Étude Problem

This is one of the étude problems for this winter. Please turn it in by uploading to Gradescope by 5pm on Thursday, with a regraded version by 5pm on Friday. We will not be lenient with upload times: because of the nature of the regrading, we will be posting solutions as close as possible to 5:01pm on Thursdays, so turning the étude in late will result in no credit. Please upload a placeholder to Gradescope well before the due date.

You may use any books, notes, or computer programs, but you may not discuss the étude with anyone—including online—until the solutions are posted. The only exception is that you can ask us for clarification, via the course staff email address. We’ve tried pretty hard to make these questions unambiguous and clear, so we’re unlikely to say much.

Please submit your étude via Gradescope.

We will deduct points from long, needlessly complex solutions, even if they are correct. Our solutions are not long, so if you find that your solution to a problem goes on and on for many pages, you should try to figure out a simpler one. We expect neat, legible études from everyone, including those enrolled Cr/N.

When a problem involves computation you must give all of the following: a clear discussion and justification of exactly what you did, the source code that produces the result, and the final numerical results or plots.

Files containing problem data can be found in the following location:

http://www.stanford.edu/~jduchi/teaching/364a/data/

Please respect the honor code. Although we allow you to work on homework assignments in small groups, you cannot discuss the études with anyone.

Some études are (quite) straightforward. Others, not so much.

Be sure you are using the most recent version of CVX, CVXPY, CVXR, or Convex.jl.

Some problems involve applications. But you do not need to know anything about the problem area to solve the problem; the problem statement contains everything you need.
6. **Kernel Support Vector Machines.** In class (and additional exercise 6.32), we saw that if we wished to fit a classifier based on a (potentially nonlinear) feature mapping \( \varphi : \mathbb{R}^n \to \mathbb{R}^N \), where the classification rule (i.e., prediction on an input \( x \)) is

\[
\hat{y} = \text{sign}(\theta^T \varphi(x)).
\]

Let \( K(x, z) = \varphi(x)^T \varphi(z) \) be the *kernel function* associated with feature mapping \( \varphi \). Given a collection of pairs \((x_i, y_i) \in \mathbb{R}^n \times \{ -1, 1 \} \), we find a classifier \( \theta \) by solving

\[
\text{minimize} \quad \sum_{i=1}^{m} f(y_i \varphi(x_i)^T \theta) + \frac{\lambda}{2} \| \theta \|_2^2 ,
\]

where \( f : \mathbb{R} \to \mathbb{R} \) is a convex, non-increasing function and \( \lambda \geq 0 \) is a regularization parameter. Equivalent to problem (1) is to solve the dual problem

\[
\text{maximize} \quad -\sum_{i=1}^{m} f^*(\alpha_i) - \frac{1}{2\lambda} \alpha^T \text{diag}(y) G \text{diag}(y) \alpha
\]

with variable \( \alpha \in \mathbb{R}^m \), where \( G \in \mathbb{S}^m \) is the *Gram matrix*, whose entries are \( G_{ij} = K(x_i, x_j) \). To recover the optimal \( \theta^* \) for problem (1) given an optimal dual variable \( \alpha^* \), we may set

\[
\theta^* = -\frac{1}{\lambda} \sum_{i=1}^{m} y_i \varphi(x_i) \alpha_i^* = \sum_{i=1}^{m} \varphi(x_i) \nu_i^*,
\]

where \( \nu^* = -\frac{1}{\lambda} \text{diag}(y) \alpha^* \).

Now, given a \( \theta \in \mathbb{R}^N \) taking the form \( \theta = \sum_{i=1}^{m} \nu_i \varphi(x_i) \), we can define the prediction function \( p_\theta : \mathbb{R}^n \to \mathbb{R} \) by

\[
p_\theta(x) = \theta^T \varphi(x) = \sum_{i=1}^{m} \nu_i \varphi(x_i)^T \varphi(x) = \sum_{i=1}^{m} K(x_i, x) \nu_i,
\]

where \( K \) is the kernel function for \( \varphi \). To make a prediction on an input \( x \), we simply take the sign \( \hat{y} = \text{sign}(p_\theta(x)) \), and we view the magnitude of \( p_\theta(x) \) as the “confidence” the classifier gives to its prediction.

Using the kernel \( K(x, z) = (1 + x^T z)^6 \) and objective \( f(t) = (1 - t)_+ = \max\{0, 1 - t\} \), implement the dual problem (2) for the problem data in `kernel_svm_data.*`. Solve the resulting problem with regularization multipliers \( \lambda = 10^{-3}, 10^{-2}, 1, 10 \). For each \( \lambda \), plot a contour plots of the resulting prediction function \( p_\theta(x) \) as a function of \( x \in [-1.5, 1.5]^2 \subset \mathbb{R}^2 \). What does increasing/decreasing the regularization \( \lambda \) do?

A few notes on implementation: numerical instability issues mean that in your code, the Gram matrix \( G \) may be slightly indefinite; in this case, you should replace \( G \) with \( G + \epsilon I \) for \( \epsilon = 10^{-6} \) or some other small constant. Different solvers may experience instability, so if one solver does not work (e.g. SCS) try another. In our solution, we also plotted the datapoints in a scatterplot to give some intuition. Be sure to include your code and the four contour plots. *Hint.* You may use that \( f^*(s) = s \) for \( s \in [-1, 0] \) and \( f^*(s) = +\infty \) otherwise.