Stanford **Spring 2024**

AA 203: Optimal and Learning-based Control
Homework #0
Not graded

Learning goals for this problem set:

Problem 1: Review stability of discrete LTI systems.

Problem 2: Review unconstrained convex optimization.

Problem 3: Review linear regression techniques, and numerical and plotting libraries in Python.

0.1 Discrete-time LTI stability. Consider the system $x_{t+1} = Ax_t + Bu_t$, where

$$A = \begin{bmatrix} 4/5 & 0 & 0 \\ 0 & \sqrt{3} & 1 \\ 0 & -1 & \sqrt{3} \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}.$$

(a) Explain whether or not this system is "open-loop stable", i.e., asymptotically stable for $u_t \equiv 0$.

(b) Design a linear feedback controller $u_t = Kx_t$ with fixed gain matrix $K \in \mathbb{R}^{2\times 3}$ such that the closed-loop system is asymptotically stable.

0.2 Poisson maximum likelihood. Suppose we observe the number of customers X to a store over N days, and we want to fit a Poisson distribution to the resulting data $\mathcal{D} \coloneqq \{x_1, x_2, \ldots, x_N\}$. The Poisson distribution is a distribution over non-negative integers with a single parameter $\lambda \geq 0$. It is often used to model arrival times of random events or count the number of random arrivals within a given amount of time. Its probability mass function is

$$\Pr(X = x) = \frac{e^{-\lambda}\lambda^x}{x!}.$$

To fit our model, we want to choose the parameter λ of the Poisson distribution to maximize the probability of the data $\mathcal{D}$. Assuming the number of customers on each day is *independent and identically distributed (IID)*, the *likelihood* of $\mathcal{D}$ is

$$p(\mathcal{D}; \lambda) \coloneqq \prod_{t=1}^{N} \Pr(X = x_t).$$

Specifically, we will maximize the *log-likelihood* of $\mathcal{D}$ by solving the optimization problem

$$\underset{\lambda \geq 0}{\text{maximize}} \log p(\mathcal{D}; \lambda).$$

(a) What property of the logarithm allows us to replace the likelihood with the log-likelihood in this maximization problem?

(b) Derive the maximum likelihood estimator $\hat{\lambda} \coloneqq \arg\max_{\lambda \geq 0} \log p(\mathcal{D}; \lambda)$.

0.3 Asteroid regression. Suppose we obtain measurements $\{(d_i, m_i)\}_{i=1}^N$ for N asteroids, where $d_i > 0$ and $m_i > 0$ are the diameter and mass, respectively, of the i-th asteroid. If the asteroids were radially symmetric and uniformly dense, then we could posit that $m \sim d^3$. However, the asteroids are not radially symmetric nor uniformly dense, yet we still suspect that d and m exhibit a cubic polynomial relationship, i.e.,

$$m = x_1 d + x_2 d^2 + x_3 d^3,$$

for some coefficients $x := (x_1, x_2, x_3) \in \mathbb{R}^3$. We do not include a constant term since the asteroid mass should be zero when its diameter is zero.

(a) Formulate this regression problem (i.e., the problem of fitting the coefficients x to the data $\{(d_i, m_i)\}_{i=1}^N$) as a convex least-squares optimization of the form

$$\underset{x}{\text{minimize}} \ \|Ax - y\|_2^2.$$

Specifically, describe how the matrix A and the vector y are formed from the data $\{(d_i, m_i)\}_{i=1}^N$.

(b) Express the optimal least-squares solution x^* in terms of A and y.

Hint: You may assume $A^\mathsf{T} A$ is invertible.

(c) Data of the form $\{(d_i, m_i)\}_{i=1}^N$ is provided in `data_asteroid_regression.csv`. Using NumPy in Python, load this data and implement the least-squares solution for x^*. Report x^* up to two decimal places for each entry.

In general, the ℓ_2-norm is susceptible to overfitting to outliers. We can find a more robust solution by solving the ℓ_1-norm optimization

$$\underset{x}{\text{minimize}} \ \|Ax - y\|_1.$$

Unlike the ℓ_2-norm problem, the ℓ_1-norm problem does not have a closed-form solution. However, we can use gradient descent to solve for x^* by iteratively producing estimates of a minimizer for the objective $f(x) := \|Ax - y\|_1$. Gradient descent is described by the update rule

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)})$$

at the k-th iteration, where $\alpha^{(k)} > 0$ is the step size.

(d) Derive the gradient of the ℓ_1-norm regression objective $f(x)$ in terms of A, y, and x.

Hint: Technically, the ℓ_1-norm is not differentiable at zero or any vector containing a zero entry. You may choose any number in the interval $[-1, 1]$ for $\frac{\partial}{\partial x_i}|x_i|$ at $x_i = 0$. The set $[-1, 1]$ is the *sub-differential* of $|x_i|$ at $x_i = 0$, and any element of this set is a *sub-gradient*.

(e) Using NumPy in Python, implement sub-gradient descent for the ℓ_1-norm regression problem for the data in `data_asteroid_regression.csv`. Initialize $x^{(0)} = 0$ and use a constant step size of $\alpha^{(k)} = 10^{-4}$ for all iterations. At each iteration, set x^* as the best solution found so far by keeping track of the objective value $f(x)$. Terminate after 10000 iterations. Report the ℓ_1-norm-optimized x^* up to two decimal places for each entry.

(f) Plot the ℓ_2-fit, ℓ_1-fit, and data on the same (d, m)-axes using Matplotlib in Python.