Prof. Yiping Lu
IEMS 304: Statistical Learning for Data Analysis
April 7, 2025

Homework 2

This homework is to give a brief reminder of R, RStudio, and statistical topics covered in IEMS 303.

Note: The homework is scored out of 100 points. The problems add up to 90 points, while the remaining ten points will be graded according to a writing rubric, given at the end of the assignment.

R/RStudio installation If you have not installed R and RStudio, follow the installation instructions outlined in `https://posit.co/download/rstudio-desktop/`. You are strongly encouraged to use R Markdown to integrate text, code, images and mathematics or you can you use the latex code we provide.

Question 1. Linear Regression with Missing Data We consider the standard regression model

$$Y = \beta_0 + \beta_1 X + \epsilon, \quad \epsilon \sim N(0, \sigma^2), \tag{1}$$

with $X \in \mathbb{R}$. In this exercise, 10% of the generated Y values are randomly set to zero. That is, if we denote the observed response by $\tilde{Y}$, then

$$\tilde{Y} = \begin{cases} Y, & \text{with probability } 0.9, \\ 0, & \text{with probability } 0.1. \end{cases} \tag{2}$$

Exercise 1: Bias of the OLS Estimator. When we run OLS on the observed data $\tilde{Y}$, Can $E[\tilde{Y} \mid X]$ be written down as a linear function X? Prove that the bias in the estimators is then given by

$$\text{Bias}(\hat{\beta}_0) = \beta_0 - 0.9\beta_0 = \frac{\beta_0}{10}, \quad \text{Bias}(\hat{\beta}_1) = \beta_1 - 0.9\beta_1 = \frac{\beta_1}{10}.$$

This means both estimators are biased downward by 10% of the true parameter value.

Write down your answer in the `sol` environment.

Exercise 2: Log-Likelihood Formulation. To account for the zero-inflation, we use a likelihood that reflects the two different ways an observation can occur. For each observation i, show that the likelihood is given by:

$$L_i(\beta_0, \beta_1, \sigma) = \begin{cases} 0.1, & \text{if } y_i = 0, \\ 0.9 \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \beta_0 - \beta_1 x_i)^2}{2\sigma^2}\right), & \text{if } y_i \neq 0. \end{cases}$$

What is the full log-likelihood for the dataset? What is the new objective function you written down for the problem? What is the algorithm looks like?

Question 2. Linear Regression with Censored Data Suppose the true (latent) model is linear:

$$Y^* = \beta_0 + \beta_1 X + \epsilon, \quad \epsilon \sim N(0, \sigma^2). \tag{3}$$

However, instead of observing Y^*, we observe

$$Y = \max\{Y^*, 0\}. \tag{4}$$

This is a censored (or Tobit) model where values below 0 are censored to 0.

Exercise 1: Bias of the OLS Estimator. Compute the Expectation. Even though the latent relationship is linear, show that the observed conditional expectation becomes

$$E[Y \mid X] = (\beta_0 + \beta_1 X)\, \Phi\left(\frac{\beta_0 + \beta_1 X}{\sigma}\right) + \sigma\, \varphi\left(\frac{\beta_0 + \beta_1 X}{\sigma}\right),$$

which is not linear in X. Here φ and Φ denote the standard normal PDF and CDF, respectively. Explain why if one naively applies OLS to Y without accounting for the censoring, the estimated coefficients will be biased.

Exercise 2: Log-Likelihood Formulation. The proper likelihood accounts for whether an observation is censored or not. For each observation i, define $\mu_i = \beta_0 + \beta_1 X_i$. Then the likelihood is given by

$$L_i(\beta_0, \beta_1, \sigma) = \begin{cases} \Phi\left(\frac{-\mu_i}{\sigma}\right), & \text{if } Y_i = 0, \\ \dfrac{1}{\sigma}\varphi\left(\frac{Y_i - \mu_i}{\sigma}\right), & \text{if } Y_i > 0. \end{cases}$$

What is the full log-likelihood $\ell(\beta_0, \beta_1, \sigma)$ and the new loss function you have?

Exercise 3: Newton Method. We aim to use Newton method to minimize the nagtive log-likelihood $\mathrm{NLL}(\beta_0, \beta_1, \sigma) = -\ell(\beta_0, \beta_1, \sigma)$., we need first to compute the hessian and gradient.

3 a) Gradient Derivation For non-censored observations ($Y_i > 0$), the log-likelihood is $\ell_i = -\log\sigma - \frac{1}{2}\log(2\pi) - \frac{(Y_i-\mu_i)^2}{2\sigma^2}$. (why?) Differentiating with respect to β_j (with $j = 0, 1$):

$$\frac{\partial \ell_i}{\partial \beta_j} = \frac{Y_i - \mu_i}{\sigma^2}\frac{\partial \mu_i}{\partial \beta_j}.$$

Since $\mu_i = \beta_0 + \beta_1 X_i$, $\quad \frac{\partial \mu_i}{\partial \beta_0} = 1$, $\quad \frac{\partial \mu_i}{\partial \beta_1} = X_i$, the contribution to the gradient of the NLL (remember we take the negative derivative) is

$$-\frac{\partial \ell_i}{\partial \beta_j} = -\frac{Y_i - \mu_i}{\sigma^2}\frac{\partial \mu_i}{\partial \beta_j}.$$

Similarly, for censored observations ($Y_i = 0$), the log-likelihood is $\ell_i = \log\Phi\left(-\frac{\mu_i}{\sigma}\right)$. (why?) Let $z_i = -\frac{\mu_i}{\sigma}$. Then,using the same steps as before show that $\frac{\partial \ell_i}{\partial \beta_j} = -\frac{1}{\sigma}\frac{\varphi(z_i)}{\Phi(z_i)}\frac{\partial \mu_i}{\partial \beta_j}$ **(provide your calculation to check my result)** and the overall gradient of the NLL is

$$\nabla \mathrm{NLL}(\beta_0, \beta_1) = -\sum_{i:Y_i>0}\frac{Y_i - \mu_i}{\sigma^2}\begin{pmatrix}1\\X_i\end{pmatrix} + \sum_{i:Y_i=0}\frac{1}{\sigma}\frac{\varphi(z_i)}{\Phi(z_i)}\begin{pmatrix}1\\X_i\end{pmatrix}.$$

3 b) Hessian Derivation for Non-censored observations ($Y_i > 0$) For non-censored observations ($Y_i > 0$), show that the hessian is

$$H_i^{\text{non-cens}} = \frac{1}{\sigma^2}\begin{pmatrix}1\\X_i\end{pmatrix}\begin{pmatrix}1 & X_i\end{pmatrix}.$$

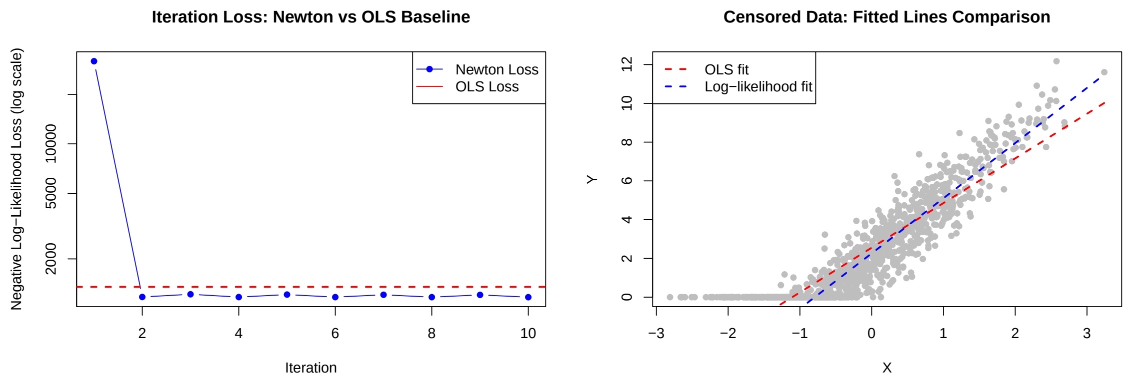
3 c) Hessian Derivation for Censored observations ($Y_i = 0$) For censored observations, we've showed the graident is $\frac{\partial \ell_i}{\partial \beta_j} = -\frac{1}{\sigma}\frac{\varphi(z_i)}{\Phi(z_i)}\frac{\partial \mu_i}{\partial \beta_j}$. Show that the Hessian contribution is

$$H_i^{\text{cens}} = \frac{1}{\sigma^2}\begin{pmatrix}1\\X_i\end{pmatrix}\begin{pmatrix}1 & X_i\end{pmatrix}\frac{\varphi(z_i)}{\Phi(z_i)^2}\Big(z_i\Phi(z_i) + \varphi(z_i)\Big)$$

based on the following Fact.

__**Fact.**__ If we define $r(z_i) = \frac{\varphi(z_i)}{\Phi(z_i)}$. Then its derivative is $r'(z_i) = -\frac{\varphi(z_i)}{\Phi(z_i)^2}\Big(z_i\Phi(z_i) + \varphi(z_i)\Big)$.

3 d) Complete the Newton Method's Code. Complete the following code using Newton method for NLL to do linear regression with censored data. If you successfully compelete the code, you are able to generate the following figures. Write down your findings.




```r
# ----------------------------
# Data Generation
# ----------------------------
set.seed(123)
n <- 1000
beta0_true <- 2; beta1_true <- 3; sigma <- 1
X <- rnorm(n)
epsilon <- rnorm(n, 0, sigma)
Y_star <- beta0_true + beta1_true * X + epsilon
# Censor at 0
Y <- pmax(Y_star, 0)

# ----------------------------
# Negative Log-Likelihood Function (NLL)
# ----------------------------
nll <- function(params, X, Y, sigma) {
  beta0 <- params[1]
  beta1 <- params[2]
  mu <- beta0 + beta1 * X
  # For censored observations: Y == 0
  cens <- (Y == 0)
  ll_cens <- sum( log(pnorm(-mu[cens] / sigma)) )
  # For non-censored observations: Y > 0
  noncens <- (Y > 0)
  ll_noncens <- sum( -log(sigma) - 0.5 * log(2*pi) - ((Y[noncens] - mu[
   noncens])^2) / (2*sigma^2) )
  return( - (ll_cens + ll_noncens) )
}

# ----------------------------
# Gradient of NLL
# ----------------------------
grad_nll <- function(params, X, Y, sigma) {
  beta0 <- params[1]
  beta1 <- params[2]
  mu <- beta0 + beta1 * X

  noncens <- (Y > 0)
  cens <- (Y == 0)
```

```

  # For non-censored observations:
  grad_non_cens_0 <- - sum((Y[noncens] - mu[noncens]) / sigma^2)
  grad_non_cens_1 <- - sum((Y[noncens] - mu[noncens]) * X[noncens] /
   sigma^2)

  # For censored observations:
  z <- -mu[cens] / sigma
  Phi_z <- pnorm(z) + 1e-10  # avoid division by zero
  factor <- dnorm(z) / (sigma * Phi_z)
  grad_cens_0 <- sum(factor)
  grad_cens_1 <- sum(factor * X[cens])

  grad0 <- grad_non_cens_0 + grad_cens_0
  grad1 <- grad_non_cens_1 + grad_cens_1

  return(c(grad0, grad1))
}

# ---------------------------
# Hessian of NLL
# ---------------------------
hessian_nll <- function(params, X, Y, sigma) {
  beta0 <- params[1]
  beta1 <- params[2]
  mu <- beta0 + beta1 * X

  H11 <- 0; H12 <- 0; H22 <- 0

  # Non-censored part:
  noncens <- (Y > 0)
  H11 <- H11 + sum( rep(1, sum(noncens)) / sigma^2 )
  H12 <- H12 + sum( X[noncens] / sigma^2 )
  H22 <- H22 + sum( (X[noncens]^2) / sigma^2 )

  # Censored part:
  cens <- (Y == 0)
  if(sum(cens) > 0){
  # ---------
  # !!!!!!!
  # PUT Your Code here
  # !!!!!!!
  # ---------
  # derivative of r(z)=phi(z)/Phi(z)
    rprime <- - (phi_z / (Phi_z^2)) * ( z * Phi_z + phi_z )

  }

  H <- matrix(c(H11, H12, H12, H22), nrow = 2)
  return(H)
}
```

```
# ----------------------------
# Newton's Method with Loss History
# ----------------------------
params_newton <- c(0, 0)   # initial guess for (beta0, beta1)
num_iter_newton <- 10
loss_history_newton <- numeric(num_iter_newton)

for(i in 1:num_iter_newton) {
  grad <- grad_nll(params_newton, X, Y, sigma)
  H <- hessian_nll(params_newton, X, Y, sigma)
  # ---------
  # !!!!!!!
  # PUT Your Code here
  # !!!!!!!
  # The code would look like params_newton <- ?
  # ---------

  loss_history_newton[i] <- nll(params_newton, X, Y, sigma)
}
newton_est <- params_newton
cat("Newton's Method Estimates (Beta0, Beta1):\n")
print(newton_est)


# ----------------------------
# OLS Estimation (ignoring censoring) as Baseline
# ----------------------------
ols_model <- lm(Y ~ X)
ols_est <- coef(ols_model)
cat("OLS Estimates (Beta0, Beta1):\n")
print(ols_est)
loss_ols <- nll(ols_est, X, Y, sigma)
cat("Negative Log-Likelihood Loss (OLS):\n")
print(loss_ols)

# ----------------------------
# Plot: Newton Iteration Loss vs OLS Baseline
# ----------------------------
# Combined Loss Plot with Log-Scale on Y-Axis: Newton, Gradient Descent
  , and OLS Baseline
plot(1:num_iter_newton, loss_history_newton, type='b', pch=16, col='
  blue', log="y",
     xlab='Iteration', ylab='Negative Log-Likelihood Loss (log scale)',
     main='Iteration Loss: Newton vs OLS Baseline')
abline(h=loss_ols, col='red', lwd=2, lty=2)
legend("topright", legend=c("Newton Loss", "OLS Loss"),
       col=c("blue", "red"), lty=c(1,1,2), pch=c(16, NA, NA))

# ----------------------------
# Plot: Fitted Lines Comparison
```

```
# ----------------------------
plot(X, Y, pch=16, col='grey',
     main='Censored Data: Fitted Lines Comparison',
     xlab='X', ylab='Y')
curve(ols_est[1] + ols_est[2]*x, add=TRUE, col='red', lwd=2, lty=2)
curve(newton_est[1] + newton_est[2]*x, add=TRUE, col='blue', lwd=2, lty
   =2)
legend("topleft", legend=c("OLS fit", "Log-likelihood fit"),
       col=c("red","blue","purple"), lty=2, lwd=2)
```

Listing 1. R code: Data generation, Newton's method, and plotting

Rubric (10)

- The text is laid out cleanly, with clear divisions between problems and sub-problems. The writing itself is well-organized, free of grammatical and other mechanical errors, and easy to follow.
- Questions which ask for a plot or table are answered with both the figure itself and the command (or commands) use to make the plot. Plots are carefully labeled, with informative and legible titles, axis labels.
- All quantitative and mathematical claims are supported by appropriate derivations, included in the text, or calculations in code. Numerical results are reported to appropriate precision.
- Code is either properly integrated with a tool like R Markdown or included as a separate R file. In the former case, both the knitted and the source file are included. In the latter case, the code is clearly divided into sections referring to particular problems. In either case, the code is indented, commented, and uses meaningful names.
- All parts of all problems are answered with actual coherent sentences, and never with raw computer code or its output. For full credit, all code runs, and the Markdown file knits (if applicable).