STAT 4360 (Introduction to Statistical Learning, Fall 2025)

Mini Project 2

Name: Van Nguyen

Question 1 (Wine Data)

(a) **Exploratory analysis.** Quality ranges from about 8 to 16. Region 3 wines show higher Quality on average. Clarity does not vary much, mostly equal to 1. Aroma, Body, Flavor, and Oakiness all rise with Quality (positive trend). Region looks important: wines from Region 1 and 2 are lower than Region 3.

(b) **Simple regressions.** Clarity and Oakiness: not significant (p-values $\approx$ 0.86 and 0.78). Aroma, Body, and Flavor: all highly significant ($p < 0.001$). Flavor has the strongest fit ($R^2 \approx 0.62$). Region: also significant. Region 3 wines score higher than Region 1, while Region 2 is lower. So overall, Aroma, Body, Flavor, and Region have a clear relationship with Quality, while Clarity and Oakiness do not.

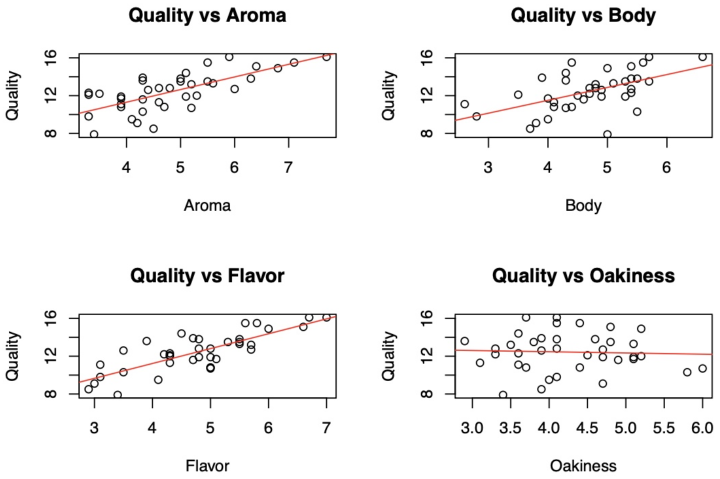

Figure 1.1: Scatterplots of Quality vs individual predictors with regression lines.

As shown in Figure 1.1, the regression lines confirm strong positive trends for Aroma, Body, and Flavor, while Clarity and Oakiness have almost flat lines (no effect).

(c) **Multiple regression.** Flavor is highly significant ($p < 0.001$). Region 2 has a significant negative effect ($p < 0.01$). Region 3 is borderline ($p \approx 0.066$). Clarity, Aroma, Body, and Oakiness are not significant. $R^2 \approx 0.84$, adjusted $R^2 \approx 0.80$. **Conclusion:** We can reject $H_0 : \beta_j = 0$ for Flavor and Region 2, but not for the others. Quality is mainly explained by Flavor and Region.

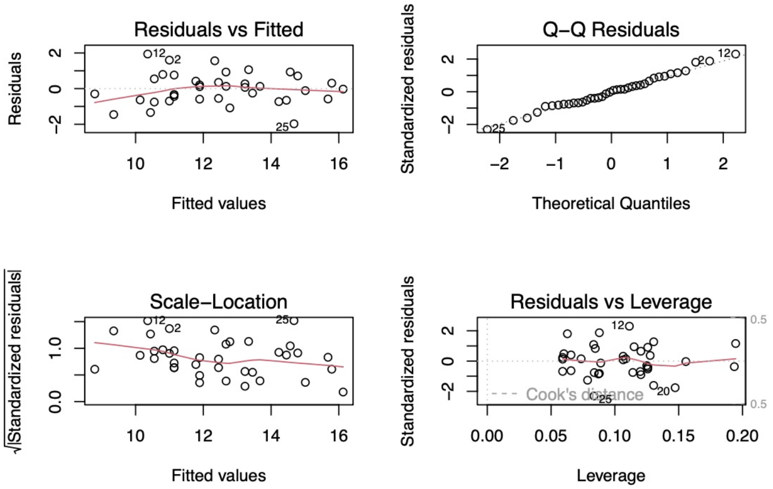

Figure 1.2: Residual diagnostics for the multiple regression model.

The diagnostic plots in Figure 1.2 show no major violations: residuals are roughly normal and evenly spread.

(d) **Reasonably good model.** From (c), only Flavor and Region were important, so I built a reduced model with these predictors. In the reduced model, Flavor is highly significant ($p < 0.001$). Region 2 (negative) and Region 3 (positive) are also significant. Testing an interaction (Flavor × Region) gave $p = 0.34$, so no strong evidence to include it. Diagnostic plots show residuals are roughly normal and spread evenly. $R^2 = 0.82$, almost as good as the full model but simpler.

$$Quality = \beta_0 + \beta_1 \cdot Flavor + \beta_2 \cdot Region2 + \beta_3 \cdot Region3 + \varepsilon$$

(e) **Final model equation.** Using coefficients from the reduced model:

$$\widehat{Quality} = 7.0943 + 1.1155 \cdot Flavor - 1.5335 \cdot Region2 + 1.2234 \cdot Region3$$

Interpretation: For Region 1 wines, Quality increases by about 1.12 for each 1-unit increase in Flavor. Region 2 wines score about 1.53 points lower, Region 3 wines about 1.22 points higher (holding Flavor constant).

(f) **Prediction.** For Region 1 wine with Flavor at its mean: predicted Quality = 12.41. 95% CI for mean response: [11.95, 12.88]. 95% PI for individual response: [10.54, 14.29]. Interpretation: On average, Region 1 wines have an average Flavor score of about 12.4, but individual wines vary ±2 points.

Question 2 (Diabetes Data)

(a) **Exploratory analysis.** From the summary, Outcome has a mean = 0.342, so about 34.2% of patients have diabetes. This matches the table counts (684/2000). On average, diabetics have higher Glucose, BMI, Insulin, and Age. Boxplots show strong separation in Glucose and BMI; BloodPressure and SkinThickness overlap more. Correlation matrix shows that the predictors are not highly correlated. Overall: Glucose and BMI are the strongest predictors, with Age and Insulin also contributing.

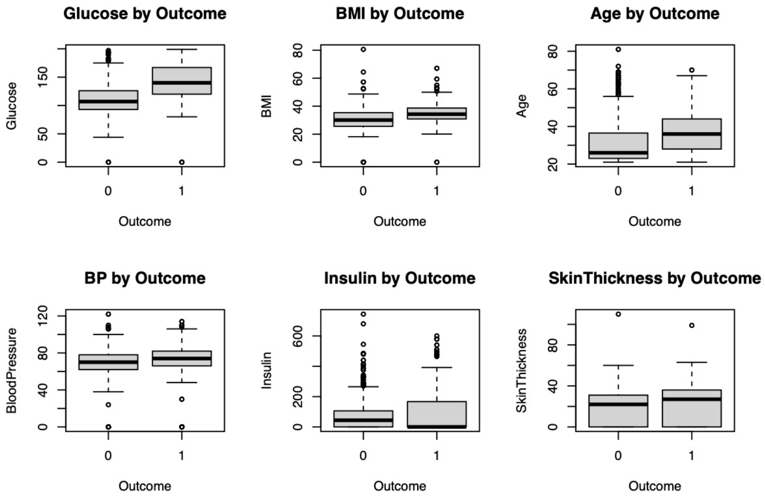

Figure 2.1: Boxplots of key predictors (Glucose, BMI, Age, Blood Pressure, Insulin, SkinThickness) by diabetes Outcome.

Figure 2.1 shows that diabetics tend to have higher Glucose, BMI, Insulin, and Age. In contrast, Blood Pressure and SkinThickness overlap heavily across groups.

(b) **LDA results.** Confusion matrix: 1174 true negatives, 298 false negatives, 142 false positives, 386 true positives. Misclassification rate = 22%. Sensitivity = 56.4%, Specificity = 89.2%. ROC curve AUC = 0.837. Observation: LDA performs reasonably well, with high specificity but lower sensitivity (misses some diabetics).

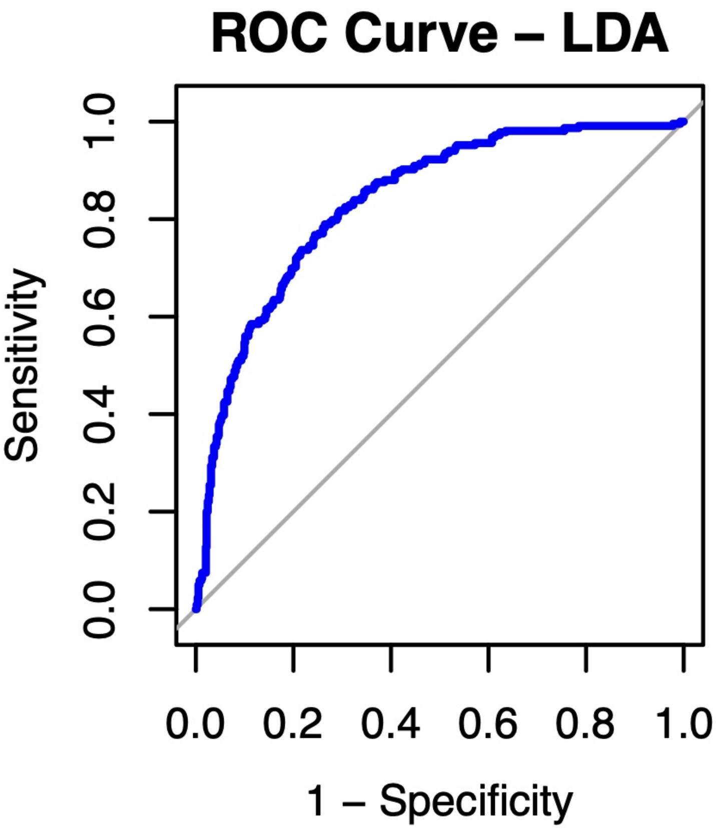

Figure 2.2: ROC curve for the LDA classifier.

The ROC curve in Figure 2.2 is well above the diagonal, consistent with the AUC of 0.837.

(c) **QDA results.** Confusion matrix: 1135 true negatives, 290 false negatives, 181 false positives, 394 true positives. Misclassification rate = 23.6%. Sensitivity = 57.6%, Specificity = 86.2%. ROC curve AUC = 0.835. Observation: QDA is slightly more sensitive but less specific than LDA. Accuracy is a bit worse overall.

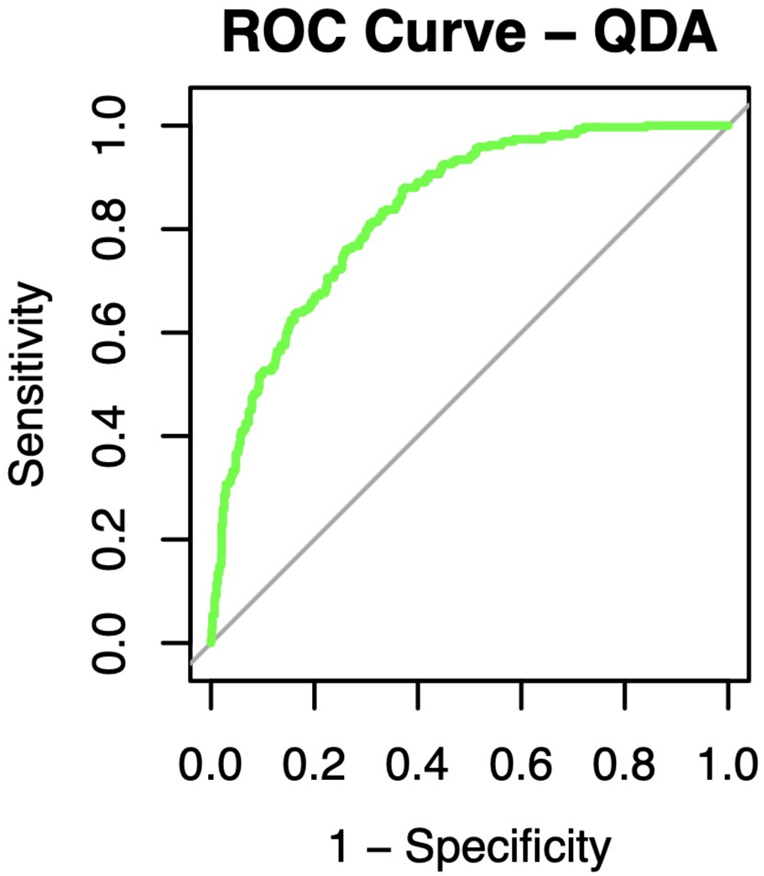

Figure 2.3: ROC curve for the QDA classifier.

The ROC curve in Figure 2.3 is similar, with AUC = 0.835, showing QDA has comparable performance to LDA.

(d) **Comparison.**

Method	Misclass. Rate	Sensitivity	Specificity	AUC
LDA	22.0%	56.4%	89.2%	0.837
QDA	23.6%	57.6%	86.2%	0.835

Both models have similar AUC (≈ 0.84). LDA has lower misclassification and higher specificity. QDA has slightly higher sensitivity. Since LDA is simpler, more stable, and slightly more accurate, I would prefer LDA here.

Bonus Question

(a) For $p = 10$, $\sigma = 1$, and $\mu = (1, 1, \ldots, 1)^T$, we simulated $N = 1000$ replications.

Results:

$$\text{Bias}_{\text{MLE}} \approx 0.09, \quad \text{Bias}_{\text{JS}} \approx 1.28$$
$$\text{Risk}_{\text{MLE}} \approx 9.84, \quad \text{Risk}_{\text{JS}} \approx 6.15$$

Observation: The MLE has almost no bias, while JS is biased. However, JS has much lower risk, showing the bias–variance tradeoff.

(b) We varied the signal strength $\mu = a \cdot (1, 1, \ldots, 1)^T$ for $a = 1, \ldots, 10$.

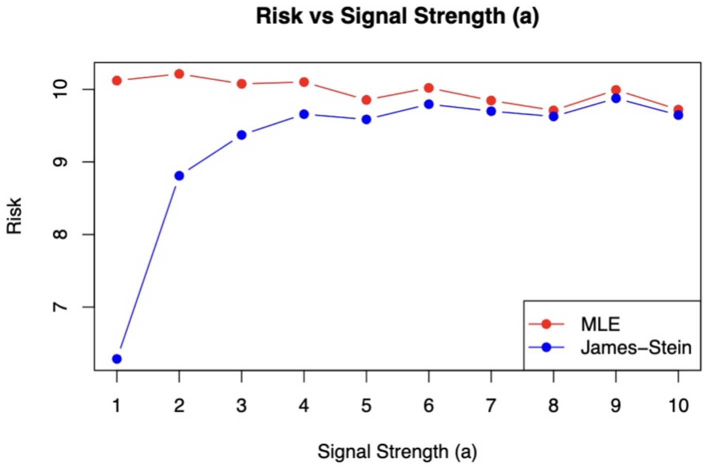

Figure 3.1: Risk vs Signal Strength a.

For small a, JS risk is much smaller than MLE. As a grows, the risks converge, so the advantage of JS disappears.

(c) We varied the noise level $\sigma \in \{0.1, 0.5, 1, 2, 5, 10\}$.

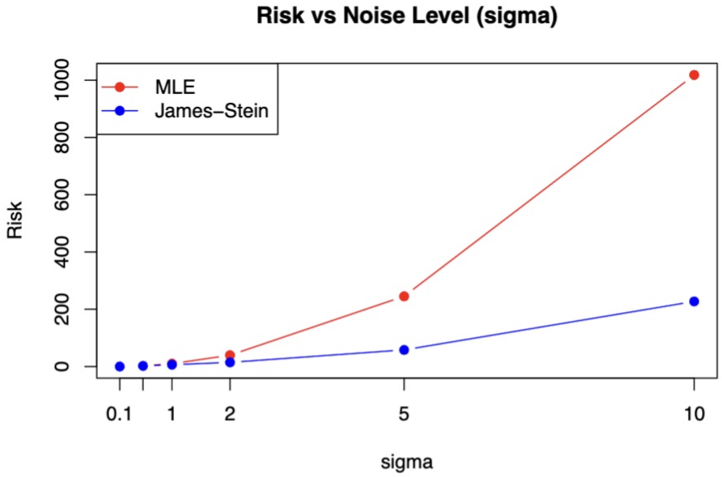

Figure 3.2: Risk vs Noise Level σ.

As σ increases, both risks grow, but MLE grows much faster. JS is more robust in high noise, confirming the theory that JS dominates MLE when $p \geq 3$.

R Code

```
---
title: ""
output: pdf_document
---

‘‘‘{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
‘‘‘

\begin{center}
\textbf{\huge Mini Project 2}\\[1em]  % Change size with \
   Large, \large, etc.
\textbf{\large of}\\[1em]
\textbf{\Large Stat 4360}
\end{center}

‘‘‘{r, echo=TRUE}

setwd("/Users/vannguyen/Downloads")
wine <- read.table("wine.txt", header = TRUE, sep = "\t")


## Question 1(a)
# Treat Region as a factor
wine$Region <- as.factor(wine$Region)

# Quick overview
str(wine)
summary(wine)

# Pairwise scatterplots to see relationships
pairs(wine[, c("Quality","Clarity","Aroma","Body","Flavor","
   Oakiness")])

# Boxplot of Quality by Region
boxplot(Quality ˜ Region, data = wine,
        main = "Wine␣Quality␣by␣Region",
        xlab = "Region", ylab = "Quality")


## Question 1(b)
# Simple linear regressions of Quality on each predictor

# Clarity
fit_clarity <- lm(Quality ˜ Clarity, data = wine)
summary(fit_clarity)

# Aroma
fit_aroma <- lm(Quality ˜ Aroma, data = wine)
summary(fit_aroma)
```

```
# Body
fit_body <- lm(Quality ~ Body, data = wine)
summary(fit_body)

# Flavor
fit_flavor <- lm(Quality ~ Flavor, data = wine)
summary(fit_flavor)

# Oakiness
fit_oakiness <- lm(Quality ~ Oakiness, data = wine)
summary(fit_oakiness)

# Region (qualitative predictor)
fit_region <- lm(Quality ~ Region, data = wine)
summary(fit_region)



# Scatterplots with regression lines
par(mfrow=c(2,2))
plot(Quality ~ Aroma, data=wine, main="Quality␣vs␣Aroma")
abline(lm(Quality ~ Aroma, data=wine), col="red")

plot(Quality ~ Body, data=wine, main="Quality␣vs␣Body")
abline(lm(Quality ~ Body, data=wine), col="red")

plot(Quality ~ Flavor, data=wine, main="Quality␣vs␣Flavor")
abline(lm(Quality ~ Flavor, data=wine), col="red")

plot(Quality ~ Oakiness, data=wine, main="Quality␣vs␣Oakiness
   ")
abline(lm(Quality ~ Oakiness, data=wine), col="red")


## Question 1(c)
# Multiple regression with all predictors
fit_all <- lm(Quality ~ Clarity + Aroma + Body + Flavor +
   Oakiness + Region,
              data = wine)
summary(fit_all)


## Question 1(d)
# Reduced model
```

~~~
fit_reduced <- lm(Quality ~ Flavor + Region, data = wine)
summary(fit_reduced)

# Interaction check
fit_interaction <- lm(Quality ~ Flavor * Region, data = wine)
summary(fit_interaction)
anova(fit_reduced, fit_interaction)

# Residual diagnostics
par(mfrow=c(2,2))
plot(fit_reduced)

# Added-variable plots
library(car)
avPlots(fit_reduced, main="Added␣Variable␣Plots␣for␣
   Predictors")




## Question 1(f)

# Use the reduced model from part (d)
fit_reduced <- lm(Quality ~ Flavor + Region, data = wine)

# Mean Flavor (from dataset)
mean_flavor <- mean(wine$Flavor)

# Create new data for Region 1 with Flavor = mean
new_obs <- data.frame(Flavor = mean_flavor,
                      Region = factor("1", levels = c("1","2"
                         ,"3")))

# Prediction and confidence intervals
predict(fit_reduced, newdata = new_obs,
        interval = "confidence", level = 0.95)

predict(fit_reduced, newdata = new_obs,
        interval = "prediction", level = 0.95)
```

```{r, echo=TRUE}
library(corrplot)
library(MASS)
library(pROC)
~~~

```
setwd("/Users/vannguyen/Downloads")
diabetes <- read.csv("diabetes.csv")

## Question 2(a)
# Quick structure and summary
str(diabetes)
summary(diabetes)

# Check distribution of the response
table(diabetes$Outcome)

# Means by Outcome (to see group differences)
aggregate(. ~ Outcome, data = diabetes, mean)

# Correlation matrix of numeric predictors
corrplot(cor(diabetes[,-9]), method="color", type="upper")

# Boxplots of key predictors by Outcome
par(mfrow=c(2,3))
boxplot(Glucose ~ Outcome, data=diabetes, main="Glucose␣by␣
   Outcome")
boxplot(BMI ~ Outcome, data=diabetes, main="BMI␣by␣Outcome")
boxplot(Age ~ Outcome, data=diabetes, main="Age␣by␣Outcome")
boxplot(BloodPressure ~ Outcome, data=diabetes, main="BP␣by␣
   Outcome")
boxplot(Insulin ~ Outcome, data=diabetes, main="Insulin␣by␣
   Outcome")
boxplot(SkinThickness ~ Outcome, data=diabetes, main="
   SkinThickness␣by␣Outcome")



## Question 2(b)
# LDA model
fit_lda <- lda(Outcome ~ ., data=diabetes)

# Predict with LDA
pred_lda <- predict(fit_lda)

# Classify using 0.5 cutoff
lda_class <- ifelse(pred_lda$posterior[,2] > 0.5, 1, 0)

# Confusion matrix
table(Predicted = lda_class, Actual = diabetes$Outcome)
```

```
# Misclassification rate
mean(lda_class != diabetes$Outcome)

# Sensitivity and Specificity
sensitivity <- sum(lda_class==1 & diabetes$Outcome==1) / sum(
   diabetes$Outcome==1)
specificity <- sum(lda_class==0 & diabetes$Outcome==0) / sum(
   diabetes$Outcome==0)
sensitivity; specificity


# ROC for LDA
roc_obj <- roc(diabetes$Outcome, pred_lda$posterior[,2],
   direction="<")

plot(roc_obj, legacy.axes=TRUE, col="blue", lwd=2,
     main="ROC␣Curve␣-␣LDA")
auc(roc_obj)



## Question 2(c)
# QDA model
fit_qda <- qda(Outcome ~ ., data=diabetes)

# Predict with QDA
pred_qda <- predict(fit_qda)

# Classify with 0.5 cutoff
qda_class <- ifelse(pred_qda$posterior[,2] > 0.5, 1, 0)

# Confusion matrix
table(Predicted = qda_class, Actual = diabetes$Outcome)

# Misclassification rate
mean(qda_class != diabetes$Outcome)

# Sensitivity and Specificity
sensitivity_qda <- sum(qda_class==1 & diabetes$Outcome==1) /
   sum(diabetes$Outcome==1)
specificity_qda <- sum(qda_class==0 & diabetes$Outcome==0) /
   sum(diabetes$Outcome==0)
sensitivity_qda; specificity_qda
```

```
# ROC curve for QDA
roc_qda <- roc(diabetes$Outcome, pred_qda$posterior[,2],
   direction="<")
plot(roc_qda, legacy.axes=TRUE, col="green", lwd=2,
     main="ROC␣Curve␣-␣QDA")
auc(roc_qda)
‘‘‘

‘‘‘{r, echo=TRUE}
## Bonus Question (a)
p <- 10
sigma <- 1
mu <- rep(1, p)
N <- 1000

# Storage
mu_mle <- matrix(0, nrow=N, ncol=p)
mu_js  <- matrix(0, nrow=N, ncol=p)

# Simulation
for(i in 1:N){
  Y <- MASS::mvrnorm(1, mu, sigma^2 * diag(p))
  # MLE
  mu_mle[i,] <- Y
  # James-Stein shrinkage
  shrink <- 1 - ( (p-2)*sigma^2 ) / sum(Y^2)
  mu_js[i,] <- shrink * Y
}

# Compute bias and risk
bias_mle <- norm(colMeans(mu_mle) - mu, type="2")
bias_js  <- norm(colMeans(mu_js)  - mu, type="2")

risk_mle <- mean(rowSums((mu_mle - matrix(mu, nrow=N, ncol=p,
    byrow=TRUE))^2))
risk_js  <- mean(rowSums((mu_js  - matrix(mu, nrow=N, ncol=p,
    byrow=TRUE))^2))

bias_mle; bias_js
risk_mle; risk_js


## Bonus Question (b)
# Risk vs Signal Strength (a)
a.values <- 1:10
```

```r
risk_mle_a <- numeric(length(a.values))
risk_js_a  <- numeric(length(a.values))

for(k in 1:length(a.values)){
  a <- a.values[k]
  mu <- rep(a, p)   # mean vector changes with a

  mu_mle <- matrix(0, nrow=N, ncol=p)
  mu_js  <- matrix(0, nrow=N, ncol=p)

  for(i in 1:N){
    Y <- MASS::mvrnorm(1, mu, sigma^2 * diag(p))
    mu_mle[i,] <- Y
    shrink <- 1 - ((p-2)*sigma^2) / sum(Y^2)
    mu_js[i,] <- shrink * Y
  }

  # record risk
  risk_mle_a[k] <- mean(rowSums((mu_mle - mu)^2))
  risk_js_a[k]  <- mean(rowSums((mu_js  - mu)^2))
}

# Plot
plot(a.values, risk_mle_a, type="b", col="red", pch=19,
     ylim=range(c(risk_mle_a, risk_js_a)),
     xlab="Signal␣Strength␣(a)", ylab="Risk",
     main="Risk␣vs␣Signal␣Strength␣(a)", xaxt="n")
lines(a.values, risk_js_a, type="b", col="blue", pch=19)
legend("bottomright", legend=c("MLE", "James Stein"),
       col=c("red","blue"), pch=19, lty=1)
axis(1, at=a.values, labels=a.values)

## Bonus Question (c)
# Risk vs Noise Level (sigma)
sigma.values <- c(0.1, 0.5, 1, 2, 5, 10)
risk_mle_s <- numeric(length(sigma.values))
risk_js_s  <- numeric(length(sigma.values))
mu <- rep(1, p)   # reset mean vector

for(k in 1:length(sigma.values)){
  sigma <- sigma.values[k]

  mu_mle <- matrix(0, nrow=N, ncol=p)
  mu_js  <- matrix(0, nrow=N, ncol=p)
```

~~~
  for(i in 1:N){
    Y <- MASS::mvrnorm(1, mu, sigma^2 * diag(p))
    mu_mle[i,] <- Y
    shrink <- 1 - ((p-2)*sigma^2) / sum(Y^2)
    mu_js[i,] <- shrink * Y
  }

  # record risk
  risk_mle_s[k] <- mean(rowSums((mu_mle - mu)^2))
  risk_js_s[k]  <- mean(rowSums((mu_js  - mu)^2))
}

# Plot
plot(sigma.values, risk_mle_s, type="b", col="red", pch=19,
     ylim=range(c(risk_mle_s, risk_js_s)),
     xlab="sigma", ylab="Risk",
     main="Risk␣vs␣Noise␣Level␣(sigma)", xaxt="n")
lines(sigma.values, risk_js_s, type="b", col="blue", pch=19)
legend("topleft", legend=c("MLE", "James-Stein"),
       col=c("red","blue"), pch=19, lty=1)
axis(1, at=sigma.values, labels=sigma.values)

```
~~~