rm(list = ls())

library(ggplot2)

library(ggrepel)

# Original data

x <- c(3.927,8.273,3.050,3.069,2.446,17.692,13.176,0.448,4.202,0.745,0.468,0.590,0.413,3.228,1.446,0.665)

xav <- c(-131.9,-1075,-760.6,-156.2,-226.7,-150,-144.7,-482,-515.6,-456.4,-1290.2,-168,-177.5, 226.3,- 324.3,- 991)

y <- c(369,3717,2811,566,784,488,446,1313,1351,1353,4608,629,607, 726, 695, 2471)

# Labels for outliers

labels <- c("A0A7P0T9Z4", "A0A087X0U3", "A0A7P0T8Q1", "E7EUV6", "E7EUS7", "A0A7P0T9A4", "A0A7P0T807", "C9J9N5", "A0A2R8Y4D1", "A0A2R8Y642", "A0A2R8Y4N0", "A0A2R8YHI4", "H0YEB1","H0Y7R2", "M0R106","F1T0D4")

# Create a data frame

df <- data.frame(x = x, y = y, label = labels)

# Calculate Pearson correlation

cor\_result <- cor.test(x, y, alternative = "two.sided", method = "pearson", exact=FALSE)

print(cor\_result)

# Function to identify pair outliers using Tukey's fence method

identify\_outliers <- function(x, y, k = 1.5) {

# Calculate Mahalanobis distances

center <- c(mean(x), mean(y))

cov\_matrix <- cov(cbind(x, y))

mahalanobis\_dist <- mahalanobis(cbind(x, y), center, cov\_matrix)

# Calculate Tukey's fences

q1 <- quantile(mahalanobis\_dist, 0.25)

q3 <- quantile(mahalanobis\_dist, 0.75)

iqr <- q3 - q1

lower\_fence <- q1 - k \* iqr

upper\_fence <- q3 + k \* iqr

# Identify outliers

outliers <- mahalanobis\_dist > upper\_fence | mahalanobis\_dist < lower\_fence

return(outliers)

}

# Identify outliers

df$outlier <- identify\_outliers(df$x, df$y)

# Print number of outliers

num\_outliers <- sum(df$outlier)

cat("Number of pair outliers detected:", num\_outliers, "\n")

# Fit linear regression model

lm\_model <- lm(y ~ x, data = df)

# Create ggplot

p <- ggplot(df, aes(x = x, y = y)) +

geom\_point(aes(color = outlier, shape = outlier), size = 2.5) + # Smaller data points

geom\_smooth(method = "lm", se = FALSE, color = "darkgreen", linetype = "solid", size = 0.5) + # Continuous line

scale\_color\_manual(values = c("steelblue", "red"),

labels = c("Normal", "Outlier"),

name = "Data Points") +

scale\_shape\_manual(values = c(16, 17),

labels = c("Normal", "Outlier"),

name = "Data Points") +

labs(title = " ",

x = " **RNA structure MFE (kcal/mol)**", y = " **cDNA Length (nt)**") +

theme\_minimal(base\_size = 13) +

theme(

legend.position = c(1, 1), # Move legend to upper right corner

legend.justification = c(1, 1), # Align legend to top-right

legend.box.just = "right",

legend.margin = margin(2, 2, 5, 5),

legend.background = element\_rect(fill = "white", color = "black", size = 0.5),

panel.grid.major = element\_line(color = "gray80", size = 0.2),

panel.grid.minor = element\_line(color = "gray90", size = 0.05)

)

# Add labels to outliers using ggrepel

p <- p + geom\_text\_repel(

data = subset(df, outlier),

aes(label = label),

box.padding = 0.5,

point.padding = 0.5,

force = 2,

max.overlaps = Inf,

size = 3.5

)

# Add regression equation

eq <- paste0("y = ", round(coef(lm\_model)[2], 4), "x + ", round(coef(lm\_model)[1], 4))

p <- p + annotate("text", x = -Inf, y = Inf, label = eq, hjust = -1.1, vjust = 16, size = 4)

# Extend the plot limits (less than before)

x\_range <- max(df$x) - min(df$x)

y\_range <- max(df$y) - min(df$y)

p <- p + coord\_cartesian(

xlim = c(min(df$x) - 0.01\* x\_range, max(df$x) + 0.01 \* x\_range),

ylim = c(min(df$y) - 0.01 \* y\_range, max(df$y) + 0.01 \* y\_range)

)

# Add Pearson correlation to the plot

cor\_text <- paste("Pearson correlation:", round(cor\_result$estimate, 4))

p <- p + annotate("text", x = -Inf, y = Inf, label = cor\_text, hjust = -0.8, vjust = 12.5, size = 4.5)

# Display the plot

print(p)