Data Science and Social Inquiry: HW4

Yu-Chang Chen and Ming-Jen Lin

November 18, 2022

Question 1: Implementing Newton's Method

In this question, we will implement Netwon's method, which is a specific version of the gradient descent algorithm, to minimize the function

$$f(x) = 0.05x^4 + 0.1x^3 - 0.75x^2 - x + 3.$$

Recall that the Newton's method uses Hessian as learning rate and iterates in the following way

$$x_{k+1} = x_k - \frac{1}{f''(x_k)} \cdot f'(x_k).$$

(a) (1 pt) Plot $f(x)$ in Python. Where is the global minimum?

Solution:

(b) (1 pt) Run the Newton's method with initial point $x_0 = 5$ and iterate 1,000 times. Plot the first 1,000 iterations on a graph. Does it converge to the global minimizer?

Solution:

(c) (1 pt) Run the Newton's method with initial point $x_0 = -1$ and iterate 1,000 times. Plot the first 1,000 iterations on a graph. Does it converge to the global minimizer?

Solution:

Question 2: Apply Gradient Descent to MLE

The maximum likelihood estimator (MLE) estimates a parameter using the maximizer of the log-likelihood function. That is,

$$\hat{\theta}_{\text{MLE}} = \arg\max_{\theta \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} ln(f(x_i|\theta)),$$

where $f(x_i|\theta)$ is the p.d.f. (or p.m.f.) that generates observations $x_1, x_2, ..., x_n$.

In the case of normal distribution with known variance $\sigma^2 = 1$, the MLE of the location parameter μ is

$$\hat{\mu}_{\text{MLE}} = \arg\max_{\theta \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} \Big[-\frac{1}{2} ln(2\pi) - \frac{1}{2}(x_i - \mu)^2) \Big]$$

Suppose that our observations x_i's are

$$3, 3, 2, 2, 4, 2, 4, 4, 3, 1.$$

Answer the following questions.

(*d*) (1 pt) Derive the Hessian of the objective function. Is it concave? [1]

Solution:

(*e*) (1 pt) Analytically solve $\hat{\mu}$ by the first order condition.

Solution:

(*f*) (1 pt) Use the Newton's method:

$$x_{k+1} = x_k + \frac{1}{f''(x_k)} \cdot f'(x_k)^2$$

to solve $\hat{\mu}_{\text{MLE}}$ numerically. How many iterations does it take to find $\hat{\mu}$?

Solution:

[1]For maximization, we prefer concave functions since local maximum must be global maximum for concave functions.

[2]Notice that to maximize a function, we update in the direction of the gradient.

Question 3: Simulating Multiple Testing

In this question, we will simulate 1,000 coins and flip each coin 100 times. Our goal is to test whether each coin i is fair or not:

$$\mathcal{H}_{i,0} : \text{Coin } i \text{ is a fair coin,}$$
$$\mathcal{H}_{i,1} : \text{Coin } i \text{ is not a fair coin.}$$

The purpose of this question is to demonstrate that, without adjustment for multiple testing, classical testing procedure may result in lots of false discovery. We'll start by constructing a "single" test for each coin.

Let $X_{i,1}, X_{i,2}, ..., X_{i,100} \overset{i.i.d.}{\sim} Bernouli(0.5)$ denote the 100 flips of coin i and $\bar{X}_i = \frac{1}{100}\sum_{j=1}^{100} X_{i,j}$ be their average. The Central Limit Theorem implies that

$$\bar{X}_i \overset{d}{\approx} \mathcal{N}\bigg(E[\bar{X}_i],\ Var(\bar{X}_i)\bigg),$$

and we can use the normal approximation to construct a t-test that rejects $\mathcal{H}_{i,0}$ if

$$|\bar{X}_i - 0.5| > c.$$

Let $\Phi(\cdot)$ be the cumulative distribution function of standard normal distribution.

(*g*) (1 pt) Calculate $E[\bar{X}_i]$ and $Var(\bar{X}_i)$.

Solution:

(*h*) If we would like the test to have size 0.05, what value of the decision cutoff c should we choose? Hint: your answer will make use of $\Phi(\cdot)$.

Solution:

Now, set `numpy.random.seed(13579)` and use `numpy.random.binomial()` to generate 1,000 $\times$ flips for 100 coins.

(*i*) Apply the test you just constructed to test $\mathcal{H}_{i,0}$ for coins. How many false discovery did you find?

Solution:

(j) Implement the Bonferroni correction for multiple testing. What is the decision cutoff for each hypothesis? How many false discovery did you find?

Solution: