \leq n.$$

In other words $C \subseteq \sqrt{n}\mathcal{E}$, which is the desired result.

7.2 *Euclidean distance matrices.* A matrix $X \in \mathbf{S}^n$ is a *Euclidean distance matrix* if its elements x_{ij} can be expressed as

$$x_{ij} = \|p_i - p_j\|_2^2, \quad i, j = 1, \ldots, n,$$

for some vectors p_1, ..., p_n (of arbitrary dimension). In this exercise we prove several classical characterizations of Euclidean distance matrices, derived by I. Schoenberg in the 1930s.

(a) Show that X is a Euclidean distance matrix if and only if

$$X = \mathbf{diag}(Y)\mathbf{1}^T + \mathbf{1}\,\mathbf{diag}(Y)^T - 2Y \tag{28}$$

for some matrix $Y \in \mathbf{S}^n_+$ (the symmetric positive semidefinite matrices of order n). Here, $\mathbf{diag}(Y)$ is the n-vector formed from the diagonal elements of Y, and $\mathbf{1}$ is the n-vector with all its elements equal to one. The equality (28) is therefore equivalent to

$$x_{ij} = y_{ii} + y_{jj} - 2y_{ij}, \quad i, j = 1, \ldots, n.$$

Hint. Y is the Gram matrix associated with the vectors p_1, ..., p_n, *i.e.*, the matrix with elements $y_{ij} = p_i^T p_j$.

(b) Show that the set of Euclidean distance matrices is a convex cone.

(c) Show that X is a Euclidean distance matrix if and only if

$$\mathbf{diag}(X) = 0, \qquad X_{22} - X_{21}\mathbf{1}^T - \mathbf{1}X_{21}^T \preceq 0. \tag{29}$$

The subscripts refer to the partitioning

$$X = \begin{bmatrix} x_{11} & X_{21}^T \\ X_{21} & X_{22} \end{bmatrix}$$

with $X_{21} \in \mathbf{R}^{n-1}$, and $X_{22} \in \mathbf{S}^{n-1}$.

Hint. The definition of Euclidean distance matrix involves only the distances $\|p_i - p_j\|_2$, so the origin can be chosen arbitrarily. For example, it can be assumed without loss of generality that $p_1 = 0$. With this assumption there is a unique Gram matrix Y for a given Euclidean distance matrix X. Find Y from (28), and relate it to the lefthand side of the inequality (29).

(d) Show that X is a Euclidean distance matrix if and only if

$$\mathbf{diag}(X) = 0, \qquad (I - \frac{1}{n}\mathbf{1}\mathbf{1}^T)X(I - \frac{1}{n}\mathbf{1}\mathbf{1}^T) \preceq 0. \tag{30}$$

Hint. Use the same argument as in part (c), but take the mean of the vectors p_k at the origin, *i.e.*, impose the condition that $p_1 + p_2 + \cdots + p_n = 0$.

(e) Suppose X is a Euclidean distance matrix. Show that the matrix $W \in \mathbf{S}^n$ with elements

$$w_{ij} = e^{-x_{ij}}, \quad i, j = 1, \ldots, n,$$

is positive semidefinite.

Hint. Use the following identity from probability theory. Define $z \sim \mathcal{N}(0, I)$. Then

$$\mathbf{E}\, e^{izx} = e^{-\frac{1}{2}\|x\|_2^2}$$

for all x, where $i = \sqrt{-1}$ and $\mathbf{E}$ denotes expectation with respect to z. (This is the characteristic function of a multivariate normal distribution.)

7.3 *Minimum total covering ball volume.* We consider a collection of n points with locations $x_1, \ldots, x_n \in \mathbf{R}^k$. We are also given a set of m groups or subsets of these points, $G_1, \ldots, G_m \subseteq \{1, \ldots, n\}$. For each group, let V_i be the volume of the smallest Euclidean ball that contains the points in group G_i. (The volume of a Euclidean ball of radius r in $\mathbf{R}^k$ is $a_k r^k$, where a_k is known constant that is positive but otherwise irrelevant here.) We let $V = V_1 + \cdots + V_m$ be the total volume of these minimal covering balls.

The points $x_{k+1}, \ldots, x_n$ are fixed (*i.e.*, they are problem data). The variables to be chosen are $x_1, \ldots, x_k$. Formulate the problem of choosing $x_1, \ldots, x_k$, in order to minimize the total minimal covering ball volume V, as a convex optimization problem. Be sure to explain any new variables you introduce, and to justify the convexity of your objective and inequality constraint functions.

7.4 *Maximum-margin multiclass classification.* In an m-category pattern classification problem, we are given m sets $C_i \subseteq \mathbf{R}^n$. Set C_i contains N_i examples of feature vectors in class i. The learning problem is to find a decision function $f : \mathbf{R}^n \to \{1, 2, \ldots, m\}$ that maps each training example to its class, and also generalizes reliably to feature vectors that are not included in the training sets C_i.

(a) A common type of decision function for two-way classification is

$$f(x) = \begin{cases} 1 & \text{if } a^T x + b > 0 \\ 2 & \text{if } a^T x + b < 0. \end{cases}$$

In the simplest form, finding f is equivalent to solving a feasibility problem: find a and b such that

$$\begin{array}{ll} a^T x + b > 0 & \text{if } x \in C_1 \\ a^T x + b < 0 & \text{if } x \in C_2. \end{array}$$

Since these strict inequalities are homogeneous in a and b, they are feasible if and only if the nonstrict inequalities

$$\begin{array}{ll} a^T x + b \geq 1 & \text{if } x \in C_1 \\ a^T x + b \leq -1 & \text{if } x \in C_2 \end{array}$$

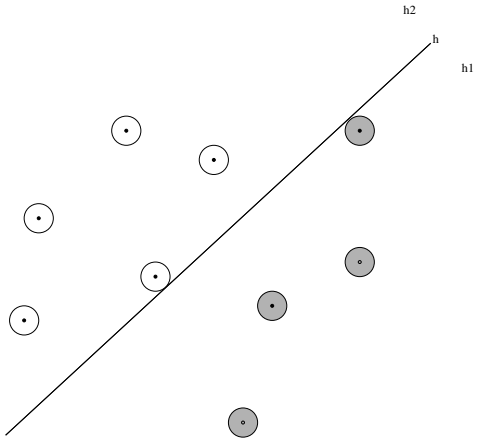
are feasible. This is a feasibility problem with $N_1 + N_2$ linear inequalities in $n + 1$ variables a, b.

As an extension that improves the robustness (*i.e.*, generalization capability) of the classifier, we can impose the condition that the decision function f classifies all points in a neighborhood of C_1 and C_2 correctly, and we can maximize the size of the neighborhood. This problem can be expressed as

$$\begin{array}{ll} \text{maximize} & t \\ \text{subject to} & a^T x + b > 0 \ \text{ if } \mathbf{dist}(x, C_1) \leq t, \\ & a^T x + b < 0 \ \text{ if } \mathbf{dist}(x, C_2) \leq t, \end{array}$$

where $\mathbf{dist}(x, C) = \min_{y \in C} \|x - y\|_2$.

This is illustrated in the figure. The centers of the shaded disks form the set C_1. The centers of the other disks form the set C_2. The set of points at a distance less than t from C_i is the union of disks with radius t and center in C_i. The hyperplane in the figure separates the two expanded sets. We are interested in expanding the circles as much as possible, until the two expanded sets are no longer separable by a hyperplane.



Since the constraints are homogeneous in a, b, we can again replace them with nonstrict inequalities

$$\begin{array}{ll} \text{maximize} & t \\ \text{subject to} & a^T x + b \geq 1 \ \text{ if } \mathbf{dist}(x, C_1) \leq t, \\ & a^T x + b \leq -1 \ \text{ if } \mathbf{dist}(x, C_2) \leq t. \end{array} \tag{31}$$

The variables are a, b, and t.

(b) Next we consider an extension to more than two classes. If $m > 2$ we can use a decision function

$$f(x) = \operatorname*{argmax}_{i=1,\ldots,m} (a_i^T x + b_i),$$

parameterized by m vectors $a_i \in \mathbf{R}^n$ and m scalars b_i. To find f, we can solve a feasibility problem: find a_i, b_i, such that

$$a_i^T x + b_i > \max_{j \neq i} (a_j^T x + b_j) \quad \text{if } x \in C_i, \quad i = 1, \ldots, m,$$

or, equivalently,

$$a_i^T x + b_i \geq 1 + \max_{j \neq i} (a_j^T x + b_j) \quad \text{if } x \in C_i, \quad i = 1, \ldots, m.$$

Similarly as in part (a), we consider a robust version of this problem:

$$\begin{array}{ll} \text{maximize} & t \\ \text{subject to} & a_i^T x + b_i \geq 1 + \max_{j \neq i} (a_j^T x + b_j) \text{ if } \mathbf{dist}(x, C_i) \leq t, \\ & \quad i = 1, \ldots, m. \end{array} \tag{32}$$

The variables in the problem are $a_i \in \mathbf{R}^n$, $b_i \in \mathbf{R}$, $i = 1, \ldots, m$, and t.

Formulate the optimization problems (31) and (32) as SOCPs (if possible), or as quasiconvex optimization problems involving SOCP feasibility problems (otherwise).

7.5 *Three-way linear classification.* We are given data

$$x^{(1)}, \ldots, x^{(N)}, \quad y^{(1)}, \ldots, y^{(M)}, \quad z^{(1)}, \ldots, z^{(P)},$$

three nonempty sets of vectors in $\mathbf{R}^n$. We wish to find three affine functions on $\mathbf{R}^n$,

$$f_i(z) = a_i^T z - b_i, \quad i = 1, 2, 3,$$

that satisfy the following properties:

$$\begin{array}{l} f_1(x^{(j)}) > \max\{f_2(x^{(j)}), f_3(x^{(j)})\}, \quad j = 1, \ldots, N, \\ f_2(y^{(j)}) > \max\{f_1(y^{(j)}), f_3(y^{(j)})\}, \quad j = 1, \ldots, M, \\ f_3(z^{(j)}) > \max\{f_1(z^{(j)}), f_2(z^{(j)})\}, \quad j = 1, \ldots, P. \end{array}$$

In words: f_1 is the largest of the three functions on the x data points, f_2 is the largest of the three functions on the y data points, f_3 is the largest of the three functions on the z data points. We can give a simple geometric interpretation: The functions f_1, f_2, and f_3 partition $\mathbf{R}^n$ into three regions,

$$\begin{array}{rcl} R_1 & = & \{z \mid f_1(z) > \max\{f_2(z), f_3(z)\}\}, \\ R_2 & = & \{z \mid f_2(z) > \max\{f_1(z), f_3(z)\}\}, \\ R_3 & = & \{z \mid f_3(z) > \max\{f_1(z), f_2(z)\}\}, \end{array}$$

defined by where each function is the largest of the three. Our goal is to find functions with $x^{(j)} \in R_1$, $y^{(j)} \in R_2$, and $z^{(j)} \in R_3$.

Pose this as a convex optimization problem. You may not use strict inequalities in your formulation.

Solve the specific instance of the 3-way separation problem given in `sep3way_data.m`, with the columns of the matrices `X`, `Y` and `Z` giving the $x^{(j)},\ j = 1, \ldots, N$, $y^{(j)},\ j = 1, \ldots, M$ and $z^{(j)},\ j = 1, \ldots, P$. To save you the trouble of plotting data points and separation boundaries, we have included the plotting code in `sep3way_data.m`. (Note that `a1`, `a2`, `a3`, `b1` and `b2` contain arbitrary numbers; you should compute the correct values using CVX.)

7.6 *Feature selection and sparse linear separation.* Suppose $x^{(1)}, \ldots, x^{(N)}$ and $y^{(1)}, \ldots, y^{(M)}$ are two given nonempty collections or classes of vectors in $\mathbf{R}^n$ that can be (strictly) separated by a hyperplane, *i.e.*, there exists $a \in \mathbf{R}^n$ and $b \in \mathbf{R}$ such that

$$a^T x^{(i)} - b \geq 1, \quad i = 1, \ldots, N, \qquad a^T y^{(i)} - b \leq -1, \quad i = 1, \ldots, M.$$

This means the two classes are (weakly) separated by the slab

$$S = \{z \mid |a^T z - b| \leq 1\},$$

which has thickness $2/\|a\|_2$. You can think of the components of $x^{(i)}$ and $y^{(i)}$ as *features*; a and b define an affine function that combines the features and allows us to distinguish the two classes.

To find the thickest slab that separates the two classes, we can solve the QP

$$\begin{array}{ll} \text{minimize} & \|a\|_2 \\ \text{subject to} & a^T x^{(i)} - b \geq 1, \quad i = 1, \ldots, N \\ & a^T y^{(i)} - b \leq -1, \quad i = 1, \ldots, M, \end{array}$$

with variables $a \in \mathbf{R}^n$ and $b \in \mathbf{R}$. (This is equivalent to the problem given in (8.23), p424, §8.6.1; see also exercise 8.23.)

In this problem we seek (a, b) that separate the two classes with a thick slab, and also has a sparse, *i.e.*, there are many j with $a_j = 0$. Note that if $a_j = 0$, the affine function $a^T z - b$ does not depend on z_j, *i.e.*, the jth feature is not used to carry out classification. So a sparse a corresponds to a classification function that is parsimonious; it depends on just a few features. So our goal is to find an affine classification function that gives a thick separating slab, and also uses as few features as possible to carry out the classification.

This is in general a hard combinatorial (bi-criterion) optimization problem, so we use the standard heuristic of solving

$$\begin{array}{ll} \text{minimize} & \|a\|_2 + \lambda \|a\|_1 \\ \text{subject to} & a^T x^{(i)} - b \geq 1, \quad i = 1, \ldots, N \\ & a^T y^{(i)} - b \leq -1, \quad i = 1, \ldots, M, \end{array}$$

where $\lambda \geq 0$ is a weight vector that controls the trade-off between separating slab thickness and (indirectly, through the ℓ_1 norm) sparsity of a.

Get the data in `sp_ln_sp_data.m`, which gives $x^{(i)}$ and $y^{(i)}$ as the columns of matrices `X` and `Y`, respectively. Find the thickness of the maximum thickness separating slab. Solve the problem above for 100 or so values of λ over an appropriate range (we recommend log spacing). For each value,

record the separation slab thickness $2/\|a\|_2$ and $\mathbf{card}(a)$, the cardinality of a (*i.e.*, the number of nonzero entries). In computing the cardinality, you can count an entry a_j of a as zero if it satisfies $|a_j| \leq 10^{-4}$. Plot these data with slab thickness on the vertical axis and cardinality on the horizontal axis.

Use this data to choose a set of 10 features out of the 50 in the data. Give the indices of the features you choose. You may have several choices of sets of features here; you can just choose one. Then find the maximum thickness separating slab that uses only the chosen features. (This is standard practice: once you've chosen the features you're going to use, you optimize again, using only those features, and without the ℓ_1 regularization.

7.7 *Thickest slab separating two sets.* We are given two sets in $\mathbf{R}^n$: a polyhedron

$$C_1 = \{x \mid Cx \preceq d\},$$

defined by a matrix $C \in \mathbf{R}^{m\times n}$ and a vector $d \in \mathbf{R}^m$, and an ellipsoid

$$C_2 = \{Pu + q \mid \|u\|_2 \leq 1\},$$

defined by a matrix $P \in \mathbf{R}^{n\times n}$ and a vector $q \in \mathbf{R}^n$. We assume that the sets are nonempty and that they do not intersect. We are interested in the optimization problem

$$\begin{array}{ll} \text{maximize} & \inf_{x\in C_1} a^T x - \sup_{x\in C_2} a^T x \\ \text{subject to} & \|a\|_2 = 1. \end{array}$$

with variable $a \in \mathbf{R}^n$.

Explain how you would solve this problem. You can answer the question by reducing the problem to a standard problem class (LP, QP, SOCP, SDP, ...), or by describing an algorithm to solve it.

Remark. The geometrical interpretation is as follows. If we choose

$$b = \frac{1}{2}(\inf_{x\in C_1} a^T x + \sup_{x\in C_2} a^T x),$$

then the hyperplane $H = \{x \mid a^T x = b\}$ is the maximum margin separating hyperplane separating C_1 and C_2. Alternatively, a gives us the thickest slab that separates the two sets.

7.8 *Bounding object position from multiple camera views.* A small object is located at unknown position $x \in \mathbf{R}^3$, and viewed by a set of m cameras. Our goal is to find a box in $\mathbf{R}^3$,

$$\mathcal{B} = \{z \in \mathbf{R}^3 \mid l \preceq z \preceq u\},$$

for which we can guarantee $x \in \mathcal{B}$. We want the smallest possible such bounding box. (Although it doesn't matter, we can use volume to judge 'smallest' among boxes.)

Now we describe the cameras. The object at location $x \in \mathbf{R}^3$ creates an image on the image plane of camera i at location

$$v_i = \frac{1}{c_i^T x + d_i}(A_i x + b_i) \in \mathbf{R}^2.$$

The matrices $A_i \in \mathbf{R}^{2\times 3}$, vectors $b_i \in \mathbf{R}^2$ and $c_i \in \mathbf{R}^3$, and real numbers $d_i \in \mathbf{R}$ are known, and depend on the camera positions and orientations. We assume that $c_i^T x + d_i > 0$. The 3×4 matrix

$$P_i = \begin{bmatrix} A_i & b_i \\ c_i^T & d_i \end{bmatrix}$$

is called the *camera matrix* (for camera i). It is often (but not always) the case the that the first 3 columns of P_i (*i.e.*, A_i stacked above c_i^T) form an orthogonal matrix, in which case the camera is called *orthographic*.

We do not have direct access to the image point v_i; we only know the (square) pixel that it lies in. In other words, the camera gives us a measurement $\hat{v}_i$ (the center of the pixel that the image point lies in); we are guaranteed that

$$\|v_i - \hat{v}_i\|_\infty \leq \rho_i/2,$$

where ρ_i is the pixel width (and height) of camera i. (We know nothing else about v_i; it could be any point in this pixel.)

Given the data A_i, b_i, c_i, d_i, $\hat{v}_i$, ρ_i, we are to find the smallest box $\mathcal{B}$ (*i.e.*, find the vectors l and u) that is guaranteed to contain x. In other words, find the smallest box in $\mathbf{R}^3$ that contains all points consistent with the observations from the camera.

(a) Explain how to solve this using convex or quasiconvex optimization. You must explain any transformations you use, any new variables you introduce, etc. If the convexity or quasiconvexity of any function in your formulation isn't obvious, be sure justify it.

(b) Solve the specific problem instance given in the file `camera_data.m`. Be sure that your final numerical answer (*i.e.*, l and u) stands out.

7.9 *Triangulation from multiple camera views.* A projective camera can be described by a linear-fractional function $f : \mathbf{R}^3 \to \mathbf{R}^2$,

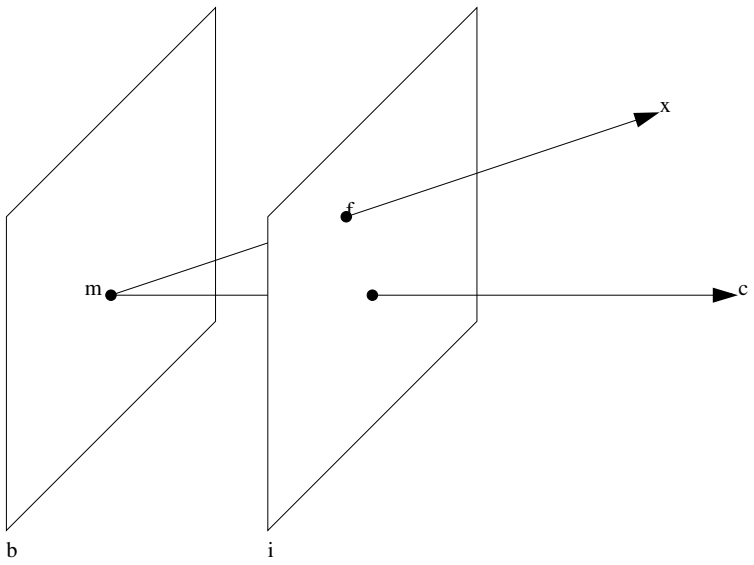
$$f(x) = \frac{1}{c^T x + d}(Ax + b), \qquad \mathbf{dom}\, f = \{x \mid c^T x + d > 0\},$$

with

$$\mathbf{rank}(\begin{bmatrix} A \\ c^T \end{bmatrix}) = 3.$$

The domain of f consists of the points in front of the camera.

Before stating the problem, we give some background and interpretation, most of which will not be needed for the actual problem.



The 3×4-matrix

$$P = \begin{bmatrix} A & b \\ c^T & d \end{bmatrix}$$

is called the *camera matrix* and has rank 3. Since f is invariant with respect to a scaling of P, we can normalize the parameters and assume, for example, that $\|c\|_2 = 1$. The numerator $c^T x + d$ is then the distance of x to the plane $\{z \mid c^T z + d = 0\}$. This plane is called the *principal plane*. The point

$$x_{\mathrm{c}} = -\begin{bmatrix} A \\ c^T \end{bmatrix}^{-1} \begin{bmatrix} b \\ d \end{bmatrix}$$

lies in the principal plane and is called the *camera center*. The ray $\{x_{\mathrm{c}} + \theta c \mid \theta \geq 0\}$, which is perpendicular to the principal plane, is the *principal axis*. We will define the *image plane* as the plane parallel to the principal plane, at a unit distance from it along the principal axis.

The point x' in the figure is the intersection of the image plane and the line through the camera center and x, and is given by

$$x' = x_{\mathrm{c}} + \frac{1}{c^T(x - x_{\mathrm{c}})}(x - x_{\mathrm{c}}).$$

Using the definition of x_{c} we can write $f(x)$ as

$$f(x) = \frac{1}{c^T(x - x_{\mathrm{c}})} A(x - x_{\mathrm{c}}) = A(x' - x_{\mathrm{c}}) = Ax' + b.$$

This shows that the mapping $f(x)$ can be interpreted as a projection of x on the image plane to get x', followed by an affine transformation of x'. We can interpret $f(x)$ as the point x' expressed in some two-dimensional coordinate system attached to the image plane.

In this exercise we consider the problem of determining the position of a point $x \in \mathbf{R}^3$ from its image in N cameras. Each of the cameras is characterized by a known linear-fractional mapping f_k and camera matrix P_k:

$$f_k(x) = \frac{1}{c_k^T x + d_k}(A_k x + b_k), \qquad P_k = \begin{bmatrix} A_k & b_k \\ c_k^T & d_k \end{bmatrix}, \qquad k = 1, \ldots, N.$$

The image of the point x in camera k is denoted $y^{(k)} \in \mathbf{R}^2$. Due to camera imperfections and calibration errors, we do not expect the equations $f_k(x) = y^{(k)}$, $k = 1, \ldots, N$, to be exactly solvable. To estimate the point x we therefore minimize the maximum error in the N equations by solving

$$\text{minimize} \quad g(x) = \max_{k=1,\ldots,N} \|f_k(x) - y^{(k)}\|_2. \tag{33}$$

(a) Show that (33) is a quasiconvex optimization problem. The variable in the problem is $x \in \mathbf{R}^3$. The functions f_k (*i.e.*, the parameters A_k, b_k, c_k, d_k) and the vectors $y^{(k)}$ are given.

(b) Solve the following instance of (33) using CVX (and bisection): $N = 4$,

$$P_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \qquad P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 10 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} 1 & 1 & 1 & -10 \\ -1 & 1 & 1 & 0 \\ -1 & -1 & 1 & 10 \end{bmatrix}, \qquad P_4 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 10 \end{bmatrix},$$

$$y^{(1)} = \begin{bmatrix} 0.98 \\ 0.93 \end{bmatrix}, \qquad y^{(2)} = \begin{bmatrix} 1.01 \\ 1.01 \end{bmatrix}, \qquad y^{(3)} = \begin{bmatrix} 0.95 \\ 1.05 \end{bmatrix}, \qquad y^{(4)} = \begin{bmatrix} 2.04 \\ 0.00 \end{bmatrix}.$$

You can terminate the bisection when a point is found with accuracy $g(x) - p^\star \leq 10^{-4}$, where $p^\star$ is the optimal value of (33).

7.10 *Projection onto the probability simplex.* In this problem you will work out a simple method for finding the Euclidean projection y of $x \in \mathbf{R}^n$ onto the probability simplex $\mathcal{P} = \{z \mid z \succeq 0,\ \mathbf{1}^T z = 1\}$.

Hints. Consider the problem of minimizing $(1/2)\|y - x\|_2^2$ subject to $y \succeq 0$, $\mathbf{1}^T y = 1$. Form the partial Lagrangian

$$L(y, \nu) = (1/2)\|y - x\|_2^2 + \nu(\mathbf{1}^T y - 1),$$

leaving the constraint $y \succeq 0$ implicit. Show that $y = (x - \nu\mathbf{1})_+$ minimizes $L(y, \nu)$ over $y \succeq 0$.

7.11 *Conformal mapping via convex optimization.* Suppose that Ω is a closed bounded region in $\mathbf{C}$ with no holes (*i.e.*, it is simply connected). The Riemann mapping theorem states that there exists a conformal mapping φ from Ω onto $D = \{z \in \mathbf{C} \mid |z| \leq 1\}$, the unit disk in the complex plane. (This means that φ is an analytic function, and maps Ω one-to-one onto D.)

One proof of the Riemann mapping theorem is based on an infinite dimensional optimization problem. We choose a point $a \in \mathbf{int}\,\Omega$ (the interior of Ω). Among all analytic functions that map $\partial\Omega$ (the boundary of Ω) into D, we choose one that maximizes the magnitude of the derivative at a. Amazingly, it can be shown that this function is a conformal mapping of Ω onto D.

We can use this theorem to construct an approximate conformal mapping, by sampling the boundary of Ω, and by restricting the optimization to a finite-dimensional subspace of analytic functions. Let $b_1, \ldots, b_N$ be a set of points in $\partial\Omega$ (meant to be a sampling of the boundary). We will search only over polynomials of degree up to n,

$$\hat{\varphi}(z) = \alpha_1 z^n + \alpha_2 z^{n-1} + \cdots + \alpha_n z + \alpha_{n+1},$$

where $\alpha_1, \ldots, \alpha_{n+1} \in \mathbf{C}$. With these approximations, we obtain the problem

$$\begin{array}{ll} \text{maximize} & |\hat{\varphi}'(a)| \\ \text{subject to} & |\hat{\varphi}(b_i)| \leq 1, \quad i = 1, \ldots, N, \end{array}$$

with variables $\alpha_1, \ldots, \alpha_{n+1} \in \mathbf{C}$. The problem data are $b_1, \ldots, b_N \in \partial\Omega$ and $a \in \mathbf{int}\,\Omega$.

(a) Explain how to solve the problem above via convex or quasiconvex optimization.

(b) Carry out your method on the problem instance given in `conf_map_data.m`. This file defines the boundary points b_i and plots them. It also contains code that will plot $\hat{\varphi}(b_i)$, the boundary of the mapped region, once you provide the values of α_j; these points should be very close to the boundary of the unit disk. (Please turn in this plot, and give us the values of α_j that you find.) The function `polyval` may be helpful.

Remarks.

- We've been a little informal in our mathematics here, but it won't matter.
- You do not need to know any complex analysis to solve this problem; we've told you everything you need to know.
- A basic result from complex analysis tells us that $\hat{\varphi}$ is one-to-one if and only if the image of the boundary does not 'loop over' itself. (We mention this just for fun; we're not asking you to verify that the $\hat{\varphi}$ you find is one-to-one.)

7.12 *Fitting a vector field to given directions.* This problem concerns a vector field on $\mathbf{R}^n$, *i.e.*, a function $F : \mathbf{R}^n \to \mathbf{R}^n$. We are given the *direction* of the vector field at points $x^{(1)}, \ldots, x^{(N)} \in \mathbf{R}^n$,

$$q^{(i)} = \frac{1}{\|F(x^{(i)})\|_2} F(x^{(i)}), \quad i = 1, \ldots, N.$$

(These directions might be obtained, for example, from samples of trajectories of the differential equation $\dot{z} = F(z)$.) The goal is to fit these samples with a vector field of the form

$$\hat{F} = \alpha_1 F_1 + \cdots + \alpha_m F_m,$$

where $F_1, \ldots, F_m : \mathbf{R}^n \to \mathbf{R}^n$ are given (basis) functions, and $\alpha \in \mathbf{R}^m$ is a set of coefficients that we will choose.

We will measure the fit using the maximum angle error,

$$J = \max_{i=1,\ldots,N} \left| \angle(q^{(i)}, \hat{F}(x^{(i)})) \right|,$$

where $\angle(z, w) = \cos^{-1}((z^T w)/\|z\|_2\|w\|_2)$ denotes the angle between nonzero vectors z and w. We are only interested in the case when J is smaller than $\pi/2$.

(a) Explain how to choose α so as to minimize J using convex optimization. Your method can involve solving multiple convex problems. Be sure to explain how you handle the constraints $\hat{F}(x^{(i)}) \neq 0$.

(b) Use your method to solve the problem instance with data given in `vfield_fit_data.m`, with an affine vector field fit, *i.e.*, $\hat{F}(z) = Az + b$. (The matrix A and vector b are the parameters α above.) Give your answer to the nearest degree, as in '$20^\circ < J^\star \leq 21^\circ$'.

This file also contains code that plots the vector field directions, and also (but commented out) the directions of the vector field fit, $\hat{F}(x^{(i)})/\|\hat{F}(x^{(i)})\|_2$. Create this plot, with your fitted vector field.

7.13 *Robust minimum volume covering ellipsoid.* Suppose z is a point in $\mathbf{R}^n$ and $\mathcal{E}$ is an ellipsoid in $\mathbf{R}^n$ with center c. The *Mahalanobis distance* of the point to the ellipsoid center is defined as

$$M(z, \mathcal{E}) = \inf\{t \geq 0 \mid z \in c + t(\mathcal{E} - c)\},$$

which is the factor by which we need to scale the ellipsoid about its center so that z is on its boundary. We have $z \in \mathcal{E}$ if and only if $M(z, \mathcal{E}) \leq 1$. We can use $(M(z, \mathcal{E}) - 1)_+$ as a measure of the Mahalanobis distance of the point z to the ellipsoid $\mathcal{E}$.

Now we can describe the problem. We are given m points $x_1, \ldots, x_m \in \mathbf{R}^n$. The goal is to find the optimal trade-off between the volume of the ellipsoid $\mathcal{E}$ and the total Mahalanobis distance of the points to the ellipsoid, *i.e.*,

$$\sum_{i=1}^{m} (M(z, \mathcal{E}) - 1)_+ .$$

Note that this can be considered a robust version of finding the smallest volume ellipsoid that covers a set of points, since here we allow one or more points to be outside the ellipsoid.

(a) Explain how to solve this problem. You must say clearly what your variables are, what problem you solve, and why the problem is convex.

(b) Carry out your method on the data given in `rob_min_vol_ellips_data.m`. Plot the optimal trade-off curve of ellipsoid volume versus total Mahalanobis distance. For some selected points on the trade-off curve, plot the ellipsoid and the points (which are in $\mathbf{R}^2$). We are only interested in the region of the curve where the ellipsoid volume is within a factor of ten (say) of the minimum volume ellipsoid that covers all the points.

Important. Depending on how you formulate the problem, you might encounter problems that are unbounded below, or where CVX encounters numerical difficulty. Just avoid these by appropriate choice of parameter.

Very important. If you use Matlab version 7.0 (which is filled with bugs) you might find that functions involving determinants don't work in CVX. If you use this version of Matlab, then you must download the file `blkdiag.m` on the course website and put it in your Matlab path before the default version (which has a bug).

7.14 *Isoperimetric problem.* We consider the problem of choosing a curve in a two-dimensional plane that encloses as much area as possible between itself and the x-axis, subject to constraints. For simplicity we will consider only curves of the form

$$\mathcal{C} = \{(x, y) \mid y = f(x)\},$$

where $f : [0, a] \to \mathbf{R}$. This assumes that for each x-value, there can only be a single y-value, which need not be the case for general curves. We require that at the end points (which are given), the

curve returns to the x-axis, so $f(0) = 0$, and $f(a) = 0$. In addition, the length of the curve cannot exceed a budget L, so we must have

$$\int_0^a \sqrt{1 + f'(x)^2}\, dx \leq L.$$

The objective is the area enclosed, which is given by

$$\int_0^a f(x)\, dx.$$

To pose this as a finite dimensional optimization problem, we discretize over the x-values. Specifically, we take $x_i = h(i-1)$, $i = 1, \ldots, N+1$, where $h = a/N$ is the discretization step size, and we let $y_i = f(x_i)$. Thus our objective becomes

$$h \sum_{i=1}^{N} y_i,$$

and our constraints can be written as

$$h \sum_{i=1}^{N} \sqrt{1 + ((y_{i+1} - y_i)/h)^2} \leq L, \quad y_1 = 0, \quad y_{N+1} = 0.$$

In addition to these constraints, we will also require that our curve passes through a set of prespecified points. Let $\mathcal{F} \subseteq \{1, \ldots, N+1\}$ be an index set. For $j \in \mathcal{F}$, we require $y_j = y_j^{\text{fixed}}$, where $y^{\text{fixed}} \in \mathbf{R}^{N+1}$ (the entries of y^{fixed} whose indices are not in $\mathcal{F}$ can be ignored). Finally, we add a constraint on maximum curvature,

$$-C \leq (y_{i+2} - 2y_{i+1} + y_i)/h^2 \leq C, \quad i = 1, \ldots, N-1.$$

Explain how to find the curve, *i.e.*, $y_1, \ldots, y_{N+1}$, that maximizes the area enclosed subject to these constraints, using convex optimization. Carry out your method on the problem instance with data given in `iso_perim_data.m`. Report the optimal area enclosed, and use the commented out code in the data file to plot your curve.

Remark (for your amusement only). The isoperimetric problem is an ancient problem in mathematics with a history dating all the way back to the tragedy of queen Dido and the founding of Carthage. The story (which is mainly the account of the poet Virgil in his epic volume *Aeneid*), goes that Dido was a princess forced to flee her home after her brother murdered her husband. She travels across the mediterranean and arrives on the shores of what is today modern Tunisia. The natives weren't very happy about the newcomers, but Dido was able to negotiate with the local King: in return for her fortune, the King promised to cede her as much land as she could mark out with the skin of a bull.

The king thought he was getting a good deal, but Dido outmatched him in mathematical skill. She broke down the skin into thin pieces of leather and sewed them into a long piece of string. Then, taking the seashore as an edge, they laid the string in a semicircle, carving out a piece of land larger than anyone imagined; and on this land, the ancient city of Carthage was born. When the king saw what she had done, he was so impressed by Dido's talent that he asked her to marry him. Dido refused, so the king built a university in the hope that he could find another woman with similar talent.

7.15 *Dual of maximum volume ellipsoid problem.* Consider the problem of computing the maximum volume ellipsoid inscribed in a nonempty bounded polyhedron

$$C = \{x \mid a_i^T x \leq b_i, i = 1, \ldots, m\}.$$

Parametrizing the ellipsoid as $\mathcal{E} = \{Bu + d \mid \|u\|_2 \leq 1\}$, with $B \in \mathbf{S}^n_{++}$ and $d \in \mathbf{R}^n$, the optimal ellipsoid can be found by solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & -\log\det B \\ \text{subject to} & \|Ba_i\|_2 + a_i^T d \leq b_i, \quad i = 1, \ldots, m \end{array}$$

with variables $B \in \mathbf{S}^n$, $d \in \mathbf{R}^n$. Derive the Lagrange dual of the equivalent problem

$$\begin{array}{ll} \text{minimize} & -\log\det B \\ \text{subject to} & \|y_i\|_2 + a_i^T d \leq b_i, \quad i = 1, \ldots, m \\ & Ba_i = y_i, \quad i = 1, \ldots, m \end{array}$$

with variables $B \in \mathbf{S}^n$, $d \in \mathbf{R}^n$, $y_i \in \mathbf{R}^n$, $i = 1, \ldots, m$.

7.16 *Fitting a sphere to data.* Consider the problem of fitting a sphere $\{x \in \mathbf{R}^n \mid \|x - x_c\|_2 = r\}$ to m points $u_1, \ldots, u_m \in \mathbf{R}^n$, by minimizing the error function

$$\sum_{i=1}^m \left(\|u_i - x_c\|_2^2 - r^2\right)^2$$

over the variables $x_c \in \mathbf{R}^n$, $r \in \mathbf{R}$.

(a) Explain how to solve this problem using convex or quasiconvex optimization. The simpler your formulation, the better. (For example: a convex formulation is simpler than a quasiconvex formulation; an LP is simpler than an SOCP, which is simpler than an SDP.) Be sure to explain what your variables are, and how your formulation minimizes the error function above.

(b) Use your method to solve the problem instance with data given in the file `sphere_fit_data.m`, with $n = 2$. Plot the fitted circle and the data points.

7.17 The *polar* of a set $C \subseteq \mathbf{R}^n$ is defined as

$$C^\circ = \{x \mid u^T x \leq 1 \; \forall u \in C\}.$$

(a) Show that C° is convex, regardless of the properties of C.

(b) Let C_1 and C_2 be two nonempty polyhedra defined by sets of linear inequalities:

$$C_1 = \{u \in \mathbf{R}^n \mid A_1 u \preceq b_1\}, \qquad C_2 = \{v \in \mathbf{R}^n \mid A_2 v \preceq b_2\}$$

with $A_1 \in \mathbf{R}^{m_1 \times n}$, $A_2 \in \mathbf{R}^{m_2 \times n}$, $b_1 \in \mathbf{R}^{m_1}$, $b_2 \in \mathbf{R}^{m_2}$. Formulate the problem of finding the Euclidean distance between C_1° and C_2°,

$$\begin{array}{ll} \text{minimize} & \|x_1 - x_2\|_2^2 \\ \text{subject to} & x_1 \in C_1^\circ \\ & x_2 \in C_2^\circ, \end{array}$$

as a QP. Your formulation should be efficient, *i.e.*, the dimensions of the QP (number of variables and constraints) should be linear in m_1, m_2, n. (In particular, formulations that require enumerating the extreme points of C_1 and C_2 are to be avoided.)

7.18 *Polyhedral cone questions.* You are given matrices $A \in \mathbf{R}^{n\times k}$ and $B \in \mathbf{R}^{n\times p}$.

Explain how to solve the following two problems using convex optimization. Your solution can involve solving multiple convex problems, as long as the number of such problems is no more than linear in the dimensions $n,\ k,\ p$.

(a) How would you determine whether $A\mathbf{R}^k_+ \subseteq B\mathbf{R}^p_+$? This means that every nonnegative linear combination of the columns of A can be expressed as a nonnegative linear combination of the columns of B.

(b) How would you determine whether $A\mathbf{R}^k_+ = \mathbf{R}^n$? This means that every vector in $\mathbf{R}^n$ can be expressed as a nonnegative linear combination of the columns of A.

7.19 *Projection on convex hull of union of ellipsoids.* Let $E_1, \ldots, E_m$ be m ellipsoids in $\mathbf{R}^n$ defined as

$$E_i = \{A_i u + b_i \mid \|u\|_2 \leq 1\}, \quad i = 1, \ldots, m,$$

with $A_i \in \mathbf{R}^{n\times n}$ and $b_i \in \mathbf{R}^n$. Consider the problem of projecting a point $a \in \mathbf{R}^n$ on the convex hull of the union of the ellipsoids:

$$\begin{array}{ll} \text{minimize} & \|x - a\|_2 \\ \text{subject to} & x \in \mathbf{conv}(E_1 \cup \cdots \cup E_m). \end{array}$$

Formulate this as a second order cone program.

7.20 *Bregman divergences.* Let $f : \mathbf{R}^n \to \mathbf{R}$ be strictly convex and differentiable. Then the *Bregman divergence* associated with f is the function $D_f : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R}$ given by

$$D_f(x, y) = f(x) - f(y) - \nabla f(y)^T(x - y).$$

(a) Show that $D_f(x, y) \geq 0$ for all $x,\ y \in \mathbf{dom}\, f$.

(b) Show that if $f = \|\cdot\|_2^2$, then $D_f(x, y) = \|x - y\|_2^2$.

(c) Show that if $f(x) = \sum_{i=1}^n x_i \log x_i$ (negative entropy), with $\mathbf{dom}\, f = \mathbf{R}^n_+$ (with $0 \log 0$ taken to be 0), then

$$D_f(x, y) = \sum_{i=1}^n \left(x_i \log(x_i/y_i) - x_i + y_i\right),$$

the *Kullback-Leibler divergence* between x and y.

(d) *Bregman projection.* The previous parts suggest that Bregman divergences can be viewed as generalized 'distances', *i.e.*, functions that measure how similar two vectors are. This suggests solving geometric problems that measure distance between vectors using a Bregman divergence rather than Euclidean distance.

Explain whether

$$\begin{array}{ll} \text{minimize} & D_f(x, y) \\ \text{subject to} & x \in \mathcal{C}, \end{array}$$

with variable $x \in \mathbf{R}^n$, is a convex optimization problem (assuming $\mathcal{C}$ is convex).

(e) *Duality.* Show that $D_g(y^*, x^*) = D_f(x, y)$, where $g = f^*$ and $z^* = \nabla f(z)$. You can assume that $\nabla f^* = (\nabla f)^{-1}$ and that f is closed.

7.21 *Ellipsoidal peeling.* In this problem, you will implement an outlier identification technique using Löwner-John ellipsoids. Given a set of points $\mathcal{D} = \{x_1, \ldots, x_N\}$ in $\mathbf{R}^n$, the goal is to identify a set $\mathcal{O} \subseteq \mathcal{D}$ that are anomalous in some sense. Roughly speaking, we think of an outlier as a point that is far away from most of the points, so we would like the points in $\mathcal{D} \setminus \mathcal{O}$ to be relatively close together, and to be relatively far apart from the points in $\mathcal{O}$.

We describe a heuristic technique for identifying $\mathcal{O}$. We start with $\mathcal{O} = \emptyset$ and find the minimum volume (Löwner-John) ellipsoid $\mathcal{E}$ containing all $x_i \notin \mathcal{O}$ (which is all x_i in the first step). Each iteration, we flag (*i.e.*, add to $\mathcal{O}$) the point that corresponds to the largest dual variable for the constraint $x_i \in \mathcal{E}$; this point will be one of the points on the boundary of $\mathcal{E}$, and intuitively, it will be the one for whom the constraint is 'most' binding. We then plot $\mathbf{vol}\,\mathcal{E}$ (on a log scale) versus $\mathbf{card}\,\mathcal{O}$ and hope that we see a sharp drop in the curve. We use the value of $\mathcal{O}$ after the drop.

The hope is that after removing a relatively small number of points, the volume of the minimum volume ellipsoid containing the remaining points will be much smaller than the minimum volume ellipsoid for $\mathcal{D}$, which means the removed points are far away from the others.

For example, suppose we have 100 points that lie in the unit ball and 3 points with (Euclidean) norm 1000. Intuitively, it is clear that it is reasonable to consider the three large points outliers. The minimum volume ellipsoid of all 103 points will have very large volume. The three points will be the first ones removed, and as soon as they are, the volume of the ellipsoid ellipsoid will drop dramatically and be on the order of the volume of the unit ball.

Run 6 iterations of the algorithm on the data given in `ellip_anomaly_data.m`. Plot $\mathbf{vol}\,\mathcal{E}$ (on a log scale) versus $\mathbf{card}\,\mathcal{O}$. In addition, on a single plot, plot all the ellipses found with the function `ellipse_draw(A,b)` along with the outliers (in red) and the remaining points (in blue).

Of course, we have chosen an example in $\mathbf{R}^2$ so the ellipses can be plotted, but one can detect outliers in $\mathbf{R}^2$ simply by inspection. In dimension much higher than 3, however, detecting outliers by plotting will become substantially more difficult, while the same algorithm can be used.

Note. In CVX, you should use `det_rootn` (which is SDP-representable and handled exactly) instead of `log_det` (which is handled using an inefficient iterative procedure).

8 Unconstrained and equality constrained minimization

8.1 *Gradient descent and nondifferentiable functions.*

(a) Let $\gamma > 1$. Show that the function

$$f(x_1, x_2) = \begin{cases} \sqrt{x_1^2 + \gamma x_2^2} & |x_2| \leq x_1 \\ \dfrac{x_1 + \gamma |x_2|}{\sqrt{1+\gamma}} & \text{otherwise} \end{cases}$$

is convex. You can do this, for example, by verifying that

$$f(x_1, x_2) = \sup \left\{ x_1 y_1 + \sqrt{\gamma} x_2 y_2 \;\middle|\; y_1^2 + y_2^2 \leq 1,\ y_1 \geq 1/\sqrt{1+\gamma} \right\}.$$

Note that f is unbounded below. (Take $x_2 = 0$ and let x_1 go to $-\infty$.)

(b) Consider the gradient descent algorithm applied to f, with starting point $x^{(0)} = (\gamma, 1)$ and an exact line search. Show that the iterates are

$$x_1^{(k)} = \gamma \left(\frac{\gamma - 1}{\gamma + 1} \right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma - 1}{\gamma + 1} \right)^k.$$

Therefore $x^{(k)}$ converges to $(0, 0)$. However, this is not the optimum, since f is unbounded below.

8.2 *A characterization of the Newton decrement.* Let $f : \mathbf{R}^n \to \mathbf{R}$ be convex and twice differentiable, and let A be a $p \times n$-matrix with rank p. Suppose $\hat{x}$ is feasible for the equality constrained problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & Ax = b. \end{array}$$

Recall that the Newton step Δx at $\hat{x}$ can be computed from the linear equations

$$\begin{bmatrix} \nabla^2 f(\hat{x}) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ u \end{bmatrix} = \begin{bmatrix} -\nabla f(\hat{x}) \\ 0 \end{bmatrix},$$

and that the Newton decrement $\lambda(\hat{x})$ is defined as

$$\lambda(\hat{x}) = (-\nabla f(\hat{x})^T \Delta x)^{1/2} = (\Delta x^T \nabla^2 f(\hat{x}) \Delta x)^{1/2}.$$

Assume the coefficient matrix in the linear equations above is nonsingular and that $\lambda(\hat{x})$ is positive. Express the solution y of the optimization problem

$$\begin{array}{ll} \text{minimize} & \nabla f(\hat{x})^T y \\ \text{subject to} & Ay = 0 \\ & y^T \nabla^2 f(\hat{x}) y \leq 1 \end{array}$$

in terms of Newton step Δx and the Newton decrement $\lambda(\hat{x})$.

8.3 *Suggestions for exercises 9.30 in* Convex Optimization. We recommend the following to generate a problem instance:

```
n = 100;
m = 200;
randn('state',1);
A=randn(m,n);
```

Of course, you should try out your code with different dimensions, and different data as well.

In all cases, be sure that your line search *first* finds a step length for which the tentative point is in $\mathbf{dom}\, f$; if you attempt to evaluate f outside its domain, you'll get complex numbers, and you'll never recover.

To find expressions for $\nabla f(x)$ and $\nabla^2 f(x)$, use the chain rule (see Appendix A.4); if you attempt to compute $\partial^2 f(x)/\partial x_i \partial x_j$, you will be sorry.

To compute the Newton step, you can use `vnt=-H\g`.

8.4 *Suggestions for exercise 9.31 in* Convex Optimization. For 9.31a, you should try out $N = 1$, $N = 15$, and $N = 30$. You might as well compute and store the Cholesky factorization of the Hessian, and then back solve to get the search directions, even though you won't really see any speedup in Matlab for such a small problem. After you evaluate the Hessian, you can find the Cholesky factorization as `L=chol(H,'lower')`. You can then compute a search step as `-L'\(L\g)`, where `g` is the gradient at the current point. Matlab will do the right thing, *i.e.*, it will first solve `L\g` using forward substitution, and then it will solve `-L'\(L\g)` using backward substitution. Each substitution is order n^2.

To fairly compare the convergence of the three methods (*i.e.*, $N = 1$, $N = 15$, $N = 30$), the horizontal axis should show the approximate total number of flops required, and not the number of iterations. You can compute the approximate number of flops using $n^3/3$ for each factorization, and $2n^2$ for each solve (where each 'solve' involves a forward substitution step and a backward substitution step).

8.5 *Efficient numerical method for a regularized least-squares problem.* We consider a regularized least squares problem with smoothing,

$$\text{minimize} \quad \sum_{i=1}^{k}(a_i^T x - b_i)^2 + \delta \sum_{i=1}^{n-1}(x_i - x_{i+1})^2 + \eta \sum_{i=1}^{n} x_i^2,$$

where $x \in \mathbf{R}^n$ is the variable, and $\delta, \eta > 0$ are parameters.

(a) Express the optimality conditions for this problem as a set of linear equations involving x. (These are called the normal equations.)

(b) Now assume that $k \ll n$. Describe an efficient method to solve the normal equations found in part (a). Give an approximate flop count for a general method that does not exploit structure, and also for your efficient method.

(c) *A numerical instance.* In this part you will try out your efficient method. We'll choose $k = 100$ and $n = 4000$, and $\delta = \eta = 1$. First, randomly generate A and b with these dimensions. Form the normal equations as in part (a), and solve them using a generic method. Next, write (short) code implementing your efficient method, and run it on your problem instance. Verify that the solutions found by the two methods are nearly the same, and also that your efficient method is much faster than the generic one.

Note: You'll need to know some things about Matlab to be sure you get the speedup from the efficient method. Your method should involve solving linear equations with tridiagonal coefficient matrix. In this case, both the factorization and the back substitution can be carried out very efficiently. The Matlab documentation says that banded matrices are recognized and exploited, when solving equations, but we found this wasn't always the case. To be sure Matlab knows your matrix is tridiagonal, you can declare the matrix as sparse, using `spdiags`, which can be used to create a tridiagonal matrix. You could also create the tridiagonal matrix conventionally, and then convert the resulting matrix to a sparse one using `sparse`.

One other thing you need to know. Suppose you need to solve a group of linear equations with the same coefficient matrix, *i.e.*, you need to compute $F^{-1}a_1, ..., F^{-1}a_m$, where F is invertible and a_i are column vectors. By concatenating columns, this can be expressed as a single matrix

$$\left[F^{-1}a_1 \ \cdots \ F^{-1}a_m\right] = F^{-1}\left[a_1 \ \cdots \ a_m\right].$$

To compute this matrix using Matlab, you should collect the righthand sides into one matrix (as above) and use Matlab's backslash operator: `F\A`. This will do the right thing: factor the matrix F once, and carry out multiple back substitutions for the righthand sides.

8.6 *Newton method for approximate total variation de-noising.* Total variation de-noising is based on the bi-criterion problem with the two objectives

$$\|x - x^{\mathrm{cor}}\|_2, \qquad \phi_{\mathrm{tv}}(x) = \sum_{i=1}^{n-1} |x_{i+1} - x_i|.$$

Here $x^{\mathrm{cor}} \in \mathbf{R}^n$ is the (given) corrupted signal, $x \in \mathbf{R}^n$ is the de-noised signal to be computed, and ϕ_{tv} is the total variation function. This bi-criterion problem can be formulated as an SOCP, or, by squaring the first objective, as a QP. In this problem we consider a method used to approximately formulate the total variation de-noising problem as an unconstrained problem with twice differentiable objective, for which Newton's method can be used.

We first observe that the Pareto optimal points for the bi-criterion total variation de-noising problem can found as the minimizers of the function

$$\|x - x^{\mathrm{cor}}\|_2^2 + \mu\phi_{\mathrm{tv}}(x),$$

where $\mu \geq 0$ is parameter. (Note that the Euclidean norm term has been squared here, and so is twice differentiable.) In *approximate total variation de-noising*, we substitute a twice differentiable approximation of the total variation function,

$$\phi_{\mathrm{atv}}(x) = \sum_{i=1}^{n-1} \left(\sqrt{\epsilon^2 + (x_{i+1} - x_i)^2} - \epsilon\right),$$

for the total variation function ϕ_{tv}. Here $\epsilon > 0$ is parameter that controls the level of approximation. In approximate total variation de-noising, we use Newton's method to minimize

$$\psi(x) = \|x - x^{\mathrm{cor}}\|_2^2 + \mu\phi_{\mathrm{atv}}(x).$$

(The parameters $\mu > 0$ and $\epsilon > 0$ are given.)

(a) Find expressions for the gradient and Hessian of ψ.

(b) Explain how you would exploit the structure of the Hessian to compute the Newton direction for ψ efficiently. (Your explanation can be brief.) Compare the approximate flop count for your method with the flop count for a generic method that does not exploit any structure in the Hessian of ψ.

(c) Implement Newton's method for approximate total variation de-noising. Get the corrupted signal $x^{\rm cor}$ from the file `approx_tv_denoising_data.m`, and compute the de-noised signal $x^\star$, using parameters $\epsilon = 0.001$, $\mu = 50$ (which are also in the file). Use line search parameters $\alpha = 0.01$, $\beta = 0.5$, initial point $x^{(0)} = 0$, and stopping criterion $\lambda^2/2 \leq 10^{-8}$. Plot the Newton decrement versus iteration, to verify asymptotic quadratic convergence. Plot the final smoothed signal $x^\star$, along with the corrupted one $x^{\rm cor}$.

8.7 Derive the Newton equation for the unconstrained minimization problem

$$\text{minimize} \quad (1/2)x^T x + \log \textstyle\sum_{i=1}^m \exp(a_i^T x + b_i).$$

Give an efficient method for solving the Newton system, assuming the matrix $A \in \mathbf{R}^{m\times n}$ (with rows a_i^T) is dense with $m \ll n$. Give an approximate flop count of your method.

8.8 We consider the equality constrained problem

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) - \log\det X \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1}. \end{array}$$

The variable is the matrix $X \in \mathbf{S}^n$. The domain of the objective function is $\mathbf{S}^n_{++}$. The matrix $C \in \mathbf{S}^n$ is a problem parameter. This problem is similar to the analytic centering problem discussed in lecture 11 (p.18–19) and pages 553-555 of the textbook. The differences are the extra linear term $\mathbf{tr}(CX)$ in the objective, and the special form of the equality constraints. (Note that the equality constraints can be written as $\mathbf{tr}(A_i X) = 1$ with $A_i = e_i e_i^T$, a matrix of zeros except for the i,i element, which is equal to one.)

(a) Show that X is optimal if and only if

$$X \succ 0, \qquad X^{-1} - C \text{ is diagonal}, \qquad \mathbf{diag}(X) = \mathbf{1}.$$

(b) The Newton step ΔX at a feasible X is defined as the solution of the Newton equations

$$X^{-1}\Delta X X^{-1} + \mathbf{diag}(w) = -C + X^{-1}, \qquad \mathbf{diag}(\Delta X) = 0,$$

with variables $\Delta X \in \mathbf{S}^n$, $w \in \mathbf{R}^n$. (Note the two meanings of the **diag** function: $\mathbf{diag}(w)$ is the diagonal matrix with the vector w on its diagonal; $\mathbf{diag}(\Delta X)$ is the vector of the diagonal elements of ΔX.) Eliminating ΔX from the first equation gives an equation

$$\mathbf{diag}(X\,\mathbf{diag}(w)X) = \mathbf{1} - \mathbf{diag}(XCX).$$

This is a set of n linear equations in n variables, so it can be written as $Hw = g$. Give a simple expression for the coefficients of the matrix H.

(c) Implement the feasible Newton method in Matlab. You can use $X = I$ as starting point. The code should terminate when $\lambda(X)^2/2 \leq 10^{-6}$, where $\lambda(X)$ is the Newton decrement.

You can use the Cholesky factorization to evaluate the cost function: if $X = LL^T$ where L is triangular with positive diagonal then $\log\det X = 2\sum_i \log L_{ii}$.

To ensure that the iterates remain feasible, the line search has to consist of two phases. Starting at $t = 1$, you first need to backtrack until $X + t\Delta X \succ 0$. Then you continue the backtracking until the condition of sufficient decrease

$$f_0(X + t\Delta X) \leq f_0(X) + \alpha t\,\mathbf{tr}(\nabla f_0(X)\Delta X)$$

is satisfied. To check that a matrix $X + t\Delta X$ is positive definite, you can use the Cholesky factorization with two output arguments (`[R, p] = chol(A)` returns $p > 0$ if A is not positive definite).

Test your code on randomly generated problems of sizes $n = 10, \ldots, 100$ (for example, using `n = 100; C = randn(n); C = C + C'`).

8.9 *Estimation of a vector from one-bit measurements.* A system of m sensors is used to estimate an unknown parameter $x \in \mathbf{R}^n$. Each sensor makes a noisy measurement of some linear combination of the unknown parameters, and quantizes the measured value to one bit: it returns $+1$ if the measured value exceeds a certain threshold, and -1 otherwise. In other words, the output of sensor i is given by

$$y_i = \mathbf{sign}(a_i^T x + v_i - b_i) = \begin{cases} 1 & a_i^T x + v_i \geq b_i \\ -1 & a_i^T x + v_i < b_i, \end{cases}$$

where a_i and b_i are known, and v_i is measurement error. We assume that the measurement errors v_i are independent random variables with a zero-mean unit-variance Gaussian distribution (*i.e.*, with a probability density $\phi(v) = (1/\sqrt{2\pi})e^{-v^2/2}$). As a consequence, the sensor outputs y_i are random variables with possible values ± 1. We will denote $\mathbf{prob}(y_i = 1)$ as $P_i(x)$ to emphasize that it is a function of the unknown parameter x:

$$\begin{aligned} P_i(x) &= \mathbf{prob}(y_i = 1) = \mathbf{prob}(a_i^T x + v_i \geq b_i) = \frac{1}{\sqrt{2\pi}} \int_{b_i - a_i^T x}^{\infty} e^{-t^2/2} dt \\ 1 - P_i(x) &= \mathbf{prob}(y_i = -1) = \mathbf{prob}(a_i^T x + v_i < b_i) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{b_i - a_i^T x} e^{-t^2/2} dt. \end{aligned}$$

The problem is to estimate x, based on observed values $\bar{y}_1$, $\bar{y}_2$, $\ldots$, $\bar{y}_m$ of the m sensor outputs.

We will apply the maximum likelihood (ML) principle to determine an estimate $\hat{x}$. In maximum likelihood estimation, we calculate $\hat{x}$ by maximizing the *log-likelihood function*

$$l(x) = \log\left(\prod_{\bar{y}_i = 1} P_i(x) \prod_{\bar{y}_i = -1} (1 - P_i(x))\right) = \sum_{\bar{y}_i = 1} \log P_i(x) + \sum_{\bar{y}_i = -1} \log(1 - P_i(x)).$$

(a) Show that the maximum likelihood estimation problem

$$\text{maximize } l(x)$$

is a convex optimization problem. The variable is x. The measured vector $\bar{y}$, and the parameters a_i and b_i are given.

(b) Solve the ML estimation problem with data defined in `one_bit_meas_data.m`, using Newton's method with backtracking line search. This file will define a matrix A (with rows a_i^T), a vector b, and a vector $\bar{y}$ with elements ± 1.

Remark. The Matlab functions `erfc` and `erfcx` are useful to evaluate the following functions:

$$\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{u} e^{-t^2/2}\,dt = \frac{1}{2}\mathbf{erfc}(-\frac{u}{\sqrt{2}}), \qquad \frac{1}{\sqrt{2\pi}}\int_{u}^{\infty} e^{-t^2/2}\,dt = \frac{1}{2}\mathbf{erfc}(\frac{u}{\sqrt{2}})$$

$$\frac{1}{\sqrt{2\pi}}e^{u^2/2}\int_{-\infty}^{u} e^{-t^2/2}\,dt = \frac{1}{2}\mathbf{erfcx}(-\frac{u}{\sqrt{2}}), \qquad \frac{1}{\sqrt{2\pi}}e^{u^2/2}\int_{u}^{\infty} e^{-t^2/2}\,dt = \frac{1}{2}\mathbf{erfcx}(\frac{u}{\sqrt{2}}).$$

8.10 *Functions with bounded Newton decrement.* Let $f : \mathbf{R}^n \to \mathbf{R}$ be a convex function with $\nabla^2 f(x) \succ 0$ for all $x \in \mathbf{dom}\, f$ and Newton decrement bounded by a positive constant c:

$$\lambda(x)^2 \leq c \quad \forall x \in \mathbf{dom}\, f.$$

Show that the function $g(x) = \exp(-f(x)/c)$ is concave.

8.11 *Monotone convergence of Newton's method.* Suppose $f : \mathbf{R} \to \mathbf{R}$ is strongly convex and smooth, and in addition, $f''' \leq 0$. Let $x^\star$ minimize f, and suppose Newton's method is initialized with $x^{(0)} < x^\star$. Show that the iterates $x^{(k)}$ converge to $x^\star$ monotonically, and that a backtracking line search always takes a step size of one, *i.e.*, $t^{(k)} = 1$.

8.12 *Infeasible start Newton method for LP centering problem.* Implement the infeasible start Newton method for solving the centering problem arising in the standard form LP,

$$\begin{array}{ll} \text{minimize} & c^T x - \sum_{i=1}^n \log x_i \\ \text{subject to} & Ax = b, \end{array}$$

with variable x. The data are $A \in \mathbf{R}^{m\times n}$, with $m < n$, $c \in \mathbf{R}^n$, and $b \in \mathbf{R}^m$. You can assume that A is full rank. This problem cannot be solved when it is infeasible or unbounded below.

Your code should accept A, b, c, and x_0, and return $x^\star$, the primal optimal point, $\nu^\star$, a dual optimal point, and the number of Newton steps executed. The initial point $x^{(0)}$ must satisfy $x^{(0)} \succ 0$, but it need not satisfy the equality constraints.

Use the block elimination method to compute the Newton step. (You can also compute the Newton step via the KKT system, and compare the result to the Newton step computed via block elimination. The two steps should be close, but if any x_i is very small, you might get a warning about the condition number of the KKT matrix.)

Plot $\|r(x,\nu)\|_2$, the norm of the concatenated primal and dual residuals, versus iteration k for various problem data and initial points, to verify that your implementation achieves quadratic convergence. As stopping criterion, you can use $\|r(x,\nu)\|_2 \leq 10^{-6}$ (which means the problem was solved) or some maximum number of iterations (say, 50) was reached, which means it was not solved (likely because the problem is either infeasible or unbounded below).

For a fixed problem instance, experiment with varying the algorithm parameters α and β, observing the effect on the total number of Newton steps required.

To generate problem data (*i.e.*, A, b, c, x_0) that are feasible, you can first generate A, then random positive vector p, and set $b = Ap$. You can be sure that the problem is not unbounded by making one row of A have positive entries. You may also want to check that A is full rank.

Test the behavior of your implementation on data instances that are not feasible, and also ones that are unbounded below.

9 Interior point methods

9.1 *Dual feasible point from analytic center.* We consider the problem

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m, \end{array} \tag{34}$$

where the functions f_i are convex and differentiable. For $u > p^\star$, define $x_{\rm ac}(u)$ as the analytic center of the inequalities

$$f_0(x) \leq u, \qquad f_i(x) \leq 0, \quad i = 1, \ldots, m,$$

i.e.,

$$x_{\rm ac}(u) = \operatorname{argmin} \left(-\log(u - f_0(x)) - \sum_{i=1}^m \log(-f_i(x)) \right).$$

Show that $\lambda \in \mathbf{R}^m$, defined by

$$\lambda_i = \frac{u - f_0(x_{\rm ac}(u))}{-f_i(x_{\rm ac}(u))}, \quad i = 1, \ldots, m$$

is dual feasible for the problem above. Express the corresponding dual objective value in terms of u, $x_{\rm ac}(u)$ and the problem parameters.

9.2 *Efficient solution of Newton equations.* Explain how you would solve the Newton equations in the barrier method applied to the quadratic program

$$\begin{array}{ll} \text{minimize} & (1/2)x^T x + c^T x \\ \text{subject to} & Ax \preceq b \end{array}$$

where $A \in \mathbf{R}^{m \times n}$ is dense. Distinguish two cases, $m \gg n$ and $n \gg m$, and give the most efficient method in each case.

9.3 *Efficient solution of Newton equations.* Describe an efficient method for solving the Newton equation in the barrier method for the quadratic program

$$\begin{array}{ll} \text{minimize} & (1/2)(x - a)^T P^{-1}(x - a) \\ \text{subject to} & 0 \preceq x \preceq \mathbf{1}, \end{array}$$

with variable $x \in \mathbf{R}^n$. The matrix $P \in \mathbf{S}^n$ and the vector $a \in \mathbf{R}^n$ are given.

Assume that the matrix P is large, positive definite, and sparse, and that P^{-1} is dense. 'Efficient' means that the complexity of the method should be much less than $O(n^3)$.

9.4 *Dual feasible point from incomplete centering.* Consider the SDP

$$\begin{array}{ll} \text{minimize} & \mathbf{1}^T x \\ \text{subject to} & W + \mathbf{diag}(x) \succeq 0, \end{array}$$

with variable $x \in \mathbf{R}^n$, and its dual

$$\begin{array}{ll} \text{maximize} & -\mathbf{tr}\, WZ \\ \text{subject to} & Z_{ii} = 1, \quad i = 1, \ldots, n \\ & Z \succeq 0, \end{array}$$

with variable $X \in \mathbf{S}^n$. (These problems arise in a relaxation of the two-way partitioning problem, described on page 219; see also exercises 5.39 and 11.23.)

Standard results for the barrier method tell us that when x is on the central path, *i.e.*, minimizes the function

$$\phi(x) = t\mathbf{1}^T x + \log\det(W + \mathbf{diag}(x))^{-1}$$

for some parameter $t > 0$, the matrix

$$Z = \frac{1}{t}(W + \mathbf{diag}(x))^{-1}$$

is dual feasible, with objective value $-\mathbf{tr}\, WZ = \mathbf{1}^T x - n/t$.

Now suppose that x is strictly feasible, but not necessarily on the central path. (For example, x might be the result of using Newton's method to minimize ϕ, but with early termination.) Then the matrix Z defined above will not be dual feasible. In this problem we will show how to construct a dual feasible $\hat{Z}$ (which agrees with Z as given above when x is on the central path), from any point x that is *near* the central path. Define $X = W + \mathbf{diag}(x)$, and let $v = -\nabla^2\phi(x)^{-1}\nabla\phi(x)$ be the Newton step for the function ϕ defined above. Define

$$\hat{Z} = \frac{1}{t}\left(X^{-1} - X^{-1}\,\mathbf{diag}(v)X^{-1}\right).$$

(a) Verify that when x is on the central path, we have $\hat{Z} = Z$.

(b) Show that $\hat{Z}_{ii} = 1$, for $i = 1, \ldots, n$.

(c) Let $\lambda(x) = \nabla\phi(x)^T\nabla^2\phi(x)^{-1}\nabla\phi(x)$ be the Newton decrement at x. Show that

$$\lambda(x) = \mathbf{tr}(X^{-1}\,\mathbf{diag}(v)X^{-1}\,\mathbf{diag}(v)) = \mathbf{tr}(X^{-1/2}\,\mathbf{diag}(v)X^{-1/2})^2.$$

(d) Show that $\lambda(x) < 1$ implies that $\hat{Z} \succ 0$. Thus, when x is near the central path (meaning, $\lambda(x) < 1$), Z is dual feasible.

9.5 *Standard form LP barrier method.* In the following three parts of this exercise, you will implement a barrier method for solving the standard form LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \quad x \succeq 0, \end{array}$$

with variable $x \in \mathbf{R}^n$, where $A \in \mathbf{R}^{m\times n}$, with $m < n$. Throughout these exercises we will assume that A is full rank, and the sublevel sets $\{x \mid Ax = b,\ x \succeq 0,\ c^T x \leq \gamma\}$ are all bounded. (If this is not the case, the centering problem is unbounded below.)

(a) *Centering step.* Implement Newton's method for solving the centering problem

$$\begin{array}{ll} \text{minimize} & c^T x - \sum_{i=1}^n \log x_i \\ \text{subject to} & Ax = b, \end{array}$$

with variable x, given a strictly feasible starting point x_0.

Your code should accept A, b, c, and x_0, and return $x^\star$, the primal optimal point, $\nu^\star$, a dual optimal point, and the number of Newton steps executed.

Use the block elimination method to compute the Newton step. (You can also compute the Newton step via the KKT system, and compare the result to the Newton step computed via block elimination. The two steps should be close, but if any x_i is very small, you might get a warning about the condition number of the KKT matrix.)

Plot $\lambda^2/2$ versus iteration k, for various problem data and initial points, to verify that your implementation gives asymptotic quadratic convergence. As stopping criterion, you can use $\lambda^2/2 \leq 10^{-6}$. Experiment with varying the algorithm parameters α and β, observing the effect on the total number of Newton steps required, for a fixed problem instance. Check that your computed $x^\star$ and $\nu^\star$ (nearly) satisfy the KKT conditions.

To generate some random problem data (*i.e.*, A, b, c, x_0), we recommend the following approach. First, generate A randomly. (You might want to check that it has full rank.) Then generate a random positive vector x_0, and take $b = Ax_0$. (This ensures that x_0 is strictly feasible.) The parameter c can be chosen randomly. To be sure the sublevel sets are bounded, you can add a row to A with all positive elements. If you want to be able to repeat a run with the same problem data, be sure to set the state for the uniform and normal random number generators.

Here are some hints that may be useful.

- We recommend computing λ^2 using the formula $\lambda^2 = -\Delta x_{\rm nt}^T \nabla f(x)$. You don't really need λ for anything; you can work with λ^2 instead. (This is important for reasons described below.)
- There can be small numerical errors in the Newton step $\Delta x_{\rm nt}$ that you compute. When x is nearly optimal, the computed value of λ^2, *i.e.*, $\lambda^2 = -\Delta x_{\rm nt}^T \nabla f(x)$, can actually be (slightly) negative. If you take the squareroot to get λ, you'll get a complex number, and you'll never recover. Moreover, your line search will never exit. However, this only happens when x is nearly optimal. So if you exit on the condition $\lambda^2/2 \leq 10^{-6}$, everything will be fine, even when the computed value of λ^2 is negative.
- For the line search, you must first multiply the step size t by β until $x + t\Delta x_{\rm nt}$ is feasible (*i.e.*, strictly positive). If you don't, when you evaluate f you'll be taking the logarithm of negative numbers, and you'll never recover.

(b) *LP solver with strictly feasible starting point.* Using the centering code from part (a), implement a barrier method to solve the standard form LP

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \quad x \succeq 0, \end{array}$$

with variable $x \in \mathbf{R}^n$, given a strictly feasible starting point x_0. Your LP solver should take as argument A, b, c, and x_0, and return $x^\star$.

You can terminate your barrier method when the duality gap, as measured by n/t, is smaller than 10^{-3}. (If you make the tolerance much smaller, you might run into some numerical trouble.) Check your LP solver against the solution found by CVX*, for several problem instances.

The comments in part (a) on how to generate random data hold here too.

Experiment with the parameter μ to see the effect on the number of Newton steps per centering step, and the total number of Newton steps required to solve the problem.

Plot the progress of the algorithm, for a problem instance with $n = 500$ and $m = 100$, showing duality gap (on a log scale) on the vertical axis, versus the cumulative total number of Newton steps (on a linear scale) on the horizontal axis.

Your algorithm should return a $2 \times k$ matrix `history`, (where k is the total number of centering steps), whose first row contains the number of Newton steps required for each centering step, and whose second row shows the duality gap at the end of each centering step. In order to get a plot that looks like the ones in the book (*e.g.*, figure 11.4, page 572), you should use the following code:

```
[xx, yy] = stairs(cumsum(history(1,:)),history(2,:));
semilogy(xx,yy);
```

(c) *LP solver.* Using the code from part (b), implement a general standard form LP solver, that takes arguments A, b, c, determines (strict) feasibility, and returns an optimal point if the problem is (strictly) feasible.

You will need to implement a phase I method, that determines whether the problem is strictly feasible, and if so, finds a strictly feasible point, which can then be fed to the code from part (b). In fact, you can use the code from part (b) to implement the phase I method.

To find a strictly feasible initial point x_0, we solve the phase I problem

$$\begin{array}{ll} \text{minimize} & t \\ \text{subject to} & Ax = b \\ & x \succeq (1-t)\mathbf{1}, \quad t \geq 0, \end{array}$$

with variables x and t. If we can find a feasible (x, t), with $t < 1$, then x is strictly feasible for the original problem. The converse is also true, so the original LP is strictly feasible if and only if $t^\star < 1$, where $t^\star$ is the optimal value of the phase I problem.

We can initialize x and t for the phase I problem with any x^0 satisfying $Ax^0 = b$, and $t^0 = 2 - \min_i x_i^0$. (Here we can assume that $\min x_i^0 \leq 0$; otherwise x^0 is already a strictly feasible point, and we are done.) You can use a change of variable $z = x + (t-1)\mathbf{1}$ to transform the phase I problem into the form in part (b).

Check your LP solver against CVX* on several numerical examples, including both feasible and infeasible instances.

9.6 *Primal and dual feasible points in the barrier method for LP.* Consider a standard form LP and its dual

$$\begin{array}{lll} \text{minimize} & c^T x & \\ \text{subject to} & Ax = b & \\ & x \succeq 0 & \end{array} \qquad \begin{array}{ll} \text{maximize} & b^T y \\ \text{subject to} & A^T y \preceq c, \end{array}$$

with $A \in \mathbf{R}^{m \times n}$ and $\mathbf{rank}(A) = m$. In the barrier method the (feasible) Newton method is applied to the equality constrained problem

$$\begin{array}{ll} \text{minimize} & tc^T x + \phi(x) \\ \text{subject to} & Ax = b, \end{array}$$

where $t > 0$ and $\phi(x) = -\sum_{i=1}^{n} \log x_i$. The Newton equation at a strictly feasible $\hat{x}$ is given by

$$\begin{bmatrix} \nabla^2 \phi(\hat{x}) & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ w \end{bmatrix} = \begin{bmatrix} -tc - \nabla \phi(\hat{x}) \\ 0 \end{bmatrix}.$$

Suppose $\lambda(\hat{x}) \leq 1$ where $\lambda(\hat{x})$ is the Newton decrement at $\hat{x}$.

(a) Show that $\hat{x} + \Delta x$ is primal feasible.

(b) Show that $y = -(1/t)w$ is dual feasible.

(c) Let $p^\star$ be the optimal value of the LP. Show that

$$c^T \hat{x} - p^\star \leq \frac{n + \lambda(\hat{x})\sqrt{n}}{t}.$$

9.7 Consider a convex optimization problem and its dual

$$\begin{array}{lll} \text{minimize} & f_0(x) & \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, m, & \end{array} \qquad \begin{array}{lll} \text{maximize} & g(\lambda) & \\ \text{subject to} & \lambda \succeq 0. & \end{array} \tag{35}$$

The centering problem in the barrier method is

$$\text{minimize} \quad tf_0(x) - \sum_{i=1}^{m} \log(-f_i(x)), \tag{36}$$

where t is a positive parameter.

(a) The centering problem can be written as

$$\begin{array}{ll} \text{minimize} & tf_0(x) - \sum_{i=1}^{m} \log(y_i) \\ \text{subject to} & f_i(x) + y_i \leq 0, \quad i = 1, \ldots, m, \end{array}$$

with variables x and y. Derive the Lagrange dual of this problem and express it in terms of the dual function $g(\lambda)$ in (35).

(b) Suppose the feasible set of the dual problem in (35) contains strictly positive λ. Show that the centering problem (36) is bounded below for any positive t.

9.8 *Standard form LP barrier method with infeasible start Newton method.* Implement the barrier method for the standard form LP,

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & Ax = b, \quad x \succeq 0, \end{array}$$

with variable $x \in \mathbf{R}^n$, where $A \in \mathbf{R}^{m \times n}$, with $m < n$, with A full rank. (Your method will of course fail if the problem is not strictly feasible, or if it is unbounded.)

Use the centering code that you developed in exercise 8.12. Your LP solver should take as argument A, b, c, and return primal and dual optimal points $x^\star$, $\nu^\star$, and $\lambda^\star$.

You can terminate your barrier method when the duality gap, as measured by n/t, is smaller than 10^{-3}. (If you make the tolerance much smaller, you might run into some numerical trouble.)

Check your LP solver against the solution found by CVX* for several problem instances. The comments in exercise 8.12 on how to generate random data hold here too.

Experiment with the parameter μ to see the effect on the number of Newton steps per centering step, and the total number of Newton steps required to solve the problem.

Plot the progress of the algorithm, for a problem instance with $n = 500$ and $m = 100$, showing duality gap (on a log scale) on the vertical axis, versus the cumulative total number of Newton steps (on a linear scale) on the horizontal axis.

Your algorithm should return a $2 \times k$ matrix `history`, (where k is the total number of centering steps), whose first row contains the number of Newton steps required for each centering step, and whose second row shows the duality gap at the end of each centering step. In order to get a plot that looks like the ones in the book (*e.g.*, figure 11.4, page 572), you should use the following code in Matlab:

```
[xx, yy] = stairs(cumsum(history(1,:)),history(2,:));
semilogy(xx,yy);
```

In Julia, with PyPlot you can use the following:

```
using PyPlot
step(cumsum(history[1,:]'),history[2,:]')
yscale("log")
```

In Python, also with PyPlot you should use:

```
import matplotlib.pyplot as plt
plt.step(np.cumsum(history[0,:]),history[1,:])
plt.yscale("log")
plt.show()
```

10 Mathematical background

10.1 *Some famous inequalities.* The Cauchy-Schwarz inequality states that

$$|a^T b| \leq \|a\|_2 \|b\|_2$$

for all vectors a, $b \in \mathbf{R}^n$ (see page 633 of the textbook).

(a) Prove the Cauchy-Schwarz inequality.
Hint. A simple proof is as follows. With a and b fixed, consider the function $g(t) = \|a + tb\|_2^2$ of the scalar variable t. This function is nonnegative for all t. Find an expression for $\inf_t g(t)$ (the minimum value of g), and show that the Cauchy-Schwarz inequality follows from the fact that $\inf_t g(t) \geq 0$.

(b) The 1-norm of a vector x is defined as $\|x\|_1 = \sum_{k=1}^n |x_k|$. Use the Cauchy-Schwarz inequality to show that

$$\|x\|_1 \leq \sqrt{n}\|x\|_2$$

for all x.

(c) The *harmonic mean* of a positive vector $x \in \mathbf{R}^n_{++}$ is defined as

$$\left(\frac{1}{n}\sum_{k=1}^n \frac{1}{x_k}\right)^{-1}.$$

Use the Cauchy-Schwarz inequality to show that the arithmetic mean $(\sum_k x_k)/n$ of a positive n-vector is greater than or equal to its harmonic mean.

10.2 *Schur complements.* Consider a matrix $X = X^T \in \mathbf{R}^{n\times n}$ partitioned as

$$X = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix},$$

where $A \in \mathbf{R}^{k\times k}$. If $\det A \neq 0$, the matrix $S = C - B^T A^{-1} B$ is called the *Schur complement* of A in X. Schur complements arise in many situations and appear in many important formulas and theorems. For example, we have $\det X = \det A \det S$. (You don't have to prove this.)

(a) The Schur complement arises when you minimize a quadratic form over some of the variables. Let $f(u, v) = (u, v)^T X (u, v)$, where $u \in \mathbf{R}^k$. Let $g(v)$ be the minimum value of f over u, *i.e.*, $g(v) = \inf_u f(u, v)$. Of course $g(v)$ can be $-\infty$.
Show that if $A \succ 0$, we have $g(v) = v^T S v$.

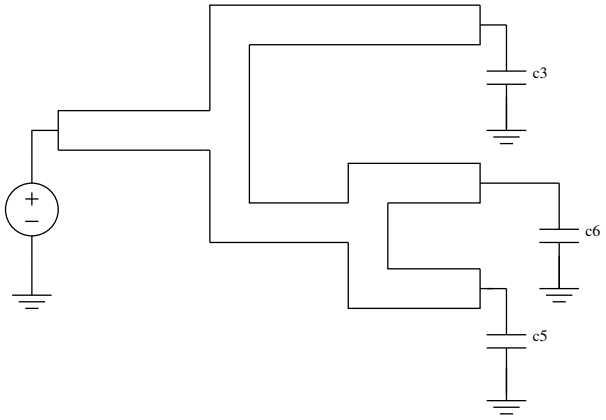
(b) The Schur complement arises in several characterizations of positive definiteness or semidefiniteness of a block matrix. As examples we have the following three theorems:

- $X \succ 0$ if and only if $A \succ 0$ and $S \succ 0$.
- If $A \succ 0$, then $X \succeq 0$ if and only if $S \succeq 0$.
- $X \succeq 0$ if and only if $A \succeq 0$, $B^T(I - AA^\dagger) = 0$ and $C - B^T A^\dagger B \succeq 0$, where $A^\dagger$ is the pseudo-inverse of A. ($C - B^T A^\dagger B$ serves as a generalization of the Schur complement in the case where A is positive semidefinite but singular.)

Prove *one* of these theorems. (You can choose which one.)

11 Circuit design

11.1 *Interconnect sizing.* In this problem we will size the interconnecting wires of the simple circuit shown below, with one voltage source driving three different capacitive loads C_{load1}, C_{load2}, and C_{load3}.



We divide the wires into 6 segments of fixed length l_i; our variables will be the widths w_i of the segments. (The height of the wires is related to the particular IC technology process, and is fixed.) The total area used by the wires is, of course,

$$A = \sum_i w_i l_i.$$

We'll take the lengths to be one, for simplicity. The wire widths must be between a minimum and maximum allowable value:

$$W_{\min} \leq w_i \leq W_{\max}.$$

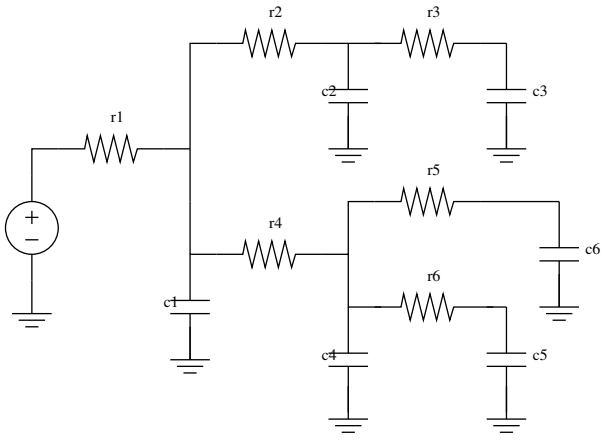
For our specific problem, we'll take $W_{\min} = 0.1$ and $W_{\max} = 10$.

Each of the wire segments will be modeled by a simple simple RC circuit, with the resistance inversely proportional to the width of the wire and the capacitance proportional to the width. (A far better model uses an extra constant term in the capacitance, but this complicates the equations.) The capacitance and resistance of the ith segment is thus

$$C_i = k_0 w_i, \qquad R_i = \rho / w_i,$$

where k_0 and ρ are positive constants, which we take to be one for simplicity. We also have $C_{load1} = 1.5$, $C_{load2} = 1$, and $C_{load3} = 5$.

Using the RC model for the wire segments yields the circuit shown below.



We will use the Elmore delay to model the delay from the source to each of the loads. The Elmore delay to loads 1, 2, and 3 are given by

$$\begin{aligned}
T_1 &= (C_3 + C_{load1})(R_1 + R_2 + R_3) + C_2(R_1 + R_2) + \\
&\quad +(C_1 + C_4 + C_5 + C_6 + C_{load2} + C_{load3})R_1 \\
T_2 &= (C_5 + C_{load2})(R_1 + R_4 + R_5) + C_4(R_1 + R_4) + \\
&\quad +(C_6 + C_{load3})(R_1 + R_4) + (C_1 + C_2 + C_3 + C_{load1})R_1 \\
T_3 &= (C_6 + C_{load3})(R_1 + R_4 + R_6) + C_4(R_1 + R_4) + \\
&\quad +(C_1 + C_2 + C_3 + C_{load1})R_1 + (C_5 + C_{load2})(R_1 + R_4).
\end{aligned}$$

Our main interest is in the maximum of these delays,

$$T = \max\{T_1, T_2, T_3\}.$$

(a) Explain how to find the optimal trade-off curve between area A and delay T.

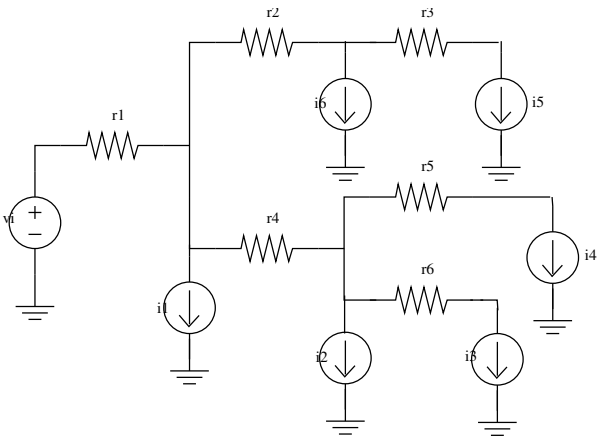
(b) *Optimal area-delay sizing.* For the specific problem parameters given, plot the area-delay trade-off curve, together with the individual Elmore delays. Comment on the results you obtain.

(c) *The simple method.* Plot the area-delay trade-off obtained when you assign all wire widths to be the same width (which varies between $W_{\min}$ and $W_{\max}$). Compare this curve to the optimal one, obtained in part (b). How much better does the optimal method do than the simple method? *Note:* for a large circuit, say with 1000 wires to size, the difference is *far larger*.

For this problem you can use the CVX in GP mode. We've also made available the function `elm_del_example.m`, which evaluates the three delays, given the widths of the wires.

11.2 *Optimal sizing of power and ground trees.* We consider a system or VLSI device with many subsystems or subcircuits, each of which needs one or more power supply voltages. In this problem we consider the case where the power supply network has a tree topology with the power supply (or external pin connection) at the root. Each node of the tree is connected to some subcircuit that draws power.

We model the power supply as a constant voltage source with value V. The m subcircuits are modeled as current sources that draw currents $i_1(t), \ldots, i_m(t)$ from the node (to ground) (see the figure below).



The subcircuit current draws have two components:

$$i_k(t) = i_k^{\mathrm{dc}} + i_k^{\mathrm{ac}}(t)$$

where i_k^{dc} is the DC current draw (which is a positive constant), and $i_k^{\mathrm{ac}}(t)$ is the AC draw (which has zero average value). We characterize the AC current draw by its RMS value, defined as

$$\mathrm{RMS}(i_k^{\mathrm{ac}}) = \left(\lim_{T\to\infty} \frac{1}{T} \int_0^T i_k^{\mathrm{ac}}(t)^2 \, dt \right)^{1/2}.$$

For each subcircuit we are given maximum values for the DC and RMS AC currents draws, *i.e.*, constants I_k^{dc} and I_k^{ac} such that

$$0 \leq i_k^{\mathrm{dc}} \leq I_k^{\mathrm{dc}}, \qquad \mathrm{RMS}(i_k^{\mathrm{ac}}) \leq I_k^{\mathrm{ac}}. \tag{37}$$

The n wires that form the distribution network are modeled as resistors R_k (which, presumably, have small value). (Since the circuit has a tree topology, we can use the following labeling convention: node k and the current source $i_k(t)$ are immediately following resistor R_k.) The resistance of the wires is given by

$$R_i = \alpha l_i / w_i,$$

where α is a constant and l_i are the lengths of the wires, which are known and fixed. The variables in the problem are the width of the wires, $w_1, \ldots, w_n$. Obviously by making the wires very wide, the resistances become very low, and we have a nearly ideal power network. The purpose of this problem is to optimally select wire widths, to minimize area while meeting certain specfications. Note that in this problem we ignore dynamics, *i.e.*, we do not model the capacitance or inductance of the wires.

As a result of the current draws and the nonzero resistance of the wires, the voltage at node k (which supplies subcircuit k) has a DC value less than the supply voltage, and also an AC voltage (which is called power supply ripple or noise). By superposition these two effects can be analyzed separately.

- The DC voltage drop $V - v_k^{\rm dc}$ at node k is equal to the sum of the voltage drops across wires on the (unique) path from node k to the root. It can be expressed as

$$V - v_k^{\rm dc} = \sum_{j=1}^{m} i_j^{\rm dc} \sum_{i \in \mathcal{N}(j,k)} R_i, \tag{38}$$

  where $\mathcal{N}(j,k)$ consists of the indices of the branches upstream from nodes j and k, *i.e.*, $i \in \mathcal{N}(j,k)$ if and only if R_i is in the path from node j to the root and in the path from node k to the root.

- The power supply noise at a node can be found as follows. The AC voltage at node k is equal to

$$v_k^{\rm ac}(t) = -\sum_{j=1}^{m} i_j^{\rm ac}(t) \sum_{i \in \mathcal{N}(j,k)} R_i.$$

  We assume the AC current draws are independent, so the RMS value of $v_k^{\rm ac}(t)$ is given by the squareroot of the sum of the squares of the RMS value of the ripple due to each other node, *i.e.*,

$$\mathrm{RMS}(v_k^{\rm ac}) = \left(\sum_{j=1}^{m} \left(\mathrm{RMS}(i_j^{\rm ac}) \sum_{i \in \mathcal{N}(j,k)} R_i \right)^2 \right)^{1/2}. \tag{39}$$

The problem is to choose wire widths w_i that minimize the total wire area $\sum_{i=k}^{n} w_k l_k$ subject to the following specifications:

- maximum allowable DC voltage drop at each node:

$$V - v_k^{\rm dc} \leq V_{\rm max}^{\rm dc}, \quad k = 1, \ldots, m, \tag{40}$$

  where $V - v_k^{\rm dc}$ is given by (38), and $V_{\rm max}^{\rm dc}$ is a given constant.

- maximum allowable power supply noise at each node:

$$\mathrm{RMS}(v_k^{\rm ac}) \leq V_{\rm max}^{\rm ac}, \quad k = 1, \ldots, m, \tag{41}$$

  where $\mathrm{RMS}(v_k^{\rm ac})$ is given by (39), and $V_{\rm max}^{\rm ac}$ is a given constant.

- upper and lower bounds on wire widths:

$$w_{\rm min} \leq w_i \leq w_{\rm max}, \quad i = 1, \ldots, n, \tag{42}$$

  where $w_{\rm min}$ and $w_{\rm max}$ are given constants.

- maximum allowable DC current density in a wire:

$$\left(\sum_{j \in \mathcal{M}(k)} i_j^{\rm dc} \right) \Big/ w_k \leq \rho_{\rm max}, \quad k = 1, \ldots, n, \tag{43}$$

  where $\mathcal{M}(k)$ is the set of all indices of nodes downstream from resistor k, *i.e.*, $j \in \mathcal{M}(k)$ if and only if R_k is in the path from node j to the root, and $\rho_{\rm max}$ is a given constant.

- maximum allowable total DC power dissipation in supply network:

$$\sum_{k=1}^{n} R_k \left(\sum_{j \in \mathcal{M}(k)} i_j^{\mathrm{dc}} \right)^2 \leq P_{\max}, \tag{44}$$

  where $P_{\max}$ is a given constant.

These specifications must be satisfied for all possible $i_k(t)$ that satisfy (37).

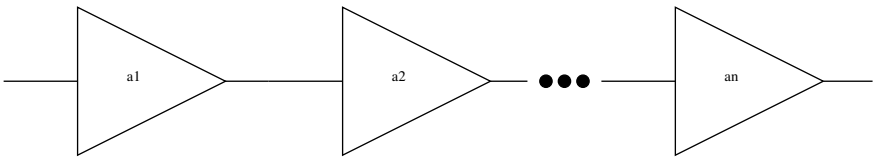
Formulate this as a convex optimization problem in the standard form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, \quad i = 1, \ldots, p \\ & Ax = b. \end{array}$$

You may introduce new variables, or use a change of variables, but you must say very clearly

- what the optimization variable x is, and how it corresponds to the problem variables w (*i.e.*, is x equal to w, does it include auxiliary variables, ...?)
- what the objective f_0 and the constraint functions f_i are, and how they relate to the objectives and specifications of the problem description
- why the objective and constraint functions are convex
- what A and b are (if applicable).

11.3 *Optimal amplifier gains.* We consider a system of n amplifiers connected (for simplicity) in a chain, as shown below. The variables that we will optimize over are the gains $a_1, \ldots, a_n > 0$ of the amplifiers. The first specification is that the overall gain of the system, *i.e.*, the product $a_1 \cdots a_n$, is equal to A^{tot}, which is given.



We are concerned about two effects: noise generated by the amplifiers, and amplifier overload. These effects are modeled as follows.

We first describe how the noise depends on the amplifier gains. Let N_i denote the noise level (RMS, or root-mean-square) at the output of the ith amplifier. These are given recursively as

$$N_0 = 0, \quad N_i = a_i \left(N_{i-1}^2 + \alpha_i^2 \right)^{1/2}, \quad i = 1, \ldots, n$$

where $\alpha_i > 0$ (which is given) is the ('input-referred') RMS noise level of the ith amplifier. The *output noise level* N_{out} of the system is given by $N_{\mathrm{out}} = N_n$, *i.e.*, the noise level of the last amplifier. Evidently N_{out} depends on the gains $a_1, \ldots, a_n$.

Now we describe the amplifier overload limits. S_i will denote the signal level at the output of the ith amplifier. These signal levels are related by

$$S_0 = S_{\text{in}}, \qquad S_i = a_i S_{i-1}, \quad i = 1, \ldots, n,$$

where $S_{\text{in}} > 0$ is the *input signal level*. Each amplifier has a maximum allowable output level $M_i > 0$ (which is given). (If this level is exceeded the amplifier will distort the signal.) Thus we have the constraints $S_i \leq M_i$, for $i = 1, \ldots, n$. (We can ignore the noise in the overload condition, since the signal levels are much larger than the noise levels.)

The *maximum output signal level* $S_{\max}$ is defined as the maximum value of S_n, over all input signal levels S_{in} that respect the the overload constraints $S_i \leq M_i$. Of course $S_{\max} \leq M_n$, but it can be smaller, depending on the gains $a_1, \ldots, a_n$.

The *dynamic range* D of the system is defined as $D = S_{\max}/N_{\text{out}}$. Evidently it is a (rather complicated) function of the amplifier gains $a_1, \ldots, a_n$.

The goal is to choose the gains a_i to maximize the dynamic range D, subject to the constraint $\prod_i a_i = A^{\text{tot}}$, and upper bounds on the amplifier gains, $a_i \leq A_i^{\max}$ (which are given).

Explain how to solve this problem as a convex (or quasiconvex) optimization problem. If you introduce new variables, or transform the variables, explain. Clearly give the objective and inequality constraint functions, explaining why they are convex if it is not obvious. If your problem involves equality constraints, give them explicitly.

Carry out your method on the specific instance with $n = 4$, and data

$$\begin{aligned} A^{\text{tot}} &= 10000, \\ \alpha &= (10^{-5}, 10^{-2}, 10^{-2}, 10^{-2}), \\ M &= (0.1, 5, 10, 10), \\ A^{\max} &= (40, 40, 40, 20). \end{aligned}$$

Give the optimal gains, and the optimal dynamic range.

11.4 *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.

11.5 *Solving nonlinear circuit equations using convex optimization.* An electrical circuit consists of b two-terminal devices (or branches) connected to n nodes, plus a so-called ground node. The goal is to compute several sets of physical quantities that characterize the circuit operation. The vector of *branch voltages* is $v \in \mathbf{R}^b$, where v_j is the voltage appearing across device j. The vector of *branch currents* is $i \in \mathbf{R}^b$, where i_j is the current flowing through device j. (The symbol i, which is often used to denote an index, is unfortunately the standard symbol used to denote current.) The vector of *node potentials* is $e \in \mathbf{R}^n$, where e_k is the potential of node k with respect to the ground node. (The ground node has potential zero by definition.)

The circuit variables v, i, and e satisfy several physical laws. Kirchhoff's current law (KCL) can be expressed as $Ai = 0$, and Kirchhoff's voltage law (KVL) can be expressed as $v = A^T e$, where $A \in \mathbf{R}^{n\times b}$ is the reduced incidence matrix, which describes the circuit topology:

$$A_{kj} = \begin{cases} -1 & \text{branch } j \text{ enters node } k \\ +1 & \text{branch } j \text{ leaves node } k \\ 0 & \text{otherwise,} \end{cases}$$

for $k = 1, \ldots, n$, $j = 1, \ldots, b$. (KCL states that current is conserved at each node, and KVL states that the voltage across each branch is the difference of the potentials of the nodes it is connected to.)

The branch voltages and currents are related by

$$v_j = \phi_j(i_j), \quad j = 1, \ldots, b,$$

where ϕ_j is a given function that depends on the *type* of device j. We will assume that these functions are continuous and nondecreasing. We give a few examples. If device j is a resistor with resistance $R_j > 0$, we have $\phi_j(i_j) = R_j i_j$ (which is called Ohm's law). If device j is a voltage source with voltage V_j and internal resistance $r_j > 0$, we have $\phi_j(i_j) = V_j + r_j i_j$. And for a more interesting example, if device j is a diode, we have $\phi_j(i_j) = V_T \log(1 + i_j/I_S)$, where I_S and V_T are known positive constants.

(a) Find a method to solve the circuit equations, *i.e.*, find v, i, and e that satisfy KCL, KVL, and the branch equations, that relies on convex optimization. State the optimization problem clearly, indicating what the variables are. Be sure to explain how solving the convex optimization problem you propose leads to choices of the circuit variables that satisfy all of the circuit

equations. You can assume that no pathologies occur in the problem that you propose, for example, it is feasible, a suitable constraint qualification holds, and so on.

Hint. You might find the function $\psi : \mathbf{R}^b \to \mathbf{R}$,

$$\psi(i_1, \ldots, i_b) = \sum_{j=1}^{b} \int_0^{i_j} \phi_j(u_j)\, du_j,$$

useful.

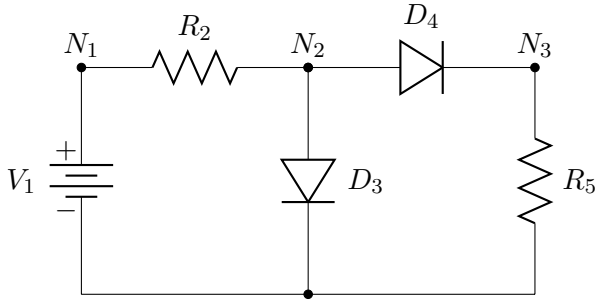
(b) Consider the circuit shown in the diagram below. Device 1 is a voltage source with parameters $V_1 = 1000$, $r_1 = 1$. Devices 2 and 5 are resistors with resistance $R_2 = 1000$, and $R_5 = 100$ respectively. Devices 3 and 4 are identical diodes with parameters $V_T = 26$, $I_S = 1$. (The units are mV, mA, and Ω.)

The nodes are labeled N_1, N_2, and N_3; the ground node is at the bottom. The incidence matrix A is

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}.$$

(The reference direction for each edge is down or to the right.)

Use the method in part (a) to compute v, i, and e. Verify that all the circuit equations hold.



12 Signal processing and communications

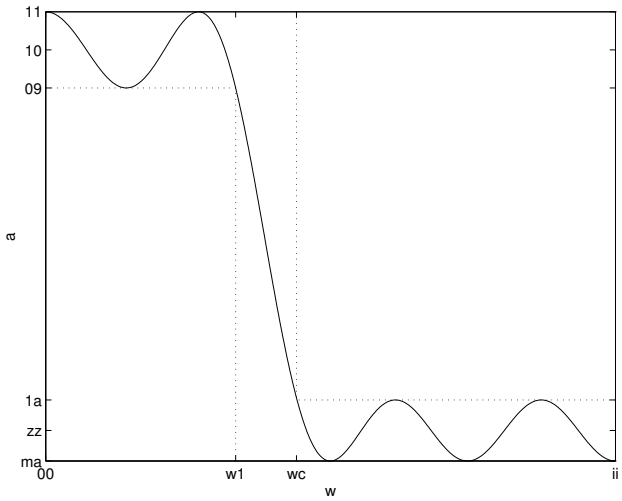
12.1 *FIR low-pass filter design.* Consider the (symmetric, linear phase) finite impulse response (FIR) filter described by its frequency response

$$H(\omega) = a_0 + \sum_{k=1}^{N} a_k \cos k\omega,$$

where $\omega \in [0, \pi]$ is the frequency. The design variables in our problems are the real coefficients $a = (a_0, \ldots, a_N) \in \mathbf{R}^{N+1}$, where N is called the order or length of the FIR filter. In this problem we will explore the design of a low-pass filter, with specifications:

- For $0 \leq \omega \leq \pi/3$, $0.89 \leq H(\omega) \leq 1.12$, *i.e.*, the filter has about $\pm$1dB ripple in the 'passband' $[0, \pi/3]$.
- For $\omega_c \leq \omega \leq \pi$, $|H(\omega)| \leq \alpha$. In other words, the filter achieves an attenuation given by α in the 'stopband' $[\omega_c, \pi]$. Here ω_c is called the filter 'cutoff frequency'.

(It is called a low-pass filter since low frequencies are allowed to pass, but frequencies above the cutoff frequency are attenuated.) These specifications are depicted graphically in the figure below.



For parts (a)–(c), explain how to formulate the given problem as a convex or quasiconvex optimization problem.

(a) *Maximum stopband attenuation.* We fix ω_c and N, and wish to maximize the stopband attenuation, *i.e.*, minimize α.

(b) *Minimum transition band.* We fix N and α, and want to minimize ω_c, *i.e.*, we set the stopband attenuation and filter length, and wish to minimize the 'transition' band (between $\pi/3$ and ω_c).

(c) *Shortest length filter.* We fix ω_c and α, and wish to find the smallest N that can meet the specifications, *i.e.*, we seek the shortest length FIR filter that can meet the specifications.

(d) *Numerical filter design.* Use CVX to find the shortest length filter that satisfies the filter specifications with

$$\omega_c = 0.4\pi, \quad \alpha = 0.0316.$$

(The attenuation corresponds to -30dB.) For this subproblem, you may sample the constraints in frequency, which means the following. Choose K large (say, 500; an old rule of thumb is that K should be at least $15N$), and set $\omega_k = k\pi/K$, $k = 0, \ldots, K$. Then replace the specifications with

- For k with $0 \leq \omega_k \leq \pi/3$, $0.89 \leq H(\omega_k) \leq 1.12$.
- For k with $\omega_c \leq \omega_k \leq \pi$, $|H(\omega_k)| \leq \alpha$.

Plot $H(\omega)$ versus ω for your design.

12.2 *SINR maximization.* Solve the following instance of problem 4.20: We have $n = 5$ transmitters, grouped into two groups: $\{1, 2\}$ and $\{3, 4, 5\}$. The maximum power for each transmitter is 3, the total power limit for the first group is 4, and the total power limit for the second group is 6. The noise σ is equal to 0.5 and the limit on total received power is 5 for each receiver. Finally, the path gain matrix is given by

$$G = \begin{bmatrix} 1.0 & 0.1 & 0.2 & 0.1 & 0.0 \\ 0.1 & 1.0 & 0.1 & 0.1 & 0.0 \\ 0.2 & 0.1 & 2.0 & 0.2 & 0.2 \\ 0.1 & 0.1 & 0.2 & 1.0 & 0.1 \\ 0.0 & 0.0 & 0.2 & 0.1 & 1.0 \end{bmatrix}.$$

Find the transmitter powers $p_1, \ldots, p_5$ that maximize the minimum SINR ratio over all receivers. Also report the maximum SINR value. Solving the problem to an accuracy of 0.05 (in SINR) is fine.

Hint. When implementing a bisection method in CVX, you will need to check feasibility of a convex problem. You can do this using `strcmpi(cvx_status, 'Solved')`.

12.3 *Power control for sum rate maximization in interference channel.* We consider the optimization problem

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} \log\left(1 + \frac{p_i}{\sum_{j\neq i} A_{ij}p_j + v_i}\right) \\ \text{subject to} & \displaystyle\sum_{i=1}^{n} p_i = 1 \\ & p_i \geq 0, \quad i = 1, \ldots, n \end{array}$$

with variables $p \in \mathbf{R}^n$. The problem data are the matrix $A \in \mathbf{R}^{n\times n}$ and the vector $v \in \mathbf{R}^n$. We assume A and v are componentwise nonnegative ($A_{ij} \geq 0$ and $v_i \geq 0$), and that the diagonal elements of A are equal to one. If the off-diagonal elements of A are zero ($A = I$), the problem has a simple solution, given by the waterfilling method. We are interested in the case where the off-diagonal elements are nonzero.

We can give the following interpretation of the problem, which is not needed below. The variables in the problem are the transmission powers in a communications system. We limit the total power to one (for simplicity; we could have used any other number). The ith term in the objective is

the Shannon capacity of the ith channel; the fraction in the argument of the log is the signal to interference plus noise ratio.

We can express the problem as

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i=1}^{n} \log\left(\frac{\sum_{j=1}^{n} B_{ij}p_j}{\sum_{j=1}^{n} B_{ij}p_j - p_i}\right) \\ \text{subject to} & \displaystyle\sum_{i=1}^{n} p_i = 1 \\ & p_i \geq 0, \;\; i = 1, \ldots, n, \end{array} \tag{45}$$

where $B \in \mathbf{R}^{n\times n}$ is defined as $B = A + v\mathbf{1}^T$, *i.e.*, $B_{ij} = A_{ij} + v_i$, $i, j = 1, \ldots, n$. Suppose B is nonsingular and

$$B^{-1} = I - C$$

with $C_{ij} \geq 0$. Express the problem above as a convex optimization problem. *Hint.* Use $y = Bp$ as variables.

12.4 *Radio-relay station placement and power allocation.* Radio relay stations are to be located at positions $x_1, \ldots, x_n \in \mathbf{R}^2$, and transmit at power $p_1, \ldots, p_n \geq 0$. In this problem we will consider the problem of simultaneously deciding on good locations *and* operating powers for the relay stations.

The received signal power S_{ij} at relay station i from relay station j is proportional to the transmit power and inversely proportional to the distance, *i.e.*,

$$S_{ij} = \frac{\alpha p_j}{\|x_i - x_j\|^2},$$

where $\alpha > 0$ is a known constant.

Relay station j must transmit a signal to relay station i at the rate (or bandwidth) $R_{ij} \geq 0$ bits per second; $R_{ij} = 0$ means that relay station j does not need to transmit any message (directly) to relay station i. The matrix of bit rates R_{ij} is given. Although it doesn't affect the problem, R would likely be sparse, *i.e.*, each relay station needs to communicate with only a few others.

To guarantee accurate reception of the signal from relay station j to i, we must have

$$S_{ij} \geq \beta R_{ij},$$

where $\beta > 0$ is a known constant. (In other words, the minimum allowable received signal power is proportional to the signal bit rate or bandwidth.)

The relay station positions $x_{r+1}, \ldots, x_n$ are fixed, *i.e.*, problem parameters. The problem variables are $x_1, \ldots, x_r$ and $p_1, \ldots, p_n$. The goal is to choose the variables to minimize the total transmit power, *i.e.*, $p_1 + \cdots + p_n$.

Explain how to solve this problem as a convex or quasiconvex optimization problem. If you introduce new variables, or transform the variables, explain. Clearly give the objective and inequality constraint functions, explaining why they are convex. If your problem involves equality constraints, express them using an affine function.

12.5 *Power allocation with coherent combining receivers.* In this problem we consider a variation on the power allocation problem described on pages 4-13 and 4-14 of the notes. In that problem we have m transmitters, each of which transmits (broadcasts) to n receivers, so the total number of receivers is mn. In this problem we have the converse: multiple transmitters send a signal to each receiver.

More specifically we have m receivers labeled $1, \ldots, m$, and mn transmitters labeled (j,k), $j = 1, \ldots, m$, $k = 1, \ldots, n$. The transmitters $(i,1), \ldots, (i,n)$ all transmit the same message to the receiver i, for $i = 1, \ldots, m$.

Transmitter (j,k) operates at power p_{jk}, which must satisfy $0 \leq p_{jk} \leq P_{\max}$, where $P_{\max}$ is a given maximum allowable transmitter power.

The path gain from transmitter (j,k) to receiver i is $A_{ijk} > 0$ (which are given and known). Thus the power received at receiver i from transmitter (j,k) is given by $A_{ijk}p_{jk}$.

For $i \neq j$, the received power $A_{ijk}p_{jk}$ represents an interference signal. The total interference-plus-noise power at receiver i is given by

$$I_i = \sum_{j \neq i,\ k=1,\ldots,n} A_{ijk}p_{jk} + \sigma$$

where $\sigma > 0$ is the known, given (self) noise power of the receivers. Note that the *powers* of the interference and noise signals add to give the total interference-plus-noise power.

The receivers use *coherent detection and combining* of the desired message signals, which means the effective received signal power at receiver i is given by

$$S_i = \left(\sum_{k=1,\ldots,n} (A_{iik}p_{ik})^{1/2} \right)^2 .$$

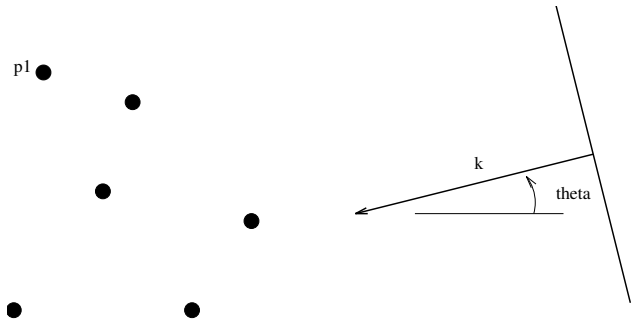
(Thus, the *amplitudes* of the desired signals add to give the effective signal amplitude.)

The total signal to interference-plus-noise ratio (SINR) for receiver i is given by $\gamma_i = S_i/I_i$.

The problem is to choose transmitter powers p_{jk} that maximize the minimum SINR $\min_i \gamma_i$, subject to the power limits.

Explain in detail how to solve this problem using convex or quasiconvex optimization. If you transform the problem by using a different set of variables, explain completely. Identify the objective function, and all constraint functions, indicating if they are convex or quasiconvex, etc.

12.6 *Antenna array weight design.* We consider an array of n omnidirectional antennas in a plane, at positions (x_k, y_k), $k = 1, \ldots, n$.



A unit plane wave with frequency ω is incident from an angle θ. This incident wave induces in the kth antenna element a (complex) signal $\exp(i(x_k \cos\theta + y_k \sin\theta - \omega t))$, where $i = \sqrt{-1}$. (For simplicity we assume that the spatial units are normalized so that the wave number is one, *i.e.*, the wavelength is $\lambda = 2\pi$.) This signal is demodulated, *i.e.*, multiplied by $e^{i\omega t}$, to obtain the baseband signal (complex number) $\exp(i(x_k \cos\theta + y_k \sin\theta))$. The baseband signals of the n antennas are combined linearly to form the output of the antenna array

$$\begin{aligned} G(\theta) &= \sum_{k=1}^{n} w_k \, e^{i(x_k \cos\theta + y_k \sin\theta)} \\ &= \sum_{k=1}^{n} \left(w_{\mathrm{re},k} \cos\gamma_k(\theta) - w_{\mathrm{im},k} \sin\gamma_k(\theta)\right) + i\left(w_{\mathrm{re},k} \sin\gamma_k(\theta) + w_{\mathrm{im},k} \cos\gamma_k(\theta)\right), \end{aligned}$$

if we define $\gamma_k(\theta) = x_k \cos\theta + y_k \sin\theta$. The complex weights in the linear combination,

$$w_k = w_{\mathrm{re},k} + i w_{\mathrm{im},k}, \quad k = 1, \ldots, n,$$

are called the *antenna array coefficients* or *shading coefficients*, and will be the design variables in the problem. For a given set of weights, the combined output $G(\theta)$ is a function of the angle of arrival θ of the plane wave. The design problem is to select weights w_i that achieve a desired directional pattern $G(\theta)$.

We now describe a basic weight design problem. We require unit gain in a target direction θ^{tar}, *i.e.*, $G(\theta^{\mathrm{tar}}) = 1$. We want $|G(\theta)|$ small for $|\theta - \theta^{\mathrm{tar}}| \geq \Delta$, where 2Δ is our beamwidth. To do this, we can minimize

$$\max_{|\theta - \theta^{\mathrm{tar}}| \geq \Delta} |G(\theta)|,$$

where the maximum is over all $\theta \in [-\pi, \pi]$ with $|\theta - \theta^{\mathrm{tar}}| \geq \Delta$. This number is called the sidelobe level for the array; our goal is to minimize the sidelobe level. If we achieve a small sidelobe level, then the array is relatively insensitive to signals arriving from directions more than Δ away from the target direction. This results in the optimization problem

$$\begin{array}{ll} \text{minimize} & \max_{|\theta - \theta^{\mathrm{tar}}| \geq \Delta} |G(\theta)| \\ \text{subject to} & G(\theta^{\mathrm{tar}}) = 1, \end{array}$$

with $w \in \mathbf{C}^n$ as variables.

The objective function can be approximated by discretizing the angle of arrival with (say) N values (say, uniformly spaced) $\theta_1, \ldots, \theta_N$ over the interval $[-\pi, \pi]$, and replacing the objective with

$$\max\{|G(\theta_k)| \mid |\theta_k - \theta^{\mathrm{tar}}| \geq \Delta\}$$

(a) Formulate the antenna array weight design problem as an SOCP.

(b) Solve an instance using CVX, with $n = 40$, $\theta^{\mathrm{tar}} = 15^\circ$, $\Delta = 15^\circ$, $N = 400$, and antenna positions generated using

```
rand('state',0);
n = 40;
x = 30 * rand(n,1);
y = 30 * rand(n,1);
```

Compute the optimal weights and make a plot of $|G(\theta)|$ (on a logarithmic scale) versus θ. *Hint.* CVX can directly handle complex variables, and recognizes the modulus `abs(x)` of a complex number as a convex function of its real and imaginary parts, so you do not need to explicitly form the SOCP from part (a). Even more compactly, you can use `norm(x,Inf)` with complex argument.

12.7 *Power allocation problem with analytic solution.* Consider a system of n transmitters and n receivers. The ith transmitter transmits with power x_i, $i = 1, \ldots, n$. The vector x will be the variable in this problem. The path gain from each transmitter j to each receiver i will be denoted A_{ij} and is assumed to be known (obviously, $A_{ij} \geq 0$, so the matrix A is elementwise nonnegative, and $A_{ii} > 0$). The signal received by each receiver i consists of three parts: the desired signal, arriving from transmitter i with power $A_{ii}x_i$, the interfering signal, arriving from the other receivers with power $\sum_{j \neq i} A_{ij}x_j$, and noise β_i (which are positive and known). We are interested in allocating the powers x_i in such a way that the signal to noise plus interference ratio at each of the receivers exceeds a level α. (Thus α is the minimum acceptable SNIR for the receivers; a typical value might be around $\alpha = 3$, *i.e.*, around 10dB). In other words, we want to find $x \succeq 0$ such that for $i = 1, \ldots, n$

$$A_{ii}x_i \geq \alpha \left(\sum_{j \neq i} A_{ij}x_j + \beta_i \right).$$

Equivalently, the vector x has to satisfy

$$x \succeq 0, \quad Bx \succeq \alpha\beta \tag{46}$$

where $B \in \mathbf{R}^{n \times n}$ is defined as

$$B_{ii} = A_{ii}, \quad B_{ij} = -\alpha A_{ij}, \; j \neq i.$$

(a) Show that (46) is feasible if and only if B is invertible and $z = B^{-1}\mathbf{1} \succeq 0$ ($\mathbf{1}$ is the vector with all components 1). Show how to construct a feasible power allocation x from z.

(b) Show how to find the largest possible SNIR, *i.e.*, how to maximize α subject to the existence of a feasible power allocation.

To solve this problem you may need the following:

Hint. Let $T \in \mathbf{R}^{n \times n}$ be a matrix with nonnegative elements, and $s \in \mathbf{R}$. Then the following are equivalent:

(a) $s > \rho(T)$, where $\rho(T) = \max_i |\lambda_i(T)|$ is the spectral radius of T.

(b) $sI - T$ is nonsingular and the matrix $(sI - T)^{-1}$ has nonnegative elements.

(c) there exists an $x \succeq 0$ with $(sI - T)x \succ 0$.

(For such s, the matrix $sI - T$ is called a *nonsingular M-matrix.*)

Remark. This problem gives an analytic solution to a very special form of transmitter power allocation problem. Specifically, there are exactly as many transmitters as receivers, and no power limits on the transmitters. One consequence is that the receiver noises β_i play no role at all in the solution — just crank up all the transmitters to overpower the noises!

12.8 *Optimizing rates and time slot fractions.* We consider a wireless system that uses time-domain multiple access (TDMA) to support n communication flows. The flows have (nonnegative) rates $r_1, \ldots, r_n$, given in bits/sec. To support a rate r_i on flow i requires transmitter power

$$p = a_i(e^{br} - 1),$$

where b is a (known) positive constant, and a_i are (known) positive constants related to the noise power and gain of receiver i.

TDMA works like this. Time is divided up into periods of some fixed duration T (seconds). Each of these T-long periods is divided into n time-slots, with durations $t_1, \ldots, t_n$, that must satisfy $t_1 + \cdots + t_n = T$, $t_i \geq 0$. In time-slot i, communications flow i is transmitted at an instantaneous rate $r = Tr_i/t_i$, so that over each T-long period, Tr_i bits from flow i are transmitted. The power required during time-slot i is $a_i(e^{bTr_i/t_i} - 1)$, so the average transmitter power over each T-long period is

$$P = (1/T)\sum_{i=1}^{n} a_i t_i (e^{bTr_i/t_i} - 1).$$

When t_i is zero, we take $P = \infty$ if $r_i > 0$, and $P = 0$ if $r_i = 0$. (The latter corresponds to the case when there is zero flow, and also, zero time allocated to the flow.)

The problem is to find rates $r \in \mathbf{R}^n$ and time-slot durations $t \in \mathbf{R}^n$ that maximize the log utility function

$$U(r) = \sum_{i=1}^{n} \log r_i,$$

subject to $P \leq P^{\max}$. (This utility function is often used to ensure 'fairness'; each communication flow gets at least some positive rate.) The problem data are a_i, b, T and $P^{\max}$; the variables are t_i and r_i.

(a) Formulate this problem as a convex optimization problem. Feel free to introduce new variables, if needed, or to change variables. Be sure to justify convexity of the objective or constraint functions in your formulation.

(b) Give the optimality conditions for your formulation. Of course we prefer simpler optimality conditions to complex ones. *Note:* We do not expect you to *solve* the optimality conditions; you can give them as a set of equations (and possibly inequalities).

Hint. With a log utility function, we cannot have $r_i = 0$, and therefore we cannot have $t_i = 0$; therefore the constraints $r_i \geq 0$ and $t_i \geq 0$ cannot be active or tight. This will allow you to simplify the optimality conditions.

12.9 *Optimal jamming power allocation.* A set of n jammers transmit with (nonnegative) powers $p_1, \ldots, p_n$, which are to be chosen subject to the constraints

$$p \succeq 0, \qquad Fp \preceq g.$$

The jammers produce interference power at m receivers, given by

$$d_i = \sum_{j=1}^{n} G_{ij} p_j, \quad i = 1, \ldots, m,$$

where G_{ij} is the (nonnegative) channel gain from jammer j to receiver i.

Receiver i has capacity (in bits/s) given by

$$C_i = \alpha \log(1 + \beta_i/(\sigma_i^2 + d_i)), \quad i = 1, \ldots, m,$$

where α, β_i, and σ_i are positive constants. (Here β_i is proportional to the signal power at receiver i and σ_i^2 is the receiver i self-noise, but you won't need to know this to solve the problem.)

Explain how to choose p to *minimize* the sum channel capacity, $C = C_1 + \cdots + C_m$, using convex optimization. (This corresponds to the most effective jamming, given the power constraints.) The problem data are F, g, G, α, β_i, σ_i.

If you change variables, or transform your problem in any way that is not obvious (for example, you form a relaxation), you must explain fully how your method works, and why it gives the solution. If your method relies on any convex functions that we have not encountered before, you must show that the functions are convex.

Disclaimer. The teaching staff does not endorse jamming, optimal or otherwise.

12.10 *2D filter design.* A symmetric convolution kernel with support $\{-(N-1), \ldots, N-1\}^2$ is characterized by N^2 coefficients

$$h_{kl}, \quad k, l = 1, \ldots, N.$$

These coefficients will be our variables. The corresponding 2D frequency response (Fourier transform) $H : \mathbf{R}^2 \to \mathbf{R}$ is given by

$$H(\omega_1, \omega_2) = \sum_{k,l=1,\ldots,N} h_{kl} \cos((k-1)\omega_1) \cos((l-1)\omega_2),$$

where ω_1 and ω_2 are the frequency variables. Evidently we only need to specify H over the region $[0, \pi]^2$, although it is often plotted over the region $[-\pi, \pi]^2$. (It won't matter in this problem, but we should mention that the coefficients h_{kl} above are not exactly the same as the impulse response coefficients of the filter.)

We will design a 2D filter (*i.e.*, find the coefficients h_{kl}) to satisfy $H(0, 0) = 1$ and to minimize the maximum response R in the rejection region $\Omega_{\text{rej}} \subset [0, \pi]^2$,

$$R = \sup_{(\omega_1, \omega_2) \in \Omega_{\text{rej}}} |H(\omega_1, \omega_2)|.$$

(a) Explain why this 2D filter design problem is convex.

(b) Find the optimal filter for the specific case with $N = 5$ and

$$\Omega_{\text{rej}} = \{(\omega_1, \omega_2) \in [0, \pi]^2 \mid \omega_1^2 + \omega_2^2 \geq W^2\},$$

with $W = \pi/4$.

You can approximate R by sampling on a grid of frequency values. Define

$$\omega^{(p)} = \pi(p-1)/M, \quad p = 1, \ldots, M.$$

(You can use $M = 25$.) We then replace the exact expression for R above with

$$\hat{R} = \max\{|H(\omega^{(p)}, \omega^{(q)})| \mid p, q = 1, \ldots, M, \ (\omega^{(p)}, \omega^{(q)}) \in \Omega_{\text{rej}}\}.$$

Give the optimal value of $\hat{R}$. Plot the optimal frequency response using `plot_2D_filt(h)`, available on the course web site, where `h` is the matrix containing the coefficients h_{kl}.

12.11 *Maximizing log utility in a wireless system with interference.* Consider a wireless network consisting of n data links, labeled $1, \ldots, n$. Link i transmits with power $P_i > 0$, and supports a data rate $R_i = \log(1 + \gamma_i)$, where γ_i is the signal-to-interference-plus-noise ratio (SINR). These SINR ratios depend on the transmit powers, as described below.

The system is characterized by the link gain matrix $G \in \mathbf{R}^{n \times n}_{++}$, where G_{ij} is the gain from the transmitter on link j to the receiver for link i. The received signal power for link i is $G_{ii}P_i$; the noise plus interference power for link i is given by

$$\sigma_i^2 + \sum_{j \neq i} G_{ij} P_j,$$

where $\sigma_i^2 > 0$ is the receiver noise power for link i. The SINR is the ratio

$$\gamma_i = \frac{G_{ii} P_i}{\sigma_i^2 + \sum_{j \neq i} G_{ij} P_j}.$$

The problem is to choose the transmit powers $P_1, \ldots, P_n$, subject to $0 < P_i \leq P_i^{\max}$, in order to maximize the log utility function

$$U(P) = \sum_{i=1}^{n} \log R_i.$$

(This utility function can be argued to yield a fair distribution of rates.) The data are G, σ_i^2, and $P_i^{\max}$.

Formulate this problem as a convex or quasiconvex optimization problem. If you make any transformations or use any steps that are not obvious, explain.

Hints.

- The function $\log \log(1 + e^x)$ is concave. (If you use this fact, you must show it.)
- You might find the new variables defined by $z_i = \log P_i$ useful.

12.12 *Spectral factorization via semidefinite programming.* A Toeplitz matrix is a matrix that has constant values on its diagonals. We use the notation

$$T_m(x_1, \ldots, x_m) = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_{m-1} & x_m \\ x_2 & x_1 & x_2 & \cdots & x_{m-2} & x_{m-1} \\ x_3 & x_2 & x_1 & \cdots & x_{m-3} & x_{m-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m-1} & x_{m-2} & x_{m-2} & \cdots & x_1 & x_2 \\ x_m & x_{m-1} & x_{m-2} & \cdots & x_2 & x_1 \end{bmatrix}$$

to denote the symmetric Toeplitz matrix in $\mathbf{S}^{m \times m}$ constructed from x_1, ..., x_m. Consider the semidefinite program

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & T_n(x_1, \ldots, x_n) \succeq e_1 e_1^T, \end{array}$$

with variable $x = (x_1, \ldots, x_n)$, where $e_1 = (1, 0, \ldots, 0)$.

(a) Derive the dual of the SDP above. Denote the dual variable as Z. (Hence $Z \in \mathbf{S}^n$ and the dual constraints include an inequality $Z \succeq 0$.)

(b) Show that $T_n(x_1, \ldots, x_n) \succ 0$ for every feasible x in the SDP above. You can do this by induction on n.

- For $n = 1$, the constraint is $x_1 \geq 1$ which obviously implies $x_1 > 0$.
- In the induction step, assume $n \geq 2$ and that $T_{n-1}(x_1, \ldots, x_{n-1}) \succ 0$. Use a Schur complement argument and the Toeplitz structure of T_n to show that $T_n(x_1, \ldots, x_n) \succeq e_1 e_1^T$ implies $T_n(x_1, \ldots, x_n) \succ 0$.

(c) Suppose the optimal value of the SDP above is finite and attained, and that Z is dual optimal. Use the result of part (b) to show that the rank of Z is at most one, *i.e.*, Z can be expressed as $Z = yy^T$ for some n-vector y. Show that y satisfies

$$\begin{aligned} y_1^2 + y_2^2 + \cdots + y_n^2 &= c_1 \\ y_1 y_2 + y_2 y_3 + \cdots + y_{n-1} y_n &= c_2/2 \\ &\vdots \\ y_1 y_{n-1} + y_2 y_n &= c_{n-1}/2 \\ y_1 y_n &= c_n/2. \end{aligned}$$

This can be expressed as an identity $|Y(\omega)|^2 = R(\omega)$ between two functions

$$\begin{aligned} Y(\omega) &= y_1 + y_2 e^{-i\omega} + y_3 e^{-3i\omega} + \cdots + y_n e^{-i(n-1)\omega} \\ R(\omega) &= c_1 + c_2 \cos\omega + c_3 \cos(2\omega) + \cdots + c_n \cos((n-1)\omega) \end{aligned}$$

(with $i = \sqrt{-1}$). The function $Y(\omega)$ is called a *spectral factor* of the trigonometric polynomial $R(\omega)$.

12.13 *Bandlimited signal recovery from zero-crossings.* Let $y \in \mathbf{R}^n$ denote a *bandlimited* signal, which means that it can be expressed as a linear combination of sinusoids with frequencies in a band:

$$y_t = \sum_{j=1}^{B} a_j \cos\left(\frac{2\pi}{n}(f_{\min} + j - 1)t\right) + b_j \sin\left(\frac{2\pi}{n}(f_{\min} + j - 1)t\right), \quad t = 1, \ldots, n,$$

where $f_{\min}$ is lowest frequency in the band, B is the bandwidth, and $a, b \in \mathbf{R}^B$ are the cosine and sine coefficients, respectively. We are given $f_{\min}$ and B, but not the coefficients a, b or the signal y.

We do not know y, but we are given its sign $s = \mathbf{sign}(y)$, where $s_t = 1$ if $y_t \geq 0$ and $s_t = -1$ if $y_t < 0$. (Up to a change of overall sign, this is the same as knowing the 'zero-crossings' of the signal, *i.e.*, when it changes sign. Hence the name of this problem.)

We seek an estimate $\hat{y}$ of y that is consistent with the bandlimited assumption and the given signs. Of course we cannot distinguish y and αy, where $\alpha > 0$, since both of these signals have the same sign pattern. Thus, we can only estimate y up to a positive scale factor. To normalize $\hat{y}$, we will require that $\|\hat{y}\|_1 = n$, *i.e.*, the average value of $|y_i|$ is one. Among all $\hat{y}$ that are consistent with the bandlimited assumption, the given signs, and the normalization, we choose the one that minimizes $\|\hat{y}\|_2$.

(a) Show how to find $\hat{y}$ using convex or quasiconvex optimization.

(b) Apply your method to the problem instance with data in `zero_crossings_data.*`. The data files also include the true signal y (which of course you cannot use to find $\hat{y}$). Plot $\hat{y}$ and y, and report the relative recovery error, $\|y - \hat{y}\|_2 / \|y\|_2$. Give one short sentence commenting on the quality of the recovery.

13 Finance

13.1 *Transaction cost.* Consider a market for some asset or commodity, which we assume is infinitely divisible, *i.e.*, can be bought or sold in quantities of shares that are real numbers (as opposed to integers). The *order book* at some time consists of a set of offers to sell or buy the asset, at a given price, up to a given quantity of shares. The N offers to sell the asset have positive prices per share $p_1^{\text{sell}}, \ldots, p_N^{\text{sell}}$, sorted in increasing order, in positive share quantities $q_1^{\text{sell}}, \ldots, q_N^{\text{sell}}$. The M offers to buy the asset have positive prices $p_1^{\text{buy}}, \ldots, p_N^{\text{buy}}$, sorted in decreasing order, and positive quantities $q_1^{\text{buy}}, \ldots, q_M^{\text{buy}}$. The price p_1^{sell} is called the (current) *ask price* for the asset; p_1^{buy} is the *bid price* for the asset. The ask price is larger than the bid price; the difference is called the *spread*. The average of the ask and bid prices is called the *mid-price*, denoted p^{mid}.

Now suppose that you want to purchase $q > 0$ shares of the asset, where $q \leq q_1^{\text{sell}} + \cdots + q_N^{\text{sell}}$, *i.e.*, your purchase quantity does not exceed the total amount of the asset currently offered for sale. Your purchase proceeds as follows. Suppose that

$$q_1^{\text{sell}} + \cdots + q_k^{\text{sell}} < q \leq q_1^{\text{sell}} + \cdots + q_{k+1}^{\text{sell}}.$$

Then you pay an amount

$$A = p_1^{\text{sell}} q_1^{\text{sell}} + \cdots + p_k^{\text{sell}} q_k^{\text{sell}} + p_{k+1}^{\text{sell}} (q - q_1^{\text{sell}} - \cdots - q_k^{\text{sell}}).$$

Roughly speaking, you work your way through the offers in the order book, from the least (ask) price, and working your way up the order book until you fill the order. We define the *transaction cost* as

$$T(q) = A - p^{\text{mid}} q.$$

This is the difference between what you pay, and what you would have paid had you been able to purchase the shares at the mid-price. It is always positive.

We handle the case of selling the asset in a similar way. Here we take $q < 0$ to mean that we sell $-q$ shares of the asset. Here you sell shares at the bid price, up to the quantity q^{buy} (or $-q$, whichever is smaller); if needed, you sell shares at the price p_2^{buy}, and so on, until all $-q$ shares are sold. Here we assume that $-q \leq q_1^{\text{buy}} + \cdots + q_M^{\text{buy}}$, *i.e.*, you are not selling more shares than the total quantity of offers to buy. Let A denote the amount you receive from the sale. Here we define the transaction cost as

$$T(q) = -p^{\text{mid}} q - A,$$

the difference between the amount you would have received had you sold the shares at the mid-price, and the amount you received. It is always positive. We set $T(0) = 0$.

(a) Show that T is a convex piecewise linear function.

(b) Show that $T(q) \geq (s/2)|q|$, where s is the spread. When would we have $T(q) = (s/2)|q|$ for all q (in the range between the total shares offered to purchase or sell)?

(c) Give an interpretation of the conjugate function $T^*(y) = \sup_q (yq - T(q))$. *Hint.* Suppose you can purchase or sell the asset in another market, at the price p^{other}.

13.2 *Risk-return trade-off in portfolio optimization.* We consider the portfolio risk-return trade-off problem of page 185, with the following data:

$$\bar{p} = \begin{bmatrix} 0.12 \\ 0.10 \\ 0.07 \\ 0.03 \end{bmatrix}, \qquad \Sigma = \begin{bmatrix} 0.0064 & 0.0008 & -0.0011 & 0 \\ 0.0008 & 0.0025 & 0 & 0 \\ -0.0011 & 0 & 0.0004 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

(a) Solve the quadratic program

$$\begin{array}{ll} \text{minimize} & -\bar{p}^T x + \mu x^T \Sigma x \\ \text{subject to} & \mathbf{1}^T x = 1, \quad x \succeq 0 \end{array}$$

for a large number of positive values of μ (for example, 100 values logarithmically spaced between 1 and 10^7). Plot the optimal values of the expected return $\bar{p}^T x$ versus the standard deviation $(x^T \Sigma x)^{1/2}$. Also make an area plot of the optimal portfolios x versus the standard deviation (as in figure 4.12).

(b) Assume the price change vector p is a Gaussian random variable, with mean $\bar{p}$ and covariance Σ. Formulate the problem

$$\begin{array}{ll} \text{maximize} & \bar{p}^T x \\ \text{subject to} & \mathbf{prob}(p^T x \leq 0) \leq \eta \\ & \mathbf{1}^T x = 1, \quad x \succeq 0, \end{array}$$

as a convex optimization problem, where $\eta < 1/2$ is a parameter. In this problem we maximize the expected return subject to a constraint on the probability of a negative return. Solve the problem for a large number of values of η between 10^{-4} and 10^{-1}, and plot the optimal values of $\bar{p}^T x$ versus η. Also make an area plot of the optimal portfolios x versus η.

Hint: The Matlab functions `erfc` and `erfcinv` can be used to evaluate

$$\Phi(x) = (1/\sqrt{2\pi}) \int_{-\infty}^{x} e^{-t^2/2}\, dt$$

and its inverse:

$$\Phi(u) = \frac{1}{2}\mathbf{erfc}(-u/\sqrt{2}).$$

Since you will have to solve this problem for a large number of values of η, you may find the command `cvx_quiet(true)` helpful.

(c) *Monte Carlo simulation.* Let x be the optimal portfolio found in part (b), with $\eta = 0.05$. This portfolio maximizes the expected return, subject to the probability of a loss being no more than 5%. Generate 10000 samples of p, and plot a histogram of the returns. Find the empirical mean of the return samples, and calculate the percentage of samples for which a loss occurs.

Hint: You can generate samples of the price change vector using

```
p=pbar+sqrtm(Sigma)*randn(4,1);
```

13.3 *Simple portfolio optimization.* We consider a portfolio optimization problem as described on pages 155 and 185–186 of *Convex Optimization*, with data that can be found in the file `simple_portfolio_data.*`.

(a) Find minimum-risk portfolios with the same expected return as the uniform portfolio ($x = (1/n)\mathbf{1}$), with risk measured by portfolio return variance, and the following portfolio constraints (in addition to $\mathbf{1}^T x = 1$):

- No (additional) constraints.
- Long-only: $x \succeq 0$.
- Limit on total short position: $\mathbf{1}^T(x_-) \leq 0.5$, where $(x_-)_i = \max\{-x_i, 0\}$.

Compare the optimal risk in these portfolios with each other and the uniform portfolio.

(b) Plot the optimal risk-return trade-off curves for the long-only portfolio, and for total short-position limited to 0.5, in the same figure. Follow the style of figure 4.12 (top), with horizontal axis showing standard deviation of portfolio return, and vertical axis showing mean return.

13.4 *Bounding portfolio risk with incomplete covariance information.* Consider the following instance of the problem described in §4.6, on p171–173 of *Convex Optimization*. We suppose that Σ_{ii}, which are the squares of the price volatilities of the assets, are known. For the off-diagonal entries of Σ, all we know is the sign (or, in some cases, nothing at all). For example, we might be given that $\Sigma_{12} \geq 0$, $\Sigma_{23} \leq 0$, etc. This means that we do not know the correlation between p_1 and p_2, but we do know that they are nonnegatively correlated (*i.e.*, the prices of assets 1 and 2 tend to rise or fall together).

Compute σ^2_{wc}, the worst-case variance of the portfolio return, for the specific case

$$x = \begin{bmatrix} 0.1 \\ 0.2 \\ -0.05 \\ 0.1 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} 0.2 & + & + & \pm \\ + & 0.1 & - & - \\ + & - & 0.3 & + \\ \pm & - & + & 0.1 \end{bmatrix},$$

where a "+" entry means that the element is nonnegative, a "−" means the entry is nonpositive, and "±" means we don't know anything about the entry. (The negative value in x represents a *short position*: you sold stocks that you didn't have, but must produce at the end of the investment period.) In addition to σ^2_{wc}, give the covariance matrix Σ_{wc} associated with the maximum risk. Compare the worst-case risk with the risk obtained when Σ is diagonal.

13.5 *Log-optimal investment strategy.* In this problem you will solve a specific instance of the log-optimal investment problem described in exercise 4.60, with $n = 5$ assets and $m = 10$ possible outcomes in each period. The problem data are defined in `log_opt_invest.*`, with the rows of the matrix `P` giving the asset return vectors p_j^T. The outcomes are equiprobable, *i.e.*, we have $\pi_j = 1/m$. Each column of the matrix `P` gives the return of the associated asset in the different posible outcomes. You can examine the columns to get an idea of the types of assets. For example, the last asset gives a fixed and certain return of 1%; the first asset is a very risky one, with occasional large return, and (more often) substantial loss.

Find the log-optimal investment strategy $x^\star$, and its associated long term growth rate $R^\star_{\mathrm{lt}}$. Compare this to the long term growth rate obtained with a uniform allocation strategy, *i.e.*, $x = (1/n)\mathbf{1}$, and also with a pure investment in each asset.

For the optimal investment strategy, and also the uniform investment strategy, plot 10 sample trajectories of the accumulated wealth, *i.e.*, $W(T) = W(0)\prod_{t=1}^{T}\lambda(t)$, for $T = 0, \ldots, 200$, with initial wealth $W(0) = 1$.

To save you the trouble of figuring out how to simulate the wealth trajectories or plot them nicely, we've included the simulation and plotting code in `log_opt_invest.*`; you just have to add the code needed to find $x^\star$.

Hint (MATLAB users only): The current version of CVX handles the logarithm via an iterative method, which can be slow and unreliable. You're better off using `geo_mean()`, which is directly handled by CVX, to solve the problem.

13.6 *Optimality conditions and dual for log-optimal investment problem.*

(a) Show that the optimality conditions for the log-optimal investment problem described in exercise 4.60 can be expressed as: $\mathbf{1}^T x = 1$, $x \succeq 0$, and for each i,

$$x_i > 0 \;\Rightarrow\; \sum_{j=1}^m \pi_j \frac{p_{ij}}{p_j^T x} = 1, \quad x_i = 0 \;\Rightarrow\; \sum_{j=1}^m \pi_j \frac{p_{ij}}{p_j^T x} \leq 1.$$

We can interpret this as follows. $p_{ij}/p_j^T x$ is a random variable, which gives the ratio of the investment gain with asset i only, to the investment gain with our mixed portfolio x. The optimality condition is that, for each asset we invest in, the expected value of this ratio is one, and for each asset we do not invest in, the expected value cannot exceed one. Very roughly speaking, this means our portfolio does as well as any of the assets that we choose to invest in, and cannot do worse than any assets that we do not invest in.

Hint. You can start from the simple criterion given in §4.2.3 or the KKT conditions.

(b) In this part we will derive the dual of the log-optimal investment problem. We start by writing the problem as

$$\begin{array}{ll} \text{minimize} & -\sum_{j=1}^m \pi_j \log y_j \\ \text{subject to} & y = P^T x, \quad x \succeq 0, \quad \mathbf{1}^T x = 1. \end{array}$$

Here, P has columns $p_1, \ldots, p_m$, and we have the introduced new variables $y_1, \ldots, y_m$, with the implicit constraint $y \succ 0$. We will associate dual variables ν, λ and ν_0 with the constraints $y = P^T x$, $x \succeq 0$, and $\mathbf{1}^T x = 1$, respectively. Defining $\tilde{\nu}_j = \nu_j/\nu_0$ for $j = 1, \ldots, m$, show that the dual problem can be written as

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^m \pi_j \log(\tilde{\nu}_j/\pi_j) \\ \text{subject to} & P\tilde{\nu} \preceq \mathbf{1}, \end{array}$$

with variable $\tilde{\nu}$. The objective here is the (negative) Kullback-Leibler divergence between the given distribution π and the dual variable $\tilde{\nu}$.

13.7 *Arbitrage and theorems of alternatives.* Consider an event (for example, a sports game, political elections, the evolution of the stock market over a certain period) with m possible outcomes. Suppose that n wagers on the outcome are possible. If we bet an amount x_j on wager j, and the outcome of the event is i ($i = 1, \ldots, m$), then our return will be equal to $r_{ij}x_j$. The return $r_{ij}x_j$ is the net gain: we pay x_j initially, and receive $(1 + r_{ij})x_j$ if the outcome of the event is i. We allow the bets x_j to be positive, negative, or zero. The interpretation of a negative bet is as follows. If $x_j < 0$, then initially we *receive* an amount of money $|x_j|$, with an obligation to *pay* $(1 + r_{ij})|x_j|$ if outcome i occurs. In that case, we lose $r_{ij}|x_j|$, *i.e.*, our net is gain $r_{ij}x_j$ (a negative number).

We call the matrix $R \in \mathbf{R}^{m\times n}$ with elements r_{ij} the *return matrix*. A *betting strategy* is a vector $x \in \mathbf{R}^n$, with as components x_j the amounts we bet on each wager. If we use a betting strategy

Country	Odds
Holland	3.5
Italy	5.0
Spain	5.5
France	6.5
Germany	7.0
England	10.0
Belgium	14.0
Sweden	16.0

Country	Odds
Czech Republic	17.0
Romania	18.0
Yugoslavia	20.0
Portugal	20.0
Norway	20.0
Denmark	33.0
Turkey	50.0
Slovenia	80.0

Table 1: Odds for the 2000 European soccer championships.

x, our total return in the event of outcome i is equal to $\sum_{j=1}^{n} r_{ij}x_j$, *i.e.*, the ith component of the vector Rx.

(a) *The arbitrage theorem.* Suppose you are given a return matrix R. Prove the following theorem: there is a betting strategy $x \in \mathbf{R}^n$ for which

$$Rx \succ 0$$

if and only if there exists no vector $p \in \mathbf{R}^m$ that satisfies

$$R^T p = 0, \qquad p \succeq 0, \qquad p \neq 0.$$

We can interpret this theorem as follows. If $Rx \succ 0$, then the betting strategy x guarantees a positive return for all possible outcomes, *i.e.*, it is a sure-win betting scheme. In economics, we say there is an *arbitrage opportunity*.

If we normalize the vector p in the second condition, so that $\mathbf{1}^T p = 1$, we can interpret it as a probability vector on the outcomes. The condition $R^T p = 0$ means that

$$\mathbf{E}\, Rx = p^T Rx = 0$$

for all x, *i.e.*, the expected return is zero for all betting strategies. In economics, p is called a risk neutral probability.

We can therefore rephrase the arbitrage theorem as follows: There is no sure-win betting strategy (or arbitrage opportunity) if and only if there is a probability vector on the outcomes that makes all bets fair (*i.e.*, the expected gain is zero).

(b) *Betting.* In a simple application, we have exactly as many wagers as there are outcomes ($n = m$). Wager i is to bet that the outcome will be i. The returns are usually expressed as *odds*. For example, suppose that a bookmaker accepts bets on the result of the 2000 European soccer championships. If the odds against Belgium winning are 14 to one, and we bet \$100 on Belgium, then we win \$1400 if they win the tournament, and we lose \$100 otherwise.

In general, if we have m possible outcomes, and the odds against outcome i are λ_i to one, then the return matrix $R \in \mathbf{R}^{m\times m}$ is given by

$$\begin{array}{ll} r_{ij} = \lambda_i & \text{if } j = i \\ r_{ij} = -1 & \text{otherwise.} \end{array}$$

Show that there is no sure-win betting scheme (or arbitrage opportunity) if

$$\sum_{i=1}^{m} \frac{1}{1+\lambda_i} = 1.$$

In fact, you can verify that if this equality is not satisfied, then the betting strategy

$$x_i = \frac{1/(1+\lambda_i)}{1 - \sum_{i=1}^{m} 1/(1+\lambda_i)}$$

always results in a profit.

The common situation in real life is

$$\sum_{i=1}^{m} \frac{1}{1+\lambda_i} > 1,$$

because the bookmakers take a cut on all bets.

13.8 *Log-optimal investment.* We consider an instance of the log-optimal investment problem described in exercise 4.60 of *Convex Optimization*. In this exercise, however, we allow x, the allocation vector, to have negative components. Investing a negative amount $x_i W(t)$ in an asset is called *shorting* the asset. It means you borrow the asset, sell it for $|x_i W(t)|$, and have an obligation to purchase it back later and return it to the lender.

(a) Let R be the $n \times m$-matrix with columns r_j:

$$R = \begin{bmatrix} r_1 & r_2 & \cdots & r_m \end{bmatrix}.$$

We assume that the elements r_{ij} of R are all positive, which implies that the log-optimal investment problem is feasible. Show the following property: if there exists a $v \in \mathbf{R}^n$ with

$$\mathbf{1}^T v = 0, \qquad R^T v \succeq 0, \qquad R^T v \neq 0 \tag{47}$$

then the log-optimal investment problem is unbounded (assuming that the probabilities p_j are all positive).

(b) Derive a Lagrange dual of the log-optimal investment problem (or an equivalent problem of your choice). Use the Lagrange dual to show that the condition in part a is also necessary for unboundedness. In other words, the log-optimal investment problem is bounded if and only if there does not exist a v satisfying (47).

(c) Consider the following small example. We have four scenarios and three investment options. The return vectors for the four scenarios are

$$r_1 = \begin{bmatrix} 2 \\ 1.3 \\ 1 \end{bmatrix}, \qquad r_2 = \begin{bmatrix} 2 \\ 0.5 \\ 1 \end{bmatrix}, \qquad r_3 = \begin{bmatrix} 0.5 \\ 1.3 \\ 1 \end{bmatrix}, \qquad r_4 = \begin{bmatrix} 0.5 \\ 0.5 \\ 1 \end{bmatrix}.$$

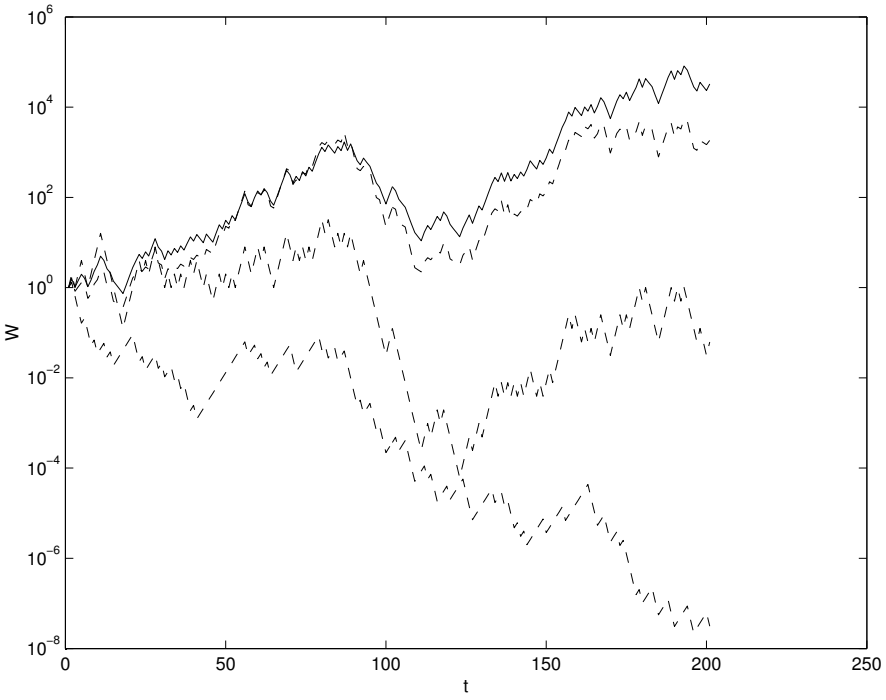
The probabilities of the three scenarios are

$$p_1 = 1/3, \qquad p_2 = 1/6, \qquad p_3 = 1/3, \qquad p_4 = 1/6.$$

The interpretation is as follows. We can invest in two stocks. The first stock doubles in value in each period with a probability 1/2, or decreases by 50% with a probability 1/2. The second stock either increases by 30% with a probability 2/3, or decreases by 50% with a probability 1/3. The fluctuations in the two stocks are independent, so we have four scenarios: both stocks go up (probability 2/6), stock 1 goes up and stock 2 goes down (probability 1/6), stock 1 goes down and stock 2 goes up (probability 1/3), both stocks go down (probability 1/6). The fractions of our capital we invest in stocks 1 and 2 are denoted by x_1 and x_2, respectively. The rest of our capital, $x_3 = 1 - x_1 - x_2$ is not invested.

What is the expected growth rate of the log-optimal strategy x? Compare with the strategies $(x_1, x_2, x_3) = (1, 0, 0)$, $(x_1, x_2, x_3) = (0, 1, 0)$ and $(x_1, x_2, x_3) = (1/2, 1/2, 0)$. (Obviously the expected growth rate for $(x_1, x_2, x_3) = (0, 0, 1)$ is zero.)

Remark. The figure below shows a simulation that compares three investment strategies over 200 periods. The solid line shows the log-optimal investment strategy. The dashed lines show the growth for strategies $x = (1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$.



13.9 *Maximizing house profit in a gamble and imputed probabilities.* A set of n participants bet on which one of m outcomes, labeled $1, \ldots, m$, will occur. Participant i offers to purchase up to $q_i > 0$ gambling contracts, at price $p_i > 0$, that the true outcome will be in the set $S_i \subset \{1, \ldots, m\}$. The house then sells her x_i contracts, with $0 \leq x_i \leq q_i$. If the true outcome j is in S_i, then participant i receives \$1 per contract, *i.e.*, x_i. Otherwise, she loses, and receives nothing. The house collects a total of $x_1 p_1 + \cdots + x_n p_n$, and pays out an amount that depends on the outcome j,

$$\sum_{i:\ j \in S_i} x_i.$$

The difference is the house profit.

(a) *Optimal house strategy.* How should the house decide on x so that its worst-case profit (over the possible outcomes) is maximized? (The house determines x after examining all the participant offers.)

(b) *Imputed probabilities.* Suppose $x^\star$ maximizes the worst-case house profit. Show that there exists a probability distribution π on the possible outcomes (*i.e.*, $\pi \in \mathbf{R}^m_+$, $\mathbf{1}^T\pi = 1$) for which $x^\star$ also maximizes the expected house profit. Explain how to find π.

Hint. Formulate the problem in part (a) as an LP; you can construct π from optimal dual variables for this LP.

Remark. Given π, the 'fair' price for offer i is $p_i^{\text{fair}} = \sum_{j\in S_i} \pi_j$. All offers with $p_i > p_i^{\text{fair}}$ will be completely filled (*i.e.*, $x_i = q_i$); all offers with $p_i < p_i^{\text{fair}}$ will be rejected (*i.e.*, $x_i = 0$).

Remark. This exercise shows how the probabilities of outcomes (*e.g.*, elections) can be guessed from the offers of a set of gamblers.

(c) *Numerical example.* Carry out your method on the simple example below with $n = 5$ participants, $m = 5$ possible outcomes, and participant offers

Participant i	p_i	q_i	S_i
1	0.50	10	{1,2}
2	0.60	5	{4}
3	0.60	5	{1,4,5}
4	0.60	20	{2,5}
5	0.20	10	{3}

Compare the optimal worst-case house profit with the worst-case house profit, if all offers were accepted (*i.e.*, $x_i = q_i$). Find the imputed probabilities.

13.10 *Optimal investment to fund an expense stream.* An organization (such as a municipality) knows its operating expenses over the next T periods, denoted $E_1, \ldots, E_T$. (Normally these are positive; but we can have negative E_t, which corresponds to income.) These expenses will be funded by a combination of investment income, from a mixture of bonds purchased at $t = 0$, and a cash account.

The bonds generate investment income, denoted $I_1, \ldots, I_T$. The cash balance is denoted $B_0, \ldots, B_T$, where $B_0 \geq 0$ is the amount of the initial deposit into the cash account. We can have $B_t < 0$ for $t = 1, \ldots, T$, which represents borrowing.

After paying for the expenses using investment income and cash, in period t, we are left with $B_t - E_t + I_t$ in cash. If this amount is positive, it earns interest at the rate $r_+ > 0$; if it is negative, we must pay interest at rate r_-, where $r_- \geq r_+$. Thus the expenses, investment income, and cash balances are linked as follows:

$$B_{t+1} = \begin{cases} (1+r_+)(B_t - E_t + I_t) & B_t - E_t + I_t \geq 0 \\ (1+r_-)(B_t - E_t + I_t) & B_t - E_t + I_t < 0, \end{cases}$$

for $t = 1, \ldots, T-1$. We take $B_1 = (1+r_+)B_0$, and we require that $B_T - E_T + I_T = 0$, which means the final cash balance, plus income, exactly covers the final expense.

The initial investment will be a mixture of bonds, labeled $1, \ldots, n$. Bond i has a price $P_i > 0$, a coupon payment $C_i > 0$, and a maturity $M_i \in \{1, \ldots, T\}$. Bond i generates an income stream

given by

$$a_t^{(i)} = \begin{cases} C_i & t < M_i \\ C_i + 1 & t = M_i \\ 0 & t > M_i, \end{cases}$$

for $t = 1, \ldots, T$. If x_i is the number of units of bond i purchased (at $t = 0$), the total investment cash flow is

$$I_t = x_1 a_t^{(1)} + \cdots + x_n a_t^{(n)}, \quad t = 1, \ldots, T.$$

We will require $x_i \geq 0$. (The x_i can be fractional; they do not need to be integers.)

The total initial investment required to purchase the bonds, and fund the initial cash balance at $t = 0$, is $x_1 P_1 + \cdots + x_n P_n + B_0$.

(a) Explain how to choose x and B_0 to minimize the total initial investment required to fund the expense stream.

(b) Solve the problem instance given in `opt_funding_data.m`. Give optimal values of x and B_0. Give the optimal total initial investment, and compare it to the initial investment required if no bonds were purchased (which would mean that all the expenses were funded from the cash account). Plot the cash balance (versus period) with optimal bond investment, and with no bond investment.

13.11 *Planning production with uncertain demand.* You must order (nonnegative) amounts $r_{\mathrm{I}}, \ldots, r_m$ of raw materials, which are needed to manufacture (nonnegative) quantities $q_1, \ldots, q_n$ of n different products. To manufacture one unit of product j requires at least A_{ij} units of raw material i, so we must have $r \succeq Aq$. (We will assume that A_{ij} are nonnegative.) The per-unit cost of the raw materials is given by $c \in \mathbf{R}_+^m$, so the total raw material cost is $c^T r$.

The (nonnegative) demand for product j is denoted d_j; the number of units of product j sold is $s_j = \min\{q_j, d_j\}$. (When $q_j > d_j$, $q_j - d_j$ is the amount of product j produced, but not sold; when $d_j > q_j$, $d_j - q_j$ is the amount of unmet demand.) The revenue from selling the products is $p^T s$, where $p \in \mathbf{R}_+^n$ is the vector of product prices. The profit is $p^T s - c^T r$. (Both d and q are real vectors; their entries need not be integers.)

You are given A, c, and p. The product demand, however, is not known. Instead, a set of K possible demand vectors, $d^{(1)}, \ldots, d^{(K)}$, with associated probabilities $\pi_1, \ldots, \pi_K$, is given. (These satisfy $\mathbf{1}^T \pi = 1$, $\pi \succeq 0$.)

You will explore two different optimization problems that arise in choosing r and q (the variables).

I. Choose r and q ahead of time. You must choose r and q, knowing only the data listed above. (In other words, you must order the raw materials, and commit to producing the chosen quantities of products, before you know the product demand.) The objective is to maximize the expected profit.

II. Choose r ahead of time, and q after d is known. You must choose r, knowing only the data listed above. Some time after you have chosen r, the demand will become known to you. This means that you will find out which of the K demand vectors is the true demand. Once you

know this, you must choose the quantities to be manufactured. (In other words, you must order the raw materials before the product demand is known; but you can choose the mix of products to manufacture after you have learned the true product demand.) The objective is to maximize the expected profit.

(a) Explain how to formulate each of these problems as a convex optimization problem. Clearly state what the variables are in the problem, what the constraints are, and describe the roles of any auxiliary variables or constraints you introduce.

(b) Carry out the methods from part (a) on the problem instance with numerical data given in `planning_data.m`. This file will define `A`, `D`, `K`, `c`, `m`, `n`, `p` and `pi`. The K columns of D are the possible demand vectors. For both of the problems described above, give the optimal value of r, and the expected profit.

13.12 *Gini coefficient of inequality.* Let $x_1, \ldots, x_n$ be a set of nonnegative numbers with positive sum, which typically represent the wealth or income of n individuals in some group. The *Lorentz curve* is a plot of the fraction f_i of total wealth held by the i poorest individuals,

$$f_i = (1/\mathbf{1}^T x) \sum_{j=1}^{i} x_{(j)}, \quad i = 0, \ldots, n,$$

versus i/n, where $x_{(j)}$ denotes the jth smallest of the numbers $\{x_1, \ldots, x_n\}$, and we take $f_0 = 0$. The Lorentz curve starts at $(0, 0)$ and ends at $(1, 1)$. Interpreted as a continuous curve (as, say, $n \to \infty$) the Lorentz curve is convex and increasing, and lies on or below the straight line joining the endpoints. The curve coincides with this straight line, *i.e.*, $f_i = (i/n)$, if and only if the wealth is distributed equally, *i.e.*, the x_i are all equal.

The *Gini coefficient* is defined as twice the area between the straight line corresponding to uniform wealth distribution and the Lorentz curve:

$$G(x) = (2/n) \sum_{i=1}^{n} ((i/n) - f_i).$$

The Gini coefficient is used as a measure of wealth or income inequality: It ranges between 0 (for equal distribution of wealth) and $1 - 1/n$ (when one individual holds all wealth).

(a) Show that G is a quasiconvex function on $x \in \mathbf{R}^n_+ \setminus \{0\}$.

(b) *Gini coefficient and marriage.* Suppose that individuals i and j get married ($i \neq j$) and therefore pool wealth. This means that x_i and x_j are both replaced with $(x_i + x_j)/2$. What can you say about the change in Gini coefficient caused by this marriage?

13.13 *Internal rate of return for cash streams with a single initial investment.* We use the notation of example 3.34 in the textbook. Let $x \in \mathbf{R}^{n+1}$ be a cash flow over n periods, with x indexed from 0 to n, where the index denotes period number. We assume that $x_0 < 0$, $x_j \geq 0$ for $j = 1, \ldots, n$, and $x_0 + \cdots + x_n > 0$. This means that there is an initial positive investment; thereafter, only payments are made, with the total of the payments exceeding the initial investment. (In the more general setting of example 3.34, we allow additional investments to be made after the initial investment.)

(a) Show that $\mathrm{IRR}(x)$ is quasilinear in this case.

(b) *Blending initial investment only streams.* Use the result in part (a) to show the following. Let $x^{(i)} \in \mathbf{R}^{n+1}$, $i = 1, \ldots, k$, be a set of k cash flows over n periods, each of which satisfies the conditions above. Let $w \in \mathbf{R}^k_+$, with $\mathbf{1}^T w = 1$, and consider the blended cash flow given by $x = w_1 x^{(1)} + \cdots + w_k x^{(k)}$. (We can think of this as investing a fraction w_i in cash flow i.) Show that $\mathrm{IRR}(x) \leq \max_i \mathrm{IRR}(x^{(i)})$. Thus, blending a set of cash flows (with initial investment only) will not improve the IRR over the best individual IRR of the cash flows.

13.14 *Efficient solution of basic portfolio optimization problem.* This problem concerns the simplest possible portfolio optimization problem:

$$\begin{array}{ll} \text{maximize} & \mu^T w - (\lambda/2) w^T \Sigma w \\ \text{subject to} & \mathbf{1}^T w = 1, \end{array}$$

with variable $w \in \mathbf{R}^n$ (the normalized portfolio, with negative entries meaning short positions), and data μ (mean return), $\Sigma \in \mathbf{S}^n_{++}$ (return covariance), and $\lambda > 0$ (the risk aversion parameter). The return covariance has the factor form $\Sigma = FQF^T + D$, where $F \in \mathbf{R}^{n \times k}$ (with rank K) is the *factor loading matrix*, $Q \in \mathbf{S}^k_{++}$ is the factor covariance matrix, and D is a diagonal matrix with positive entries, called the *idiosyncratic risk* (since it describes the risk of each asset that is independent of the factors). This form for Σ is referred to as a 'k-factor risk model'. Some typical dimensions are $n = 2500$ (assets) and $k = 30$ (factors).

(a) What is the flop count for computing the optimal portfolio, if the low-rank plus diagonal structure of Σ is *not* exploited? You can assume that $\lambda = 1$ (which can be arranged by absorbing it into Σ).

(b) Explain how to compute the optimal portfolio more efficiently, and give the flop count for your method. You can assume that $k \ll n$. You do not have to give the best method; any method that has linear complexity in n is fine. You can assume that $\lambda = 1$.

Hints. You may want to introduce a new variable $y = F^T w$ (which is called the vector of factor exposures). You may want to work with the matrix

$$G = \begin{bmatrix} \mathbf{1} & F \\ 0 & -I \end{bmatrix} \in \mathbf{R}^{(n+k) \times (1+k)},$$

treating it as dense, ignoring the (little) exploitable structure in it.

(c) Carry out your method from part (b) on some randomly generated data with dimensions $n = 2500$, $k = 30$. For comparison (and as a check on your method), compute the optimal portfolio using the method of part (a) as well. Give the (approximate) CPU time for each method, using `tic` and `toc`. *Hints.* After you generate D and Q randomly, you might want to add a positive multiple of the identity to each, to avoid any issues related to poor conditioning. Also, to be able to invert a block diagonal matrix efficiently, you'll need to recast it as sparse.

(d) *Risk return trade-off curve.* Now suppose we want to compute the optimal portfolio for M values of the risk aversion parameter λ. Explain how to do this efficiently, and give the complexity in terms of M, n, and k. Compare to the complexity of using the method of part (b) M times. *Hint.* Show that the optimal portfolio is an affine function of $1/\lambda$.

13.15 *Sparse index tracking.* The (weekly, say) return of n stocks is given by a random variable $r \in \mathbf{R}^n$, with mean $\overline{r}$ and covariance $\mathbf{E}(r-\overline{r})(r-\overline{r})^T = \Sigma \succ 0$. An index (such as S&P 500 or Wilshire 5000) is a weighted sum of these returns, given by $z = c^T r$, where $c \in \mathbf{R}^n_+$. (For example, the vector c is nonzero only for the stocks in the index, and the coefficients c_i might be proportional to some measure of market capitalization of stock i.) We will assume that the index weights $c \in \mathbf{R}^n$, as well as the return mean and covariance $\overline{r}$ and Σ, are known and fixed.

Our goal is to find a *sparse* weight vector $w \in \mathbf{R}^n$, which can include negative entries (meaning, short positions), so that the RMS index tracking error, defined as

$$E = \left(\frac{\mathbf{E}(z - w^T r)^2}{\mathbf{E}\, z^2} \right)^{1/2},$$

does not exceed 0.10 (*i.e.*, 10%). Of course, taking $w = c$ results in $E = 0$, but we are interested in finding a weight vector with (we hope) many fewer nonzero entries than c has.

Remark. This is the idea behind an *index fund*: You find a sparse portfolio that replicates or tracks the return of the index (within some error tolerance). Acquiring (and rebalancing) the sparse tracking portfolio will incur smaller transactions costs than trading in the full index.

(a) Propose a (simple) heuristic method for finding a sparse weight vector w that satisfies $E \leq 0.10$.

(b) Carry out your method on the problem instance given in `sparse_idx_track_data.m`. Give $\mathbf{card}(w)$, the number of nonzero entries in w. (To evaluate $\mathbf{card}(w)$, use `sum(abs(w)>0.01)`, which treats weight components smaller than 0.01 as zero.) (You might want to compare the index weights and the weights you find by typing `[c w]`. No need to print or turn in the resulting output, though.)

13.16 *Option price bounds.* In this problem we use the methods and results of example 5.10 to give bounds on the arbitrage-free price of an option. (See exercise 5.38 for a simple version of option pricing.) We will use all the notation and definitions from Example 5.10.

We consider here options on an underlying asset (such as a stock); these have a payoff or value that depends on S, the value of the underlying asset at the end of the investment period. We will assume that the underying asset can only take on m different values, $S^{(1)}, \ldots, S^{(m)}$. These correspond to the m possible scenarios or outcomes described in Example 5.10.

A risk-free asset has value $r > 1$ in every scenario.

A *put option* at *strike price* K gives the owner the right to sell one unit of the underlying stock at price K. At the end of the investment period, if the stock is trading at a price S, then the put option has payoff $(K-S)_+ = \max\{0, K-S\}$ (since the option is exercised only if $K > S$). Similarly a *call option* at strike price K gives the buyer the right to buy a unit of stock at price K. A call option has payoff $(S-K)_+ = \max\{0, S-K\}$.

A *collar* is an option with payoff

$$\begin{cases} C - S_0 & S > C \\ S - S_0 & F \leq S \leq C \\ F - S_0 & S < F \end{cases}$$

where F is the *floor*, C is the *cap* and S_0 is the price of the underlying at the start of the investment period. This option limits both the upside and downside of payoff.

Now we consider a specific problem. The price of the risk-free asset, with $r = 1.05$, is 1. The price of the underlying asset is $S_0 = 1$. We will use $m = 200$ scenarios, with $S^{(i)}$ uniformly spaced from $S^{(1)} = 0.5$ to $S^{(200)} = 2$. The following options are traded on an exchange, with prices listed below.

Type	Strike	Price
Call	1.1	0.06
Call	1.2	0.03
Put	0.8	0.02
Put	0.7	0.01.

A collar with floor $F = 0.9$ and cap $C = 1.15$ is not traded on an exchange. Find the range of prices for this collar, consistent with the absence of arbitrage and the prices given above.

13.17 *Portfolio optimization with qualitative return forecasts.* We consider the risk-return portfolio optimization problem described on pages 155 and 185 of the book, with one twist: We don't precisely know the mean return vector $\bar{p}$. Instead, we have a range of possible values for each asset, *i.e.*, we have $l,\ u \in \mathbf{R}^n$ with $l \preceq \bar{p} \preceq u$. We use l and u to encode various qualitative forecasts we have about the mean return vector $\bar{p}$. For example, $l_7 = 0.02$ and $u_7 = 0.20$ means that we believe the mean return for asset 7 is between 2% and 20%.

Define the *worst-case mean return* R^{wc}, as a function of portfolio vector x, as the worst (minimum) value of $\bar{p}^T x$, over all $\bar{p}$ consistent with the given bounds l and u.

(a) Explain how to find a portfolio x that maximizes R^{wc}, subject to a budget constraint and risk limit,

$$\mathbf{1}^T x = 1, \qquad x^T \Sigma x \leq \sigma_{\max}^2,$$

where $\Sigma \in \mathbf{S}^n_{++}$ and $\sigma_{\max} \in \mathbf{R}_{++}$ are given.

(b) Solve the problem instance given in `port_qual_forecasts_data.m`. Give the optimal worst-case mean return achieved by the optimal portfolio $x^\star$.

In addition, construct a portfolio x^{mid} that maximizes $c^T x$ subject to the budget constraint and risk limit, where $c = (1/2)(l + u)$. This is the optimal portfolio assuming that the mean return has the midpoint value of the forecasts. Compare the midpoint mean returns $c^T x^{\mathrm{mid}}$ and $c^T x^\star$, and the worst-case mean returns of x^{mid} and $x^\star$.

Briefly comment on the results.

13.18 *De-leveraging.* We consider a multi-period portfolio optimization problem, with n assets and T time periods, where $x_t \in \mathbf{R}^n$ gives the holdings (say, in dollars) at time t, with negative entries denoting, as usual, short positions. For each time period the return vector has mean $\mu \in \mathbf{R}^n$ and covariance $\Sigma \in \mathbf{S}^n_{++}$. (These are known.)

The initial portfolio x_0 maximizes the risk-adjusted expected return $\mu^T x - \gamma x^T \Sigma x$, where $\gamma > 0$, subject to the leverage limit constraint $\|x\|_1 \leq L^{\mathrm{init}}$, where $L^{\mathrm{init}} > 0$ is the given initial leverage limit. (There are several different ways to measure leverage; here we use the sum of the total short and long positions.) The final portfolio x_T maximizes the risk-adjusted return, subject to

$\|x\|_1 \leq L^{\text{new}}$, where $L^{\text{new}} > 0$ is the given final leverage limit (with $L^{\text{new}} < L^{\text{init}}$). This uniquely determines x_0 and x_T, since the objective is strictly concave.

The question is how to move from x_0 to x_T, *i.e.*, how to choose $x_1, \ldots, x_{T-1}$. We will do this so as to maximize the objective

$$J = \sum_{t=1}^{T} \left(\mu^T x_t - \gamma x_t^T \Sigma x_t - \phi(x_t - x_{t-1}) \right),$$

which is the total risk-adjusted expected return, minus the total transaction cost. The transaction cost function ϕ has the form

$$\phi(u) = \sum_{i=1}^{n} \left(\kappa_i |u_i| + \lambda_i u_i^2 \right),$$

where $\kappa \succeq 0$ and $\lambda \succeq 0$ are known parameters. We will require that $\|x_t\|_1 \leq L^{\text{init}}$, for $t = 1, \ldots, T-1$. In other words, the leverage limit is the initial leverage limit up until the deadline T, when it drops to the new lower value.

(a) Explain how to find the portfolio sequence $x_1^\star, \ldots, x_{T-1}^\star$ that maximizes J subject to the leverage limit constraints.

(b) Find the optimal portfolio sequence $x_t^\star$ for the problem instance with data given in `deleveraging_data.m`. Compare this sequence with two others: $x_t^{\text{lp}} = x_0$ for $t = 1, \ldots, T-1$ (*i.e.*, one that does all trading at the last possible period), and the linearly interpolated portfolio sequence

$$x_t^{\text{lin}} = (1 - t/T)x_0 + (t/T)x_T, \quad t = 1, \ldots, T-1.$$

For each of these three portfolio sequences, give the objective value obtained, and plot the risk and transaction cost adjusted return,

$$\mu^T x_t - \gamma x_t^T \Sigma x_t - \phi(x_t - x_{t-1}),$$

and the leverage $\|x_t\|_1$, versus t, for $t = 0, \ldots, T$. Also, for each of the three portfolio sequences, generate a single plot that shows how the holdings $(x_t)_i$ of the n assets change over time, for $i = 1, \ldots, n$.

Give a *very short* (one or two sentence) intuitive explanation of the results.

13.19 *Worst-case variance.* Suppose Z is a random variable on $\mathbf{R}^n$ with covariance matrix $\Sigma \in \mathbf{S}_+^n$. Let $c \in \mathbf{R}^n$. The variance of $Y = c^T Z$ is $\mathbf{var}(Y) = c^T \Sigma c$. We define the *worst-case variance* of Y, denoted $\mathbf{wcvar}(Y)$, as the maximum possible value of $c^T \tilde{\Sigma} c$, over all $\tilde{\Sigma} \in \mathbf{S}_+^n$ that satisfy $\Sigma_{ii} = \tilde{\Sigma}_{ii}$, $i = 1, \ldots, n$. In other words, the worst-case variance of Y is the maximum possible variance, if we are allowed to arbitrarily change the correlations between Z_i and Z_j. Of course we have $\mathbf{wcvar}(Y) \geq \mathbf{var}(Y)$.

(a) Find a simple expression for $\mathbf{wcvar}(Y)$ in terms of c and the diagonal entries of Σ. You must justify your expression.

(b) *Portfolio optimization.* Explain how to find the portfolio $x \in \mathbf{R}^n$ that maximizes the expected return $\mu^T x$ subject to a limit on risk, $\mathbf{var}(r^T x) = x^T \Sigma x \leq R$, and a limit on worst-case risk $\mathbf{wcvar}(r^T x) \leq R^{\text{wc}}$, where $R > 0$ and $R^{\text{wc}} > R$ are given. Here $\mu = \mathbf{E}\, r$ and $\Sigma = \mathbf{E}(r - \mu)(r - \mu)^T$ are the (given) mean and covariance of the (random) return vector $r \in \mathbf{R}^n$.

(c) Carry out the method of part (b) for the problem instance with data given in `wc_risk_portfolio_opt_data.m`. Also find the optimal portfolio when the worst-case risk limit is ignored. Find the expected return and worst-case risk for these two portfolios.

Remark. If a portfolio is highly leveraged, and the correlations in the returns change drastically, you (the portfolio manager) can be in big trouble, since you are now exposed to much more risk than you thought you were. And yes, this (almost exactly) has happened.

13.20 *Risk budget allocation.* Suppose an amount $x_i > 0$ is invested in n assets, labeled $i = 1, ..., n$, with asset return covariance matrix $\Sigma \in \mathbf{S}^n_{++}$. We define the *risk* of the investments as the standard deviation of the total return, $R(x) = (x^T \Sigma x)^{1/2}$.

We define the (relative) *risk contribution* of asset i (in the portfolio x) as

$$\rho_i = \frac{\partial \log R(x)}{\partial \log x_i} = \frac{\partial R(x)}{R(x)} \frac{x_i}{\partial x_i}, \quad i = 1, \ldots, n.$$

Thus ρ_i gives the fractional increase in risk per fractional increase in investment i. We can express the risk contributions as

$$\rho_i = \frac{x_i (\Sigma x)_i}{x^T \Sigma x}, \quad i = 1, \ldots, n,$$

from which we see that $\sum_{i=1}^n \rho_i = 1$. For general x, we can have $\rho_i < 0$, which means that a small increase in investment i decreases the risk. Desirable investment choices have $\rho_i > 0$, in which case we can interpret ρ_i as the fraction of the total risk contributed by the investment in asset i. Note that the risk contributions are homogeneous of degree zero, *i.e.*, scaling x by a positive constant does not affect ρ_i.

In the *risk budget allocation problem*, we are given Σ and a set of desired risk contributions $\rho_i^{\text{des}} > 0$ with $\mathbf{1}^T \rho^{\text{des}} = 1$; the goal is to find an investment mix $x \succ 0$, $\mathbf{1}^T x = 1$, with these risk contributions. When $\rho^{\text{des}} = (1/n)\mathbf{1}$, the problem is to find an investment mix that achieves so-called *risk parity*.

(a) Explain how to solve the risk budget allocation problem using convex optimization.
Hint. Minimize $(1/2)x^T \Sigma x - \sum_{i=1}^n \rho_i^{\text{des}} \log x_i$.

(b) Find the investment mix that achieves risk parity for the return covariance matrix

$$\Sigma = \begin{bmatrix} 6.1 & 2.9 & -0.8 & 0.1 \\ 2.9 & 4.3 & -0.3 & 0.9 \\ -0.8 & -0.3 & 1.2 & -0.7 \\ 0.1 & 0.9 & -0.7 & 2.3 \end{bmatrix}.$$

For your convenience, this is contained in `risk_alloc_data.m`.

13.21 *Portfolio rebalancing.* We consider the problem of rebalancing a portfolio of assets over multiple periods. We let $h_t \in \mathbf{R}^n$ denote the vector of our dollar value holdings in n assets, at the beginning of period t, for $t = 1, \ldots, T$, with negative entries meaning short positions. We will work with the portfolio weight vector, defined as $w_t = h_t/(\mathbf{1}^T h_t)$, where we assume that $\mathbf{1}^T h_t > 0$, *i.e.*, the total portfolio value is positive.

The *target portfolio weight vector* $w^\star$ is defined as the solution of the problem

$$\begin{array}{ll} \text{maximize} & \mu^T w - \frac{\gamma}{2} w^T \Sigma w \\ \text{subject to} & \mathbf{1}^T w = 1, \end{array}$$

where $w \in \mathbf{R}^n$ is the variable, μ is the mean return, $\Sigma \in \mathbf{S}^n_{++}$ is the return covariance, and $\gamma > 0$ is the risk aversion parameter. The data μ, Σ, and γ are given. In words, the target weights maximize the risk-adjusted expected return.

At the beginning of each period t we are allowed to rebalance the portfolio by buying and selling assets. We call the post-trade portfolio weights $\tilde{w}_t$. They are found by solving the (rebalancing) problem

$$\begin{array}{ll} \text{maximize} & \mu^T w - \frac{\gamma}{2} w^T \Sigma w - \kappa^T |w - w_t| \\ \text{subject to} & \mathbf{1}^T w = 1, \end{array}$$

with variable $w \in \mathbf{R}^n$, where $\kappa \in \mathbf{R}^n_+$ is the vector of (so-called linear) transaction costs for the assets. (For example, these could model bid/ask spread.) Thus, we choose the post-trade weights to maximize the risk-adjusted expected return, minus the transactions costs associated with rebalancing the portfolio. Note that the pre-trade weight vector w_t is known at the time we solve the problem. If we have $\tilde{w}_t = w_t$, it means that no rebalancing is done at the beginning of period t; we simply hold our current portfolio. (This happens if $w_t = w^\star$, for example.)

After holding the rebalanced portfolio over the investment period, the dollar value of our portfolio becomes $h_{t+1} = \mathbf{diag}(r_t)\tilde{h}_t$, where $r_t \in \mathbf{R}^n_{++}$ is the (random) vector of asset returns over period t, and $\tilde{h}_t$ is the post-trade portfolio given in dollar values (which you do not need to know). The next weight vector is then given by

$$w_{t+1} = \frac{\mathbf{diag}\,(r_t)\tilde{w}_t}{r_t^T \tilde{w}_t}.$$

(If $r_t^T \tilde{w}_t \leq 0$, which means our portfolio has negative value after the investment period, we have gone bust, and all trading stops.) The standard model is that r_t are IID random variables with mean and covariance μ and Σ, but this is not relevant in this problem.

(a) *No-trade condition.* Show that $\tilde{w}_t = w_t$ is optimal in the rebalancing problem if

$$\gamma\,|\Sigma(w_t - w^\star)| \preceq \kappa$$

holds, where the absolute value on the left is elementwise.

Interpretation. The lefthand side measures the deviation of w_t from the target portfolio $w^\star$; when this deviation is smaller than the cost of trading, you do not rebalance.

Hint. Find dual variables, that with $w = w_t$ satisfy the KKT conditions for the rebalancing problem.

(b) Starting from $w_1 = w^\star$, compute a sequence of portfolio weights $\tilde{w}_t$ for $t = 1, \ldots, T$. For each t, find $\tilde{w}_t$ by solving the rebalancing problem (with w_t a known constant); then generate a vector of returns r_t (using our supplied function) to compute w_{t+1} (The sequence of weights is random, so the results won't be the same each time you run your script. But they should look similar.)

Report the fraction of periods in which the no-trade condition holds and the fraction of periods in which the solution has only zero (or negligible) trades, defined as $\|\tilde{w}_t - w_t\|_\infty \leq 10^{-3}$. Plot the sequence $\tilde{w}_t$ for $t = 1, 2, \ldots, T$.

The file `portf_weight_rebalance_data.*` provides the data, a function to generate a (random) vector r_t of market returns, and the code to plot the sequence $\tilde{w}_t$. (The plotting code also draws a dot for every non-negligible trade.)

Carry this out for two values of κ, $\kappa = \kappa_1$ and $\kappa = \kappa_2$. Briefly comment on what you observe.

Hint. In CVXPY we recommend using the solver ECOS. But if you use SCS you should increase the default accuracy, by passing `eps=1e-4` to the `cvxpy.Problem.solve()` method.

13.22 *Portfolio optimization using multiple risk models.* Let $w \in \mathbf{R}^n$ be a vector of portfolio weights, where negative values correspond to short positions, and the weights are normalized such that $\mathbf{1}^T w = 1$. The expected return of the portfolio is $\mu^T w$, where $\mu \in \mathbf{R}^n$ is the (known) vector of expected asset returns. As usual we measure the risk of the portfolio using the variance of the portfolio return. However, in this problem we do not know the covariance matrix Σ of the asset returns; instead we assume that Σ is one of M (known) covariance matrices $\Sigma^{(k)} \in \mathbf{S}^n_{++}$, $k = 1, \ldots, M$. We can think of the $\Sigma^{(k)}$ as representing M different risk models, associated with M different market regimes (say). For a weight vector w, there are M different possible values of the risk: $w^T \Sigma^{(k)} w$, $k = 1, \ldots, M$. The worst-case risk, across the different models, is given by $\max_{k=1,\ldots,M} w^T \Sigma^{(k)} w$. (This is the same as the worst-case risk over all covariance matrices in the convex hull of $\Sigma^{(1)}, \ldots, \Sigma^{(M)}$.)

We will choose the portfolio weights in order to maximize the expected return, adjusted by the worst-case risk, *i.e.*, as the solution $w^\star$ of the problem

$$\begin{array}{ll} \text{maximize} & \mu^T w - \gamma \max_{k=1,\ldots,M} w^T \Sigma^{(k)} w \\ \text{subject to} & \mathbf{1}^T w = 1, \end{array}$$

with variable w, where $\gamma > 0$ is a given risk-aversion parameter. We call this the mean-worst-case-risk portfolio problem.

(a) Show that there exist $\gamma_1, \ldots, \gamma_M \geq 0$ such that $\sum_{k=1}^M \gamma_k = \gamma$ and the solution $w^\star$ of the mean-worst-case-risk portfolio problem is also the solution of the problem

$$\begin{array}{ll} \text{maximize} & \mu^T w - \sum_{k=1}^M \gamma_k w^T \Sigma^{(k)} w \\ \text{subject to} & \mathbf{1}^T w = 1, \end{array}$$

with variable w.

Remark. The result above has a beautiful interpretation: We can think of the γ_k as allocating our total risk aversion γ in the mean-worst-case-risk portfolio problem across the M different regimes.

Hint. The values γ_k are not easy to find: you have to solve the mean-worst-case-risk problem to get them. Thus, this result does not help us solve the mean-worst-case-risk problem; it simply gives a nice interpretation of its solution.

(b) Find the optimal portfolio weights for the problem instance with data given in `multi_risk_portfolio_data.*`. Report the weights and the values of γ_k, $k = 1, \ldots, M$. Give the M possible values of the risk associated with your weights, and the worst-case risk.

13.23 *Computing market-clearing prices.* We consider n commodities or goods, with $p \in \mathbf{R}^n_{++}$ the vector of prices (per unit quantity) of them. The (nonnegative) demand for the products is a function of the prices, which we denote $D : \mathbf{R}^n \to \mathbf{R}^n$, so $D(p)$ is the demand when the product prices are p. The (nonnegative) supply of the products (*i.e.*, the amounts that manufacturers are willing to produce) is also a function of the prices, which we denote $S : \mathbf{R}^n \to \mathbf{R}^n$, so $S(p)$ is the supply when the product prices are p. We say that the market *clears* if $S(p) = D(p)$, *i.e.*, supply equals demand, and we refer to p in this case as a set of *market-clearing prices*.

Elementary economics courses consider the special case $n = 1$, *i.e.*, a single commodity, so supply and demand can be plotted (vertically) against the price (on the horizontal axis). It is assumed that demand decreases with increasing price, and supply increases; the market clearing price can be found 'graphically', as the point where the supply and demand curves intersect. In this problem we examine some cases in which market-clearing prices (for the general case $n > 1$) can be computed using convex optimization.

We assume that the demand function is *Hicksian*, which means it has the form $D(p) = \nabla E(p)$, where $E : \mathbf{R}^n \to \mathbf{R}$ is a differentiable function that is concave and increasing in each argument, called the *expenditure function*. (While not relevant in this problem, Hicksian demand arises from a model in which consumers make purchases by maximizing a concave utility function.)

We will assume that the producers are independent, so $S(p)_i = S_i(p_i)$, $i = 1, \ldots, n$, where $S_i : \mathbf{R} \to \mathbf{R}$ is the supply function for good i. We will assume that the supply functions are positive and increasing on their domain $\mathbf{R}_+$.

(a) Explain how to use convex optimization to find market-clearing prices under the assumptions given above. (You do not need to worry about technical details like zero prices, or cases in which there are no market-clearing prices.)

(b) Compute market-clearing prices for the specific case with $n = 4$,

$$E(p) = \left(\prod_{i=1}^{4} p_i\right)^{1/4},$$

$$S(p) = (0.2p_1 + 0.5, 0.02p_2 + 0.1, 0.04p_3, 0.1p_4 + 0.2).$$

Give the market-clearing prices and the demand and supply (which should match) at those prices.

Hint: In CVX and CVXPY, `geo_mean` gives the geometric mean of the entries of a vector argument. Julia does not yet have a vector argument `geom_mean` function, but you can get the geometric mean of 4 variables a, b, c, d using `geomean(geomean(a, b), geomean(c, d))`.

13.24 *Funding an expense stream.* Your task is to fund an expense stream over n time periods. We consider an expense stream $e \in \mathbf{R}^n$, so that e_t is our expenditure at time t.

One possibility for funding the expense stream is through our bank account. At time period t, the account has balance b_t and we withdraw an amount w_t. (A negative withdrawal represents a deposit.) The value of our bank account accumulates with an interest rate ρ per time period, less withdrawals:

$$b_{t+1} = (1 + \rho)b_t - w_t.$$

We assume the account value must be nonnegative, so that $b_t \geq 0$ for all t.

We can also use other investments to fund our expense stream, which we purchase at the initial time period $t = 1$, and which pay out over the n time periods. The amount each investment type pays out over the n time periods is given by the *payout matrix* P, defined so that P_{tj} is the amount investment type j pays out at time period t per dollar invested. There are m investment types, and we purchase $x_j \geq 0$ dollars of investment type j. In time period t, the total payout of all investments purchased is therefore given by $(Px)_t$.

In each time period, the sum of the withdrawals and the investment payouts must cover the expense stream, so that

$$w_t + (Px)_t \geq e_t$$

for all $t = 1, \ldots, n$.

The total amount we invest to fund the expense stream is the sum of the initial account balance, and the sum total of the investments purchased: $b_1 + \mathbf{1}^T x$.

(a) Show that the minimum initial investment that funds the expense stream can be found by solving a convex optimization problem.

(b) Using the data in `expense_stream_data.*`, carry out your method in part (a). On three graphs, plot the expense stream, the payouts from the m investment types (so m different curves), and the bank account balance, all as a function of the time period t. Report the minimum initial investment, and the initial investment required when no investments are purchased (so $x = 0$).

14 Mechanical and aerospace engineering

14.1 *Optimal design of a tensile structure.* A tensile structure is modeled as a set of n masses in $\mathbf{R}^2$, some of which are fixed, connected by a set of N springs. The masses are in equilibrium, with spring forces, connection forces for the fixed masses, and gravity balanced. (This equilibrium occurs when the position of the masses minimizes the total energy, defined below.)

We let $(x_i, y_i) \in \mathbf{R}^2$ denote the position of mass i, and $m_i > 0$ its mass value. The first p masses are fixed, which means that $x_i = x_i^{\text{fixed}}$ and $y_i = y_i^{\text{fixed}}$, for $i = 1, \ldots, p$. The gravitational potential energy of mass i is gm_iy_i, where $g \approx 9.8$ is the gravitational acceleration.

Suppose spring j connects masses r and s. Its elastic potential energy is

$$(1/2)k_j\left((x_r - x_s)^2 + (y_r - y_s)^2\right),$$

where $k_j \geq 0$ is the stiffness of spring j.

To describe the topology, *i.e.*, which springs connect which masses, we will use the incidence matrx $A \in \mathbf{R}^{n\times N}$, defined as

$$A_{ij} = \begin{cases} 1 & \text{head of spring } j \text{ connects to mass } i \\ -1 & \text{tail of spring } j \text{ connects to mass } i \\ 0 & \text{otherwise.} \end{cases}$$

Here we arbitrarily choose a head and tail for each spring, but in fact the springs are completely symmetric, and the choice can be reversed without any effect. (Hopefully you will discover why it is convenient to use the incidence matrix A to specify the topology of the system.)

The total energy is the sum of the gravitational energies, over all the masses, plus the sum of the elastic energies, over all springs. The equilibrium positions of the masses is the point that minimizes the total energy, subject to the constraints that the first p positions are fixed. (In the equilibrium positions, the total force on each mass is zero.) We let $E_{\min}$ denote the total energy of the system, in its equilibrium position. (We assume the energy is bounded below; this occurs if and only if each mass is connected, through some set of springs with positive stiffness, to a fixed mass.)

The total energy $E_{\min}$ is a measure of the stiffness of the structure, with larger $E_{\min}$ corresponding to stiffer. (We can think of $E_{\min} = -\infty$ as an infinitely unstiff structure; in this case, at least one mass is not even supported against gravity.)

(a) Suppose we know the fixed positions $x_1^{\text{fixed}}, \ldots, x_p^{\text{fixed}}$, $y_1^{\text{fixed}}, \ldots, y_p^{\text{fixed}}$, the mass values $m_1, \ldots, m_n$, the spring topology A, and the constant g. You are to choose nonnegative $k_1, \ldots, k_N$, subject to a budget constraint $\mathbf{1}^Tk = k_1 + \cdots + k_N = k^{\text{tot}}$, where k^{tot} is given. Your goal is to maximize $E_{\min}$.

Explain how to do this using convex optimization.

(b) Carry out your method for the problem data given in `tens_struct_data.m`. This file defines all the needed data, and also plots the equilibrium configuration when the stiffness is evenly distributed across the springs (*i.e.*, $k = (k^{\text{tot}}/N)\mathbf{1}$).

Report the optimal value of $E_{\min}$. Plot the optimized equilibrium configuration, and compare it to the equilibrium configuration with evenly distributed stiffness. (The code for doing this is in the file `tens_struct_data.m`, but commented out.)

14.2 *Equilibrium position of a system of springs.* We consider a collection of n masses in $\mathbf{R}^2$, with locations $(x_1, y_1), \ldots, (x_n, y_n)$, and masses $m_1, \ldots, m_n$. (In other words, the vector $x \in \mathbf{R}^n$ gives the x-coordinates, and $y \in \mathbf{R}^n$ gives the y-coordinates, of the points.) The masses m_i are, of course, positive.

For $i = 1, \ldots, n-1$, mass i is connected to mass $i+1$ by a spring. The potential energy in the ith spring is a function of the (Euclidean) distance $d_i = \|(x_i, y_i) - (x_{i+1}, y_{i+1})\|_2$ between the ith and $(i+1)$st masses, given by

$$E_i = \begin{cases} 0 & d_i < l_i \\ (k_i/2)(d_i - l_i)^2 & d_i \geq l_i \end{cases}$$

where $l_i \geq 0$ is the rest length, and $k_i > 0$ is the stiffness, of the ith spring. The gravitational potential energy of the ith mass is gm_iy_i, where g is a positive constant. The total potential energy of the system is therefore

$$E = \sum_{i=1}^{n-1} E_i + gm^T y.$$

The locations of the first and last mass are fixed. The equilibrium location of the other masses is the one that minimizes E.

(a) Show how to find the equilibrium positions of the masses $2, \ldots, n-1$ using convex optimization. Be sure to justify convexity of any functions that arise in your formulation (if it is not obvious). The problem data are m_i, k_i, l_i, g, x_1, y_1, x_n, and y_n.

(b) Carry out your method to find the equilibrium positions for a problem with $n = 10$, $m_i = 1$, $k_i = 10$, $l_i = 1$, $x_1 = y_1 = 0$, $x_n = y_n = 10$, with g varying from $g = 0$ (no gravity) to $g = 10$ (say). Verify that the results look reasonable. Plot the equilibrium configuration for several values of g.

14.3 *Elastic truss design.* In this problem we consider a truss structure with m bars connecting a set of nodes. Various external forces are applied at each node, which cause a (small) displacement in the node positions. $f \in \mathbf{R}^n$ will denote the vector of (components of) external forces, and $d \in \mathbf{R}^n$ will denote the vector of corresponding node displacements. (By 'corresponding' we mean if f_i is, say, the z-coordinate of the external force applied at node k, then d_i is the z-coordinate of the displacement of node k.) The vector f is called a *loading* or *load*.

The structure is linearly elastic, *i.e.*, we have a linear relation $f = Kd$ between the vector of external forces f and the node displacements d. The matrix $K = K^T \succ 0$ is called the *stiffness matrix* of the truss. Roughly speaking, the 'larger' K is (*i.e.*, the stiffer the truss) the smaller the node displacement will be for a given loading.

We assume that the geometry (unloaded bar lengths and node positions) of the truss is fixed; we are to design the cross-sectional areas of the bars. These cross-sectional areas will be the design variables x_i, $i = 1, \ldots, m$. The stiffness matrix K is a linear function of x:

$$K(x) = x_1K_1 + \cdots + x_mK_m,$$

where $K_i = K_i^T \succeq 0$ depend on the truss geometry. You can assume these matrices are given or known. The total weight W_{tot} of the truss also depends on the bar cross-sectional areas:

$$W_{\text{tot}}(x) = w_1x_1 + \cdots + w_mx_m,$$

where $w_i > 0$ are known, given constants (density of the material times the length of bar i). Roughly speaking, the truss becomes stiffer, but also heavier, when we increase x_i; there is a tradeoff between stiffness and weight.

Our goal is to design the stiffest truss, subject to bounds on the bar cross-sectional areas and total truss weight:

$$l \leq x_i \leq u, \quad i = 1, \ldots, m, \qquad W_{\text{tot}}(x) \leq W,$$

where l, u, and W are given. You may assume that $K(x) \succ 0$ for all feasible vectors x. To obtain a specific optimization problem, we must say how we will measure the stiffness, and what model of the loads we will use.

(a) There are several ways to form a scalar measure of how stiff a truss is, for a given load f. In this problem we will use the *elastic stored energy*

$$\mathcal{E}(x, f) = \frac{1}{2} f^T K(x)^{-1} f$$

to measure the stiffness. Maximizing stiffness corresponds to minimizing $\mathcal{E}(x, f)$.

Show that $\mathcal{E}(x, f)$ is a convex function of x on $\{x \mid K(x) \succ 0\}$.
Hint. Use Schur complements to prove that the epigraph is a convex set.

(b) We can consider several different scenarios that reflect our knowledge about the possible loadings f that can occur. The simplest is that f is a single, fixed, known loading. In more sophisticated formulations, the loading f might be a random vector with known distribution, or known only to lie in some set $\mathcal{F}$, etc.

Show that each of the following four problems is a convex optimization problem, with x as variable.

- *Design for a fixed known loading.* The vector f is known and fixed. The design problem is

$$\begin{array}{ll} \text{minimize} & \mathcal{E}(x, f) \\ \text{subject to} & l \leq x_i \leq u, \quad i = 1, \ldots, m \\ & W_{\text{tot}}(x) \leq W. \end{array}$$

- *Design for multiple loadings.* The vector f can take any of N known values $f^{(i)}$, $i = 1, \ldots, N$, and we are interested in the worst-case scenario. The design problem is

$$\begin{array}{ll} \text{minimize} & \max_{i=1,\ldots,N} \mathcal{E}(x, f^{(i)}) \\ \text{subject to} & l \leq x_i \leq u, \quad i = 1, \ldots, m \\ & W_{\text{tot}}(x) \leq W. \end{array}$$

- *Design for worst-case, unknown but bounded load.* Here we assume the vector f can take arbitrary values in a ball $B = \{f \mid \|f\|_2 \leq \alpha\}$, for a given value of α. We are interested in minimizing the worst-case stored energy, *i.e.*,

$$\begin{array}{ll} \text{minimize} & \sup_{\|f\|_2 \leq \alpha} \mathcal{E}(x, f^{(i)}) \\ \text{subject to} & l \leq x_i \leq u, \quad i = 1, \ldots, m \\ & W_{\text{tot}}(x) \leq W. \end{array}$$

- *Design for a random load with known statistics.* We can also use a stochastic model of the uncertainty in the load, and model the vector f as a random variable with known mean and covariance:

$$\mathbf{E}\, f = f^{(0)}, \quad \mathbf{E}(f - f^{(0)})(f - f^{(0)})^T = \Sigma.$$

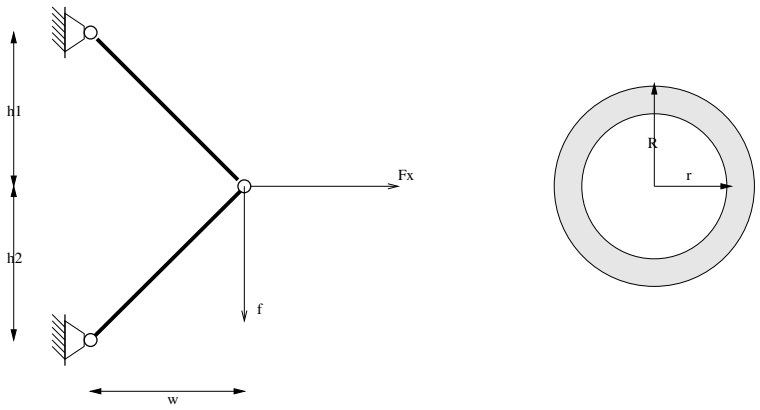
In this case we would be interested in minimizing the expected stored energy, *i.e.*,

$$\begin{array}{ll} \text{minimize} & \mathbf{E}\, \mathcal{E}(x, f^{(i)}) \\ \text{subject to} & l \leq x_i \leq u, \quad i = 1, \ldots, m \\ & W_{\text{tot}}(x) \leq W. \end{array}$$

Hint. If v is a random vector with zero mean and covariance Σ, then $\mathbf{E}\, v^T A v = \mathbf{E}\, \mathbf{tr}\, A v v^T = \mathbf{tr}\, A\, \mathbf{E}\, v v^T = \mathbf{tr}\, A\Sigma$.

(c) Formulate the four problems in (b) as semidefinite programming problems.

14.4 *A structural optimization problem* [Bazaraa, Sherali, and Shetty]. The figure shows a two-bar truss with height $2h$ and width w. The two bars are cylindrical tubes with inner radius r and outer radius R. We are interested in determining the values of r, R, w, and h that minimize the weight of the truss subject to a number of constraints. The structure should be strong enough for two loading scenarios. In the first scenario a vertical force F_1 is applied to the node; in the second scenario the force is horizontal with magnitude F_2.



The weight of the truss is proportional to the total volume of the bars, which is given by

$$2\pi(R^2 - r^2)\sqrt{w^2 + h^2}$$

This is the cost function in the design problem.

The first constraint is that the truss should be strong enough to carry the load F_1, *i.e.*, the stress caused by the external force F_1 must not exceed a given maximum value. To formulate this constraint, we first determine the forces in each bar when the structure is subjected to the vertical load F_1. From the force equilibrium and the geometry of the problem we can determine that the magnitudes of the forces in two bars are equal and given by

$$\frac{\sqrt{w^2 + h^2}}{2h} F_1.$$

The maximum force in each bar is equal to the cross-sectional area times the maximum allowable stress σ (which is a given constant). This gives us the first constraint:

$$\frac{\sqrt{w^2+h^2}}{2h}F_1 \leq \sigma\pi(R^2-r^2).$$

The second constraint is that the truss should be strong enough to carry the load F_2. When F_2 is applied, the magnitudes of the forces in two bars are again equal and given by

$$\frac{\sqrt{w^2+h^2}}{2w}F_2,$$

which gives us the second constraint:

$$\frac{\sqrt{w^2+h^2}}{2w}F_2 \leq \sigma\pi(R^2-r^2).$$

We also impose limits $w_{\min} \leq w \leq w_{\max}$ and $h_{\min} \leq h \leq h_{\max}$ on the width and the height of the structure, and limits $1.1r \leq R \leq R_{\max}$ on the outer radius.

In summary, we obtain the following problem:

$$\begin{array}{ll}
\text{minimize} & 2\pi(R^2-r^2)\sqrt{w^2+h^2} \\
\text{subject to} & \dfrac{\sqrt{w^2+h^2}}{2h}F_1 \leq \sigma\pi(R^2-r^2) \\
& \dfrac{\sqrt{w^2+h^2}}{2w}F_2 \leq \sigma\pi(R^2-r^2) \\
& w_{\min} \leq w \leq w_{\max} \\
& h_{\min} \leq h \leq h_{\max} \\
& 1.1r \leq R \leq R_{\max} \\
& R > 0, \quad r > 0, \quad w > 0, \quad h > 0.
\end{array}$$

The variables are R, r, w, h.

Formulate this as a geometric programming problem.

14.5 *Optimizing the inertia matrix of a 2D mass distribution.* An object has density $\rho(z)$ at the point $z = (x, y) \in \mathbf{R}^2$, over some region $\mathcal{R} \subset \mathbf{R}^2$. Its mass $m \in \mathbf{R}$ and center of gravity $c \in \mathbf{R}^2$ are given by

$$m = \int_{\mathcal{R}} \rho(z)\, dxdy, \qquad c = \frac{1}{m}\int_{\mathcal{R}} \rho(z) z\, dxdy,$$

and its inertia matrix $M \in \mathbf{R}^{2\times 2}$ is

$$M = \int_{\mathcal{R}} \rho(z)(z-c)(z-c)^T\, dxdy.$$

(You do not need to know the mechanics interpretation of M to solve this problem, but here it is, for those interested. Suppose we rotate the mass distribution around a line passing through the

center of gravity in the direction $q \in \mathbf{R}^2$ that lies in the plane where the mass distribution is, at angular rate ω. Then the total kinetic energy is $(\omega^2/2)q^T M q$.)

The goal is to choose the density ρ, subject to $0 \leq \rho(z) \leq \rho^{\max}$ for all $z \in \mathcal{R}$, and a fixed total mass $m = m^{\text{given}}$, in order to maximize $\lambda_{\min}(M)$.

To solve this problem numerically, we will discretize $\mathcal{R}$ into N pixels each of area a, with pixel i having constant density ρ_i and location (say, of its center) $z_i \in \mathbf{R}^2$. We will assume that the integrands above don't vary too much over the pixels, and from now on use instead the expressions

$$m = a\sum_{i=1}^{N} \rho_i, \quad c = \frac{a}{m}\sum_{i=1}^{N} \rho_i z_i, \quad M = a\sum_{i=1}^{N} \rho_i (z_i - c)(z_i - c)^T.$$

The problem below refers to these discretized expressions.

(a) Explain how to solve the problem using convex (or quasiconvex) optimization.

(b) Carry out your method on the problem instance with data in `inertia_dens_data.m`. This file includes code that plots a density. Give the optimal inertia matrix and its eigenvalues, and plot the optimal density.

14.6 *Truss loading analysis.* A truss (in 2D, for simplicity) consists of a set of n nodes, with positions $p^{(1)}, \ldots, p^{(n)} \in \mathbf{R}^2$, connected by a set of m bars with tensions $t_1, \ldots, t_m \in \mathbf{R}$ ($t_j < 0$ means bar j operates in compression).

Each bar puts a force on the two nodes which it connects. Suppose bar j connects nodes k and l. The tension in this bar applies a force

$$\frac{t_j}{\|p^{(l)} - p^{(k)}\|_2}(p^{(l)} - p^{(k)}) \in \mathbf{R}^2$$

to node k, and the opposite force to node l. In addition to the forces imparted by the bars, each node has an external force acting on it. We let $f^{(i)} \in \mathbf{R}^2$ be the external force acting on node i. For the truss to be in equilibrium, the total force on each node, *i.e.*, the sum of the external force and the forces applied by all of the bars that connect to it, must be zero. We refer to this constraint as force balance.

The tensions have given limits, $T_j^{\min} \leq t_j \leq T_j^{\max}$, with $T_j^{\min} \leq 0$ and $T_j^{\max} \geq 0$, for $j = 1, \ldots, m$. (For example, if bar j is a cable, then it can only apply a nonnegative tension, so $T_j^{\min} = 0$, and we interpret $T_j^{\max}$ as the maximum tension the cable can carry.)

The first p nodes, $i = 1, \ldots, p$, are *free*, while the remaining $n - p$ nodes, $i = p + 1, \ldots, n$, are *anchored* (*i.e.*, attached to a foundation). We will refer to the external forces on the free nodes as *load forces*, and external forces at the anchor nodes as *anchor forces*. The anchor forces are unconstrained. (More accurately, the foundations at these points are engineered to withstand any total force that the bars attached to it can deliver.) We will assume that the load forces are just dead weight, *i.e.*, have the form

$$f^{(i)} = \begin{bmatrix} 0 \\ -w_i \end{bmatrix}, \quad i = 1, \ldots, p,$$

where $w_i \geq 0$ is the weight supported at node i.

The set of weights $w \in \mathbf{R}^p_+$ is *supportable* if there exists a set of tensions $t \in \mathbf{R}^m$ and anchor forces $f^{(p+1)}, \ldots, f^{(n)}$ that, together with the given load forces, satisfy the force balance equations and respect the tension limits. (The tensions and anchor forces in a real truss will adjust themselves to have such values when the load forces are applied.) If there does not exist such a set of tensions and anchor forces, the set of load forces is said to be *unsupportable*. (In this case, a real truss will fail, or collapse, when the load forces are applied.)

Finally, we get to the questions.

(a) Explain how to find the maximum total weight, $\mathbf{1}^T w$, that is supportable by the truss.

(b) Explain how to find the minimum total weight that is not supportable by the truss. (Here we mean: Find the minimum value of $\mathbf{1}^T w$, for which $(1+\epsilon)w$ is not supportable, for all $\epsilon > 0$.)

(c) Carry out the methods of parts (a) and (b) on the data given in `truss_load_data.m`. Give the critical total weights from parts (a) and (b), as well as the individual weight vectors.

Notes.

- In parts (a) and (b), we don't need a fully formal mathematical justification; a clear argument or explanation of anything not obvious is fine.
- The force balance equations can be expressed in the compact and convenient form

$$At + \begin{bmatrix} f^{\text{load,x}} \\ f^{\text{load,y}} \\ f^{\text{anch}} \end{bmatrix} = 0,$$

where

$$\begin{aligned} f^{\text{load,x}} &= (f_1^{(1)}, \ldots, f_1^{(p)}) \in \mathbf{R}^p, \\ f^{\text{load,y}} &= (f_2^{(1)}, \ldots, f_2^{(p)}) \in \mathbf{R}^p, \\ f^{\text{anch}} &= (f_1^{(p+1)}, \ldots, f_1^{(n)}, f_2^{(p+1)}, \ldots, f_2^{(n)}) \in \mathbf{R}^{2(n-p)}, \end{aligned}$$

and $A \in \mathbf{R}^{2n \times m}$ is a matrix that can be found from the geometry data (truss topology and node positions). You may refer to A in your solutions to parts (a) and (b). For part (c), *we have very kindly provided the matrix A for you in the m-file*, to save you the time and trouble of working out the force balance equations from the geometry of the problem.

14.7 *Quickest take-off.* This problem concerns the braking and thrust profiles for an airplane during take-off. For simplicity we will use a discrete-time model. The position (down the runway) and the velocity in time period t are p_t and v_t, respectively, for $t = 0, 1, \ldots$. These satisfy $p_0 = 0$, $v_0 = 0$, and $p_{t+1} = p_t + hv_t$, $t = 0, 1, \ldots$, where $h > 0$ is the sampling time period. The velocity updates as

$$v_{t+1} = (1-\eta)v_t + h(f_t - b_t), \quad t = 0, 1, \ldots,$$

where $\eta \in (0, 1)$ is a friction or drag parameter, f_t is the engine thrust, and b_t is the braking force, at time period t. These must satisfy

$$0 \leq b_t \leq \min\{B^{\max}, f_t\}, \quad 0 \leq f_t \leq F^{\max}, \quad t = 0, 1, \ldots,$$

as well as a constraint on how fast the engine thrust can be changed,

$$|f_{t+1} - f_t| \leq S, \quad t = 0, 1, \ldots.$$

Here $B^{\max}$, $F^{\max}$, and S are given parameters. The initial thrust is $f_0 = 0$. The take-off time is $T^{\text{to}} = \min\{t \mid v_t \geq V^{\text{to}}\}$, where V^{to} is a given take-off velocity. The take-off position is $P^{\text{to}} = p_{T^{\text{to}}}$, the position of the aircraft at the take-off time. The length of the runway is $L > 0$, so we must have $P^{\text{to}} \leq L$.

(a) Explain how to find the thrust and braking profiles that minimize the take-off time T^{to}, respecting all constraints. Your solution can involve solving more than one convex problem, if necessary.

(b) Solve the quickest take-off problem with data

$$h = 1, \quad \eta = 0.05, \quad B^{\max} = 0.5, \quad F^{\max} = 4, \quad S = 0.8, \quad V^{\text{to}} = 40, \quad L = 300.$$

Plot p_t, v_t, f_t, and b_t versus t. Comment on what you see. Report the take-off time and take-off position for the profile you find.

14.8 *Optimal spacecraft landing.* We consider the problem of optimizing the thrust profile for a spacecraft to carry out a landing at a target position. The spacecraft dynamics are

$$m\ddot{p} = f - mge_3,$$

where $m > 0$ is the spacecraft mass, $p(t) \in \mathbf{R}^3$ is the spacecraft position, with 0 the target landing position and $p_3(t)$ representing height, $f(t) \in \mathbf{R}^3$ is the thrust force, and $g > 0$ is the gravitational acceleration. (For simplicity we assume that the spacecraft mass is constant. This is not always a good assumption, since the mass decreases with fuel use. We will also ignore any atmospheric friction.) We must have $p(T^{\text{td}}) = 0$ and $\dot{p}(T^{\text{td}}) = 0$, where T^{td} is the touchdown time. The spacecraft must remain in a region given by

$$p_3(t) \geq \alpha \|(p_1(t), p_2(t))\|_2,$$

where $\alpha > 0$ is a given minimum glide slope. The initial position $p(0)$ and velocity $\dot{p}(0)$ are given.

The thrust force $f(t)$ is obtained from a single rocket engine on the spacecraft, with a given maximum thrust; an attitude control system rotates the spacecraft to achieve any desired direction of thrust. The thrust force is therefore characterized by the constraint $\|f(t)\|_2 \leq F^{\max}$. The fuel use rate is proportional to the thrust force magnitude, so the total fuel use is

$$\int_0^{T^{\text{td}}} \gamma \|f(t)\|_2 \, dt,$$

where $\gamma > 0$ is the fuel consumption coefficient. The thrust force is discretized in time, *i.e.*, it is constant over consecutive time periods of length $h > 0$, with $f(t) = f_k$ for $t \in [(k-1)h, kh)$, for $k = 1, \ldots, K$, where $T^{\text{td}} = Kh$. Therefore we have

$$v_{k+1} = v_k + (h/m)f_k - hge_3, \quad p_{k+1} = p_k + (h/2)(v_k + v_{k+1}),$$

where p_k denotes $p((k-1)h)$, and v_k denotes $\dot{p}((k-1)h)$. We will work with this discrete-time model. For simplicity, we will impose the glide slope constraint only at the times $t = 0, h, 2h, \ldots, Kh$.

(a) *Minimum fuel descent.* Explain how to find the thrust profile $f_1, \ldots, f_K$ that minimizes fuel consumption, given the touchdown time $T^{\mathrm{td}} = Kh$ and discretization time h.

(b) *Minimum time descent.* Explain how to find the thrust profile that minimizes the touchdown time, *i.e.*, K, with h fixed and given. Your method can involve solving several convex optimization problems.

(c) Carry out the methods described in parts (a) and (b) above on the problem instance with data given in `spacecraft_landing_data.*`. Report the optimal total fuel consumption for part (a), and the minimum touchdown time for part (b). The data files also contain plotting code (commented out) to help you visualize your solution. Use the code to plot the spacecraft trajectory and thrust profiles you obtained for parts (a) and (b).

Hints.

- In Julia, the plot will come out rotated.

Remarks. If you'd like to see the ideas of this problem in action, watch these videos:

- `http://www.youtube.com/watch?v=2t15vP1PyoA`
- `https://www.youtube.com/watch?v=orUjSkc2pG0`
- `https://www.youtube.com/watch?v=1B6oiLNyKKI`
- `https://www.youtube.com/watch?v=ZCBE8ocOkAQ`

14.9 *Feedback gain optimization.* A system (such as an industrial plant) is characterized by $y = Gu + v$, where $y \in \mathbf{R}^n$ is the output, $u \in \mathbf{R}^n$ is the input, and $v \in \mathbf{R}^n$ is a disturbance signal. The matrix $G \in \mathbf{R}^{n \times n}$, which is known, is called the system input-output matrix. The input signal u is found using a linear feedback (control) policy: $u = Fy$, where $F \in \mathbf{R}^{n \times n}$ is the feedback (gain) matrix, which is what we need to determine. From the equations given above, we have

$$y = (I - GF)^{-1}v, \qquad u = F(I - GF)^{-1}v.$$

(You can simply assume that $I - GF$ will be invertible.)

The disturbance v is random, with $\mathbf{E}\, v = 0$, $\mathbf{E}\, vv^T = \sigma^2 I$, where σ is known. The objective is to minimize $\max_{i=1,\ldots,n} \mathbf{E}\, y_i^2$, the maximum mean square value of the output components, subject to the constraint that $\mathbf{E}\, u_i^2 \leq 1$, $i = 1, \ldots, n$, *i.e.*, each input component has a mean square value not exceeding one. The variable to be chosen is the matrix $F \in \mathbf{R}^{n \times n}$.

(a) Explain how to use convex (or quasi-convex) optimization to find an optimal feedback gain matrix. As usual, you must fully explain any change of variables or other transformations you carry out, and why your formulation solves the problem described above. A few comments:

- You can assume that matrices arising in your change of variables are invertible; you do not need to worry about the special cases when they are not.
- You can assume that G is invertible if you need to, but we will deduct a few points from these answers.

(b) Carry out your method for the problem instance with data

$$\sigma = 1, \qquad G = \begin{bmatrix} 0.3 & -0.1 & -0.9 \\ -0.6 & 0.3 & -0.3 \\ -0.3 & 0.6 & 0.2 \end{bmatrix}.$$

Give an optimal F, and the associated optimal objective value.

14.10 *Fuel use as function of distance and speed.* A vehicle uses fuel at a rate $f(s)$, which is a function of the vehicle speed s. We assume that $f : \mathbf{R} \to \mathbf{R}$ is a positive increasing convex function, with $\mathbf{dom}\, f = \mathbf{R}_+$. The physical units of s are m/s (meters per second), and the physical units of $f(s)$ are kg/s (kilograms per second).

(a) Let $g(d, t)$ be the total fuel used (in kg) when the vehicle moves a distance $d \geq 0$ (in meters) in time $t > 0$ (in seconds) at a constant speed. Show that g is convex.

(b) Let $h(d)$ be the minimum fuel used (in kg) to move a distance d (in m) at a constant speed s (in m/s). Show that h is convex.

14.11 *Minimum time speed profile along a road.* A vehicle of mass $m > 0$ moves along a road in $\mathbf{R}^3$, which is piecewise linear with given knot points $p_1, \ldots, p_{N+1} \in \mathbf{R}^3$, starting at p_1 and ending at p_{N+1}. We let $h_i = (p_i)_3$, the z-coordinate of the knot point; these are the heights of the knot points (above sea-level, say). For your convenience, these knot points are equidistant, *i.e.*, $\|p_{i+1} - p_i\|_2 = d$ for all i. (The points give an arc-length parametrization of the road.) We let $s_i > 0$ denote the (constant) vehicle speed as it moves along road segment i, from p_i to p_{i+1}, for $i = 1, \ldots, N$, and $s_{N+1} \geq 0$ denote the vehicle speed after it passes through knot point p_{N+1}. Our goal is to minimize the total time to traverse the road, which we denote T.

We let $f_i \geq 0$ denote the total fuel burnt while traversing the ith segment. This fuel burn is turned into an increase in vehicle energy given by ηf_i, where $\eta > 0$ is a constant that includes the engine efficiency and the energy content of the fuel. While traversing the ith road segment the vehicle is subject to a drag force, given by $C_D s_i^2$, where $C_D > 0$ is the coefficient of drag, which results in an energy loss $dC_D s_i^2$.

We derive equations that relate these quantities via energy balance:

$$\frac{1}{2} m s_{i+1}^2 + mgh_{i+1} = \frac{1}{2} m s_i^2 + mgh_i + \eta f_i - dC_D s_i^2, \quad i = 1, \ldots, N,$$

where $g = 9.8$ is the gravitational acceleration. The lefthand side is the total vehicle energy (kinetic plus potential) after it passes through knot point p_{i+1}; the righthand side is the total vehicle energy after it passes through knot point p_i, plus the energy gain from the fuel burn, minus the energy lost to drag. To set up the first vehicle speed s_1 requires an additional initial fuel burn f_0, with $\eta f_0 = \frac{1}{2} m s_1^2$.

Fuel is also used to power the on-board system of the vehicle. The total fuel used for this purpose is f_{ob}, where $\eta f_{\text{ob}} = TP$, where $P > 0$ is the (constant) power consumption of the on-board system. We have a fuel capacity constraint: $\sum_{i=0}^{N} f_i + f_{\text{ob}} \leq F$, where $F > 0$ is the total initial fuel.

The problem data are m, d, $h_1, \ldots, h_{N+1}$, η, C_D, P, and F. (You don't need the knot points p_i.)

(a) Explain how to find the fuel burn levels $f_0, \ldots, f_N$ that minimize the time T, subject to the constraints.

(b) Carry out the method described in part (a) for the problem instance with data given in `min_time_speed_data.m`. Give the optimal time $T^\star$, and compare it to the time T^{unif} achieved if the fuel for propulsion were burned uniformly, *i.e.*, $f_0 = \cdots = f_N$. For each of these cases, plot speed versus distance along the road, using the plotting code in the data file as a template.

14.12 *Least-cost road grading.* A road is to be built along a given path. We must choose the height of the roadbed (say, above sea level) along the path, minimizing the total cost of grading, subject to some constraints. The cost of grading (*i.e.*, moving earth to change the height of the roadbed from the existing elevation) depends on the difference in height between the roadbed and the existing elevation. When the roadbed is below the existing elevation it is called a *cut*; when it is above it is called a *fill*. Each of these incurs engineering costs; for example, fill is created in a series of *lifts*, each of which involves dumping just a few inches of soil and then compacting it. Deeper cuts and higher fills require more work to be done on the road shoulders, and possibly, the addition of reinforced concrete structures to stabilize the earthwork. This explains why the marginal cost of cuts and fills increases with their depth/height.

We will work with a discrete model, specifying the road height as h_i, $i = 1, \ldots, n$, at points equally spaced a distance d from each other along the given path. These are the variables to be chosen. (The heights $h_1, \ldots, h_n$ are called a *grading plan*.) We are given e_i, $i = 1, \ldots, n$, the existing elevation, at the points. The grading cost is

$$C = \sum_{i=1}^{n} \left(\phi^{\text{fill}}((h_i - e_i)_+) + \phi^{\text{cut}}((e_i - h_i)_+) \right),$$

where ϕ^{fill} and ϕ^{cut} are the fill and cut cost functions, respectively, and $(a)_+ = \max\{a, 0\}$. The fill and cut functions are increasing and convex. The goal is to minimize the grading cost C.

The road height is constrained by given limits on the first, second, and third derivatives:

$$\begin{aligned} |h_{i+1} - h_i|/d &\leq D^{(1)}, \quad i = 1, \ldots, n-1 \\ |h_{i+1} - 2h_i + h_{i-1}|/d^2 &\leq D^{(2)}, \quad i = 2, \ldots, n-1 \\ |h_{i+1} - 3h_i + 3h_{i-1} - h_{i-2}|/d^3 &\leq D^{(3)}, \quad i = 3, \ldots, n-1, \end{aligned}$$

where $D^{(1)}$ is the maximum allowable road slope, $D^{(2)}$ is the maximum allowable curvature, and $D^{(3)}$ is the maximum allowable third derivative.

(a) Explain how to find the optimal grading plan.

(b) Find the optimal grading plan for the problem with data given in `road_grading_data.m`, and fill and cut cost functions

$$\phi^{\text{fill}}(u) = 2(u)_+^2 + 30(u)_+, \qquad \phi^{\text{cut}} = 12(u)_+^2 + (u)_+.$$

Plot $h_i - e_i$ for the optimal grading plan and report the associated cost.

(c) Suppose the optimal grading problem with $n = 1000$ can be solved on a particular machine (say, with one, or just a few, cores) in around one second. Assuming the author of the software took EE364a, about how long will it take to solve the optimal grading problem with $n = 10000$? Give a very brief justification of your answer, no more than a few sentences.

14.13 *Lightest structure that resists a set of loads.* We consider a mechanical structure in 2D (for simplicity) which consists of a set of m nodes, with known positions $p_1, \ldots, p_m \in \mathbf{R}^2$, connected by a set of n bars (also called struts or elements), with cross-sectional areas $a_1, \ldots, a_n \in \mathbf{R}_+$, and internal tensions $t_1, \ldots, t_n \in \mathbf{R}$.

Bar j is connected between nodes r_j and s_j. (The indices $r_1, \ldots, r_n$ and $s_1, \ldots, s_n$ give the structure topology.) The length of bar j is $L_j = \|p_{r_j} - p_{s_j}\|_2$, and the total volume of the bars is $V = \sum_{j=1}^n a_j L_j$. (The total weight is proportional to the total volume.)

Bar j applies a force $(t_j/L_j)(p_{r_j} - p_{s_j}) \in \mathbf{R}^2$ to node s_j and the negative of this force to node r_j. Thus, positive tension in a bar pulls its two adjacent nodes towards each other; negative tension (also called compression) pushes them apart. The ratio of the tension in a bar to its cross-sectional area is limited by its yield strength, which is symmetric in tension and compression: $|t_j| \leq \sigma a_j$, where $\sigma > 0$ is a known constant that depends on the material.

The nodes are divided into two groups: free and fixed. We will take nodes $1, \ldots, k$ to be free, and nodes $k+1, \ldots, m$ to be fixed. Roughly speaking, the fixed nodes are firmly attached to the ground, or a rigid structure connected to the ground; the free ones are not.

A *loading* consists of a set of external forces, $f_1, \ldots, f_k \in \mathbf{R}^2$ applied to the free nodes. Each free node must be in equilibrium, which means that the sum of the forces applied to it by the bars and the external force is zero. The structure can *resist* a loading (without collapsing) if there exists a set of bar tensions that satisfy the tension bounds and force equilibrium constraints. (For those with knowledge of statics, these conditions correspond to a structure made entirely with pin joints.)

Finally, we get to the problem. You are given a set of M loadings, *i.e.*, $f_1^{(i)}, \ldots, f_k^{(i)} \in \mathbf{R}^2$, $i = 1, \ldots, M$. The goal is to find the bar cross-sectional areas that minimize the structure volume V while resisting all of the given loadings. (Thus, you are to find *one* set of bar cross-sectional areas, and M sets of tensions.) Using the problem data provided in `lightest_struct_data.m`, report $V^\star$ and V^{unif}, the smallest feasible structure volume when all bars have the same cross-sectional area. The node positions are given as a $2 \times m$ matrix `P`, and the loadings as a $2 \times k \times M$ array `F`. Use the code included in the data file to visualize the structure with the bar cross-sectional areas that you find, and provide the plot in your solution.

Hint. You might find the graph incidence matrix $A \in \mathbf{R}^{m \times n}$ useful. It is defined as

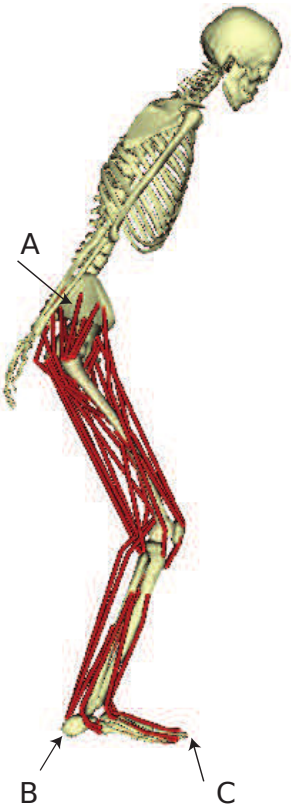
$$A_{ij} = \begin{cases} +1 & i = r_j \\ -1 & i = s_j \\ 0 & \text{otherwise.} \end{cases}$$

Remark. You could reasonably ask, 'Does a mechanical structure really solve a convex optimization problem to determine whether it should collapse?'. It sounds odd, but the answer is, yes it does.

14.14 *Maintaining static balance.* In this problem we study a human's ability to maintain balance against an applied external force. We will use a planar (two-dimensional) model to characterize the set of push forces a human can sustain before he or she is unable to maintain balance. We model the human as a linkage of 4 body segments, which we consider to be rigid bodies: the foot, lower leg, upper leg, and pelvis (into which we lump the upper body). The pose is given by the joint angles, but this won't matter in this problem, since we consider a fixed pose. A set of 40 muscles act on the body segments; each of these develops a (scalar) tension t_i that satisfies $0 \leq t_i \leq T_i^{\max}$, where $T_i^{\max}$ is the maximum possible tension for muscle i. (The maximum muscle tensions depend on the pose, and the person, but here they are known constants.) An external pushing force $f^{\text{push}} \in \mathbf{R}^2$ acts on the pelvis. Two (ground contact) forces act on the foot: $f^{\text{heel}} \in \mathbf{R}^2$ and $f^{\text{toe}} \in \mathbf{R}^2$. (These are shown at right.) These must satisfy

$$|f_1^{\text{heel}}| \leq \mu f_2^{\text{heel}}, \quad |f_1^{\text{toe}}| \leq \mu f_2^{\text{toe}},$$

where $\mu > 0$ is the coefficient of friction of the ground. There are also joint forces that act at the joints between the body segments, and gravity forces for each body segment, but we won't need them explicitly in this problem.



To maintain balance, the net force and torque on each each body segment must be satisfied. These equations can be written out from the geometry of the body (*e.g.*, attachment points for the muscles) and the pose. They can be reduced to a set of 6 linear equations:

$$A^{\text{musc}} t + A^{\text{toe}} f^{\text{toe}} + A^{\text{heel}} f^{\text{heel}} + A^{\text{push}} f^{\text{push}} = b,$$

where $t \in \mathbf{R}^{40}$ is the vector of muscle tensions, and A^{musc}, A^{toe}, A^{heel}, and A^{push} are known matrices and $b \in \mathbf{R}^6$ is a known vector. These data depend on the pose, body weight and dimensions, and muscle lines of action. Fortunately for you, our biomechanics expert Apoorva has worked them out; you will find them in `static_balance_data.*` (along with $T^{\max}$ and μ).

We say that the push force f^{push} can be *resisted* if there exist muscle tensions and ground contact forces that satisfy the constraints above. (This raises a philosophical question: Does a person solve an optimization to decide whether he or she should lose their balance? In any case, this approach makes good predictions.)

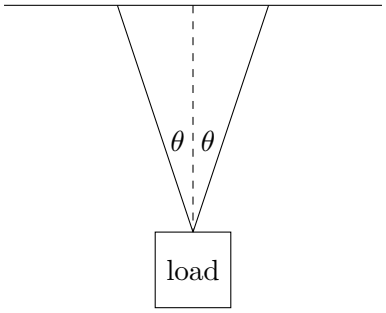
Find $\mathcal{F}^{\text{res}} \subset \mathbf{R}^2$, the set of push forces that can be resisted. Plot it as a shaded region.

Hints. Show that $\mathcal{F}^{\text{res}}$ is a convex set. For the given data, $0 \in \mathcal{F}^{\text{res}}$. Then for $\theta = 1^\circ, 2^\circ, \ldots, 360^\circ$, determine the maximum push force, applied in the direction θ, that can be resisted. To make a filled region on a plot, you can use the command `fill()` in Matlab. For Python and Julia, `fill()` is also available through PyPlot. In Julia, make sure to use the ECOS solver with `solver = ECOSSolver(verbose=false)`.

Remark. A person can resist a much larger force applied to the hip than you might think.

14.15 *Minimum time maneuver for a crane.* A crane manipulates a load with mass $m > 0$ in two

dimensions using two cables attached to the load. The cables maintain angles $\pm\theta$ with respect to vertical, as shown below.



The (scalar) tensions T^{left} and T^{right} in the two cables are independently controllable, from 0 up to a given maximum tension $T^{\max}$. The total force on the load is

$$F = T^{\text{left}} \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix} + T^{\text{right}} \begin{bmatrix} \sin\theta \\ \cos\theta \end{bmatrix} + mg,$$

where $g = (0, -9.8)$ is the acceleration due to gravity. The acceleration of the load is then F/m.

We approximate the motion of the load using

$$p_{i+1} = p_i + hv_i, \quad v_{i+1} = v_i + (h/m)F_i, \quad i = 1, 2, \ldots,$$

where $p_i \in \mathbf{R}^2$ is the position of the load, $v_i \in \mathbf{R}^2$ is the velocity of the load, and $F_i \in \mathbf{R}^2$ is the force on the load, at time $t = ih$. Here $h > 0$ is a small (given) time step.

The goal is to move the load, which is initially at rest at position p^{init} to the position p^{des}, also at rest, in minimum time. In other words, we seek the smallest k for which

$$p_1 = p^{\text{init}}, \quad p_k = p^{\text{des}}, \quad v_1 = v_k = (0, 0)$$

is possible, subject to the constraints described above.

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

(b) Carry out the method of part (a) for the problem instance with

$$m = 0.1, \quad \theta = 15^\circ, \quad T^{\max} = 2, \quad p^{\text{init}} = (0, 0), \quad p^{\text{des}} = (10, 2),$$

with time step $h = 0.1$. Report the minimum time $k^\star$. Plot the tensions versus time, and the load trajectory, *i.e.*, the points $p_1, \ldots, p_k$ in $\mathbf{R}^2$. Does the load move along the line segment between p^{init} and p^{des} (*i.e.*, the shortest path from p^{init} and p^{des})? Comment briefly.

14.16 *Thermodynamic potentials.* We consider a mixture of k chemical species. The *internal energy* of the mixture is

$$U(S, V, N_1, \ldots, N_k),$$

where S is the entropy of the mixture, V is the volume occupied by the mixture, and N_i is the quantity (in moles) of chemical species i. We assume the function U is convex. (Real internal

energy functions satisfy this and other interesting properties, but we won't need any others for this problem.) The *enthalpy* H, the *Helmoltz free energy* A, and the *Gibbs free energy* G are defined as

$$\begin{aligned} H(S,P,N_1,\ldots,N_k) &= \inf_V \left(U(S,V,N_1,\ldots,N_k) - PV\right), \\ A(T,V,N_1,\ldots,N_k) &= \inf_S \left(U(S,V,N_1,\ldots,N_k) + TS\right), \\ G(T,P,N_1,\ldots,N_k) &= \inf_{S,V} \left(U(S,V,N_1,\ldots,N_k) + TS - PV\right). \end{aligned}$$

The variables T and P can be interpreted physically as the temperature and pressure of the mixture. These four functions are called *thermodynamic potentials*. We refer to the arguments S, V, and $N_1,\ldots N_k$ as the extensive variables, and the arguments T and P as the intensive variables.

(a) Show that H, A, and G are convex in the extensive variables, when the intensive variables are fixed.

(b) Show that H, A, and G are concave in the intensive variables, when the extensive variables are fixed.

(c) We consider a simple reaction involving three species,

$$2[\text{species 1}] \rightleftharpoons [\text{species 2}] + [\text{species 3}],$$

carried out at temperature T_{react} and volume V_{react}. The Helmholtz free energy of the mixture is

$$A(T,V,N_1,N_2,N_3) = T\sum_{j=1}^{3} N_j(s_{0,j} - Rc_j) + TR\sum_{j=1}^{3} N_j \log\left(N_j\left(\frac{V_0}{V}\right)\left(\frac{T_0}{T}\right)^{c_j}\right),$$

where R, V_0, T_0, $s_{0,j}$, and c_j, for $j = 1,\ldots,k$, are known, positive constants. The equilibrium molar quantities $N_1^\star$, $N_2^\star$, and $N_3^\star$ of the three species are those that minimize $A(T_{\text{react}}, V_{\text{react}}, N_1, N_2, N_3)$ subject to the stoichiometry constraints

$$N_1 = N_{1,\text{init}} - 2z, \quad N_2 = N_{2,\text{init}} + z, \quad N_3 = N_{3,\text{init}} + z,$$

where $N_{j,\text{init}}$ is the initial quantity of species j, and the variable z gives the amount of the reaction that has proceeded. For the values of T_{react}, V_{react}, R, V_0, T_0, $s_{0,j}$, and c_j given in `thermo_potentials_data.*`, report the equilibrium molar quantities $N_1^\star$, $N_2^\star$, and $N_3^\star$.

Note: Julia users might want the ECOS solver. Include `using ECOS`, and solve by using `solve!(prob, ECOSSolver())`.

14.17 *Elastic stored energy in a spring.* A spring is a mechanical device that exerts a force F that depends on its extension x: $F = \phi(x)$, where $\phi : \mathbf{R} \to \mathbf{R}$. The domain $\mathbf{dom}\,\phi$ is an interval $[x^{\min}, x^{\max}]$ containing 0, where $x^{\min}$ ($x^{\max}$) is the minimum (maximum) possible extension of the spring. When $x > 0$, the spring is said to be extended, and when $x < 0$, it is said to be in compression. The force exerted by the spring must be *restoring*, which means that $F \geq 0$ when $x \geq 0$, and $F \leq 0$ when $x \leq 0$. (Our sign convention is that a positive force F opposes a positive extension x.) This implies that $F = 0$ when $x = 0$, *i.e.*, zero force is developed when the spring is not extended or compressed.

The simplest spring is a Hooke (linear) spring, with $\phi(x) = Kx$, where $K > 0$ is the *spring constant*. (The constant $1/K$ is called the spring *compliance*.)

A spring is called *monotonic* if the function ϕ is nondecreasing, *i.e.*, larger extension leads to a stronger restoring force. Many, but not all, springs are monotonic. A classic example is a compound bow, which has a force that first increases with x, and then decreases to a small value at the extension x where it is fully drawn. (This decrease in force from the maximum is called the *let off* of the bow.)

The elastic stored energy in the spring is

$$E(x) = \int_0^x \phi(x)\, dx,$$

with domain $[x^{\min}, x^{\max}]$.

Show that E is quasi-convex. Show that E is convex if and only if the spring is monotonic. You may assume ϕ is differentiable.

15 Graphs and networks

15.1 A *hypergraph* with nodes $1, \ldots, m$ is a set of nonempty subsets of $\{1, 2, \ldots, m\}$, called *edges*. An ordinary graph is a special case in which the edges contain no more than two nodes.

We consider a hypergraph with m nodes and assume coordinate vectors $x_j \in \mathbf{R}^p$, $j = 1, \ldots, m$, are associated with the nodes. Some nodes are fixed and their coordinate vectors x_j are given. The other nodes are free, and their coordinate vectors will be the optimization variables in the problem. The objective is to place the free nodes in such a way that some measure of the physical size of the nets is small.

As an example application, we can think of the nodes as modules in an integrated circuit, placed at positions $x_j \in \mathbf{R}^2$. Every edge is an interconnect network that carries a signal from one module to one or more other modules.

To define a measure of the size of a net, we store the vectors x_j as columns of a matrix $X \in \mathbf{R}^{p\times m}$. For each edge S in the hypergraph, we use X_S to denote the $p \times |S|$ submatrix of X with the columns associated with the nodes of S. We define

$$f_S(X) = \inf_y \|X_S - y\mathbf{1}^T\|. \tag{48}$$

as the *size* of the edge S, where $\|\cdot\|$ is a matrix norm, and $\mathbf{1}$ is a vector of ones of length $|S|$.

(a) Show that the optimization problem

$$\text{minimize} \quad \textstyle\sum_{\text{edges } S} f_S(X)$$

is convex in the free node coordinates x_j.

(b) The size $f_S(X)$ of a net S obviously depends on the norm used in the definition (48). We consider five norms.

- *Frobenius norm:*
$$\|X_s - y\mathbf{1}^T\|_F = \left(\sum_{j\in S}\sum_{i=1}^{p}(x_{ij} - y_i)^2\right)^{1/2}.$$
- *Maximum Euclidean column norm:*
$$\|X_S - y\mathbf{1}^T\|_{2,1} = \max_{j\in S}\left(\sum_{i=1}^{p}(x_{ij} - y_i)^2\right)^{1/2}.$$
- *Maximum column sum norm:*
$$\|X_S - y\mathbf{1}^T\|_{1,1} = \max_{j\in S}\sum_{i=1}^{p}|x_{ij} - y_i|.$$
- *Sum of absolute values norm:*
$$\|X_s - y\mathbf{1}^T\|_{\text{sav}} = \sum_{j\in S}\sum_{i=1}^{p}|x_{ij} - y_i|$$

- *Sum-row-max norm:*

$$\|X_s - y\mathbf{1}^T\|_{\text{srm}} = \sum_{i=1}^{p} \max_{j\in S} |x_{ij} - y_i|$$

For which of these norms does f_S have the following interpretations?

(i) $f_S(X)$ is the radius of the smallest Euclidean ball that contains the nodes of S.

(ii) $f_S(X)$ is (proportional to) the perimeter of the smallest rectangle that contains the nodes of S:

$$f_S(X) = \frac{1}{4}\sum_{i=1}^{p}(\max_{j\in S} x_{ij} - \min_{j\in S} x_{ij}).$$

(iii) $f_S(X)$ is the squareroot of the sum of the squares of the Euclidean distances to the mean of the coordinates of the nodes in S:

$$f_S(X) = \left(\sum_{j\in S} \|x_j - \bar{x}\|_2^2\right)^{1/2} \quad \text{where} \quad \bar{x}_i = \frac{1}{|S|}\sum_{k\in S} x_{ik}, \quad i = 1, \ldots, p.$$

(iv) $f_S(X)$ is the sum of the ℓ_1-distances to the (coordinate-wise) median of the coordinates of the nodes in S:

$$f_S(X) = \sum_{j\in S} \|x_j - \hat{x}\|_1 \quad \text{where} \quad \hat{x}_i = \text{median}(\{x_{ik} \mid k \in S\}), \quad i = 1, \ldots, p.$$

15.2 Let $W \in \mathbf{S}^n$ be a symmetric matrix with nonnegative elements w_{ij} and zero diagonal. We can interpret W as the representation of a weighted undirected graph with n nodes. If $w_{ij} = w_{ji} > 0$, there is an edge between nodes i and j, with weight w_{ij}. If $w_{ij} = w_{ji} = 0$ then nodes i and j are not connected. The *Laplacian* of the weighted graph is defined as

$$L(W) = -W + \mathbf{diag}(W\mathbf{1}).$$

This is a symmetric matrix with elements

$$L_{ij}(W) = \begin{cases} \sum_{k=1}^n w_{ik} & i = j \\ -w_{ij} & i \neq j. \end{cases}$$

The Laplacian has the useful property that

$$y^T L(W) y = \sum_{i\leq j} w_{ij}(y_i - y_j)^2$$

for all vectors $y \in \mathbf{R}^n$.

(a) Show that the function $f : \mathbf{S}^n \to \mathbf{R}$,

$$f(W) = \inf_{\mathbf{1}^T x = 0} n\lambda_{\max}\left(L(W) + \mathbf{diag}(x)\right),$$

is convex.

(b) Give a simple argument why $f(W)$ is an upper bound on the optimal value of the combinatorial optimization problem

$$\begin{array}{ll} \text{maximize} & y^T L(W) y \\ \text{subject to} & y_i \in \{-1, 1\}, \quad i = 1, \ldots, n. \end{array}$$

This problem is known as the *max-cut* problem, for the following reason. Every vector y with components ± 1 can be interpreted as a partition of the nodes of the graph in a set $S = \{i \mid y_i = 1\}$ and a set $T = \{i \mid y_i = -1\}$. Such a partition is called a *cut* of the graph. The objective function in the max-cut problem is

$$y^T L(W) y = \sum_{i \leq j} w_{ij} (y_i - y_j)^2.$$

If y is ± 1-vector corresponding to a partition in sets S and T, then $y^T L(W) y$ equals four times the sum of the weights of the edges that join a point in S to a point in T. This is called the weight of the cut defined by y. The solution of the max-cut problem is the cut with the maximum weight.

(c) The function f defined in part 1 can be evaluated, for a given W, by solving the optimization problem

$$\begin{array}{ll} \text{minimize} & n\lambda_{\max}(L(W) + \mathbf{diag}(x)) \\ \text{subject to} & \mathbf{1}^T x = 0, \end{array}$$

with variable $x \in \mathbf{R}^n$. Express this problem as an SDP.

(d) Derive an alternative expression for $f(W)$, by taking the dual of the SDP in part 3. Show that the dual SDP is equivalent to the following problem:

$$\begin{array}{ll} \text{maximize} & \displaystyle\sum_{i \leq j} w_{ij} \|p_i - p_j\|_2^2 \\ \text{subject} & \|p_i\|_2 = 1, \quad i = 1, \ldots, n, \end{array}$$

with variables $p_i \in \mathbf{R}^n$, $i = 1, \ldots, n$. In this problem we place n points p_i on the unit sphere in $\mathbf{R}^n$ in such a way that the weighted sum of their squared pair-wise distances is maximized.

15.3 *Utility versus latency trade-off in a network.* We consider a network with m edges, labeled $1, \ldots, m$, and n flows, labeled $1, \ldots, n$. Each flow has an associated nonnegative flow rate f_j; each edge or link has an associated positive capacity c_i. Each flow passes over a fixed set of links (its route); the total traffic t_i on link i is the sum of the flow rates over all flows that pass through link i. The flow routes are described by a routing matrix $R \in \mathbf{R}^{m \times n}$, defined as

$$R_{ij} = \begin{cases} 1 & \text{flow } j \text{ passes through link } i \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the vector of link traffic, $t \in \mathbf{R}^m$, is given by $t = Rf$. The link capacity constraint can be expressed as $Rf \preceq c$. The (logarithmic) network utility is defined as $U(f) = \sum_{j=1}^n \log f_j$.

The (average queuing) delay on link i is given by

$$d_i = \frac{1}{c_i - t_i}$$

(multiplied by a constant, that doesn't matter to us). We take $d_i = \infty$ for $t_i = c_i$. The delay or latency for flow j, denoted l_j, is the sum of the link delays over all links that flow j passes through. We define the maximum flow latency as

$$L = \max\{l_1, \ldots, l_n\}.$$

We are given R and c; we are to choose f.

(a) How would you find the flow rates that maximize the utility U, ignoring flow latency? (In particular, we allow $L = \infty$.) We'll refer to this maximum achievable utility as $U^{\max}$.

(b) How would you find the flow rates that minimize the maximum flow latency L, ignoring utility? (In particular, we allow $U = -\infty$.) We'll refer to this minimum achievable latency as $L^{\min}$.

(c) Explain how to find the optimal trade-off between utility U (which we want to maximize) and latency L (which we want to minimize).

(d) Find $U^{\max}$, $L^{\min}$, and plot the optimal trade-off of utility versus latency for the network with data given in `net_util_data.m`, showing $L^{\min}$ and $U^{\max}$ on the same plot. Your plot should cover the range from $L = 1.1L^{\min}$ to $L = 11L^{\min}$. Plot U vertically, on a linear scale, and L horizontally, using a log scale.

Note. For parts (a), (b), and (c), your answer can involve solving one or more convex optimization problems. But if there is a simpler solution, you should say so.

15.4 *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 $P^{\max} = P_n$).

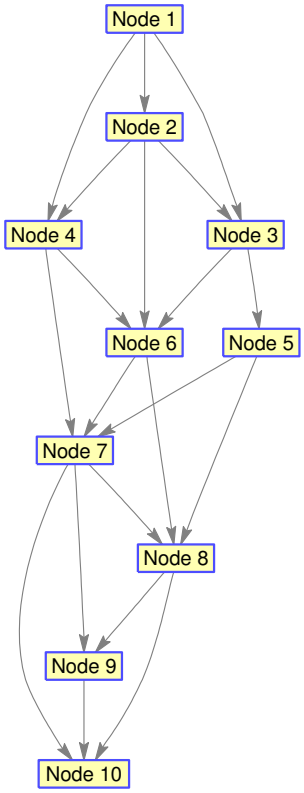
(b) Carry out your method on the problem instance given in `interdict_alloc_data.m`. The data file contains the data a, $x^{\max}$, B, and the graph incidence matrix $A \in \mathbf{R}^{n\times m}$, where

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

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.)



15.5 *Network sizing.* We consider a network with n directed arcs. The flow through arc k is denoted x_k and can be positive, negative, or zero. The flow vector x must satisfy the network constraint $Ax = b$ where A is the node-arc incidence matrix and b is the external flow supplied to the nodes.

Each arc has a positive capacity or width y_k. The quantity $|x_k|/y_k$ is the flow density in arc k. The cost of the flow in arc k depends on the flow density and the width of the arc, and is given by $y_k\phi_k(|x_k|/y_k)$, where ϕ_k is convex and nondecreasing on $\mathbf{R}_+$.

(a) Define $f(y, b)$ as the optimal value of the network flow optimization problem

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{k=1}^{n} y_k\,\phi_k(|x_k|/y_k) \\ \text{subject to} & Ax = b \end{array}$$

with variable x, for given values of the arc widths $y \succ 0$ and external flows b. Is f a convex function (jointly in y, b)? Carefully explain your answer.

(b) Suppose b is a discrete random vector with possible values $b^{(1)}, \ldots, b^{(m)}$. The probability that $b = b^{(j)}$ is π_j. Consider the problem of sizing the network (selecting the arc widths y_k) so that

the expected cost is minimized:

$$\text{minimize} \quad g(y) + \mathbf{E}\, f(y, b). \tag{49}$$

The variable is y. Here g is a convex function, representing the installation cost, and $\mathbf{E}\, f(y, b)$ is the expected optimal network flow cost

$$\mathbf{E}\, f(y, b) = \sum_{j=1}^{m} \pi_j f(y, b^{(j)}),$$

where f is the function defined in part 1. Is (49) a convex optimization problem?

15.6 *Maximizing algebraic connectivity of a graph.* Let $G = (V, E)$ be a weighted undirected graph with $n = |V|$ nodes, $m = |E|$ edges, and weights $w_1, \ldots, w_m \in \mathbf{R}_+$ on the edges. If edge k connects nodes i and j, then define $a_k \in \mathbf{R}^n$ as $(a_k)_i = 1$, $(a_k)_j = -1$, with other entries zero. The *weighted Laplacian* (matrix) of the graph is defined as

$$L = \sum_{k=1}^{m} w_k a_k a_k^T = A\, \mathbf{diag}(w) A^T,$$

where $A = [a_1 \cdots a_m] \in \mathbf{R}^{n \times m}$ is the *incidence matrix* of the graph. Nonnegativity of the weights implies $L \succeq 0$.

Denote the eigenvalues of the Laplacian L as

$$\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n,$$

which are functions of w. The minimum eigenvalue λ_1 is always zero, while the second smallest eigenvalue λ_2 is called the *algebraic connectivity* of G and is a measure of the connectedness of a graph: The larger λ_2 is, the better connected the graph is. It is often used, for example, in analyzing the robustness of computer networks.

Though not relevant for the rest of the problem, we mention a few other examples of how the algebraic connectivity can be used. These results, which relate graph-theoretic properties of G to properties of the spectrum of L, belong to a field called *spectral graph theory*. For example, $\lambda_2 > 0$ if and only if the graph is connected. The eigenvector v_2 associated with λ_2 is often called the *Fiedler vector* and is widely used in a graph partitioning technique called *spectral partitioning*, which assigns nodes to one of two groups based on the sign of the relevant component in v_2. Finally, λ_2 is also closely related to a quantity called the *isoperimetric number* or *Cheeger constant* of G, which measures the degree to which a graph has a bottleneck.

The problem is to choose the edge weights $w \in \mathbf{R}_+^m$, subject to some linear inequalities (and the nonnegativity constraint) so as to maximize the algebraic connectivity:

$$\begin{array}{ll} \text{maximize} & \lambda_2 \\ \text{subject to} & w \succeq 0, \quad Fw \preceq g, \end{array}$$

with variable $w \in \mathbf{R}^m$. The problem data are A (which gives the graph topology), and F and g (which describe the constraints on the weights).

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

(b) *Numerical example.* Solve the problem instance given in `max_alg_conn_data.m`, which uses $F = \mathbf{1}^T$ and $g = 1$ (so the problem is to allocate a total weight of 1 to the edges of the graph). Compare the algebraic connectivity for the graph obtained with the optimal weights $w^\star$ to the one obtained with $w^{\text{unif}} = (1/m)\mathbf{1}$ (*i.e.*, a uniform allocation of weight to the edges).

Use the function `plotgraph(A,xy,w)` to visualize the weighted graphs, with weight vectors $w^\star$ and w^{unif}. You will find that the optimal weight vector $v^\star$ has some zero entries (which due to the finite precision of the solver, will appear as small weight values); you may want to round small values (say, those under 10^{-4}) of $w^\star$ to exactly zero. Use the `gplot` function to visualize the original (given) graph, and the subgraph associated with nonzero weights in $w^\star$.

Briefly comment on the following (incorrect) intuition: "The more edges a graph has, the more connected it is, so the optimal weight assignment should make use of all available edges."

15.7 *Graph isomorphism via linear programming.* An (undirected) graph with n vertices can be described by its adjacency matrix $A \in \mathbf{S}^n$, given by

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

Two (undirected) graphs are *isomorphic* if we can permute the vertices of one so it is the same as the other (*i.e.*, the same pairs of vertices are connected by edges). If we describe them by their adjacency matrices A and B, isomorphism is equivalent to the existence of a permutation matrix $P \in \mathbf{R}^{n\times n}$ such that $PAP^T = B$. (Recall that a matrix P is a permutation matrix if each row and column has exactly one entry 1, and all other entries 0.) Determining if two graphs are isomorphic, and if so, finding a suitable permutation matrix P, is called the *graph isomorphism problem*.

Remarks (not needed to solve the problem). It is not currently known if the graph isomorphism problem is NP-complete or solvable in polynomial time. The graph isomorphism problem comes up in several applications, such as determining if two descriptions of a molecule are the same, or whether the physical layout of an electronic circuit correctly reflects the given circuit schematic diagram.

(a) Find a set of linear equalities and inequalities on $P \in \mathbf{R}^{n\times n}$, that together with the Boolean constraint $P_{ij} \in \{0, 1\}$, are necessary and sufficient for P to be a permutation matrix satisfying $PAP^T = B$. Thus, the graph isomorphism problem is equivalent to a Boolean feasibility LP.

(b) Consider the relaxed version of the Boolean feasibility LP found in part (a), *i.e.*, the LP that results when the constraints $P_{ij} \in \{0, 1\}$ are replaced with $P_{ij} \in [0, 1]$. When this LP is infeasible, we can be sure that the two graphs are not isomorphic. If a solution of the LP is found that satisfies $P_{ij} \in \{0, 1\}$, then the graphs are isomporphic and we have solved the graph isomorphism problem. This of course does not always happen, even if the graphs are isomorphic.

A standard trick to encourage the entries of P to take on the values 0 and 1 is to add a random linear objective to the relaxed feasibility LP. (This doesn't change whether the problem is feasible or not.) In other words, we minimize $\sum_{i,j} W_{ij}P_{ij}$, where W_{ij} are chosen randomly (say, from $\mathcal{N}(0, 1)$). (This can be repeated with different choices of W.)

Carry out this scheme for the two isomorphic graphs with adjacency matrices A and B given in `graph_isomorphism_data.*` to find a permutation matrix P that satisfies $PAP^T = B$. Report the permutation vector, given by the matrix-vector product Pv, where $v = (1, 2, \ldots, n)$. Verify that all the required conditions on P hold. To check that the entries of the solution of the LP are (close to) $\{0, 1\}$, report $\max_{i,j} P_{ij}(1 - P_{ij})$. And yes, you might have to try more than one instance of the randomized method described above before you find a permutation that establishes isomorphism of the two graphs.

16 Energy and power

16.1 *Power flow optimization with 'N − 1' reliability constraint.* We model a network of power lines as a graph with n nodes and m edges. The power flow along line j is denoted p_j, which can be positive, which means power flows along the line in the direction of the edge, or negative, which means power flows along the line in the direction opposite the edge. (In other words, edge orientation is only used to determine the direction in which power flow is considered positive.) Each edge can support power flow in either direction, up to a given maximum capacity $P_j^{\max}$, *i.e.*, we have $|p_j| \leq P_j^{\max}$.

Generators are attached to the first k nodes. Generator i provides power g_i to the network. These must satisfy $0 \leq g_i \leq G_i^{\max}$, where $G_i^{\max}$ is a given maximum power available from generator i. The power generation costs are $c_i > 0$, which are given; the total cost of power generation is $c^T g$.

Electrical loads are connected to the nodes $k+1, \ldots, n$. We let $d_i \geq 0$ denote the demand at node $k+i$, for $i = 1, \ldots, n-k$. We will consider these loads as given. In this simple model we will neglect all power losses on lines or at nodes. Therefore, power must balance at each node: the total power flowing into the node must equal the sum of the power flowing out of the node. This power balance constraint can be expressed as

$$Ap = \begin{bmatrix} -g \\ d \end{bmatrix},$$

where $A \in \mathbf{R}^{n \times m}$ is the node-incidence matrix of the graph, defined by

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

In the basic power flow optimization problem, we choose the generator powers g and the line flow powers p to minimize the total power generation cost, subject to the constraints listed above. The (given) problem data are the incidence matrix A, line capacities $P^{\max}$, demands d, maximum generator powers $G^{\max}$, and generator costs c.

In this problem we will add a basic (and widely used) reliability constraint, commonly called an '$N-1$ constraint'. (N is not a parameter in the problem; '$N-1$' just means 'all-but-one'.) This states that the system can still operate even if any one power line goes out, by re-routing the line powers. The case when line j goes out is called 'failure contingency j'; this corresponds to replacing $P_j^{\max}$ with 0. The requirement is that there must exist a contingency power flow vector $p^{(j)}$ that satisfies all the constraints above, with $p_j^{(j)} = 0$, using the same given generator powers. (This corresponds to the idea that power flows can be re-routed quickly, but generator power can only be changed more slowly.) The '$N-1$ reliability constraint' requires that for each line, there is a contingency power flow vector. The '$N-1$ reliability constraint' is (implicitly) a constraint on the generator powers.

The questions below concern the specific instance of this problem with data given in `rel_pwr_flow_data.*`. (Executing this file will also generate a figure showing the network you are optimizating.) Especially for part (b) below, you must explain exactly how you set up the problem as a convex optimization problem.

(a) *Nominal optimization.* Find the optimal generator and line power flows for this problem instance (without the $N-1$ reliability constraint). Report the optimal cost and generator powers. (You do not have to give the power line flows.)

(b) *Nominal optimization with $N-1$ reliability constraint.* Minimize the nominal cost, but you must choose generator powers that meet the $N-1$ reliability requirement as well. Report the optimal cost and generator powers. (You do not have to give the nominal power line flows, or any of the contingency flows.)

16.2 *Optimal generator dispatch.* In the *generator dispatch problem*, we schedule the electrical output power of a set of generators over some time interval, to minimize the total cost of generation while exactly meeting the (assumed known) electrical demand. One challenge in this problem is that the generators have dynamic constraints, which couple their output powers over time. For example, every generator has a maximum rate at which its power can be increased or decreased.

We label the generators $i = 1, \ldots, n$, and the time periods $t = 1, \ldots, T$. We let $p_{i,t}$ denote the (nonnegative) power output of generator i at time interval t. The (positive) electrical demand in period t is d_t. The total generated power in each period must equal the demand:

$$\sum_{i=1}^{n} p_{i,t} = d_t, \quad t = 1, \ldots, T.$$

Each generator has a minimum and maximum allowed output power:

$$P_i^{\min} \leq p_{i,t} \leq P_i^{\max}, \quad i = 1, \ldots, n, \quad t = 1, \ldots, T.$$

The cost of operating generator i at power output u is $\phi_i(u)$, where ϕ_i is an increasing strictly convex function. (Assuming the cost is mostly fuel cost, convexity of ϕ_i says that the thermal efficiency of the generator decreases as its output power increases.) We will assume these cost functions are quadratic: $\phi_i(u) = \alpha_i u + \beta_i u^2$, with α_i and β_i positive.

Each generator has a maximum ramp-rate, which limits the amount its power output can change over one time period:

$$|p_{i,t+1} - p_{i,t}| \leq R_i, \quad i = 1, \ldots, n, \quad t = 1, \ldots, T-1.$$

In addition, changing the power output of generator i from u_t to u_{t+1} incurs an additional cost $\psi_i(u_{t+1} - u_t)$, where ψ_i is a convex function. (This cost can be a real one, due to increased fuel use during a change of power, or a fictitious one that accounts for the increased maintenance cost or decreased lifetime caused by frequent or large changes in power output.) We will use the power change cost functions $\psi_i(v) = \gamma_i|v|$, where γ_i are positive.

Power plants with large capacity (*i.e.*, $P_i^{\max}$) are typically more efficient (*i.e.*, have smaller α_i, β_i), but have smaller ramp-rate limits, and higher costs associated with changing power levels. Small gas-turbine plants ('peakers') are less efficient, have less capacity, but their power levels can be rapidly changed.

The total cost of operating the generators is

$$C = \sum_{i=1}^{n} \sum_{t=1}^{T} \phi_i(p_{i,t}) + \sum_{i=1}^{n} \sum_{t=1}^{T-1} \psi_i(p_{i,t+1} - p_{i,t}).$$

Choosing the generator output schedules to minimize C, while respecting the constraints described above, is a convex optimization problem. The problem data are d_t (the demands), the generator power limits $P_i^{\min}$ and $P_i^{\max}$, the ramp-rate limits R_i, and the cost function parameters α_i, β_i, and γ_i. We will assume that problem is feasible, and that $p_{i,t}^\star$ are the (unique) optimal output powers.

(a) *Price decomposition.* Show that there are power prices $Q_1, \ldots, Q_T$ for which the following holds: For each i, $p_{i,t}^\star$ solves the optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T} \left(\phi_i(p_{i,t}) - Q_t p_{i,t}\right) + \sum_{t=1}^{T-1} \psi_i(p_{i,t+1} - p_{i,t}) \\ \text{subject to} & P_i^{\min} \leq p_{i,t} \leq P_i^{\max}, \quad t = 1, \ldots, T \\ & |p_{i,t+1} - p_{i,t}| \leq R_i, \quad t = 1, \ldots, T-1. \end{array}$$

The objective here is the portion of the objective for generator i, minus the revenue generated by the sale of power at the prices Q_t. Note that this problem involves *only* generator i; it can be solved independently of the other generators (once the prices are known). How would you find the prices Q_t?

You do not have to give a full formal proof; but you must explain your argument fully. You are welcome to use results from the text book.

(b) Solve the generator dispatch problem with the data given in `gen_dispatch_data.m`, which gives (fake, but not unreasonable) demand data for 2 days, at 15 minute intervals. This file includes code to plot the demand, optimal generator powers, and prices. (You must replace these variables with their correct values.) Comment on anything you see in your solution that might at first seem odd. Using the prices found, solve the problems in part (a) for the generators separately, to be sure they give the optimal powers (up to some small numerical errors).

Remark. While beyond the scope of this course, we mention that there are very simple price update mechanisms that adjust the prices in such a way that when the generators independently schedule themselves using the prices (as described above), we end up with the total power generated in each period matching the demand, *i.e.*, the optimal solution of the whole (coupled) problem. This gives a decentralized method for generator dispatch.

16.3 *Optimizing a portfolio of energy sources.* We have n different energy sources, such as coal-fired plants, several wind farms, and solar farms. Our job is to size each of these, *i.e.*, to choose its capacity. We will denote by c_i the capacity of plant i; these must satisfy $c_i^{\min} \leq c_i \leq c_i^{\max}$, where $c_i^{\min}$ and $c_i^{\max}$ are given minimum and maximum values.

Each generation source has a cost to build and operate (including fuel, maintenance, government subsidies and taxes) over some time period. We lump these costs together, and assume that the cost is proportional to c_i, with (given) coefficient b_i. Thus, the total cost to build and operate the energy sources is $b^T c$ (in, say, \$/hour).

Each generation source is characterized by an availability a_i, which is a random variable with values in $[0, 1]$. If source i has capacity c_i, then the power available from the plant is $c_i a_i$; the total power available from the portfolio of energy sources is $c^T a$, which is a random variable. A coal fired plant has $a_i = 1$ almost always, with $a_i < 1$ when one of its units is down for maintenance. A wind farm, in contrast, is characterized by strong fluctations in availability with $a_i = 1$ meaning a strong wind

is blowing, and $a_i = 0$ meaning no wind is blowing. A solar farm has $a_i = 1$ only during peak sun hours, with no cloud cover; at other times (such as night) we have $a_i = 0$.

Energy demand $d \in \mathbf{R}_+$ is also modeled as a random variable. The components of a (the availabilities) and d (the demand) are *not* independent. Whenever the total power available falls short of the demand, the additional needed power is generated by (expensive) peaking power plants at a fixed positive price p. The average cost of energy produced by the peakers is

$$\mathbf{E}\, p(d - c^T a)_+,$$

where $x_+ = \max\{0, x\}$. This average cost has the same units as the cost $b^T c$ to build and operate the plants.

The objective is to choose c to minimize the overall cost

$$C = b^T c + \mathbf{E}\, p(d - c^T a)_+.$$

Sample average approximation. To solve this problem, we will minimize a cost function based on a sample average of peaker cost,

$$C^{\mathrm{sa}} = b^T c + \frac{1}{N} \sum_{j=1}^{N} p(d^{(j)} - c^T a^{(j)})_+$$

where $(a^{(j)}, d^{(j)})$, $j = 1, \ldots, N$, are (given) samples from the joint distribution of a and d. (These might be obtained from historical data, weather and demand forecasting, and so on.)

Validation. After finding an optimal value of c, based on the set of samples, you should double check or validate your choice of c by evaluating the overall cost on another set of (validation) samples, $(\tilde{a}^{(j)}, \tilde{d}^{(j)})$, $j = 1, \ldots, N^{\mathrm{val}}$,

$$C^{\mathrm{val}} = b^T c + \frac{1}{N^{\mathrm{val}}} \sum_{j=1}^{N^{\mathrm{val}}} p(\tilde{d}^{(j)} - c^T \tilde{a}^{(j)})_+.$$

(These could be another set of historical data, held back for validation purposes.) If $C^{\mathrm{sa}} \approx C^{\mathrm{val}}$, our confidence that each of them is approximately the optimal value of C is increased.

Finally we get to the problem. Get the data in `energy_portfolio_data.m`, which includes the required problem data, and the samples, which are given as a $1 \times N$ row vector `d` for the scalars $d^{(j)}$, and an $n \times N$ matrix `A` for $a^{(j)}$. A second set of samples is given for validation, with the names `d_val` and `A_val`.

Carry out the optimization described above. Give the optimal cost obtained, C^{sa}, and compare to the cost evaluated using the validation data set, C^{val}.

Compare your solution with the following naive ('certainty-equivalent') approach: Replace a and d with their (sample) means, and then solve the resulting optimization problem. Give the optimal cost obtained, C^{ce} (using the average values of a and d). Is this a lower bound on the optimal value of the original problem? Now evaluate the cost for these capacities on the validation set, $C^{\mathrm{ce,val}}$. Make a brief statement.

16.4 *Optimizing processor speed.* A set of n tasks is to be completed by n processors. The variables to be chosen are the processor speeds $s_1, \ldots, s_n$, which must lie between a given minimum value $s_{\min}$ and a maximum value $s_{\max}$. The computational load of task i is α_i, so the time required to complete task i is $\tau_i = \alpha_i/s_i$.

The power consumed by processor i is given by $p_i = f(s_i)$, where $f : \mathbf{R} \to \mathbf{R}$ is positive, increasing, and convex. Therefore, the total energy consumed is

$$E = \sum_{i=1}^{n} \frac{\alpha_i}{s_i} f(s_i).$$

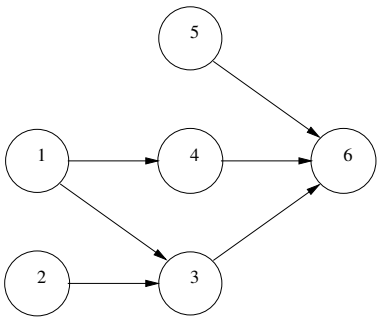
(Here we ignore the energy used to transfer data between processors, and assume the processors are powered down when they are not active.)

There is a set of *precedence constraints* for the tasks, which is a set of m ordered pairs $\mathcal{P} \subseteq \{1, \ldots, n\} \times \{1, \ldots, n\}$. If $(i, j) \in \mathcal{P}$, then task j cannot start until task i finishes. (This would be the case, for example, if task j requires data that is computed in task i.) When $(i, j) \in \mathcal{P}$, we refer to task i as a *precedent* of task j, since it must precede task j. We assume that the precedence constraints define a directed acyclic graph (DAG), with an edge from i to j if $(i, j) \in \mathcal{P}$.

If a task has no precedents, then it starts at time $t = 0$. Otherwise, each task starts as soon as all of its precedents have finished. We let T denote the time for all tasks to be completed.

To be sure the precedence constraints are clear, we consider the very small example shown below, with $n = 6$ tasks and $m = 6$ precedence constraints.

$$\mathcal{P} = \{(1,4),\ (1,3),\ (2,3),\ (3,6),\ (4,6),\ (5,6)\}.$$



In this example, tasks 1, 2, and 5 start at time $t = 0$ (since they have no precedents). Task 1 finishes at $t = \tau_1$, task 2 finishes at $t = \tau_2$, and task 5 finishes at $t = \tau_5$. Task 3 has tasks 1 and 2 as precedents, so it starts at time $t = \max\{\tau_1, \tau_2\}$, and ends τ_3 seconds later, at $t = \max\{\tau_1, \tau_2\} + \tau_3$. Task 4 completes at time $t = \tau_1 + \tau_4$. Task 6 starts when tasks 3, 4, and 5 have finished, at time $t = \max\{\max\{\tau_1, \tau_2\} + \tau_3, \tau_1 + \tau_4, \tau_5\}$. It finishes τ_6 seconds later. In this example, task 6 is the last task to be completed, so we have

$$T = \max\{\max\{\tau_1, \tau_2\} + \tau_3, \tau_1 + \tau_4, \tau_5\} + \tau_6.$$

(a) Formulate the problem of choosing processor speeds (between the given limits) to minimize completion time T, subject to an energy limit $E \leq E_{\max}$, as a convex optimization problem.

The data in this problem are $\mathcal{P}$, $s_{\min}$, $s_{\max}$, $\alpha_1, \ldots, \alpha_n$, $E_{\max}$, and the function f. The variables are $s_1, \ldots, s_n$.

Feel free to change variables or to introduce new variables. Be sure to explain clearly why your formulation of the problem is convex, and why it is equivalent to the problem statement above.

Important:

- Your formulation must be convex for any function f that is positive, increasing, and convex. You cannot make any further assumptions about f.
- This problem refers to the general case, not the small example described above.

(b) Consider the specific instance with data given in `proc_speed_data.m`, and processor power

$$f(s) = 1 + s + s^2 + s^3.$$

The precedence constraints are given by an $m \times 2$ matrix `prec`, where m is the number of precedence constraints, with each row giving one precedence constraint (the first column gives the precedents).

Plot the optimal trade-off curve of energy E versus time T, over a range of T that extends from its minimum to its maximum possible value. (These occur when all processors operate at $s_{\max}$ and $s_{\min}$, respectively, since T is monotone nonincreasing in s.) On the same plot, show the energy-time trade-off obtained when all processors operate at the same speed $\bar{s}$, which is varied from $s_{\min}$ to $s_{\max}$.

Note: In this part of the problem there is no limit $E^{\max}$ on E as in part (a); you are to find the optimal trade-off of E versus T.

16.5 *Minimum energy processor speed scheduling.* A single processor can adjust its speed in each of T time periods, labeled $1, \ldots, T$. Its speed in period t will be denoted s_t, $t = 1, \ldots, T$. The speeds must lie between given (positive) minimum and maximum values, $S^{\min}$ and $S^{\max}$, respectively, and must satisfy a slew-rate limit, $|s_{t+1} - s_t| \leq R$, $t = 1, \ldots, T-1$. (That is, R is the maximum allowed period-to-period change in speed.) The energy consumed by the processor in period t is given by $\phi(s_t)$, where $\phi : \mathbf{R} \to \mathbf{R}$ is increasing and convex. The total energy consumed over all the periods is $E = \sum_{t=1}^{T} \phi(s_t)$.

The processor must handle n jobs, labeled $1, \ldots, n$. Each job has an availability time $A_i \in \{1, \ldots, T\}$, and a deadline $D_i \in \{1, \ldots, T\}$, with $D_i \geq A_i$. The processor cannot start work on job i until period $t = A_i$, and must complete the job by the end of period D_i. Job i involves a (nonnegative) total work W_i. You can assume that in each time period, there is at least one job available, *i.e.*, for each t, there is at least one i with $A_i \leq t$ and $D_i \geq t$.

In period t, the processor allocates its effort across the n jobs as θ_t, where $\mathbf{1}^T \theta_t = 1$, $\theta_t \succeq 0$. Here θ_{ti} (the ith component of θ_t) gives the fraction of the processor effort devoted to job i in period t. Respecting the availability and deadline constraints requires that $\theta_{ti} = 0$ for $t < A_i$ or $t > D_i$. To complete the jobs we must have

$$\sum_{t=A_i}^{D_i} \theta_{ti} s_t \geq W_i, \quad i = 1, \ldots, n.$$

(a) Formulate the problem of choosing the speeds $s_1, \ldots, s_T$, and the allocations $\theta_1, \ldots, \theta_T$, in order to minimize the total energy E, as a convex optimization problem. The problem data are $S^{\min}$, $S^{\max}$, R, ϕ, and the job data, A_i, D_i, W_i, $i = 1, \ldots, n$. Be sure to justify any change of variables, or introduction of new variables, that you use in your formulation.

(b) Carry out your method on the problem instance described in `proc_sched_data.m`, with quadratic energy function $\phi(s_t) = \alpha + \beta s_t + \gamma s_t^2$. (The parameters α, β, and γ are given in the data file.) Executing this file will also give a plot showing the availability times and deadlines for the jobs.

Give the energy obtained by your speed profile and allocations. Plot these using the command `bar((s*ones(1,n)).*theta,1,'stacked')`, where s is the $T \times 1$ vector of speeds, and θ is the $T \times n$ matrix of allocations with components θ_{ti}. This will show, at each time period, how much effective speed is allocated to each job. The top of the plot will show the speed s_t. (You don't need to turn in a color version of this plot; B&W is fine.)

16.6 *AC power flow analysis via convex optimization.* This problem concerns an AC (alternating current) power system consisting of m transmission lines that connect n nodes. We describe the topology by the node-edge incidence matrix $A \in \mathbf{R}^{n \times m}$, where

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

The power flow on line j is p_j (with positive meaning in the direction of the line as defined in A, negative meaning power flow in the opposite direction).

Node i has voltage phase angle ϕ_i, and external power input s_i. (If a generator is attached to node i we have $s_i > 0$; if a load is attached we have $s_i < 0$; if the node has neither, $s_i = 0$.) Neglecting power losses in the lines, and assuming power is conserved at each node, we have $Ap = s$. (We must have $\mathbf{1}^T s = 0$, which means that the total power pumped into the network by generators balances the total power pulled out by the loads.)

The line power flows are a nonlinear function of the difference of the phase angles at the nodes they connect to:

$$p_j = \kappa_j \sin(\phi_k - \phi_l),$$

where line j goes from node k to node l. Here κ_j is a known positive constant (related to the inductance of the line). We can write this in matrix form as $p = \mathbf{diag}(\kappa) \sin(A^T \phi)$, where sin is applied elementwise.

The *DC power flow equations* are

$$Ap = s, \qquad p = \mathbf{diag}(\kappa) \sin(A^T \phi).$$

In the power analysis problem, we are given s, and want to find p and ϕ that satisfy these equations. We are interested in solutions with voltage phase angle differences that are smaller than $\pm 90^\circ$. (Under normal conditions, real power lines are never operated with voltage phase angle differences more than $\pm 20^\circ$ or so.)

You will show that the DC power flow equations can be solved by solving the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{i=j}^{m} \psi_j(p_j) \\ \text{subject to} & Ap = s, \end{array}$$

with variable p, where

$$\psi_j(u) = \int_0^u \sin^{-1}(v/\kappa_j)\, dv = u \sin^{-1}(u/\kappa_j) + \kappa_j(\sqrt{1-(u/\kappa_j)^2} - 1),$$

with domain $\mathbf{dom}\, \psi_j = (-\kappa_j, \kappa_j)$. (The second expression will be useless in this problem.)

(a) Show that the problem above is convex.

(b) Suppose the problem above has solution $p^\star$, with optimal dual variable $\nu^\star$ associated with the equality constraint $Ap = s$. Show that $p^\star$, $\phi = \nu^\star$ solves the DC power flow equation. *Hint.* Write out the optimality conditions for the problem above.

16.7 *Power transmission with losses.* A power transmission grid is modeled as a set of n nodes and m directed edges (which represent transmission lines), with topology described by the node-edge incidence matrix $A \in \mathbf{R}^{n\times m}$, defined by

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

We let $p_j^{\text{in}} \geq 0$ denote the power that flows into the tail of edge j, and $p_j^{\text{out}} \geq 0$ the power that emerges from the head of edge j, for $j = 1, \ldots, m$. Due to transmission losses, the power that flows into each edge is more than the power that emerges:

$$p_j^{\text{in}} = p_j^{\text{out}} + \alpha(L_j/R_j^2)(p_j^{\text{out}})^2, \quad j = 1, \ldots, m,$$

where $L_j > 0$ is the length of transmission line j, $R_j > 0$ is the radius of the conductors on line j, and $\alpha > 0$ is a constant. (The second term on the righthand side above is the transmission line power loss.) In addition, each edge has a maximum allowed input power, that also depends on the conductor radius: $p_j^{\text{in}} \leq \sigma R_j^2$, $j = 1, \ldots, m$, where $\sigma > 0$ is a constant.

Generators are attached to nodes $i = 1, \ldots, k$, and loads are attached to nodes $i = k+1, \ldots, n$. We let g_i denote the (nonnegative) power injected into node i by its generator, for $i = 1, \ldots, k$. We let l_i denote the (nonnegative) power pulled from node i by the load, for $i = k+1, \ldots, n$. These load powers are known and fixed.

We must have power balance at each node. For $i = 1, \ldots, k$, the sum of all power entering the node from incoming transmission lines, plus the power supplied by the generator, must equal the sum of all power leaving the node on outgoing transmission lines:

$$\sum_{j\in\mathcal{E}(i)} p_j^{\text{out}} + g_i = \sum_{j\in\mathcal{L}(i)} p_j^{\text{in}}, \quad i = 1, \ldots, k,$$

where $\mathcal{E}(i)$ ($\mathcal{L}(i)$) is the set of edge indices for edges entering (leaving) node i. For the load nodes $i = k+1, \ldots, n$ we have a similar power balance condition:

$$\sum_{j\in\mathcal{E}(i)} p_j^{\text{out}} = \sum_{j\in\mathcal{L}(i)} p_j^{\text{in}} + l_i, \quad i = k+1, \ldots, n.$$

Each generator can vary its power g_i over a given range $[0, G_i^{\max}]$, and has an associated cost of generation $\phi_i(g_i)$, where ϕ_i is convex and strictly increasing, for $i = 1, \ldots, k$.

(a) *Minimum total cost of generation.* Formulate the problem of choosing generator and edge input and output powers, so as to minimize the total cost of generation, as a convex optimization problem. (All other quantities described above are known.) Be sure to explain any additional variables or terms you introduce, and to justify any transformations you make.

Hint: You may find the matrices $A_+ = (A)_+$ and $A_- = (-A)_+$ helpful in expressing the power balance constraints.

(b) *Marginal cost of power at load nodes.* The (marginal) cost of power at node i, for $i = k + 1, \ldots, n$, is the partial derivative of the minimum total power generation cost, with respect to varying the load power l_i. (We will simply assume these partial derivatives exist.) Explain how to find the marginal cost of power at node i, from your formulation in part (a).

(c) *Optimal sizing of lines.* Now suppose that you can optimize over generator powers, edge input and output powers (as above), *and* the power line radii R_j, $j = 1, \ldots, m$. These must lie between given limits, $R_j \in [R_j^{\min}, R_j^{\max}]$ ($R_j^{\min} > 0$), and we must respect a total volume constraint on the lines,

$$\sum_{j=1}^{m} L_j R_j^2 \leq V^{\max}.$$

Formulate the problem of choosing generator and edge input and output powers, as well as power line radii, so as to minimize the total cost of generation, as a convex optimization problem. (Again, explain anything that is not obvious.)

(d) *Numerical example.* Using the data given in `ptrans_loss_data.m`, find the minimum total generation cost and the marginal cost of power at nodes $k + 1, \ldots, n$, for the case described in parts (a) and (b) (*i.e.*, using the fixed given radii R_j), and also for the case described in part (c), where you are allowed to change the transmission line radii, keeping the same total volume as the original lines. For the generator costs, use the quadratic functions

$$\phi_i(g_i) = a_i g_i + b_i g_i^2, \quad i = 1, \ldots, k,$$

where $a,\ b \in \mathbf{R}_+^k$. (These are given in the data file.)

Remark: In the m-file, we give you a load vector $l \in \mathbf{R}^{n-k}$. For consistency, the ith entry of this vector corresponds to the load at node $k + i$.

16.8 *Utility/power trade-off in a wireless network.* In this problem we explore the trade-off between total utility and total power usage in a wireless network in which the link transmit powers can be adjusted. The network consists of a set of nodes and a set of links over which data can be transmitted. There are n routes, each corresponding to a sequence of links from a source to a destination node. Route j has a data flow rate $f_j \in \mathbf{R}_+$ (in units of bits/sec, say). The total utility (which we want to maximize) is

$$U(f) = \sum_{j=1}^{n} U_j(f_j),$$

where $U_j : \mathbf{R} \to \mathbf{R}$ are concave increasing functions.

The network topology is specified by the routing matrix $R \in \mathbf{R}^{m\times n}$, defined as

$$R_{ij} = \begin{cases} 1 & \text{route } j \text{ passes over link } i \\ 0 & \text{otherwise.} \end{cases}$$

The total traffic on a link is the sum of the flows that pass over the link. The traffic (vector) is thus $t = Rf \in \mathbf{R}^m$. The traffic on each link cannot exceed the capacity of the link, *i.e.*, $t \preceq c$, where $c \in \mathbf{R}^m_+$ is the vector of link capacities.

The link capacities, in turn, are functions of the link transmit powers, given by $p \in \mathbf{R}^m_+$, which cannot exceed given limits, *i.e.*, $p \preceq p^{\max}$. These are related by

$$c_i = \alpha_i \log(1 + \beta_i p_i),$$

where α_i and β_i are positive parameters that characterize link i. The second objective (which we want to minimize) is $P = \mathbf{1}^T p$, the total (transmit) power.

(a) Explain how to find the optimal trade-off curve of total utility and total power, using convex or quasiconvex optimization.

(b) Plot the optimal trade-off curve for the problem instance with $m = 20$, $n = 10$, $U_j(x) = \sqrt{x}$ for $j = 1, \ldots, n$, $p_i^{\max} = 10$, $\alpha_i = \beta_i = 1$ for $i = 1, \ldots, m$, and network topology generated using

```
rand('seed',3);
R = round(rand(m,n));
```

Your plot should have the total power on the horizontal axis.

16.9 *Energy storage trade-offs.* We consider the use of a storage device (say, a battery) to reduce the total cost of electricity consumed over one day. We divide the day into T time periods, and let p_t denote the (positive, time-varying) electricity price, and u_t denote the (nonnegative) usage or consumption, in period t, for $t = 1, \ldots, T$. Without the use of a battery, the total cost is $p^T u$.

Let q_t denote the (nonnegative) energy stored in the battery in period t. For simplicity, we neglect energy loss (although this is easily handled as well), so we have $q_{t+1} = q_t + c_t$, $t = 1, \ldots, T-1$, where c_t is the charging of the battery in period t; $c_t < 0$ means the battery is discharged. We will require that $q_1 = q_T + c_T$, *i.e.*, we finish with the same battery charge that we start with. With the battery operating, the net consumption in period t is $u_t + c_t$; we require this to be nonnegative (*i.e.*, we do not pump power back into the grid). The total cost is then $p^T(u + c)$.

The battery is characterized by three parameters: The capacity Q, where $q_t \leq Q$; the maximum charge rate C, where $c_t \leq C$; and the maximum discharge rate D, where $c_t \geq -D$. (The parameters Q, C, and D are nonnegative.)

(a) Explain how to find the charging profile $c \in \mathbf{R}^T$ (and associated stored energy profile $q \in \mathbf{R}^T$) that minimizes the total cost, subject to the constraints.

(b) Solve the problem instance with data p and u given in `storage_tradeoff_data.*`, $Q = 35$, and $C = D = 3$. Plot u_t, p_t, c_t, and q_t versus t.

(c) *Storage trade-offs.* Plot the minimum total cost versus the storage capacity Q, using p and u from `storage_tradeoff_data.*`, and charge/discharge limits $C = D = 3$. Repeat for charge/discharge limits $C = D = 1$. (Put these two trade-off curves on the same plot.) Give an interpretation of the endpoints of the trade-off curves.

16.10 *Cost-comfort trade-off in air conditioning.* A heat pump (air conditioner) is used to cool a residence to temperature T_t in hour t, on a day with outside temperature T_t^{out}, for $t = 1, \ldots, 24$. These temperatures are given in Kelvin, and we will assume that $T_t^{\text{out}} \geq T_t$.

A total amount of heat $Q_t = \alpha(T_t^{\text{out}} - T_t)$ must be removed from the residence in hour t, where α is a positive constant (related to the quality of thermal insulation).

The electrical energy required to pump out this heat is given by $E_t = Q_t/\gamma_t$, where

$$\gamma_t = \eta \frac{T_t}{T_t^{\text{out}} - T_t}$$

is the *coefficient of performance* of the heat pump and $\eta \in (0, 1]$ is the efficiency constant. The efficiency is typically around 0.6 for a modern unit; the theoretical limit is $\eta = 1$. (When $T_t = T_t^{\text{out}}$, we take $\gamma_t = \infty$ and $E_t = 0$.)

Electrical energy prices vary with the hour, and are given by $P_t > 0$ for $t = 1, \ldots, 24$. The total energy cost is $C = \sum_t P_t E_t$. We will assume that the prices are known.

Discomfort is measured using a piecewise-linear function of temperature,

$$D_t = (T_t - T^{\text{ideal}})_+,$$

where T^{ideal} is an ideal temperature, below which there is no discomfort. The total daily discomfort is $D = \sum_{t=1}^{24} D_t$. You can assume that $T^{\text{ideal}} < T_t^{\text{out}}$.

To get a point on the optimal cost-comfort trade-off curve, we will minimize $C + \lambda D$, where $\lambda > 0$. The variables to be chosen are $T_1, \ldots, T_{24}$; all other quantities described above are given.

Show that this problem has an analytical solution of the form $T_t = \psi(P_t, T_t^{\text{out}})$, where $\psi : \mathbf{R}^2 \to \mathbf{R}$. The function ψ can depend on the constants α, η, T^{ideal}, λ. Give ψ explicitly. You are free (indeed, encouraged) to check your formula using CVX, with made up values for the constants.

Disclaimer. The focus of this course is *not* on deriving 19th century pencil and paper solutions to problems. But every now and then, a practical problem will actually have an analytical solution. This is one of them.

16.11 *Optimal electric motor drive currents.* In this problem you will design the drive current waveforms for an AC (alternating current) electric motor. The motor has a magnetic rotor which spins with constant angular velocity $\omega \geq 0$ inside the stationary stator. The stator contains three circuits (called *phase windings*) with (vector) current waveform $i : \mathbf{R} \to \mathbf{R}^3$ and (vector) voltage waveform $v : \mathbf{R} \to \mathbf{R}^3$, which are 2π-periodic functions of the angular position θ of the rotor. The circuit dynamics are

$$v(\theta) = Ri(\theta) + \omega L \frac{d}{d\theta} i(\theta) + \omega k(\theta),$$

where $R \in \mathbf{S}^3_{++}$ is the resistance matrix, $L \in \mathbf{S}^3_{++}$ is the inductance matrix, and $k : \mathbf{R} \to \mathbf{R}^3$, a 2π-periodic function of θ, is the back-EMF waveform (which encodes the electromagnetic coupling

between the rotor permanent magnets and the phase windings). The angular velocity ω, the matrices R and L, and the back-EMF waveform k, are known.

We must have $|v_i(\theta)| \leq v^{\text{supply}}$, $i = 1, 2, 3$, where v^{supply} is the (given) supply voltage. The output torque of the motor at rotor position θ is $\tau(\theta) = k(\theta)^T i(\theta)$. We will require the torque to have a given constant nonnegative value: $\tau(\theta) = \tau^{\text{des}}$ for all θ.

The average power loss in the motor is

$$P^{\text{loss}} = \frac{1}{2\pi} \int_0^{2\pi} i(\theta)^T R i(\theta)\, d\theta.$$

The mechanical output power is $P^{\text{out}} = \omega \tau^{\text{des}}$, and the motor efficiency is

$$\eta = P^{\text{out}}/(P^{\text{out}} + P^{\text{loss}}).$$

The objective is to choose the current and voltage waveforms to maximize η.

Discretization. To solve this problem we consider a discretized version in which θ takes on the N values $\theta = h, 2h, \ldots, Nh$, where $h = 2\pi/N$. We impose the voltage and torque constraints for these values of θ. We approximate the power loss as

$$P^{\text{loss}} = (1/N) \textstyle\sum_{j=1}^{N} i(jh)^T R i(jh).$$

The circuit dynamics are approximated as

$$v(jh) = Ri(jh) + \omega L \frac{i((j+1)h) - i(jh)}{h} + \omega k(jh), \quad j = 1, \ldots, N,$$

where here we take $i((N+1)h) = i(h)$ (by periodicity).

Find optimal (discretized) current and voltage waveforms for the problem instance with data given in `ac_motor_data.m`. The back-EMF waveform is given as a $3 \times N$ matrix `K`. Plot the three current waveform components on one plot, and the three voltage waveforms on another. Give the efficiency obtained.

17 Miscellaneous applications

17.1 *Earth mover's distance.* In this exercise we explore a general method for constructing a distance between two probability distributions on a finite set, called the *earth mover's distance*, *Wasserstein metric*, or *Dubroshkin metric*. Let x and y be two probability distributions on $\{1, \ldots, n\}$, *i.e.*, $\mathbf{1}^T x = \mathbf{1}^T y = 1$, $x \succeq 0$, $y \succeq 0$. We imagine that x_i is the amount of earth stored at location i; our goal is to move the earth between locations to obtain the distribution given by y. Let C_{ij} be the cost of moving one unit of earth from location j to location i. We assume that $C_{ii} = 0$, and $C_{ij} = C_{ji} > 0$ for $i \neq j$. (We allow $C_{ij} = \infty$, which means that earth cannot be moved directly from node j to node i.) Let $S_{ij} \geq 0$ denote the amount of earth moved from location j to location i. The total cost is $\sum_{i,j=1}^n S_{ij} C_{ij} = \mathbf{tr}\, C^T S$. The shipment matrix S must satisfy the balance equations,

$$\sum_{j=1}^n S_{ij} = y_i, \quad i = 1, \ldots, n, \qquad \sum_{i=1}^n S_{ij} = x_j, \quad j = 1, \ldots, n,$$

which we can write compactly as $S\mathbf{1} = y$, $S^T\mathbf{1} = x$. (The first equation states that the total amount shipped into location i equals y_i; the second equation states that the total shipped out from location j is x_j.) The earth mover's distance between x and y, denoted $d(x, y)$, is given by the minimal cost of earth moving required to transform x to y, *i.e.*, the optimal value of the problem

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}\, C^T S \\ \text{subject to} & S_{ij} \geq 0, \quad i, j = 1, \ldots, n \\ & S\mathbf{1} = y, \quad S^T\mathbf{1} = x, \end{array}$$

with variables $S \in \mathbf{R}^{n \times n}$.

We can also give a probability interpretation of $d(x, y)$. Consider a random variable Z on $\{1, \ldots, n\}^2$ with values C_{ij}. We seek the joint distribution S that minimizes the expected value of the random variable Z, with given marginals x and y.

The earth mover's distance is used to compare, for example, 2D images, with C_{ij} equal to the distance between pixels i and j. If x and y represent two photographs of the same scene, from slightly different viewpoints and with an offset in camera position (say), $d(x, y)$ will be small, but the distance between x and y measured by most common norms (*e.g.*, $\|x - y\|_1$) will be large.

(a) Show that d satisfies the following. *Symmetry*: $d(x, y) = d(y, x)$; *Nonnegativity*: $d(x, y) \geq 0$; *Definiteness*: $d(x, x) = 0$, and $d(x, y) > 0$ for $x \neq y$.

(b) Show that $d(x, y)$ is the optimal value of the problem

$$\begin{array}{ll} \text{maximize} & \nu^T x + \mu^T y \\ \text{subject to} & \nu_i + \mu_j \leq C_{ij}, \quad i, j = 1, \ldots, n, \end{array}$$

with variables $\nu,\ \mu \in \mathbf{R}^n$.

17.2 *Radiation treatment planning.* In radiation treatment, radiation is delivered to a patient, with the goal of killing or damaging the cells in a tumor, while carrying out minimal damage to other tissue. The radiation is delivered in beams, each of which has a known pattern; the level of each beam can be adjusted. (In most cases multiple beams are delivered at the same time, in one 'shot', with the

treatment organized as a sequence of 'shots'.) We let b_j denote the level of beam j, for $j = 1, \ldots, n$. These must satisfy $0 \leq b_j \leq B^{\max}$, where $B^{\max}$ is the maximum possible beam level. The exposure area is divided into m voxels, labeled $i = 1, \ldots, m$. The dose d_i delivered to voxel i is linear in the beam levels, *i.e.*, $d_i = \sum_{j=1}^{n} A_{ij} b_j$. Here $A \in \mathbf{R}_+^{m \times n}$ is a (known) matrix that characterizes the beam patterns. We now describe a simple radiation treatment planning problem.

A (known) subset of the voxels, $\mathcal{T} \subset \{1, \ldots, m\}$, corresponds to the tumor or target region. We require that a minimum radiation dose D^{target} be administered to each tumor voxel, *i.e.*, $d_i \geq D^{\text{target}}$ for $i \in \mathcal{T}$. For all other voxels, we would like to have $d_i \leq D^{\text{other}}$, where D^{other} is a desired maximum dose for non-target voxels. This is generally not feasible, so instead we settle for minimizing the penalty

$$E = \sum_{i \notin \mathcal{T}} ((d_i - D^{\text{other}})_+)^2,$$

where $(\cdot)_+$ denotes the nonnegative part. We can interpret E as the sum of the squares of the nontarget excess doses.

(a) Show that the treatment planning problem is convex. The optimization variable is $b \in \mathbf{R}^n$; the problem data are $B^{\max}$, A, $\mathcal{T}$, D^{target}, and D^{other}.

(b) Solve the problem instance with data given in the file `treatment_planning_data.m`. Here we have split the matrix A into `Atarget`, which contains the rows corresponding to the target voxels, and `Aother`, which contains the rows corresponding to other voxels. Give the optimal value. Plot the dose histogram for the target voxels, and also for the other voxels. Make a brief comment on what you see. *Remark.* The beam pattern matrix in this problem instance is randomly generated, but similar results would be obtained with realistic data.

17.3 *Flux balance analysis in systems biology.* Flux balance analysis is based on a very simple model of the reactions going on in a cell, keeping track only of the gross rate of consumption and production of various chemical species within the cell. Based on the known stoichiometry of the reactions, and known upper bounds on some of the reaction rates, we can compute bounds on the other reaction rates, or cell growth, for example.

We focus on m metabolites in a cell, labeled $M_1, \ldots, M_m$. There are n reactions going on, labeled $R_1, \ldots, R_n$, with nonnegative reaction rates $v_1, \ldots, v_n$. Each reaction has a (known) stoichiometry, which tells us the rate of consumption and production of the metabolites per unit of reaction rate. The stoichiometry data is given by the *stoichiometry matrix* $S \in \mathbf{R}^{m \times n}$, defined as follows: S_{ij} is the rate of production of M_i due to unit reaction rate $v_j = 1$. Here we consider consumption of a metabolite as negative production; so $S_{ij} = -2$, for example, means that reaction R_j causes metabolite M_i to be consumed at a rate $2v_j$.

As an example, suppose reaction R_1 has the form $M_1 \to M_2 + 2M_3$. The consumption rate of M_1, due to this reaction, is v_1; the production rate of M_2 is v_1; and the production rate of M_3 is $2v_1$. (The reaction R_1 has no effect on metabolites $M_4, \ldots, M_m$.) This corresponds to a first column of S of the form $(-1, 1, 2, 0, \ldots, 0)$.

Reactions are also used to model flow of metabolites into and out of the cell. For example, suppose that reaction R_2 corresponds to the flow of metabolite M_1 into the cell, with v_2 giving the flow rate. This corresponds to a second column of S of the form $(1, 0, \ldots, 0)$.

The last reaction, R_n, corresponds to biomass creation, or cell growth, so the reaction rate v_n is the cell growth rate. The last column of S gives the amounts of metabolites used or created per unit of cell growth rate.

Since our reactions include metabolites entering or leaving the cell, as well as those converted to biomass within the cell, we have conservation of the metabolites, which can be expressed as $Sv = 0$. In addition, we are given upper limits on *some* of the reaction rates, which we express as $v \preceq v^{\max}$, where we set $v_j^{\max} = \infty$ if no upper limit on reaction rate j is known. The goal is to find the maximum possible cell growth rate (*i.e.*, largest possible value of v_n) consistent with the constraints

$$Sv = 0, \qquad v \succeq 0, \qquad v \preceq v^{\max}.$$

The questions below pertain to the data found in `fba_data.m`.

(a) Find the maximum possible cell growth rate $G^\star$, as well as optimal Lagrange multipliers for the reaction rate limits. How sensitive is the maximum growth rate to the various reaction rate limits?

(b) *Essential genes and synthetic lethals.* For simplicity, we'll assume that each reaction is controlled by an associated gene, *i.e.*, gene G_i controls reaction R_i. Knocking out a set of genes associated with some reactions has the effect of setting the reaction rates (or equivalently, the associated $v^{\max}$ entries) to zero, which of course reduces the maximum possible growth rate. If the maximum growth rate becomes small enough or zero, it is reasonable to guess that knocking out the set of genes will kill the cell. An *essential gene* is one that when knocked out reduces the maximum growth rate below a given threshold $G^{\min}$. (Note that G_n is always an essential gene.) A *synthetic lethal* is a pair of non-essential genes that when knocked out reduces the maximum growth rate below the threshold. Find all essential genes and synthetic lethals for the given problem instance, using the threshold $G^{\min} = 0.2G^\star$.

17.4 *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`. `ad_disp_data.py`. 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.

17.5 *Ranking by aggregating preferences.* We have n objects, labeled $1, \ldots, n$. Our goal is to assign a real valued rank r_i to the objects. A *preference* is an ordered pair (i, j), meaning that object i is preferred over object j. The ranking $r \in \mathbf{R}^n$ and preference (i, j) are *consistent* if $r_i \geq r_j + 1$. (This sets the scale of the ranking: a gap of one in ranking is the threshold for preferring one item over another.) We define the *preference violation* of preference (i, j) with ranking $r \in \mathbf{R}^n$ as

$$v = (r_j + 1 - r_i)_+ = \max\{r_j + 1 - r_i, 0\}.$$

We have a set of m preferences among the objects, $(i^{(1)}, j^{(1)}), \ldots, (i^{(m)}, j^{(m)})$. (These may come from several different evaluators of the objects, but this won't matter here.)

We will select our ranking r as a minimizer of the total preference violation penalty, defined as

$$J = \sum_{k=1}^m \phi(v^{(k)}),$$

where $v^{(k)}$ is the preference violation of $(i^{(k)}, j^{(k)})$ with r, and ϕ is a nondecreasing convex penalty function that satisfies $\phi(u) = 0$ for $u \leq 0$.

(a) Make a (simple, please) suggestion for ϕ for each of the following two situations:

  (i) We don't mind some small violations, but we really want to avoid large violations.

  (ii) We want as many preferences as possible to be consistent with the ranking, but will accept some (hopefully, few) larger preference violations.

(b) Find the rankings obtained using the penalty functions proposed in part (a), on the data set found in `rank_aggr_data.m`. Plot a histogram of preference violations for each case and *briefly* comment on the differences between them. Give the number of positive preference violations for each case. (Use `sum(v>0.001)` to determine this number.)

Remark. The objects could be candidates for a position, papers at a conference, movies, websites, courses at a university, and so on. The preferences could arise in several ways. Each of a set of evaluators provides some preferences, for example by rank ordering a subset of the objects. The problem can be thought of as aggregating the preferences given by the evaluators, to come up with a composite ranking.

17.6 *Time release formulation.* A patient is treated with a drug (say, in pill form) at different times. Each treatment (or pill) contains (possibly) different amounts of various formulations of the drug. Each of the formulations, in turn, has a characteristic pattern as to how quickly it releases the drug into the bloodstream. The goal is to optimize the blend of formulations that go into each treatment, in order to achieve a desired drug concentration in the bloodstream over time.

We will use discrete time, $t = 1, 2, \ldots, T$, representing hours (say). There will be K treatments, administered at known times $1 = \tau_1 < \tau_2 < \cdots < \tau_K < T$. We have m drug formulations; each treatment consists of a mixture of these m formulations. We let $a^{(k)} \in \mathbf{R}_+^m$ denote the amounts of the m formulations in treatment k, for $k = 1, \ldots, K$.

Each formulation i has a time profile $p_i(t) \in \mathbf{R}_+$, for $t = 1, 2, \ldots$. If an amount $a_i^{(k)}$ of formulation i from treatment k is administered at time t_0, the drug concentration in the bloodstream (due to this formulation) is given by $a_i^{(k)} p_i(t - t_0)$ for $t > t_0$, and 0 for $t \leq t_0$. To simplify notation, we will define $p_i(t)$ to be zero for $t = 0, -1, -2, \ldots$. We assume the effects of the different formulations and different treatments are additive, so the total bloodstream drug concentration is given by

$$c(t) = \sum_{k=1}^{K} \sum_{i=1}^{m} p_i(t - \tau_k) a_i^{(k)}, \quad t = 1, \ldots, T.$$

(This is just a vector convolution.) Recall that $p_i(t - \tau_k) = 0$ for $t \leq \tau_k$, which means that the effect of treatment k does not show up until time $\tau_k + 1$.

We require that $c(t) \leq c^{\max}$ for $t = 1, \ldots, T$, where $c^{\max}$ is a given maximum permissible concentration. We define the therapeutic time T^{ther} as

$$T^{\text{ther}} = \min\{t \mid c(\tau) \geq c^{\min} \text{ for } \tau = t, \ldots, T\},$$

with $T^{\text{ther}} = \infty$ if $c(t) < c^{\min}$ for $t = 1, \ldots, T$. Here, $c^{\min}$ is the minimum concentration for the drug to have therapeutic value. Thus, T^{ther} is the first time at which the drug concentration reaches, and stays above, the minimum therapeutic level.

Finally, we get to the problem. The optimization variables are the treatment formulation vectors $a^{(1)}, \ldots, a^{(K)}$. There are two objectives: T^{ther} (which we want to be small), and

$$J^{\text{ch}} = \sum_{k=1}^{K-1} \|a^{(k+1)} - a^{(k)}\|_\infty$$

(which we also want to be small). This second objective is a penalty for changing the formulation amounts in the treatments.

The rest of the problem concerns the specific instance with data given in the file `time_release_form_data.m`. This gives data for $T = 168$ (one week, starting from 8AM Monday morning), with treatments occuring 3 times each day, at 8AM, 2PM, and 11PM, so we have a total of $K = 21$ treatments. We have $m = 6$ formulations, with profiles with length 96 (*i.e.*, $p_i(t) = 0$ for $t > 96$).

- Explain how to find the optimal trade-off curve of T^{ther} versus J^{ch}. Your method may involve solving several convex optimization problems.
- Plot the trade-off curve over a reasonable range, and be sure to explain or at least comment on the endpoints of the trade-off curve.
- Plot the treatment formulation amounts versus k, and the bloodstream concentration versus t, for the two trade-off curve endpoints, and one corresponding to $T^{\text{ther}} = 8$.

Warning. We've found that CVX can experience numerical problems when solving this problem (depending on how it is formulated). In one case, `cvx_status` is "Solved/Inaccurate" when in fact the problem has been solved (just not to the tolerances SeDuMi likes to see). You can ignore this status, taking it to mean Optimal. You can also try switching to the SDPT3 solver. In any case, please do not spend much time worrying about, or dealing with, these numerical problems.

17.7 *Sizing a gravity feed water supply network.* A water supply network connects water supplies (such as reservoirs) to consumers via a network of pipes. Water flow in the network is due to gravity (as opposed to pumps, which could also be added to the formulation). The network is composed of a set of n nodes and m directed edges between pairs of nodes. The first k nodes are supply or reservoir nodes, and the remaining $n-k$ are consumer nodes. The edges correspond to the pipes in the water supply network.

We let $f_j \geq 0$ denote the water flow in pipe (edge) j, and h_i denote the (known) altitude or height of node i (say, above sea level). At nodes $i = 1, \ldots, k$, we let $s_i \geq 0$ denote the flow into the network from the supply. For $i = 1, \ldots, n-k$, we let $c_i \geq 0$ denote the water flow taken out of the network (by consumers) at node $k+i$. Conservation of flow can be expressed as

$$Af = \begin{bmatrix} -s \\ c \end{bmatrix},$$

where $A \in \mathbf{R}^{n \times m}$ is the incidence matrix for the supply network, given by

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

We assume that each edge is oriented from a node of higher altitude to a node of lower altitude; if edge j goes from node i to node l, we have $h_i > h_l$. The pipe flows are determined by

$$f_j = \frac{\alpha \theta_j R_j^2 (h_i - h_l)}{L_j},$$

where edge j goes from node i to node l, $\alpha > 0$ is a known constant, $L_j > 0$ is the (known) length of pipe j, $R_j > 0$ is the radius of pipe j, and $\theta_j \in [0, 1]$ corresponds to the valve opening in pipe j.

Finally, we have a few more constraints. The supply feed rates are limited: we have $s_i \leq S_i^{\max}$. The pipe radii are limited: we have $R_j^{\min} \leq R_j \leq R_j^{\max}$. (These limits are all known.)

(a) *Supportable consumption vectors.* Suppose that the pipe radii are fixed and known. We say that $c \in \mathbf{R}_+^{n-k}$ is supportable if there is a choice of f, s, and θ for which all constraints and conditions above are satisfied. Show that the set of supportable consumption vectors is a polyhedron, and explain how to determine whether or not a given consumption vector is supportable.

(b) *Optimal pipe sizing.* You must select the pipe radii R_j to minimize the cost, which we take to be (proportional to) the total volume of the pipes, $L_1R_1^2+\cdots+L_mR_m^2$, subject to being able to support a set of consumption vectors, denoted $c^{(1)},\ldots,c^{(N)}$, which we refer to as consumption scenarios. (This means that any consumption vector in the convex hull of $\{c^{(1)},\ldots,c^{(N)}\}$ will be supportable.) Show how to formulate this as a convex optimization problem. *Note.* You are asked to choose *one* set of pipe radii, and N sets of valve parameters, flow vectors, and source vectors; one for each consumption scenario.

(c) Solve the instance of the optimal pipe sizing problem with data defined in the file `grav_feed_network_data.m`, and report the optimal value and the optimal pipe radii. The columns of the matrix C in the data file are the consumption vectors $c^{(1)},\ldots,c^{(N)}$.

Hint. $-A^Th$ gives a vector containing the height differences across the edges.

17.8 *Optimal political positioning.* A political constituency is a group of voters with similar views on a set of political issues. The electorate (*i.e.*, the set of voters in some election) is partitioned (by a political analyst) into K constituencies, with (nonnegative) populations $P_1,\ldots,P_K$. A candidate in the election has an initial or prior position on each of n issues, but is willing to consider (presumably small) deviations from her prior positions in order to maximize the total number of votes she will receive. We let $x_i \in \mathbf{R}$ denote the change in her position on issue i, measured on some appropriate scale. (You can think of $x_i < 0$ as a move to the 'left' and $x_i > 0$ as a move to the 'right' on the issue, if you like.) The vector $x \in \mathbf{R}^n$ characterizes the changes in her position on all issues; $x = 0$ represents the prior positions. On each issue she has a limit on how far in each direction she is willing to move, which we express as $l \preceq x \preceq u$, where $l \prec 0$ and $u \succ 0$ are given.

The candidate's position change x affects the fraction of voters in each constituency that will vote for her. This fraction is modeled as a logistic function,

$$f_k = g(w_k^Tx + v_k), \quad k = 1,\ldots,K.$$

Here $g(z) = 1/(1+\exp(-z))$ is the standard logistic function, and $w_k \in \mathbf{R}^n$ and $v_k \in \mathbf{R}$ are given data that characterize the views of constituency k on the issues. Thus the total number of votes the candidate will receive is

$$V = P_1f_1 + \cdots + P_Kf_K.$$

The problem is to choose x (subject to the given limits) so as to maximize V. The problem data are l, u, and P_k, w_k, and v_k for $k = 1,\ldots,K$.

(a) *The general political positioning problem.* Show that the objective function V need not be quasiconcave. (This means that the general optimal political positioning problem is not a quasiconvex problem, and therefore also not a convex problem.) In other words, choose problem data for which V is not a quasiconcave function of x.

(b) *The partisan political positioning problem.* Now suppose the candidate focuses only on her core constituencies, *i.e.*, those for which a significant fraction will vote for her. In this case we interpret the K constituencies as her core constituencies; we assume that $v_k \geq 0$, which means that with her prior position $x = 0$, at least half of each of her core constituencies will vote for her. We add the constraint that $w_k^Tx + v_k \geq 0$ for each k, which means that she will not take positions that alienate a majority of voters from any of her core constituencies.

Show that the partisan political positioning problem (*i.e.*, maximizing V with the additional assumptions and constraints) is convex.

(c) *Numerical example.* Find the optimal positions for the partisan political positioning problem with data given in `opt_pol_pos_data.m`. Report the number of votes from each constituency under the politician's prior positions ($x = 0$) and optimal positions, as well as the total number of votes V in each case.

You may use the function

$$g_{\text{approx}}(z) = \min\{1,\ g(i) + g'(i)(z - i) \text{ for } i = 0, 1, 2, 3, 4\}$$

as an approximation of g for $z \geq 0$. (The function g_{approx} is also an upper bound on g for $z \geq 0$.) For your convenience, we have included function definitions for g and g_{approx} (`g` and `gapx`, respectively) in the data file. You should report the results (votes from each constituency and total) using g, but be sure to check that these numbers are close to the results using g_{approx} (say, within one percent or so).

17.9 *Resource allocation in stream processing.* A large data center is used to handle a stream of J types of jobs. The traffic (number of instances per second) of each job type is denoted $t \in \mathbf{R}^J_+$. Each instance of each job type (serially) invokes or calls a set of processes. There are P types of processes, and we describe the job-process relation by the $P \times J$ matrix

$$R_{pj} = \begin{cases} 1 & \text{job } j \text{ invokes process } p \\ 0 & \text{otherwise.} \end{cases}$$

The process loads (number of instances per second) are given by $\lambda = Rt \in \mathbf{R}^P$, *i.e.*, λ_p is the sum of the traffic from the jobs that invoke process p.

The latency of a process or job type is the average time that it takes one instance to complete. These are denoted $l^{\text{proc}} \in \mathbf{R}^P$ and $l^{\text{job}} \in \mathbf{R}^J$, respectively, and are related by $l^{\text{job}} = R^T l^{\text{proc}}$, *i.e.*, l^{job}_j is the sum of the latencies of the processes called by j. Job latency is important to users, since l^{job}_j is the average time the data center takes to handle an instance of job type j. We are given a maximum allowed job latency: $l^{\text{job}} \preceq l^{\max}$.

The process latencies depend on the process load and also how much of n different resources are made available to them. These resources might include, for example, number of cores, disk storage, and network bandwidth. Here, we represent amounts of these resources as (nonnegative) real numbers, so $x_p \in \mathbf{R}^n_+$ represents the resources allocated to process p. The process latencies are given by

$$l^{\text{proc}}_p = \psi_p(x_p, \lambda_p), \quad p = 1, \ldots, P,$$

where $\psi_p : \mathbf{R}^n \times \mathbf{R} \to \mathbf{R} \cup \{\infty\}$ is a known (extended-valued) convex function. These functions are nonincreasing in their first (vector) arguments, and nondecreasing in their second arguments (*i.e.*, more resources or less load cannot increase latency). We interpret $\psi_p(x_p, \lambda_p) = \infty$ to mean that the resources given by x_p are not sufficient to handle the load λ_p.

We wish to allocate a total resource amount $x^{\text{tot}} \in \mathbf{R}^n_{++}$ among the P processes, so we have $\sum_{p=1}^P x_p \preceq x^{\text{tot}}$. The goal is to minimize the objective function

$$\sum_{j=1}^J w_j (t^{\text{tar}}_j - t_j)_+,$$

where t_j^{tar} is the target traffic level for job type j, $w_j > 0$ give the priorities, and $(u)_+$ is the nonnegative part of a vector, *i.e.*, $u_i = \max\{u_i, 0\}$. (Thus the objective is a weighted penalty for missing the target job traffic.) The variables are $t \in \mathbf{R}_+^J$ and $x_p \in \mathbf{R}_+^n$, $p = 1, \ldots, P$. The problem data are the matrix R, the vectors $l^{\max}$, x^{tot}, t^{tar}, and w, and the functions ψ_p, $p = 1, \ldots, P$.

(a) Explain why this is a convex optimization problem.

(b) Solve the problem instance with data given in `res_alloc_stream_data.m`, with latency functions

$$\psi_p(x_p, \lambda_p) = \begin{cases} 1/(a_p^T x_p - \lambda_p) & a_p^T x_p > \lambda_p, \quad x_p \succeq x_p^{\min} \\ \infty & \text{otherwise} \end{cases}$$

where $a_p \in \mathbf{R}_{++}^n$ and $x_p^{\min} \in \mathbf{R}_{++}^n$ are given data. The vectors a_p and $x_p^{\min}$ are stored as the columns of the matrices `A` and `x_min`, respectively.

Give the optimal objective value and job traffic. Compare the optimal job traffic with the target job traffic.

17.10 *Optimal parimutuel betting.* In *parimutuel betting*, participants bet nonnegative amounts on each of n outcomes, exactly one of which will actually occur. (For example, the outcome can be which of n horses wins a race.) The total amount bet by all participants on all outcomes is called the *pool* or *tote*. The house takes a commission from the pool (typically around 20%), and the remaining pool is divided among those who bet on the outcome that occurs, in proportion to their bets on the outcome. This problem concerns the choice of the amount to bet on each outcome.

Let $x_i \geq 0$ denote the amount we bet on outcome i, so the total amount we bet on all outcomes is $\mathbf{1}^T x$. Let $a_i > 0$ denote the amount bet by all other participants on outcome i, so after the house commission, the remaining pool is $P = (1-c)(\mathbf{1}^T a + \mathbf{1}^T x)$, where $c \in (0, 1)$ is the house commission rate. Our *payoff* if outcome i occurs is then

$$p_i = \left(\frac{x_i}{x_i + a_i}\right) P.$$

The goal is to choose x, subject to $\mathbf{1}^T x = B$ (where B is the total amount to be bet, which is given), so as to maximize the expected utility

$$\sum_{i=1}^n \pi_i U(p_i),$$

where π_i is the probability that outcome i occurs, and U is a concave increasing utility function, with $U(0) = 0$. You can assume that a_i, π_i, c, B, and the function U are known.

(a) Explain how to find an optimal x using convex or quasiconvex optimization. If you use a change of variables, be sure to explain how your variables are related to x.

(b) Suggest a fast method for computing an optimal x. You can assume that U is strictly concave, and that scalar optimization problems involving U (such as evaluating the conjugate of $-U$) are easily and quickly solved.

Remarks.

- To carry out this betting strategy, you'd need to know a_i, and then be the last participant to place your bets (so that a_i don't subsequently change). You'd also need to know the probabilities π_i. These could be estimated using sophisticated machine learning techniques or insider information.
- The formulation above assumes that the total amount to bet (*i.e.*, B) is known. If it is not known, you could solve the problem above for a range of values of B and use the value of B that yields the largest optimal expected utility.

17.11 *Perturbing a Hamiltonian to maximize an energy gap.* A finite dimensional approximation of a quantum mechanical system is described by its Hamiltonian matrix $H \in \mathbf{S}^n$. We label the eigenvalues of H as $\lambda_1 \leq \cdots \leq \lambda_n$, with corresponding orthonormal eigenvectors $v_1, \ldots, v_n$. In this context the eigenvalues are called the energy levels of the system, and the eigenvectors are called the eigenstates. The eigenstate v_1 is called the ground state, and λ_1 is the ground energy. The energy gap (between the ground and next state) is $\eta = \lambda_2 - \lambda_1$.

By changing the environment (say, applying external fields), we can perturb a nominal Hamiltonian matrix to obtain the perturbed Hamiltonian, which has the form

$$H = H^{\text{nom}} + \sum_{i=1}^{k} x_i H_i.$$

Here $H^{\text{nom}} \in \mathbf{S}^n$ is the nominal (unperturbed) Hamiltonian, $x \in \mathbf{R}^k$ gives the strength or value of the perturbations, and $H_1, \ldots, H_k \in \mathbf{S}^n$ characterize the perturbations. We have limits for each perturbation, which we express as $|x_i| \leq 1$, $i = 1, \ldots, k$. The problem is to choose x to maximize the gap η of the perturbed Hamiltonian, subject to the constraint that the perturbed Hamiltonian H has the same ground state (up to scaling, of course) as the unperturbed Hamiltonian H^{nom}. The problem data are the nominal Hamiltonian matrix H^{nom} and the perturbation matrices $H_1, \ldots, H_k$.

(a) Explain how to formulate this as a convex or quasiconvex optimization problem. If you change variables, explain the change of variables clearly.

(b) Carry out the method of part (a) for the problem instance with data given in `hamiltonian_gap_data.*`. Give the optimal perturbations, and the energy gap for the nominal and perturbed systems. The data H_i are given as a cell array; `H{i}` gives H_i.

17.12 *Theory-applications split in a course.* A professor teaches an advanced course with 20 lectures, labeled $i = 1, \ldots, 20$. The course involves some interesting theoretical topics, and many practical applications of the theory. The professor must decide how to split each lecture between theory and applications. Let T_i and A_i denote the fraction of the ith lecture devoted to theory and applications, for $i = 1, \ldots, 20$. (We have $T_i \geq 0$, $A_i \geq 0$, and $T_i + A_i = 1$.)

A certain amount of theory has to be covered before the applications can be taught. We model this in a crude way as

$$A_1 + \cdots + A_i \leq \phi(T_1 + \cdots + T_i), \quad i = 1, \ldots, 20,$$

where $\phi : \mathbf{R} \to \mathbf{R}$ is a given nondecreasing function. We interpret $\phi(u)$ as the cumulative amount of applications that can be covered, when the cumulative amount of theory covered is u. We will use the simple form $\phi(u) = a(u - b)_+$, with $a, b > 0$, which means that no applications can be

covered until b lectures of the theory is covered; after that, each lecture of theory covered opens the possibility of covering a lectures on applications.

The theory-applications split affects the emotional state of students differently. We let s_i denote the emotional state of a student after lecture i, with $s_i = 0$ meaning neutral, $s_i > 0$ meaning happy, and $s_i < 0$ meaning unhappy. Careful studies have shown that s_i evolves via a linear recursion (dynamics)

$$s_i = (1-\theta)s_{i-1} + \theta(\alpha T_i + \beta A_i), \quad i = 1, \ldots, 20,$$

with $s_0 = 0$. Here α and β are parameters (naturally interpreted as how much the student likes or dislikes theory and applications, respectively), and $\theta \in [0, 1]$ gives the emotional volatility of the student (*i.e.*, how quickly he or she reacts to the content of recent lectures). The student's terminal emotional state is s_{20}.

Now consider a specific instance of the problem, with course material parameters $a = 2$, $b = 3$, and three groups of students, with emotional dynamics parameters given as follows.

	Group 1	Group 2	Group 3
θ	0.05	0.1	0.3
α	-0.1	0.8	-0.3
β	1.4	-0.3	0.7

Find (four different) theory-applications splits that maximize the terminal emotional state of the first group, the terminal emotional state of the second group, the terminal emotional state of the third group, and, finally, the minimum of the terminal emotional states of all three groups.

For each case, plot T_i and the emotional state s_i for the three groups, versus i. Report the numerical values of the terminal emotional states for each group, for each of the four theory-applications splits.

17.13 *Lyapunov analysis of a dynamical system.* We consider a discrete-time time-varying linear dynamical system with state $x_t \in \mathbf{R}^n$. The state propagates according to the linear recursion $x_{t+1} = A_t x_t$, for $t = 0, 1, \ldots$, where the matrices A_t are unknown but satisfy $A_t \in \mathcal{A} = \{A^{(1)}, \ldots, A^{(K)}\}$, where $A^{(1)}, \ldots, A^{(K)}$ are known. (In computer science, this would be called a non-deterministic linear automaton.) We call the sequence $x_0, x_1, \ldots$ a *trajectory* of the system. There are infinitely many trajectories, one for each sequence $A_0, A_1, \ldots$.

The *Lyapunov exponent* κ of the system is defined as

$$\kappa = \sup_{A_0, A_1, \ldots} \limsup_{t \to \infty} \|x_t\|_2^{1/t}.$$

(If you don't know what sup and lim sup mean, you can replace them with max and lim, respectively.) Roughly speaking, this means that all trajectories grow no faster than κ^t. When $\kappa < 1$, the system is called *exponentially stable*.

It is a hard problem to determine the Lyapunov exponent of the system, or whether the system is exponentially stable, given the data $A^{(1)}, \ldots, A^{(K)}$. In this problem we explore a powerful method for computing an upper bound on the Lyapunov exponent.

(a) Let $P \in \mathbf{S}^n_{++}$ and define $V(x) = x^T P x$. Suppose V satisfies

$$V(A^{(i)}x) \leq \gamma^2 V(x) \text{ for all } x \in \mathbf{R}^n, \ i = 1, \ldots, K.$$

Show that $\kappa \leq \gamma$. Thus γ is an upper bound on the Lyapunov exponent κ. (The function V is called a quadratic *Lyapunov function* for the system.)

(b) Explain how to use convex or quasiconvex optimization to find a matrix $P \in \mathbf{S}^n_{++}$ with the smallest value of γ, *i.e.*, with the best upper bound on κ. You must justify your formulation.

(c) Carry out the method of part (b) for the specific problem with data given in `lyap_exp_bound_data.m`. Report the best upper bound on κ, to a tolerance of 0.01. The data $A^{(i)}$ are given as a cell array; `A{i}` gives $A^{(i)}$.

(d) *Approximate worst-case trajectory simulation.* The quadratic Lyapunov function found in part (c) can be used to generate sequences of A_t that tend to result in large values of $\|x_t\|_2^{1/t}$. Start from a random vector x_0. At each t, generate x_{t+1} by choosing $A_t = A^{(i)}$ that maximizes $V(A^{(i)}x_t)$, where P is computed from part (c). Do this for 50 time steps, and generate 5 such trajectories. Plot $\|x_t\|_2^{1/t}$ and γ against t to verify that the bound you obtained in the previous part is valid. Report the lower bound on the Lyapunov exponent that the trajectories suggest.

17.14 *Optimal material blending.* A standard industrial operation is to blend or mix raw materials (typically fluids such as different grades of crude oil) to create blended materials or products. This problem addresses optimizing the blending operation. We produce n blended materials from m raw materials. Each raw and blended material is characterized by a vector that gives the concentration of each of q constituents (such as different octane hydrocarbons). Let $c_1, \ldots, c_m \in \mathbf{R}^q_+$ and $\tilde{c}_1, \ldots, \tilde{c}_n \in \mathbf{R}^q_+$ be the concentration vectors of the raw materials and the blended materials, respectively. We have $\mathbf{1}^T c_j = \mathbf{1}^T \tilde{c}_i = 1$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$. The raw material concentrations are given; the blended product concentrations must lie between some given bounds, $\tilde{c}_i^{\min} \preceq \tilde{c}_i \preceq \tilde{c}_i^{\max}$.

Each blended material is created by pumping raw materials (continuously) into a vat or container where they are mixed to produce the blended material (which continuously flows out of the mixing vat). Let $f_{ij} \geq 0$ denote the flow of raw material j (say, in kg/s) into the vat for product i, for $i = 1, \ldots, n$, $j = 1, \ldots, m$. These flows are limited by the total availability of each raw material: $\sum_{i=1}^n f_{ij} \leq F_j$, $j = 1, \ldots, m$, where $F_j > 0$ is the maximum total flow of raw material j available. Let $\tilde{f}_i \geq 0$ denote the flow rates of the blended materials. These also have limits: $\tilde{f}_i \leq \tilde{F}_i$, $i = 1, \ldots, n$.

The raw and blended material flows are related by the (mass conservation) equations

$$\sum_{j=1}^m f_{ij} c_j = \tilde{f}_i \tilde{c}_i, \quad i = 1, \ldots, n.$$

(The lefthand side is the vector of incoming constituent mass flows and the righthand side is the vector of outgoing constituent mass flows.)

Each raw and blended material has a (positive) price, p_j, $j = 1, \ldots, m$ (for the raw materials), and $\tilde{p}_i$, $i = 1, \ldots, n$ (for the blended materials). We pay for the raw materials, and get paid for the blended materials. The total profit for the blending process is

$$-\sum_{i=1}^n \sum_{j=1}^m f_{ij} p_j + \sum_{i=1}^n \tilde{f}_i \tilde{p}_i.$$

The goal is to choose the variables f_{ij}, $\tilde{f}_i$, and $\tilde{c}_i$ so as to maximize the profit, subject to the constraints. The problem data are c_j, $\tilde{c}_i^{\min}$, $\tilde{c}_i^{\max}$, F_j, $\tilde{F}_i$, p_j, and $\tilde{p}_j$.

(a) Explain how to solve this problem using convex or quasi-convex optimization. You must justify any change of variables or problem transformation, and explain how you recover the solution of the blending problem from the solution of your proposed problem.

(b) Carry out the method of part (a) on the problem instance given in `material_blending_data.*`. Report the optimal profit, and the associated values of f_{ij}, $\tilde{f}_i$, and $\tilde{c}_i$.

17.15 *Optimal evacuation planning.* We consider the problem of evacuating people from a dangerous area in a way that minimizes risk exposure. We model the area as a connected graph with n nodes and m edges; people can assemble or collect at the nodes, and travel between nodes (in either direction) over the edges. We let $q_t \in \mathbf{R}_+^n$ denote the vector of the numbers of people at the nodes, in time period t, for $t = 1, \ldots, T$, where T is the number of periods we consider. (We will consider the entries of q_t as real numbers, not integers.) The initial population distribution q_1 is given. The nodes have capacity constraints, given by $q_t \preceq Q$, where $Q \in \mathbf{R}_+^n$ is the vector of node capacities.

We use the incidence matrix $A \in \mathbf{R}^{n \times m}$ to describe the graph. We assign an arbitrary reference direction to each edge, and take

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

The population dynamics are given by $q_{t+1} = Af_t + q_t$, $t = 1, \ldots, T-1$ where $f_t \in \mathbf{R}^m$ is the vector of population movement (flow) across the edges, for $t = 1, \ldots, T-1$. A positive flow denotes movement in the direction of the edge; negative flow denotes population flow in the reverse direction. Each edge has a capacity, *i.e.*, $|f_t| \preceq F$, where $F \in \mathbf{R}_+^m$ is the vector of edge capacities, and $|f_t|$ denotes the elementwise absolute value of f_t.

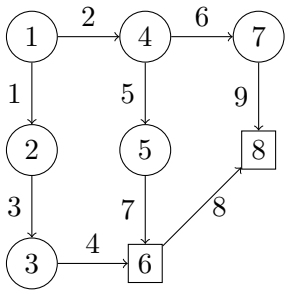
An *evacuation plan* is a sequence $q_1, q_2, \ldots, q_T$ and $f_1, f_2, \ldots, f_{T-1}$ obeying the constraints above. The goal is to find an evacuation plan that minimizes the total risk exposure, defined as

$$R_{\text{tot}} = \sum_{t=1}^{T} \left(r^T q_t + s^T q_t^2 \right) + \sum_{t=1}^{T-1} \left(\tilde{r}^T |f_t| + \tilde{s}^T f_t^2 \right),$$

where $r, s \in \mathbf{R}_+^n$ are given vectors of risk exposure coefficients associated with the nodes, and $\tilde{r}, \tilde{s} \in \mathbf{R}_+^m$ are given vectors of risk exposure coefficients associated with the edges. The notation q_t^2 and f_t^2 refers to elementwise squares of the vectors. Roughly speaking, the risk exposure is a quadratic function of the occupancy of a node, or the (absolute value of the) flow of people along an edge. The linear terms can be interpreted as the risk exposure per person; the quadratic terms can be interpreted as the additional risk associated with crowding.

A subset of nodes have zero risk ($r_i = s_i = 0$), and are designated as *safe nodes*. The population is considered *evacuated* at time t if $r^T q_t + s^T q_t^2 = 0$. The *evacuation time* t_{evac} of an evacuation plan is the smallest such t. We will assume that T is sufficiently large and that the total capacity of the safe nodes exceeds the total initial population, so evacuation is possible.

Use CVX* to find an optimal evacuation plan for the problem instance with data given in `opt_evac_data.*`. (We display the graph below, with safe nodes denoted as squares.)



Report the associated optimal risk exposure $R^\star_{\text{tot}}$. Plot the time period risk

$$R_t = r^T q_t + s^T q_t^2 + \tilde{r}^T |f_t| + \tilde{s}^T f_t^2$$

versus time. (For $t = T$, you can take the edge risk to be zero.) Plot the node occupancies q_t, and edge flows f_t versus time. Briefly comment on the results you see. Give the evacuation time t_{evac} (considering any $r^T q_t + s^T q_t^2 \leq 10^{-4}$ to be zero).

Hint. With CVXPY, use the ECOS solver with `p.solve(solver=cvxpy.ECOS)`.

17.16 *Ideal preference point.* A set of K choices for a decision maker is parametrized by a set of vectors $c^{(1)}, \ldots, c^{(K)} \in \mathbf{R}^n$. We will assume that the entries c_i of each choice are normalized to lie in the range $[0, 1]$. The *ideal preference point model* posits that there is an ideal choice vector c^{ideal} with entries in the range $[0, 1]$; when the decision maker is asked to choose between two candidate choices c and $\tilde{c}$, she will choose the one that is closest (in Euclidean norm) to her ideal point. Now suppose that the decision maker has chosen between all $K(K-1)/2$ pairs of given choices $c^{(1)}, \ldots, c^{(K)}$. The decisions are represented by a list of pairs of integers, where the pair (i, j) means that $c^{(i)}$ is chosen when given the choices $c^{(i)}, c^{(j)}$. You are given these vectors and the associated choices.

(a) How would you determine if the decision maker's choices are consistent with the ideal preference point model?

(b) Assuming they are consistent, how would you determine the bounding box of ideal choice vectors consistent with her decisions? (That is, how would you find the minimum and maximum values of c_i^{ideal}, for c^{ideal} consistent with being the ideal preference point.)

(c) Carry out the method of part (b) using the data given in `ideal_pref_point_data.*`. These files give the points $c^{(1)}, \ldots, c^{(K)}$ and the choices, and include the code for plotting the results. Report the width and the height of the bounding box and include your plot.

17.17 *Matrix equilibration.* We say that a matrix is ℓ_p *equilibrated* if each of its rows has the same ℓ_p norm, and each of its columns has the same ℓ_p norm. (The row and column ℓ_p norms are related by m, n, and p.) Suppose we are given a matrix $A \in \mathbf{R}^{m \times n}$. We seek diagonal invertible matrices $D \in \mathbf{R}^{m \times m}$ and $E \in \mathbf{R}^{n \times n}$ for which DAE is ℓ_p equilibrated.

(a) Explain how to find D and E using convex optimization. (Some matrices cannot be equilibrated. But you can assume that all entries of A are nonzero, which is enough to guarantee that it can be equilibrated.)

(b) Equilibrate the matrix A given in the file `matrix_equilibration_data.*`, with

$$m = 20, \quad n = 10, \quad p = 2.$$

Print the row ℓ_p norms and the column ℓ_p norms of the equilibrated matrix as vectors to check that each matches.

Hints.

- Work with the matrix B, with $B_{ij} = |A_{ij}|^p$.
- Consider the problem of minimizing $\sum_{i=1}^m \sum_{j=1}^n B_{ij} e^{u_i + v_j}$ subject to $\mathbf{1}^T u = 0$, $\mathbf{1}^T v = 0$. (Several variations on this idea will work.)
- We have found that expressing the terms in the objective as $e^{\log B_{ij} + u_i + v_j}$ leads to fewer numerical problems.

17.18 *Approximations of the PSD cone.* A symmetric matrix is positive semidefinite if and only if all its principal minors are nonnegative. Here we consider approximations of the positive-semidefinite cone produced by partially relaxing this condition.

Denote by $K_{1,n}$ the cone of matrices whose 1×1 principal minors (*i.e.*, diagonal elements) are nonnegative, so that

$$K_{1,n} = \{X \in \mathbf{S}^n \mid X_{ii} \geq 0 \text{ for all } i\}.$$

Similarly, denote by $K_{2,n}$ the cone of matrices whose 1×1 and 2×2 principal minors are nonnegative:

$$K_{2,n} = \left\{ X \in \mathbf{S}^n \;\middle|\; \begin{bmatrix} X_{ii} & X_{ij} \\ X_{ij} & X_{jj} \end{bmatrix} \succeq 0, \text{ for all } i \neq j \right\},$$

i.e., the cone of symmetric matrices with positive semidefinite 2×2 principal submatrices. These two cones are convex (and in fact, proper), and satisfy the relation:

$$K_{1,n}^* \subseteq K_{2,n}^* \subseteq \mathbf{S}_+^n \subseteq K_{2,n} \subseteq K_{1,n},$$

where $K_{1,n}^*$ and $K_{2,n}^*$ are the dual cones of $K_{1,n}$ and $K_{2,n}$, respectively. (The last two inclusions are immediate, and the first two inclusions follow from the second bullet on page 53 of the text.)

(a) Give an explicit characterization of $K_{1,n}^*$.

(b) Give an explicit characterization of $K_{2,n}^*$.
Hint: You can use the fact that if $K = K_1 \cap \cdots \cap K_m$, then $K^* = K_1^* + \cdots + K_m^*$.

(c) Consider the problem

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}\, CX \\ \text{subject to} & \mathbf{tr}\, AX = b \\ & X \in K \end{array}$$

with variable $X \in \mathbf{S}^n$. The problem parameters are $C \in \mathbf{S}^n$, $A \in \mathbf{S}^n$, $b \in \mathbf{R}$, and the cone $K \subseteq \mathbf{S}^n$. Using the data in `psd_cone_approx_data.*`, solve this problem five times, each time replacing K with one of the five cones $K_{1,n}$, $K_{2,n}$, $\mathbf{S}_+^n$, $K_{2,n}^*$, and $K_{1,n}^*$. Report the five different optimal values you obtain.
Note: Python users who run into numerical difficulties might want to use the SCS solver by using `prob.solve(solver=cvxpy.SCS)`.

Note: For parts (a) and (b), the shorter and clearer your description is, the more points you will receive. At the very least, it should be possible to implement your description in CVX.*.