# Convex Optimization
## Homework 6

Spring 1400
Due date: Ordibehesht 24th

---

1. Consider a binary classification problem with data in pairs $(x_i, y_i) \in \mathbf{R}^n \times \{-1, 1\}$, where we represent a classifer mapping $x \in \mathbf{R}^n$ to $\{\pm 1\}$ by $\theta \in \mathbf{R}^n$, predicting $\hat{y} = \text{sign}\left(x^T\theta\right)$. Given $m$ observations $(x_i, y_i), i = 1, \ldots, m$, we solve the convex optimization problem

$$\text{minimize} \sum_{i=1}^{m} f\left(y_i x_i^T \theta\right) + \frac{\lambda}{2}\|\theta\|_2^2 \tag{1}$$

to find a classifier $\theta$, where $\lambda > 0$ is a regularization parameter and $f : \mathbf{R} \to \mathbf{R}_+$ is a non-increasing convex function. Let $X = [x_1 \cdots x_m]^T \in \mathbf{R}^{m \times n}$ denote the data matrix.

(a) Show that a dual of problem (1) is

$$\text{maximize} \quad -\sum_{i=1}^{m} f^*\left(\alpha_i\right) - \frac{1}{2\lambda}\|X^T \operatorname{diag}(y)\alpha\|_2^2$$

with variable $\alpha \in \mathbf{R}^m$. Hint. Introduce the equality constraints $z_i = y_i x_i^T \theta$. Using duality,

(b) show that the solution $\theta^\star$ to the original problem is of the form

$$\theta^\star = \sum_{i=1}^{m} \nu_i x_i = X^T \nu$$

for some vector $\nu \in \mathbf{R}^m$. Specify your vector $\nu$.

In many scenarios, it is useful to consider functions of $x$, including (but not limited to) polynomials, Fourier transforms, or other nonlinearities. In this case, for a feature mapping $\varphi : \mathbf{R}^n \to \mathbf{R}^N$ we instead predict with $\theta \in \mathbf{R}^N$ and use $\hat{y} = \text{sign}\left(\theta^T \varphi(x)\right)$. For example, if

$$\varphi(x) = \left(1, x_1, \ldots, x_n, x_1^2, x_1x_2, x_1x_3, \ldots, x_1x_n, x_2x_1, \ldots, x_{n-1}x_n, x_n^2\right) \in \mathbf{R}^{1+n+n^2}$$

we can represent all quadratic functions of the input vector $x \in \mathbf{R}^n$. Instead of solving problem (1), we wish to find a classifier based on a (nonlinear) transformation of the $x_i$ vectors, replacing $x_i$ by $\varphi\left(x_i\right) \in \mathbf{R}^N$, that is, we wish to solve

$$\text{minimize} \sum_{i=1}^{m} f\left(y_i \varphi\left(x_i\right)^T \theta\right) + \frac{\lambda}{2}\|\theta\|_2^2. \tag{2}$$

By part (b), the solution to this must satisfy $\theta^\star = \sum_{i=1}^{m} \nu_i \varphi\left(x_i\right)$ for some $\nu \in \mathbf{R}^m$, and therefore we may classify a new instance $x$ by evaluating

$$\varphi(x)^T \theta^\star = \sum_{i=1}^{m} \varphi(x)^T \varphi\left(x_i\right) \nu_i$$

The kernel trick works as follows. In statistical machine learning parlance, a kernel function is a symmetric function $K : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R}$ that can be written as $K(x, z) = \varphi(x)^T \varphi(z)$ for some mapping $\varphi : \mathbf{R}^n \to \mathbf{R}^N$, where $N$ may even be infinite. In many cases, our choice of $\varphi(\theta)$ in problem (2) may be efficiently evaluated by such a kernel, even when $N$ is large. These mappings can allow one to introduce nonlinearities in classification rules that are quite effective.

(c) Suppose we have a kernel $K$ for $\varphi$, so that we can write $\varphi(x)^T \theta^\star = \sum_{i=1}^{m} K\left(x, x_i\right) \nu_i$. Let $G \in \mathbf{S}_+^m$ be the Gram matrix whose entries are $G_{i,j} = K\left(x_i, x_j\right)$. Show how to implicitly find $\theta^\star$ using this Gram matrix and the dual problem in part (a). More specifically, show how to

find the parameters $\nu$ in part (b) without ever explicitly computing $\varphi(x_i)$.
For completeness, we enumerate a few different kernel functions to highlight why these ideas may be important. For $k \in \mathbf{Z}_+$ and $x \in \mathbf{R}^n$ define the tensor

$$X = x^{\otimes k} \in \underbrace{\mathbf{R}^n \times \mathbf{R}^n \times \cdots \times \mathbf{R}^n}_{k \text{ times}}$$

to have entries $X_{i_1,\ldots,i_k} = x_{i_1} x_{i_2} \cdots x_{i_k}$, and let $\mathrm{vec}(X) \in \mathbf{R}^{n^k}$ be the vectorized version of $X$, that is, we simply stack all entries of $X$ on one another. (We say $x^{\otimes 0} = 1$. ) Then for any $x, z \in \mathbf{R}^n$ and degree $d \in \mathbf{Z}_+$ we have

$$\left(1 + x^T z\right)^d = \varphi(x)^T \varphi(z) \text{ where } \varphi(x) = \left[\sqrt{\binom{d}{k}}\, \mathrm{vec}\left(x^{\otimes k}\right)\right]_{k=0}^{d}.$$

Note that the dimension of $\varphi(x)$ is $\sum_{i=0}^{d} n^i = \frac{n^{d+1}-1}{n-1}$, and the structure of $\varphi$ shows that the kernel $K(x,z) = \left(1 + x^T z\right)^d$ can represent all polynomials of $x$ up to degree $d$.

(d) Compare the computational time to evaluate $\varphi(x)^T \theta^\star = \sum_{i=1}^{m} \varphi(x)^T \varphi(x_i)\, \nu_i$ by explicitly using the vector representation $\varphi(x)$ to that using the kernel $K(x,z) = \left(1 + x^T z\right)^d$.

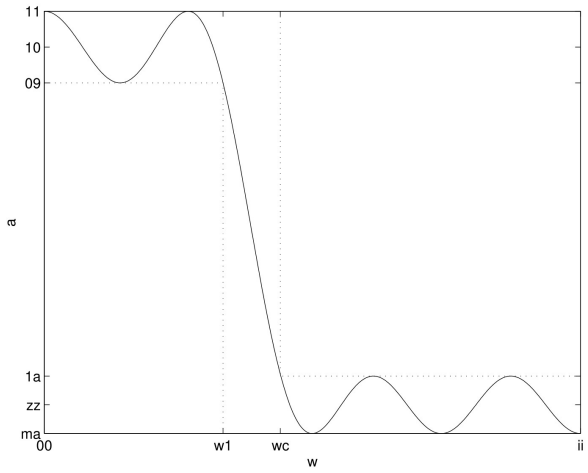
2. Consider the (symmetric, linear phase) finite impulse response (FIR) filter described by its frequency response

$$H(\omega) = a_0 + \sum_{k=1}^{N} a_k \cos k\omega$$

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 $0 \le \omega \le \pi/3, 0.89 \le H(\omega) \le 1.12$, i.e., the filter has about $\pm 1$ dB ripple in the 'passband' $[0, \pi/3]$.
- For $\omega_c \le \omega \le \pi, |H(\omega)| \le \alpha$. In other words, the filter achieves an attenuation given by $\alpha$ in the 'stopband' $[\omega_c, \pi]$. Here $\omega_c$ 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 $\omega_c$ and $N$, and wish to maximize the stopband attenuation, i.e., minimize $\alpha$.

(b) *Minimum transition band.* We fix $N$ and $\alpha$, and want to minimize $\omega_c$, i.e., we set the stopband attenuation and filter length, and wish to minimize the 'transition' band (between $\pi/3$ and $\omega_c$)

(c) *Shortest length filter.* We fix $\omega_c$ and $\alpha$, 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 $-30$ dB.) 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 $\omega$ for your design.

3. 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.

4. Show that the following problem is quasiconvex:

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

where

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

and the domain of the objective function is defined as

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

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

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. Optimal operation of a microgrid. We consider a small electrical microgrid that consists of a photovoltaic (PV) array, a storage device (battery), a load, and a connection to an external grid. We will optimize the operation of the microgrid over one day, in 15 minute increments, so all powers, and the battery charge, are represented as vectors in $\mathbf{R}^{96}$. The load power is $p^{\text{ld}}$ , which is nonnegative and known. The power that we take from the external grid is $p^{\text{grid}}$ ; $p_i^{\text{grid}} \geq 0$ means we are consuming power from the grid, and $p_i^{\text{grid}} < 0$ means we are sending power back into the grid, in time period i. The PV array output, which is nonnegative and known, is denoted as $p^{\text{pv}}$. The battery power is $p^{\text{batt}}$ , with $p_i^{\text{batt}} \geq 0$ meaning the battery is discharging, and $p_i^{\text{batt}} < 0$ meaning the battery is charging. These powers must balance in all periods, i.e., we have

$$p^{\text{ld}} = p^{\text{grid}} + p^{\text{batt}} + p^{\text{pv}}$$

(This is called the power balance constraint. The lefthand side is the load power, and the righthand side is the sum of the power coming from the grid, the battery, and the PV array.) All powers are given in kW.

The battery state of charge is given by $q \in \mathbf{R}^{96}$. It must satisfy $0 \leq q_i \leq Q$ for all $i$, where $Q$ is the battery capacity (in kWh ). The battery dynamics are

$$q_{i+1} = q_i - (1/4)p_i^{\text{batt}} , \quad i = 1, \ldots, 95, \quad q_1 = q_{96} - (1/4)p_{96}^{\text{batt}} .$$

(The last equation means that we seek a periodic operation of the microgrid.) The battery power must satisfy $-C \leq p_i^{\text{batt}} \leq D$ for all $i$, where $C$ and $D$ are (positive) known maximum charge and maximum discharge rates.
When we buy power (i.e., $p_i^{\text{grid}} \geq 0$ ) we pay for it at the rate of $R_i^{\text{buy}}$ (in \$/kWh ). When we sell power to the grid ( i.e., $p_i^{\text{grid}} < 0$ ) we are paid for it at the rate of $R_i^{\text{sell}}$ . These (positive) prices vary with time period, and are known. The total cost of the grid power (in \$ ) is

$$(1/4)\left(R^{\text{buy}}\right)^T \left(p^{\text{grid}}\right)_+ - (1/4)\left(R^{\text{sell}}\right)^T \left(p^{\text{grid}}\right)_-$$

where $\left(p^{\text{grid}}\right)_+ = \max\left\{p^{\text{grid}}, 0\right\}$ and $\left(p^{\text{grid}}\right)_- = \max\left\{-p^{\text{grid}}, 0\right\}$ (elementwise). You can assume that $R_i^{\text{buy}} > R_i^{\text{sell}} > 0$, i.e., in every period, you pay at a higher rate to consume power from the grid than you are paid when you send power back into the grid. The data for the problem are

$$p^{\text{ld}}, \quad p^{\text{pv}}, \quad Q, \quad C, \quad D, \quad R^{\text{buy}} , \quad R^{\text{sell}}$$

(a) Explain how to find the powers and battery state of charge that minimize the total cost of the grid power. Carry out your method using the data given in `microgrid_data.*`. Report the optimal cost of the grid power. Plot $p^{\text{grid}}, p^{\text{load}}, p^{\text{pv}}, p^{\text{batt}}$ , and $q$ versus $i$. Note. For CVXPY, you might need to specify `solver=cvx.ECOS` when you call the `solve()` method.

(b) Price and payments. Let $\nu \in \mathbf{R}^{96}$ denote the optimal dual variable associated with the power balance constraint. The vector $4\nu$ can be interpreted as the (time-varying) price of electricity at the microgrid, and is called the locational marginal price (LMP). The LMP is in \$/kWh, and is generally positive; the factor 4 converts between 15 minute power intervals and per kWh prices. Find and plot the LMP, along with the grid buy and sell prices, versus $i$. Make a very brief comment comparing the LMP prices with the buy and sell grid prices. Hint. Depending on how you express the power balance constraint, your software might return $-\nu$ instead of $\nu$. Feel free to use $-4\nu$ instead of $\nu$, or to switch the left-hand and righ-hand sides of your power balance constraint.

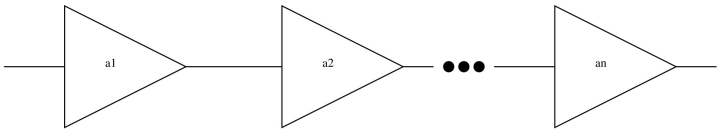
(c) The LMPs can be used as a system for payments among the load, the PV array, the battery, and the grid. The load pays $\nu^T p^{\text{ld}}$ ; the PV array is paid $\nu^T p^{\text{pv}}$; the battery is paid $\nu^T p^{\text{batt}}$ ; and the grid is paid $\nu^T p^{\text{grid}}$ . Note carefully the directions of these payments. Also note that the battery and grid, whose powers can have either sign, can be paid in some time intervals and pay in others. Use this pricing scheme to calculate the LMP payments made by the load, and to the PV array, the battery, and the grid. If all goes well, these payments will balance, i.e., the load will pay an amount equal to the sum of the others.

When you execute the script that contains the data, it will create plots showing the various powers and prices versus time. Your are welcome to use these as templates for plotting your results. You are very welcome to look inside the script to see how the data is generated.

*Remark.* (Not needed to solve the problem.) The given data is approximately consistent with a group of ten houses, a common or pooled PV array of around 100 panels, and two Tesla Powerwall batteries.

# Optional Questions

1. 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 $A^{\text{tot}}$, 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 $i$ th 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 $i$ th amplifier. The output noise level $N_{\text{out}}$ of the system is given by $N_{\text{out}} = N_n$, i.e., the noise level of the last amplifier. Evidently $N_{\text{out}}$ 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

$$S_0 = S_{\text{in}}, \quad S_i = a_i S_{i-1}, \quad i = 1, \ldots, n$$

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 $S_{\text{in}}$ 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

$$\begin{aligned} A^{\text{tot}} &= 10000 \\ \alpha &= \left(10^{-5}, 10^{-2}, 10^{-2}, 10^{-2}\right), \\ M &= (0.1, 5, 10, 10), \\ A^{\max} &= (40, 40, 40, 20). \end{aligned}$$

Give the optimal gains, and the optimal dynamic range.

2. 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 $\mathcal{D}\backslash\mathcal{O}$ 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 $\operatorname{vol}\mathcal{E}$ (on a log scale) versus $\operatorname{card}\mathcal{O}$ 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 $\operatorname{vol}\mathcal{E}$ (on a log scale) versus $\operatorname{card}\mathcal{O}$. 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).