Problem Set 1 Answers1

Kaley Burg

February 11, 2024

Instructions

- Please show your work! You may lose points by simply writing in the answer. If the problem requires you to execute commands in `R`, please include the code you used to get your answers. Please also include the `.R` file that contains your code. If you are not sure if work needs to be shown for a particular problem, please ask.
- Your homework should be submitted electronically on GitHub in `.pdf` form.
- This problem set is due before 23:59 on Sunday February 11, 2024. No late assignments will be accepted.

Question 1

The Kolmogorov-Smirnov test uses cumulative distribution statistics test the similarity of the empirical distribution of some observed data and a specified PDF, and serves as a goodness of fit test. The test statistic is created by:

$$D = \max_{i=1:n} \left\{ \frac{i}{n} - F_{(i)}, F_{(i)} - \frac{i-1}{n} \right\}$$

where F is the theoretical cumulative distribution of the distribution being tested and $F_{(i)}$ is the ith ordered value. Intuitively, the statistic takes the largest absolute difference between the two distribution functions across all x values. Large values indicate dissimilarity and the rejection of the hypothesis that the empirical distribution matches the queried theoretical distribution. The p-value is calculated from the Kolmogorov- Smirnoff CDF:

$$p(D \leq d) = \frac{\sqrt{2\pi}}{d} \sum_{k=1}^{\infty} e^{-(2k-1)^2\pi^2/(8d^2)}$$

which generally requires approximation methods (see Marsaglia, Tsang, and Wang 2003). This so-called non-parametric test (this label comes from the fact that the distribution of the test statistic does not depend on the distribution of the data being tested) performs

poorly in small samples, but works well in a simulation environment. Write an `R` function that implements this test where the reference distribution is normal. Using `R` generate 1,000 Cauchy random variables (`rcauchy(1000, location = 0, scale = 1)`) and perform the test (remember, use the same seed, something like `set.seed(123)`, whenever you're generating your own data).

As a hint, you can create the empirical distribution and theoretical CDF using this code:

```
# create empirical distribution of observed data
ECDF <- ecdf(data)
empiricalCDF <- ECDF(data)
# generate test statistic
D <- max(abs(empiricalCDF - pnorm(data)))
```

My Answer:

- First I created my data

```
set.seed(123)
```

```
data <- (rcauchy(1000, location = 0, scale = 1))
```

- Then I created my Kolmogorov-Smirnoff CDF function:

```
kolsmir_pval <- function(data) {

  set.seed(123)

  # create empirical distribution of observed data

  ECDF <- ecdf(data)
  empiricalCDF <- ECDF(data)

  #generating the test statistic

  D <- max(abs(empiricalCDF - pnorm(data)))

  #setting n
  n <- length(data)

  #initializing sum portion of the formula to 0
  sum1 <- 0

  #for loop for 1 to 1000 (instead of infinity)
  for (k in 1:1000) {
    sum1 <- sum1 + exp((-((2*k-1)^2)*(pi^2))/((8*D^2)))
  }

  #then we take the sum we created above and multiply by the other part
  #of the formula

```

```
  #this gives the overall p value
  p <- (((2*pi)^(1/2))/D)*sum1


  #making it print like it would in ks.test

  cat("D =", D, ", p-value =", p)

  return(list(D = D, p = p))
}
```

- Lastly, I compared this with the built in ks.test function in R to check my answer

```
ks_result <- ks.test(data, pnorm)
```

- The results of both my function and the ks.test output are shown in the tables below:

Table 1: Kolmogorov-Smirnov Test - My function

Statistic	Value
D	0.1347281
p	5.652523×10^{-29}

Table 2: Kolmogorov-Smirnov Test - ks.test Output

Statistic	Value
D	0.13573
p-value	2.22×10^{-16}

- Note: the results are slightly different between my function and ks.test due to internal differences in the function, but they are similar enough to show that my function was implemented correctly.
  Interpretation:

- As stated above, the D statistic is the largest absolute difference between the two distributions across all x values.

- As shown in the tables above, the D statistic is approximately 0.13 and the p-value associated with this statistic is very close to 0.

- This test overall assesses whether a sample comes from a specific distribution. Based on the p-statistic we obtain, we can draw the conclusion that we can **reject the null**

hypothesis that the sample is drawn from the reference (normal) distribution. This value would be significant at the $\alpha < 0.001$ level, due to how small the obtained p-statistic is

Question 2

Estimate an OLS regression in `R` that uses the Newton-Raphson algorithm (specifically `BFGS`, which is a quasi-Newton method), and show that you get the equivalent results to using `lm`. Use the code below to create your data.

```
set.seed (123)
data <- data.frame(x = runif(200, 1, 10))
data$y <- 0 + 2.75*data$x + rnorm(200, 0, 1.5)
```

My Answer:

- First I ran the code given to us, which sets the seed and creates the data:

```
set.seed(123)
data2 <- data.frame(x = runif(200, 1, 10))
data2$y <- 0 + 2.75*data2$x + rnorm(200, 0, 1.5)
```

- Then I created a linear likelihood function to use in my Newton-Raphson algorithm (or more specifically, BFGS). This code came from the class slides and is utilizing the following function

$$\log(L) = \sum_{i=1}^{n} -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(\sigma^2) - \frac{1}{2\sigma^2}(y_i - x_i\beta)^2$$

```
linear_lik <- function(theta, y, X) {
  n <- nrow(X)
  k <- ncol(X)
  beta <- theta[1:k]
  sigma2 <- theta[k + 1]^2
  e <- y - X%*%beta
  logl <- -.5*n*log(2*pi) -.5*n*log(sigma2) - ((t(e) %*% e) / (2 * sigma2
    ) )
  return(-logl)
}
```

- We assume that the points follow a normal probability distribution with mean $x\beta$ and variance σ^2, i.e., $y \sim N(x\beta, \sigma^2)$

- So essentially, the code above is finding the parameters β and σ that maximize the probability for all points (x_i, y_i)

- Then I used this function to find the maximum point on the surface created by the previous function using the optim function, using the BFGS algorithm to approximate the Newton-Raphson algorithm (this code came from the class slides):

```
linear_MLE <- optim(
  fn = linear_lik,
  par = c(1, 1, 1),
  hessian = TRUE,
  y = data2$y,
  X = cbind(1, data2$x),
  method = "BFGS"
)
```

- For reference (not necessary for the overall answer/explanation) the surface itself can be visualized with this code (also from class slides):

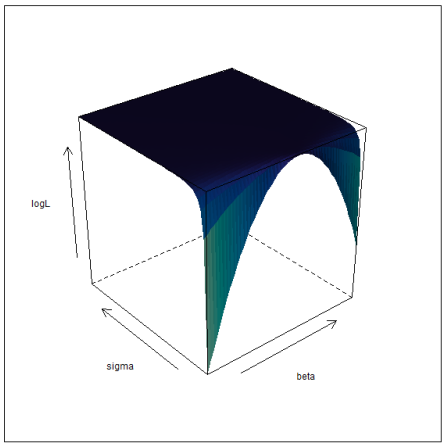
```
surface <- list()
k <- 0
for (beta in seq(0, 5, 0.1)) {
  for (sigma in seq(0.1, 5, 0.1)) {
    k <- k + 1
    logL <- linear_lik(theta = c(0, beta, sigma), y = data2$y, X = cbind
    (1, data2$x))
    surface[[k]] <- data.frame(beta = beta, sigma = sigma, logL = -logL)
  }
}

surface <- do.call(rbind, surface)

library(lattice)

png(filename = "plot.png", width = 800, height = 600)
wireframe(logL ~ beta * sigma, surface, shade = TRUE)
dev.off()
```

- Which produces this plot:



- This function therefore produces an estimation of the parameters that specify this maximum point

Table 3: BFGS results

Parameter	Estimate
Intercept (α)	0.1398324
β	2.7265559
σ	-1.4390716
σ^2	2.070927

- Then I compared the parameters created by my function with the results of the lm function

```
linear_MLE$par

lm_model <- summary(lm(y~x , data2))
```

Table 4: OLS (lm) results

Variable	Estimate	Std. Error	t value	Pr(>—t—)
(Intercept)	0.13919	0.25276	0.551	0.582
x(β)	2.72670	0.04159	65.564	$< 2e-16$

- We can see in the highlighted terms that we are getting the same estimates for both the intercept (α) and β because both the BFGS algorithm and the OLS function are, in theory, consistent estimators of the true underlying parameter of interest
  - In this case, an on average effect of the covariate on the outcome
- At some point the answers we get using the different methods would not be the same because they are different estimators
- For instance, OLS is closed form in the sense that we have the data and we plug it in to get estimates, while MLE is more complex as it uses numeric approximations
- So, in theory if we had a large amount of data, they will get closer to true estimate, but MLE is **not an unbiased estimator**