Homework 4

Due: 5:00 PM EDT on Wed Dec 15, 2021

Instructions: The homework consists of written as well as programming exercises. Your homework is your own – while you are free to discuss the problems with other students, each of you must submit your own writeup, code, etc. There is a zero-tolerance policy regarding cheating and submission of identical work.

Submitting your homework consists of two parts:

1. A **single PDF** of your written work as well as *the output of your code* should be uploaded to Gradescope (which can be accessed via Brightspace). When uploading to Gradescope, please make sure to mark which pages contain the solutions to which problems. This saves everyone a lot of time when grading.

2. Any code that you write for the assignment must also be submitted via GitHub Classroom. You will receive an invitation URL for the assignment via email. It is *your responsibility* to setup a Github account and join the GitHub Classroom.

   The code you submit *must be easy* to execute. For example, if it is Matlab code in the file `prob1.m`, the code should run when I execute the command `matlab prob1` from the command line. If the code is in C or Fortran, for example, you must include a script which compiles and executes the code. If you do not submit code for a particular problem, you will not receive credit even if you submitted the output of your code.

---

1. Imagine a container with a divider in the middle, and a total of m balls distributed in some way between the left and right sides. At each step, one of the m balls is chosen uniformly at random and moved to the other side of the divider (i.e. moved from left to right side, or vice versa). Denote by X_n the number of balls on the left side at time n.

   - (**3 points**) Show that X_n is a Markov Chain, and define its state space.
   - (**3 points**) Calculate the transition probabilities for X_n.
   - (**4 points**) Show that the stationary distribution for this chain is π, with the ith entry equal to:
   $$\pi(i) = \frac{1}{2^m}\binom{m}{i}.$$

2. (**10 points**) Suppose that we are interested in generating samples for a random variable X defined as follows:
$$X|\mu, \sigma^2 \sim N(\mu, \sigma^2),$$
$$\mu \sim \text{Exp}(1),$$
$$\sigma^2 \sim \chi^2_2.$$

Write a program (from scratch!) that uses a Monte Carlo algorithm to generate samples for X. Use your code to estimate $\mathbb{E}(X)$. Can you compute the exact value of $\mathbb{E}(X)$? If so, show that the Monte Carlo convergence rate is $O(1/\sqrt{n})$, where n is the number of samples drawn.

Note: You are only allowed to use a built-in Uniform(0,1) random number generator from your programming language of choice when writing your code. No additional software libraries are allowed.

3. Consider the probability density function

$$f(x) = \frac{x^4 e^{-x}}{24}, \qquad \text{for } x > 0.$$

- (**5 points**) Write a code to generate 50 samples from this density.
- (**7 points**) Write a code that computes the kernel density estimator of f, denoted by $\hat{f}_h$ (where h denotes the bandwidth of the kernel) using the kernel

$$K(x) = \frac{315}{256}\left(1 - x^2\right)^4, \qquad \text{for } x \in (-1, 1).$$

  Outside of the interval $(-1, 1)$, $K = 0$. In this case, we explicitly know the underlying density from which the samples were drawn. The L^2 error between the kernel density estimator and the true density is

$$\epsilon_h = \sqrt{\int_{-\infty}^{\infty} |f(x) - \hat{f}_h(x)|^2 \, dx}.$$

  Next, write a code that evaluates ϵ_h for $h = 0.1, 0.25, 0.5, 1.0, 2.5, 5.0, 10.0$. You are allowed to use build-in numerical integration routines if your programming environment has them (although writing your own might be much more efficient...).

  Hint: When you scale K by the bandwidth h, don't forget to scale its domain as well (i.e. its argument x will no longer be defined on $(-1, 1)$, it will depend on the value of h).
- (**8 points**) Finally, write an optimization code that computes the value of h that minimizes ϵ_h for *your specific samples from the density f*.