EE364a Homework 3 solutions

3.42 *Approximation width.* Let $f_0, \dots, f_n : \mathbf{R} \to \mathbf{R}$ be given continuous functions. We consider the problem of approximating f_0 as a linear combination of $f_1, \dots, f_n$. For $x \in \mathbf{R}^n$, we say that $f = x_1 f_1 + \cdots + x_n f_n$ approximates f_0 with tolerance $\epsilon > 0$ over the interval $[0, T]$ if $|f(t) - f_0(t)| \le \epsilon$ for $0 \le t \le T$. Now we choose a fixed tolerance $\epsilon > 0$ and define the *approximation width* as the largest T such that f approximates f_0 over the interval $[0, T]$:

$$W(x) = \sup\{T \mid |x_1 f_1(t) + \cdots + x_n f_n(t) - f_0(t)| \le \epsilon \text{ for } 0 \le t \le T\}.$$

Show that W is quasiconcave.

Solution. To show that W is quasiconcave we show that the sets $\{x \mid W(x) \ge \alpha\}$ are convex for all α. We have $W(x) \ge \alpha$ if and only if

$$-\epsilon \le x_1 f_1(t) + \cdots + x_n f_n(t) - f_0(t) \le \epsilon$$

for all $t \in [0, \alpha)$. Therefore the set $\{x \mid W(x) \ge \alpha\}$ is an intersection of infinitely many halfspaces (two for each t), hence a convex set.

3.54 *Log-concavity of Gaussian cumulative distribution function.* The cumulative distribution function of a Gaussian random variable,

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2}\, dt,$$

is log-concave. This follows from the general result that the convolution of two log-concave functions is log-concave. In this problem we guide you through a simple self-contained proof that f is log-concave. Recall that f is log-concave if and only if $f''(x) f(x) \le f'(x)^2$ for all x.

(a) Verify that $f''(x) f(x) \le f'(x)^2$ for $x \ge 0$. That leaves us the hard part, which is to show the inequality for $x < 0$.

(b) Verify that for any t and x we have $t^2/2 \ge -x^2/2 + xt$.

(c) Using part (b) show that $e^{-t^2/2} \le e^{x^2/2 - xt}$. Conclude that

$$\int_{-\infty}^{x} e^{-t^2/2}\, dt \le e^{x^2/2} \int_{-\infty}^{x} e^{-xt}\, dt.$$

(d) Use part (c) to verify that $f''(x) f(x) \le f'(x)^2$ for $x \le 0$.

Solution. The derivatives of f are

$$f'(x) = e^{-x^2/2}/\sqrt{2\pi}, \qquad f''(x) = -x e^{-x^2/2}/\sqrt{2\pi}.$$

(a) $f''(x) \leq 0$ for $x \geq 0$.

(b) Since $t^2/2$ is convex we have

$$t^2/2 \geq x^2/2 + x(t-x) = xt - x^2/2.$$

This is the general inequality

$$g(t) \geq g(x) + g'(x)(t-x),$$

which holds for any differentiable convex function, applied to $g(t) = t^2/2$. Another (easier?) way to establish $t^2/2 \leq -x^2/2 + xt$ is to note that

$$t^2/2 + x^2/2 - xt = (1/2)(x-t)^2 \geq 0.$$

Now just move $x^2/2 - xt$ to the other side.

(c) Take exponentials and integrate.

(d) This basic inequality reduces to

$$-xe^{-x^2/2} \int_{-\infty}^{x} e^{-t^2/2}\, dt \leq e^{-x^2}$$

i.e.,

$$\int_{-\infty}^{x} e^{-t^2/2}\, dt \leq \frac{e^{-x^2/2}}{-x}.$$

This follows from part (c) because

$$\int_{-\infty}^{x} e^{-xt}\, dt = \frac{e^{-x^2}}{-x}.$$

3.57 Show that the function $f(X) = X^{-1}$ is matrix convex on $\mathbf{S}^n_{++}$.

Solution. We must show that for arbitrary $v \in \mathbf{R}^n$, the function

$$g(X) = v^T X^{-1} v.$$

is convex in X on $\mathbf{S}^n_{++}$. This follows from example 3.4.

4.1 Consider the optimization problem

$$\begin{array}{ll} \text{minimize} & f_0(x_1, x_2) \\ \text{subject to} & 2x_1 + x_2 \geq 1 \\ & x_1 + 3x_2 \geq 1 \\ & x_1 \geq 0, \quad x_2 \geq 0. \end{array}$$

Make a sketch of the feasible set. For each of the following objective functions, give the optimal set and the optimal value.

(a) $f_0(x_1, x_2) = x_1 + x_2$.

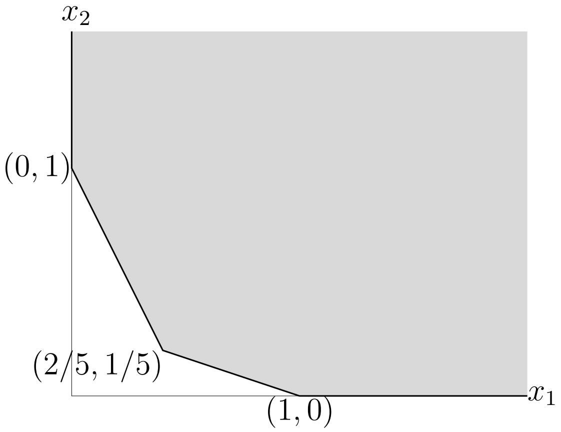
(b) $f_0(x_1, x_2) = -x_1 - x_2$.

(c) $f_0(x_1, x_2) = x_1$.

(d) $f_0(x_1, x_2) = \max\{x_1, x_2\}$.

(e) $f_0(x_1, x_2) = x_1^2 + 9x_2^2$.

Solution. The feasible set is shown in the figure.



(a) $x^\star = (2/5, 1/5)$.

(b) Unbounded below.

(c) $X_{\rm opt} = \{(0, x_2) \mid x_2 \geq 1\}$.

(d) $x^\star = (1/3, 1/3)$.

(e) $x^\star = (1/2, 1/6)$. This is optimal because it satisfies $2x_1 + x_2 = 7/6 > 1$, $x_1 + 3x_2 = 1$, and

$$\nabla f_0(x^\star) = (1, 3)$$

is perpendicular to the line $x_1 + 3x_2 = 1$.

4.4 [P. Parrilo] *Symmetries and convex optimization.* Suppose $\mathcal{G} = \{Q_1, \ldots, Q_k\} \subseteq \mathbf{R}^{n \times n}$ is a group, *i.e.*, closed under products and inverse. We say that the function $f : \mathbf{R}^n \to \mathbf{R}$ is $\mathcal{G}$*-invariant*, or *symmetric with respect to* $\mathcal{G}$, if $f(Q_i x) = f(x)$ holds for all x and $i = 1, \ldots, k$. We define $\overline{x} = (1/k) \sum_{i=1}^k Q_i x$, which is the average of x over its $\mathcal{G}$-orbit. We define the *fixed subspace* of $\mathcal{G}$ as

$$\mathcal{F} = \{x \mid Q_i x = x, \ i = 1, \ldots, k\}.$$

(a) Show that for any $x \in \mathbf{R}^n$, we have $\overline{x} \in \mathcal{F}$.

(b) Show that if $f : \mathbf{R}^n \to \mathbf{R}$ is convex and $\mathcal{G}$-invariant, then $f(\overline{x}) \leq f(x)$.

(c) We say the optimization problem

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

is $\mathcal{G}$-invariant if the objective f_0 is $\mathcal{G}$-invariant, and the feasible set is $\mathcal{G}$-invariant, which means

$$f_1(x) \leq 0, \ldots, f_m(x) \leq 0 \implies f_1(Q_i x) \leq 0, \ldots, f_m(Q_i x) \leq 0,$$

for $i = 1, \ldots, k$. Show that if the problem is convex and $\mathcal{G}$-invariant, and there exists an optimal point, then there exists an optimal point in $\mathcal{F}$. In other words, we can adjoin the equality constraints $x \in \mathcal{F}$ to the problem, without loss of generality.

(d) As an example, suppose f is convex and symmetric, *i.e.*, $f(Px) = f(x)$ for every permutation P. Show that if f has a minimizer, then it has a minimizer of the form $\alpha \mathbf{1}$. (This means to minimize f over $x \in \mathbf{R}^n$, we can just as well minimize $f(t\mathbf{1})$ over $t \in \mathbf{R}$.)

Solution.

(a) We first observe that when you multiply each Q_i by some fixed Q_j, you get a permutation of the Q_i's:

$$Q_j Q_i = Q_{\sigma(i)}, \quad i = 1, \ldots, k,$$

where σ is a permutation. This is a basic result in group theory, but it's easy enough for us to show it. First we note that by closedness, each $Q_j Q_i$ is equal to some Q_s. Now suppose that $Q_j Q_i = Q_k Q_i = Q_s$. Multiplying by Q_i^{-1} on the right, we see that $Q_j = Q_k$. Thus the mapping from the index i to the index s is one-to-one, *i.e.*, a permutation.

Now we have

$$Q_j \overline{x} = (1/k) \sum_{i=1}^k Q_j Q_i x = (1/k) \sum_{i=1}^k Q_{\sigma(i)} x = (1/k) \sum_{i=1}^k Q_i x = \overline{x}.$$

This holds for j, so we have $\overline{x} \in \mathcal{F}$.

(b) Using convexity and invariance of f,

$$f(\overline{x}) \leq (1/k) \sum_{i=1}^k f(Q_i x) = (1/k) \sum_{i=1}^k f(x) = f(x).$$

(c) Suppose $x^\star$ is an optimal solution. Then $\overline{x^\star}$ is feasible, with

$$\begin{array}{rcl} f_0(\overline{x^\star}) & = & f_0((1/k)\sum_{i=1}^k Q_i x) \\ & \leq & (1/k)\sum_{i=1}^k f_0(Q_i x) \\ & = & f_0(x^\star). \end{array}$$

Therefore $\overline{x^\star}$ is also optimal.

(d) Suppose $x^\star$ is a minimizer of f. Let $\overline{x} = (1/n!)\sum_P Px^\star$, where the sum is over all permutations. Since $\overline{x}$ is invariant under any permutation, we conclude that $\overline{x} = \alpha\mathbf{1}$ for some $\alpha \in \mathbf{R}$. By Jensen's inequality we have

$$f(\overline{x}) \leq (1/n!)\sum_P f(Px^\star) = f(x^\star),$$

which shows that $\overline{x}$ is also a minimizer.

4.8 *Some simple LPs.* Give an explicit solution of each of the following LPs.

(a) *Minimizing a linear function over an affine set.*

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

Solution. We distinguish three possibilities.

- The problem is infeasible ($b \notin \mathcal{R}(A)$). The optimal value is ∞.
- The problem is feasible, and c is orthogonal to the nullspace of A. We can decompose c as
$$c = A^T\lambda + \hat{c}, \qquad A\hat{c} = 0.$$
($\hat{c}$ is the component in the nullspace of A; $A^T\lambda$ is orthogonal to the nullspace.) If $\hat{c} = 0$, then on the feasible set the objective function reduces to a constant:
$$c^T x = \lambda^T Ax + \hat{c}^T x = \lambda^T b.$$
The optimal value is $\lambda^T b$. All feasible solutions are optimal.
- The problem is feasible, and c is not in the range of A^T ($\hat{c} \neq 0$). The problem is unbounded ($p^\star = -\infty$). To verify this, note that $x = x_0 - t\hat{c}$ is feasible for all t; as t goes to infinity, the objective value decreases unboundedly.

In summary,

$$p^\star = \begin{cases} +\infty & b \notin \mathcal{R}(A) \\ \lambda^T b & c = A^T\lambda \text{ for some } \lambda \\ -\infty & \text{otherwise.} \end{cases}$$

(b) *Minimizing a linear function over a halfspace.*

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

where $a \neq 0$.

Solution. This problem is always feasible. The vector c can be decomposed into a component parallel to a and a component orthogonal to a:

$$c = a\lambda + \hat{c},$$

with $a^T \hat{c} = 0$.

- If $\lambda > 0$, the problem is unbounded below. Choose $x = -ta$, and let t go to infinity:
$$c^T x = -tc^T a = -t\lambda a^T a \to -\infty$$
and
$$a^T x - b = -ta^T a - b \leq 0$$
for large t, so x is feasible for large t. Intuitively, by going very far in the direction $-a$, we find feasible points with arbitrarily negative objective values.
- If $\hat{c} \neq 0$, the problem is unbounded below. Choose $x = ba - t\hat{c}$ and let t go to infinity.
- If $c = a\lambda$ for some $\lambda \leq 0$, the optimal value is $c^T ab = \lambda b$.

In summary, the optimal value is

$$p^\star = \begin{cases} \lambda b & c = a\lambda \text{ for some } \lambda \leq 0 \\ -\infty & \text{otherwise.} \end{cases}$$

(c) *Minimizing a linear function over a rectangle.*

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

where l and u satisfy $l \preceq u$.

Solution. The objective and the constraints are separable: The objective is a sum of terms $c_i x_i$, each dependent on one variable only; each constraint depends on only one variable. We can therefore solve the problem by minimizing over each component of x independently. The optimal $x_i^\star$ minimizes $c_i x_i$ subject to the constraint $l_i \leq x_i \leq u_i$. If $c_i > 0$, then $x_i^\star = l_i$; if $c_i < 0$, then $x_i^\star = u_i$; if $c_i = 0$, then any x_i in the interval $[l_i, u_i]$ is optimal. Therefore, the optimal value of the problem is

$$p^\star = l^T c^+ + u^T c^-,$$

where $c_i^+ = \max\{c_i, 0\}$ and $c_i^- = \max\{-c_i, 0\}$.

(d) *Minimizing a linear function over the probability simplex.*

$$\begin{array}{ll} \text{minimize} & c^T x \\ \text{subject to} & \mathbf{1}^T x = 1, \quad x \succeq 0. \end{array}$$

What happens if the equality constraint is replaced by an inequality $\mathbf{1}^T x \leq 1$?

We can interpret this LP as a simple portfolio optimization problem. The vector x represents the allocation of our total budget over different assets, with x_i the fraction invested in asset i. The return of each investment is fixed and given by $-c_i$, so our total return (which we want to maximize) is $-c^T x$. If we replace the budget constraint $\mathbf{1}^T x = 1$ with an inequality $\mathbf{1}^T x \leq 1$, we have the option of not investing a portion of the total budget.

Solution. Suppose the components of c are sorted in increasing order with

$$c_1 = c_2 = \cdots = c_k < c_{k+1} \leq \cdots \leq c_n.$$

We have

$$c^T x \geq c_1(\mathbf{1}^T x) = c_{\min}$$

for all feasible x, with equality if and only if

$$x_1 + \cdots + x_k = 1, \qquad x_1 \geq 0, \ldots, x_k \geq 0, \qquad x_{k+1} = \cdots = x_n = 0.$$

We conclude that the optimal value is $p^\star = c_1 = c_{\min}$. In the investment interpretation this choice is quite obvious. If the returns are fixed and known, we invest our total budget in the investment with the highest return.

If we replace the equality with an inequality, the optimal value is equal to

$$p^\star = \min\{0, c_{\min}\}.$$

(If $c_{\min} \leq 0$, we make the same choice for x as above. Otherwise, we choose $x = 0$.)

(e) *Minimizing a linear function over a unit box with a total budget constraint.*

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

where α is an integer between 0 and n. What happens if α is not an integer (but satisfies $0 \leq \alpha \leq n$)? What if we change the equality to an inequality $\mathbf{1}^T x \leq \alpha$?

Solution. We first consider the case of integer α. Suppose

$$c_1 \leq \cdots \leq c_{i-1} < c_i = \cdots = c_\alpha = \cdots = c_k < c_{k+1} \leq \cdots \leq c_n.$$

The optimal value is

$$c_1 + c_2 + \cdots + c_\alpha$$

i.e., the sum of the smallest α elements of c. x is optimal if and only if

$$x_1 = \cdots = x_{i-1} = 1, \qquad x_i + \cdots + x_k = \alpha - i + 1, \qquad x_{k+1} = \cdots = x_n = 0.$$

If α is not an integer, the optimal value is

$$p^\star = c_1 + c_2 + \cdots + c_{\lfloor \alpha \rfloor} + c_{1+\lfloor \alpha \rfloor}(\alpha - \lfloor \alpha \rfloor).$$

In the case of an inequality constraint $\mathbf{1}^T x \leq \alpha$, with α an integer between 0 and n, the optimal value is the sum of the α smallest nonpositive coefficients of c.

4.17 *Optimal activity levels.* We consider the selection of n nonnegative activity levels, denoted $x_1, \ldots, x_n$. These activities consume m resources, which are limited. Activity j consumes $A_{ij}x_j$ of resource i, where A_{ij} are given. The total resource consumption is additive, so the total of resource i consumed is $c_i = \sum_{j=1}^n A_{ij}x_j$. (Ordinarily we have $A_{ij} \geq 0$, *i.e.*, activity j consumes resource i. But we allow the possibility that $A_{ij} < 0$, which means that activity j actually *generates* resource i as a by-product.) Each resource consumption is limited: we must have $c_i \leq c_i^{\max}$, where $c_i^{\max}$ are given. Each activity generates revenue, which is a piecewise-linear concave function of the activity level:

$$r_j(x_j) = \begin{cases} p_j x_j & 0 \leq x_j \leq q_j \\ p_j q_j + p_j^{\text{disc}}(x_j - q_j) & x_j \geq q_j. \end{cases}$$

Here $p_j > 0$ is the basic price, $q_j > 0$ is the quantity discount level, and p_j^{disc} is the quantity discount price, for (the product of) activity j. (We have $0 < p_j^{\text{disc}} < p_j$.) The total revenue is the sum of the revenues associated with each activity, *i.e.*, $\sum_{j=1}^n r_j(x_j)$. The goal is to choose activity levels that maximize the total revenue while respecting the resource limits. Show how to formulate this problem as an LP.

Solution. The basic problem can be expressed as

$$\begin{array}{ll} \text{maximize} & \sum_{j=1}^n r_j(x_j) \\ \text{subject to} & x \succeq 0 \\ & Ax \preceq c^{\max}. \end{array}$$

This is a convex optimization problem since the objective is concave and the constraints are a set of linear inequalities. To transform it to an equivalent LP, we first express the revenue functions as

$$r_j(x_j) = \min\{p_j x_j, p_j q_j + p_j^{\text{disc}}(x_j - q_j)\},$$

which holds since r_j is concave. It follows that $r_j(x_j) \geq u_j$ if and only if

$$p_j x_j \geq u_j, \qquad p_j q_j + p_j^{\text{disc}}(x_j - q_j) \geq u_j.$$

We can form an LP as

$$\begin{array}{ll}\text{maximize} & \mathbf{1}^T u \\ \text{subject to} & x \succeq 0 \\ & Ax \preceq c^{\max} \\ & p_j x_j \geq u_j, \quad p_j q_j + p_j^{\text{disc}}(x_j - q_j) \geq u_j, \quad j = 1, \ldots, n,\end{array}$$

with variables x and u.

To show that this LP is equivalent to the original problem, let us fix x. The last set of constraints in the LP ensure that $u_i \leq r_i(x)$, so we conclude that for every feasible x, u in the LP, the LP objective is less than or equal to the total revenue. On the other hand, we can always take $u_i = r_i(x)$, in which case the two objectives are equal.

Solutions to additional exercises

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

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

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.

Solution. The following Matlab/CVX code solves the problem. (Here we write the problem in a form close to its original statement, and let CVX do the work of reformulating it as an LP!)

```
A=[ 1 2 0 1;
    0 0 3 1;
    0 3 1 1;
    2 1 2 5;
    1 0 3 2];

cmax=[100;100;100;100;100];
p=[3;2;7;6];
pdisc=[2;1;4;2];
q=[4; 10 ;5; 10];

cvx_begin
  variable x(4)
  maximize( sum(min(p.*x,p.*q+pdisc.*(x-q))) )
  subject to
    x >= 0;
    A*x <= cmax
cvx_end

x
r=min(p.*x,p.*q+pdisc.*(x-q))
```

```
totr=sum(r)
avgPrice=r./x
```

The result of the code is

```
x =

    4.0000
   22.5000
   31.0000
    1.5000


r =

   12.0000
   32.5000
  139.0000
    9.0000


totr =

  192.5000


avgPrice =

    3.0000
    1.4444
    4.4839
    6.0000
```

We notice that the 3rd activity level is the highest and is also the one with the highest basic price. Since it also has a high discounted price its activity level is higher than the discount quantity level and it produces the highest contribution to the total revenue. The 4th activity has a discounted price which is substantially lower then the basic price and its activity is therefore lower that the discount quantity level. Moreover it require the use of a lot of resources and therefore its activity level is low.

2. *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`

Solution.

(a) The lefthand side is correctly identified as convex, but equality constraints are only valid with affine left and right hand sides. Since the norm of a vector is zero if and only if the vector is zero, we can express the constraint as `x+2*y==0; x-y==0`, or simply `x==0; y==0`.

(b) The problem is that `square()` can only accept affine arguments, because it is convex, but not increasing. To correct this use `square_pos()` instead:

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

We can also reformulate this constraint by introducing an additional variable.

```
variable t
square( x+y ) <= t
square( t ) <= x - y
```

Note that, in general, decomposing the objective by introducing new variables doesn't need to work. It works in this case because the outer `square` function is convex and monotonic over $\mathbf{R}_+$.

Alternatively, we can rewrite the constraint as

```
( x + y )^4 <= x - y
```

(c) $1/x$ isn't convex, unless you restrict the domain to $\mathbf{R}_{++}$. We can write this one as inv_pos(x)+inv_pos(y)<=1. The inv_pos function has domain $\mathbf{R}_{++}$ so the constraints $x > 0$, $y > 0$ are (implicitly) included.

(d) The problem is that norm() can only accept affine argument since it is convex but not increasing. One way to correct this is to introduce new variables u and v:

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

Decomposing the objective by introducing new variables work here because norm is convex and monotonic over $\mathbf{R}^2_+$, and in particular over $(1, \infty) \times (2, \infty)$.

(e) xy isn't concave, so this isn't going to work as stated. But we can express the constraint as x>=inv_pos(y). (You can switch around x and y here.) Another solution is to write the constraint as geomean([x,y])>=1. We can also give an LMI representation:

```
[ x 1; 1 y ] == semidefinite(2)
```

(f) This fails when we attempt to divide a convex function by a concave one. We can write this as

```
quad_over_lin(x+y,sqrt(y)) <= x-y+5
```

This works because quad_over_lin is monotone descreasing in the second argument, so it can accept a concave function here, and sqrt is concave.

(g) The function $x^3 + y^3$ is convex for $x \geq 0$, $y \geq 0$. But x^3 isn't convex for $x < 0$, so cvx is going to reject this statement. One way to rewrite this constraint is

```
quad_pos_over_lin(square(x),x) + quad_pos_over_lin(square(y),y) <= 1
```

This works because quad_pos_over_lin is convex and increasing in its first argument, hence accepts a convex function in its first argument. (The function quad_over_lin, however, is not increasing in its first argument, and so won't work.)

Alternatively, and more simply, we can rewrite the constraint as

```
pow_pos(x,3) + pow_pos(y,3) <= 1
```

(h) The problem here is that xy isn't concave, which causes cvx to reject the statement. To correct this, notice that

$$\sqrt{xy - z^2} = \sqrt{y(x - z^2/y)},$$

so we can reformulate the constraint as

```
x+z <= 1+geomean([x-quad_over_lin(z,y),y])
```

This works, since `geomean` is concave and nondecreasing in each argument. It therefore accepts a concave function in its first argument.

We can check our reformulations by writing the following feasibility problem in `cvx` (which is obviously infeasible)

```
cvx_begin
    variables x y u v z
    x == 0;
    y == 0;
    ( x + y )^4 <= x - y;
    inv_pos(x) + inv_pos(y)  <= 1;
    norm( [ u ; v ] ) <= 3*x + y;
    max( x , 1 ) <= u;
    max( y , 2 ) <= v;
    x >= inv_pos(y);
    x >= 0;
    y >= 0;
    quad_over_lin(x + y , sqrt(y)) <= x - y + 5;
    pow_pos(x,3) + pow_pos(y,3) <= 1;
    x+z <= 1+geomean([x-quad_over_lin(z,y),y])
cvx_end
```

3. *The illumination problem.* This exercise concerns the illumination problem described in lecture 1 (pages 9–11). We'll take $I_{\text{des}} = 1$ and $p_{\max} = 1$, so the problem is

$$\begin{array}{ll} \text{minimize} & f_0(p) = \max_{k=1,\ldots,n} |\log(a_k^T p)| \\ \text{subject to} & 0 \leq p_j \leq 1, \quad j = 1, \ldots, m, \end{array} \tag{1}$$

with variable $p \in \mathbf{R}^n$. You will compute several approximate solutions, and compare the results to the exact solution, for a specific problem instance.

As mentioned in the lecture, the problem is equivalent to

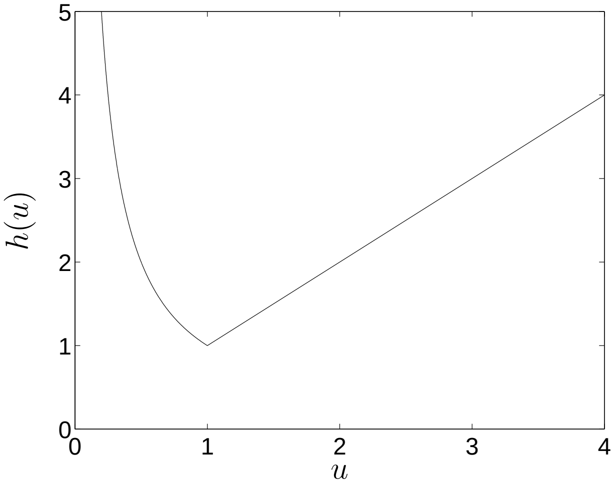
$$\begin{array}{ll} \text{minimize} & \max_{k=1,\ldots,n} h(a_k^T p) \\ \text{subject to} & 0 \leq p_j \leq 1, \quad j = 1, \ldots, m, \end{array} \tag{2}$$

where $h(u) = \max\{u, 1/u\}$ for $u > 0$. The function h, shown in the figure below, is nonlinear, nondifferentiable, and convex. To see the equivalence between (1) and (2), we note that

$$\begin{aligned} f_0(p) &= \max_{k=1,\ldots,n} |\log(a_k^T p)| \\ &= \max_{k=1,\ldots,n} \max\{\log(a_k^T p), \log(1/a_k^T p)\} \end{aligned}$$

$$
\begin{aligned}
&= \log \max_{k=1,\ldots,n} \max\{a_k^T p, 1/a_k^T p\} \\
&= \log \max_{k=1,\ldots,n} h(a_k^T p),
\end{aligned}
$$

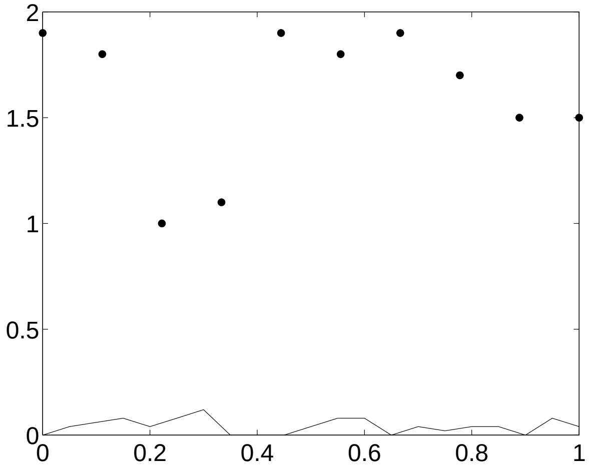
and since the logarithm is a monotonically increasing function, minimizing f_0 is equivalent to minimizing $\max_{k=1,\ldots,n} h(a_k^T p)$.



The problem instance. The specific problem data are for the geometry shown below, using the formula

$$a_{kj} = r_{kj}^{-2} \max\{\cos\theta_{kj}, 0\}$$

from the lecture. There are 10 lamps ($m = 10$) and 20 patches ($n = 20$). We take $I_{\text{des}} = 1$ and $p_{\max} = 1$. The problem data are given in the file `illum_data.m` on the course website. Running this script will construct the matrix A (which has rows a_k^T), and plot the lamp/patch geometry as shown below.



Equal lamp powers. Take $p_j = \gamma$ for $j = 1, \ldots, m$. Plot $f_0(p)$ versus γ over the interval $[0, 1]$. Graphically determine the optimal value of γ, and the associated objective value.

You can evaluate the objective function $f_0(p)$ in Matlab as `max(abs(log(A*p)))`.

Least-squares with saturation. Solve the least-squares problem

$$\text{minimize} \quad \textstyle\sum_{k=1}^{n} (a_k^T p - 1)^2 = \|Ap - \mathbf{1}\|_2^2.$$

If the solution has negative values for some p_i, set them to zero; if some values are greater than 1, set them to 1. Give the resulting value of $f_0(p)$.

Least-squares solutions can be computed using the Matlab backslash operator: `A\b` returns the solution of the least-squares problem

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

Regularized least-squares. Solve the regularized least-squares problem

$$\text{minimize} \quad \textstyle\sum_{k=1}^{n} (a_k^T p - 1)^2 + \rho \sum_{j=1}^{m} (p_j - 0.5)^2 = \|Ap - \mathbf{1}\|_2^2 + \rho \|p - (1/2)\mathbf{1}\|_2^2,$$

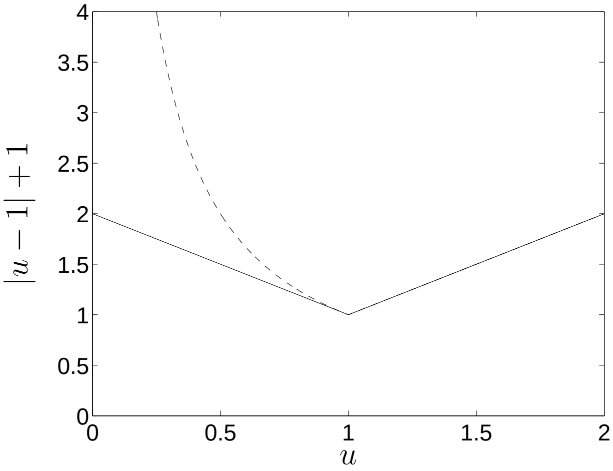
where $\rho > 0$ is a parameter. Increase ρ until all coefficients of p are in the interval $[0, 1]$. Give the resulting value of $f_0(p)$.

You can use the backslash operator in Matlab to solve the regularized least-squares problem.

Chebyshev approximation. Solve the problem

$$\begin{array}{ll} \text{minimize} & \max_{k=1,\ldots,n} |a_k^T p - 1| = \|Ap - \mathbf{1}\|_\infty \\ \text{subject to} & 0 \leq p_j \leq 1, \quad j = 1, \ldots, m. \end{array}$$

We can think of this problem as obtained by approximating the nonlinear function $h(u)$ by a piecewise-linear function $|u - 1| + 1$. As shown in the figure below, this is a good approximation around $u = 1$.



You can solve the Chebyshev approximation problem using `cvx`. The (convex) function $\|Ap-\mathbf{1}\|_\infty$ can be expressed in `cvx` as `norm(A*p-ones(n,1),inf)`. Give the resulting value of $f_0(p)$.

Exact solution. Finally, use `cvx` to solve

$$\begin{array}{ll} \text{minimize} & \max_{k=1,\ldots,n} \max(a_k^T p, 1/a_k^T p) \\ \text{subject to} & 0 \leq p_j \leq 1, \quad j=1,\ldots,m \end{array}$$

exactly. You may find the `inv_pos()` function useful. Give the resulting (optimal) value of $f_0(p)$.

Solution: The following Matlab script finds the approximate solutions using the heuristic methods proposed, as well as the exact solution.

```
% illum_sol: finds approximate and exact solutions of
%            the illumination problem

clear all;

% load input data
illum_data;

% heuristic method 1: equal lamp powers
% ----------------------------------------
nopts=1000;
p = logspace(-3,0,nopts);
f = zeros(size(p));
for k=1:nopts
   f(k) = max(abs(log(A*p(k)*ones(m,1))));
```

```
end;
[val_equal,imin] = min(f);
p_equal = p(imin)*ones(m,1);

% heuristic method 2: least-squares with saturation
% -------------------------------------------------
p_ls_sat = A\ones(n,1);
p_ls_sat = max(p_ls_sat,0);             % rounding negative p_i to 0
p_ls_sat = min(p_ls_sat,1);             % rounding p_i > 1 to 1
val_ls_sat = max(abs(log(A*p_ls_sat)));

% heuristic method 3: regularized least-squares
% ---------------------------------------------
rhos = linspace(1e-3,1,nopts);
crit = [];
for j=1:nopts
    p = [A; sqrt(rhos(j))*eye(m)]\[ones(n,1); sqrt(rhos(j))*0.5*ones(m,1)];
    crit = [ crit norm(p-0.5,inf) ];
end
idx = find(crit <= 0.5);
rho = rhos(idx(1));                     % smallest rho s.t. p is in [0,1]
p_ls_reg = [A; sqrt(rho)*eye(m)]\[ones(n,1); sqrt(rho)*0.5*ones(m,1)];
val_ls_reg = max(abs(log(A*p_ls_reg)));

% heuristic method 4: chebyshev approximation
% -------------------------------------------
cvx_begin
   variable p_cheb(m)
   minimize(norm(A*p_cheb-1, inf))
   subject to
      p_cheb >= 0
      p_cheb <= 1
cvx_end
val_cheb = max(abs(log(A*p_cheb)));

% exact solution:
% ---------------
cvx_begin
   variable p_exact(m)
   minimize(max([A*p_exact; inv_pos(A*p_exact)]))
   subject to
       p_exact >= 0
       p_exact <= 1
cvx_end
val_exact = max(abs(log(A*p_exact)));

% Results
% -------
[p_equal p_ls_sat p_ls_reg p_cheb p_exact]
[val_equal val_ls_sat val_ls_reg val_cheb val_exact]
```

The results are summarized in the following table.

	method 1	method 2	method 3	method 4	exact
$f_0(p)$	0.4693	0.8628	0.4439	0.4198	0.3575
p_1	0.3448	1	0.5004	1	1
p_2	0.3448	0	0.4777	0.1165	0.2023
p_3	0.3448	1	0.0833	0	0
p_4	0.3448	0	0.0002	0	0
p_5	0.3448	0	0.4561	1	1
p_6	0.3448	1	0.4354	0	0
p_7	0.3448	0	0.4597	1	1
p_8	0.3448	1	0.4307	0.0249	0.1882
p_9	0.3448	0	0.4034	0	0
p_{10}	0.3448	1	0.4526	1	1