Applied Stats II — Problem Set 1

Name: Qi Liu Student ID: 25340516

February 7, 2026

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 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 and perform the test.

Hint (from handout)

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

```

Answer: method

I implement the KS statistic by comparing the empirical CDF (ECDF) of the observed sample to the reference Normal CDF evaluated at the same points. The test statistic D is the maximum absolute difference between these two CDFs. To assess significance, I compute an asymptotic p-value approximation based on the Kolmogorov distribution (using $\lambda = \sqrt{n}D$ and a truncated series). I also validate my D against R's built-in `ks.test`.

Answer: R code (functions + simulation + comparison)

```r
# ---- KS statistic D for testing against N(0,1) ----
ks_D_against_norm <- function(x) {
  x <- sort(as.numeric(x))
  n <- length(x)
  F0 <- pnorm(x)
  i <- seq_len(n)
  D_plus  <- max(i / n - F0)
  D_minus <- max(F0 - (i - 1) / n)
  max(D_plus, D_minus)
}

# ---- Asymptotic p-value approximation (Kolmogorov series) ----
ks_pvalue_asymptotic <- function(D, n, K = 200) {
  if (!is.finite(D) || D <= 0) return(NA_real_)
  lambda <- sqrt(n) * D
  k <- 1:K
  pval <- 2 * sum(((-1)^(k - 1)) * exp(-2 * (k^2) * (lambda^2)))
  pval <- max(min(pval, 1), 0)
  pval
}

# ---- Wrapper ----
my_ks_test_norm <- function(x, K = 200) {
  D <- ks_D_against_norm(x)
  pval <- ks_pvalue_asymptotic(D, n = length(x), K = K)
  list(D = D, p_value_series = pval, K = K, n = length(x))
}

# ---- Data + run ----
set.seed(123)
x_cauchy <- rcauchy(1000, location = 0, scale = 1)

q1_res <- my_ks_test_norm(x_cauchy, K = 500)
ks_builtin <- ks.test(x_cauchy, "pnorm")

```

Answer: output (from console)

```
My KS D statistic (vs N(0,1)): 0.1357281
My p-value (series approx, K = 500 ): 1.994304e-16
```

```
Built-in ks.test results:
ks.test D: 0.1357281
ks.test p-value: 1.994304e-16

Diagnostics:
Absolute difference in D (my D vs ks.test D): 0
```

Conclusion

The p-value is approximately 1.99×10^{-16}, which is far below $\alpha = 0.05$. Therefore, I reject H_0 that the sample is drawn from $N(0, 1)$. This is expected because the data were generated from a Cauchy distribution, which has much heavier tails than the Normal distribution.

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.

Answer: data generation

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

```

Answer: OLS as minimising SSE

OLS can be obtained by minimising the sum of squared errors (SSE):

$$SSE(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2 .$$

I define an SSE objective function and then minimise it using `optim(..., method="BFGS")`. I compare the resulting coefficients and SSE to `lm(y ~x)`.

Answer: SSE + BFGS code + comparison with `lm`

```r
fit_lm <- lm(y ~ x, data = data)
coef_lm <- coef(fit_lm)

sse_ols <- function(par, x, y) {
  b0 <- par[1]
  b1 <- par[2]
```

```r
    resid <- y - (b0 + b1 * x)
    sum(resid^2)
  }

  init <- c(0, 0)
  opt_bfgs <- optim(
  par = init,
  fn = sse_ols,
  x = data$x,
  y = data$y,
  method = "BFGS",
  control = list(reltol = 1e-12)
  )

  coef_bfgs <- opt_bfgs$par
  names(coef_bfgs) <- c("(Intercept)", "x")

  sse_lm <- sum(residuals(fit_lm)^2)
  sse_bfgs <- opt_bfgs$value

```

Answer: output (from console)

```
lm() coefficients:
(Intercept) 0.1391874
x           2.7266985

BFGS (optim) coefficients:
(Intercept) 0.1391778
x           2.7267000

SSE comparison:
SSE from lm():   414.4577
SSE from BFGS:   414.4577
Abs diff in SSE: 3.026514e-09

Coefficient comparison (BFGS - lm):
(Intercept) -9.610088e-06
x            1.453862e-06

all.equal(BFGS, lm) for coefficients:
[1] "Mean relative difference: 3.86058e-06"
```

Conclusion

The BFGS estimates match the `lm()` estimates up to negligible numerical rounding error (about 10^{-6}), and the SSE values are essentially identical. This confirms that minimising

SSE using BFGS recovers the OLS solution.