Convex Optimization - Homework 3

Report plots, comments and theoretical results in pdf or similar. Send code with requested functions and a main script with standard examples of your functions and which reproduces all the main experiments. Please use either Julia or MATLAB.

Support Vector Machine Problem

Given n data points $x_i \in \mathbb{R}^d$ with labels $y_i \in \{-1, 1\}$ and a regularization parameter $C > 0$, consider the Support Vector Machine problem

$$\begin{array}{lll} \text{minimize} & \frac{1}{2}\|w\|_2^2 + C\mathbf{1}^T z & \\ \text{subject to} & y_i(w^T x_i) \geq 1 - z_i, \quad i = 1, \dots, n & (\lambda_i) \\ & z \geq 0 & (\pi) \end{array} \tag{SVM}$$

in the primal variables $w \in \mathbb{R}^d$, $z \in \mathbb{R}^n$ and dual variables λ and π.

1. Derive the dual problem of SVM.
2. Give strictly feasible points for primal and dual.
3. Implement the barrier method (using logarithmic barrier) to solve (SVM) given a data matrix $X = (x_1^T, ..., x_n^T) \in \mathbb{R}^{n \times d}$, a vector of labels $y \in \{-1, 1\}^n$ and a regularization parameter $C > 0$ by
   - coding the generic functions `[Q,p,A,b] = transform_svm_primal(C,X,y)` and `[Q,p,A,b] = transform_svm_dual(C,X,y)`, which writes the primal and the dual SVM problem as particular instances of a quadratic problem

     $$\min \phi(x) = \frac{1}{2}x^T Q x + p^T x \quad \text{s.t. } Ax \leq b. \tag{QP}$$

   - coding the function `[x_new,gap] = Newton(t,x,f,grad,hess)` which performs a Newton step on the function $f(x, t)$ (using a backtracking line search with appropriate parameters) where t is fixed. The function $f(x, t)$ is defined to be

     $$f(x, t) = t\phi(x) + \mathcal{B}(x).$$

where $\mathcal{B}(x)$ is a barrier function. The parameter `t` is the parameter of the barrier (see lectures), `x` is the current point, `f` is a function where `f(x,t)` returns the value $f(x,t)$, `grad` is a function where `grad(x,t)` returns the vector $\nabla_x f(x,t)$ and `hess` is a function where `hess(x,t)` returns the matrix $\nabla_x^2 f(x,t)$. The output `x_new` is the point after performing a Newton step on `x` and `gap` is an upper bound for $f(\texttt{x\_new}, t) - f(x^*(t), t)$, where $x^*(t)$ is the solution of $\min_x f(x,t)$ for fixed t.
Note: this function should call **one and only one** time `grad(x,t)` and `hess(x,t)`. However, `f(x,t)` can be called as many time as desired.

- coding a generic function `[x_seq] = centering_step(Q,p,A,b,x,t,tol)` which solves the centering step of (QP) using Newton's method, given the initial point x, objective function parameters (Q, p), linear constraints parameters of the problem (A, b), the barrier method parameter t and a precision criterion ϵ. The function should output the sequence of iterates $x_{k=1,\ldots}$, and the last x satisfying $f(x) - f(x^*(t)) \leq \texttt{tol}$, where $x^*(t)$ is the optimal solution of the barrier problem with parameter t.
- coding a function `[x_sol,x_seq] = barr_method(Q,p,A,b,x_0,mu,tol)` which implements the barrier method to solve QP given the inputs (Q, p, A, b) and the initial point `x_0` (which should be strictly feasible). The input `mu` is the increment of the barrier at each iteration (see lecture notes). The output `x_sol` must be feasible and satisfies
$$\phi(\texttt{x\_sol}) - \phi(x^*) \leq \texttt{tol}$$
where x^* is the solution of QP. The function also outputs `x_seq`, the sequence of variables $(x_i)_{i=1,\ldots}$.

4. Plot primal objective function values versus iterations in semilog-scale for different values of the barrier method parameter $\mu = 2, 15, 50, 100$ and comment the results.

5. Plot dual objective function values versus iterations in log-scale. Plot the duality gap versus iterations in semilog-scale.

6. Test your code on artificial data :
   - code a function `[X,y] = generate_data(...)` which generate two clouds of points for $d = 2$ with labels $+1$ and -1 respectively, by picking bivariate Gaussian samples with different moments.
   - use your primal optimization algorithm to find the optimal linear classifier *Add one dimension to your data points in order to account for the offset if your data is not centered.*
   - plot the two clouds of points with the original labels and the two clouds of points with the labels output by the SVM and the corresponding linear classifier.
   - try different values of C and measure out-of-sample performance.