CME 252: Homework 1
Due: Saturday, October 10, 11:59pm

AJ Friend

This homework should be doable based on the CVXPY code you've already seen in the lectures up to this point. Below, I'll give some pointers on Python, NumPy, and CVXPY. The code will also include some examples of Python syntax, in case you're not familiar with it. The point is to test your ability to model problems with CVXPY, and **not** on your knowledge of Python and NumPy, so please do ask questions if you get stuck.

Check your solutions with `grade_hw1.py`.

Submit your completed `hw1.py` to this Dropbox File Request link: https://www.dropbox.com/request/V1uTRGdaGB7Fr23d3aEk

Problem 1: Feasibility with CVXPY

Given a matrix $A \in \mathbf{R}^{m \times n}$ and vector $b \in \mathbf{R}^m$, determine if there is an $x \in \mathbf{R}^n$ such that $Ax = b$ and $x \geq 0$. If such an x exists, return an example as a NumPy `array`. If no such x exists, return the Python value `None`. Fill in `prob1(A, b)` in `hw1.py`.

Problem 2: Membership in a convex hull

Fill in `prob2(X, a)` in `hw1.py` to determine if the point $a \in \mathbf{R}^n$ is contained within the polytope given by the points $x_1, \ldots, x_m \in \mathbf{R}^n$. These points are given as the columns of the matrix $X \in \mathbf{R}^{n \times m}$. Return `True` if a is in the convex hull, and `False` otherwise.

Problem 3: Intersection of polytopes

We saw two different ways of representing a polytope in lecture: as an intersection of halfspaces ($Ax \leq b$) and as the convex hull of a set of points.

Given a polytope described as $\{x \mid Ax \leq b\}$, and the polytope given as the convex hull of the points $x_1, \ldots, x_m \in \mathbf{R}^n$ (given as the columns of matrix X), determine if the two polytopes intersect. That is, determine if there is an $x \in \mathbf{R}^n$ such that x is contained in each polytope.

Fill in `prob3(A, b, X)` in `hw1.py`. Have your function return `True` if the two intersect and `False` if they do not.

Tips

- After forming a problem and solving it with CVXPY as in

  ```
  import cvxpy as cvx
  prob = cvx.Problem(cvx.Minimize(0), constraints)
  prob.solve()
  ```

  the variable `prob.status` provides a string describing the solver status. There are a few possible values, but for this homework, we'll just consider two: `'optimal'` and `'infeasible'`. In the case of feasibility problems like this one, `'optimal'` means that the solver found a feasible point (some point satisfying all the constraints), stored in `x.value` (if `x` was your problem variable). `'infeasible'` means that no such point exists.

- `x.value` returns a NumPy `matrix` object. The way these interact with NumPy `array`s can be confusing, so I'll often just convert them with something like `x = np.array(x.value).flatten()`. The `flatten()` function just converts to a one-dimensional array, instead of a two-dimensional array with dimensions `(n,1)` or `(1,n)`.

- Matrix multiplication is written differently in CVXPY and NumPy.

  – CVXPY: `A*x`
  – NumPy: `A.dot(x)`

  Most of the time, you'll be working within CVXPY, so you won't have to switch between the two very often. I just mention it in case you try playing with the arrays outside of CVXPY.