Correlation Analysis of 3 Sound Sensors: 1inch Hole, 6mm Hole, No Hole

KRIISH HATE

2025-03-20

```r
knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# R Script: Sound Sensor Analysis
# ================================
# This script reads sound sensor data from a TXT file,
# converts raw ADC values to dB, plots a correlation graph with a best-fit li
ne,
# and computes linear regression parameters, displaying them on the graph.
# Sensor 1 (open) 2 (small hole 6mm) 3 (large hole 1 inch)

# ------------------------------
# Load necessary libraries
# ------------------------------
# The ggplot2 library is used to create visualizations
library(ggplot2)
```

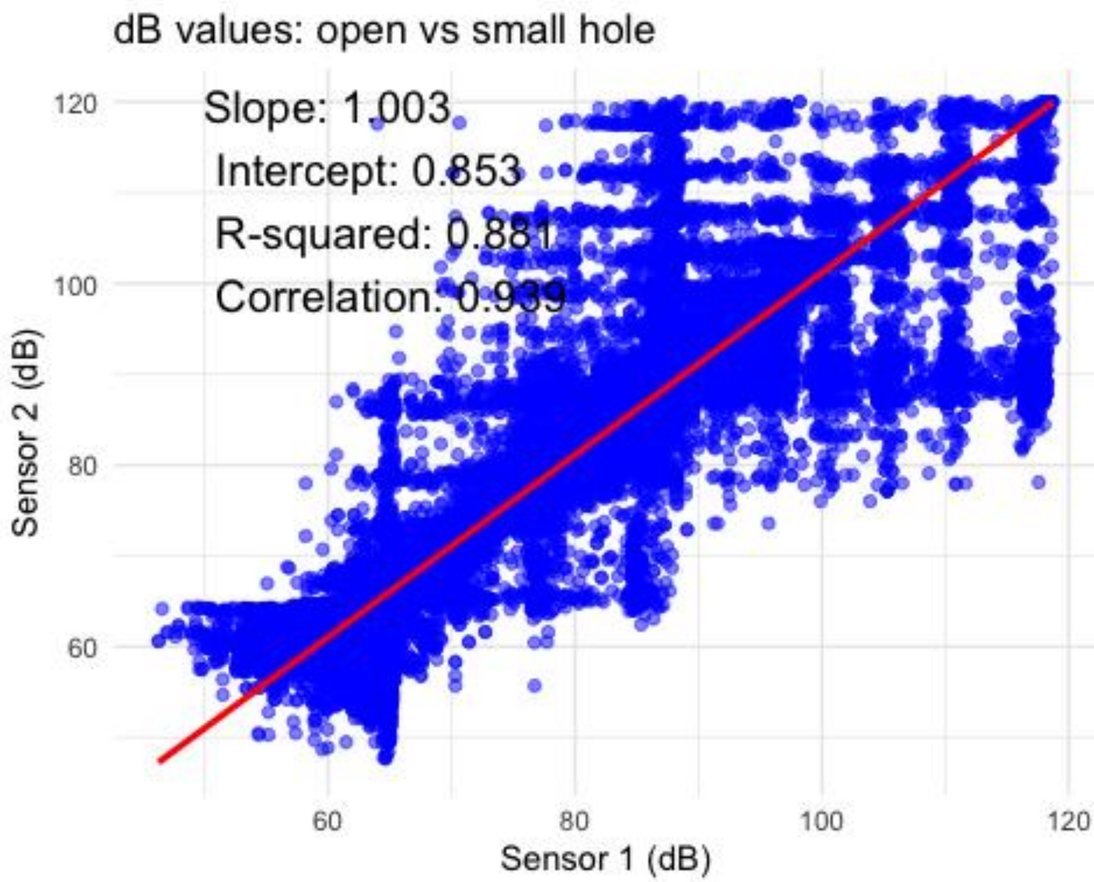
```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```r
# ------------------------------
# Read the data from the TXT file
# ------------------------------
# Define the file path for the dataset
file_path <- "LOG.TXT"  # Ensure this file is in the working directory

# Read the file into a data frame
# read.table() reads a comma-separated file without a header
# The resulting data frame will have four columns: Timestamp, Sensor1, Sensor
2, Sensor3
data <- read.table(file_path, header = FALSE, sep = ",",
                   col.names = c("Timestamp", "Sensor1", "Sensor2", "Sensor3"
))

# ------------------------------
# Define constants
# ------------------------------
# Set the reference voltage used for ADC conversion
VREF <- 6.14

# ------------------------------
# Convert raw ADC values to dB
```

```r
# ------------------------------
# Convert raw ADC values to voltage
# Formula: ( (read value) / (ADC max value) ) * voltage reference
data$Sensor1_Voltage <- ((data$Sensor1) / (32768.0)) * VREF
data$Sensor2_Voltage <- ((data$Sensor2) / (32768.0)) * VREF
data$Sensor3_Voltage <- ((data$Sensor3) / (32768.0)) * VREF

# ------------------------------
# Convert voltage to decibels (dB)
# ------------------------------
# Formula: dB = Voltage * 50 (Scaling factor)
data$Sensor1_dB <- data$Sensor1_Voltage * 50.0
data$Sensor2_dB <- data$Sensor2_Voltage * 50.0
data$Sensor3_dB <- data$Sensor3_Voltage * 50.0

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ===============================
# Plot Sensor 1 vs. Sensor 2 dB Values with Statistics
# ===============================
# ------------------------------
# Compute linear regression
# ------------------------------
# Fit a linear regression model where:
# Sensor2_dB = slope * Sensor1_dB + intercept
model <- lm(Sensor2_dB ~ Sensor1_dB, data = data)

# Display the regression summary in the console
summary(model)

##
## Call:
## lm(formula = Sensor2_dB ~ Sensor1_dB, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -40.698  -0.666   0.290   1.099  52.528
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.8526592  0.0592494   14.39   <2e-16 ***
## Sensor1_dB  1.0034210  0.0007542 1330.37   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.076 on 238508 degrees of freedom
## Multiple R-squared:  0.8812, Adjusted R-squared:  0.8812
## F-statistic: 1.77e+06 on 1 and 238508 DF,  p-value: < 2.2e-16

# Extract regression parameters
# Extract the slope (coefficient for Sensor1_dB)
```

```
slope <- coef(model)[2]
# Extract the y-intercept
intercept <- coef(model)[1]
# Compute the R-squared value, indicating model fit
r_squared <- summary(model)$r.squared
# Compute correlation coefficient as square root of R-squared
correlation_coefficient <- sqrt(r_squared)

# ------------------------------
# Create text annotation for the plot
# ------------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# ------------------------------
# Determine annotation position on the plot
# ------------------------------
# Set the x-position for annotation as 5% from the left side
text_x_pos <- min(data$Sensor1_dB) + (max(data$Sensor1_dB) - min(data$Sensor1
_dB)) * 0.05
# Set the y-position for annotation as 85% from the bottom
text_y_pos <- min(data$Sensor2_dB) + (max(data$Sensor2_dB) - min(data$Sensor2
_dB)) * 0.85

# ------------------------------
# Create a scatter plot with best-fit line
# ------------------------------
# Generate a scatter plot using ggplot2
plot <- ggplot(data, aes(x = Sensor1_dB, y = Sensor2_dB)) +
  # Add scatter points in blue with 50% transparency
  geom_point(alpha = 0.5, color = "blue") +
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("dB values: open vs small hole ") +
  # Set x-axis label
  xlab("Sensor 1 (dB)") +
  # Set y-axis label
  ylab("Sensor 2 (dB)") +
  # Use a minimal theme for cleaner visualization
  theme_minimal()
```

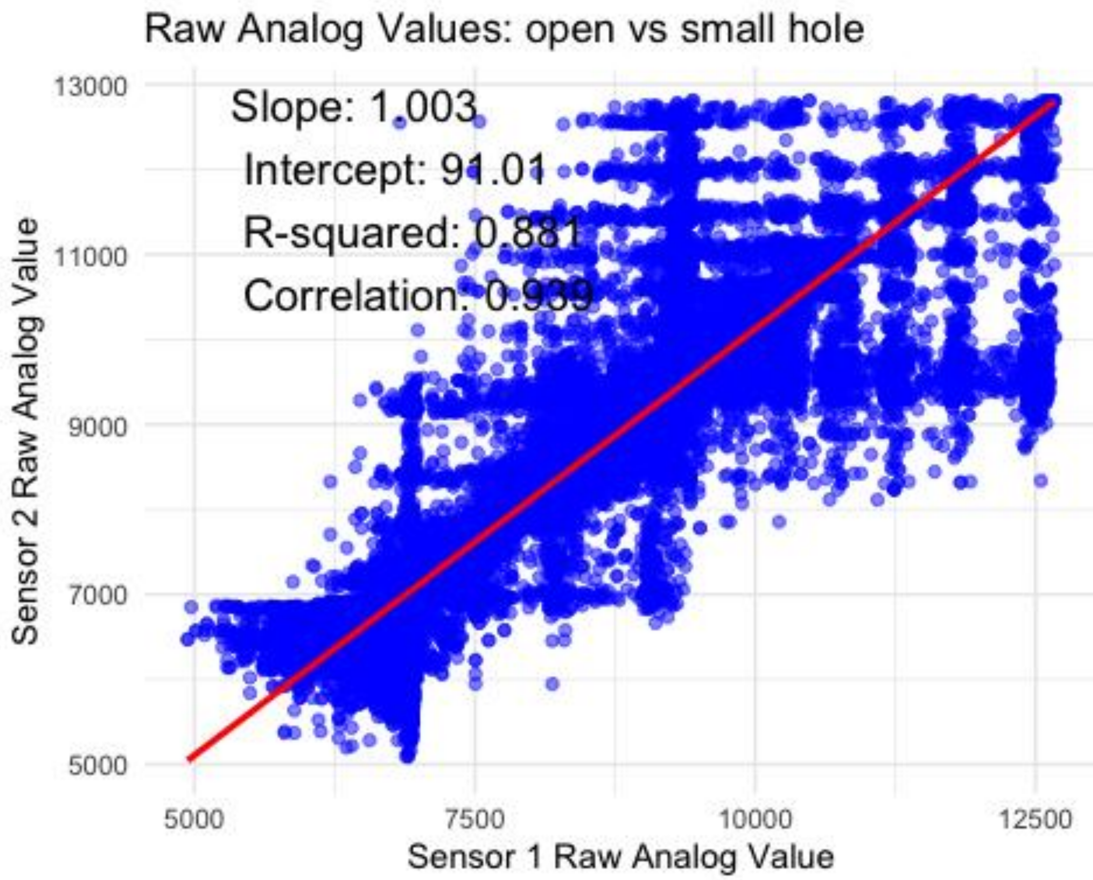
```
# -------------------------------
# Display the plot
# -------------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```




```
# -------------------------------
# Save the plot as a JPEG file
# -------------------------------
ggsave("Sensor_Correlation_dB.jpeg", plot = plot, width = 6, height = 4, dpi
= 300)

## `geom_smooth()` using formula = 'y ~ x'

# -------------------------------
# Print regression results to the console
# -------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results ---\n")
```

```
##
## --- Regression Results ---

cat("Slope:", slope, "\n")

## Slope: 1.003421

cat("Intercept:", intercept, "\n")

## Intercept: 0.8526592

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8812448

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9387464

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Raw Sensor 1 vs. Sensor 2 Analog Values with Statistics
# ================================
# -------------------------------
# Compute Linear Regression
# -------------------------------
# Fit a linear regression model where:
# Sensor2 = slope * Sensor1 + intercept
model <- lm(Sensor2 ~ Sensor1, data = data)

# Extract regression parameters
slope <- coef(model)[2]  # The slope of the regression line
intercept <- coef(model)[1]  # The y-intercept of the regression line
r_squared <- summary(model)$r.squared  # R-squared value, indicating model fi
t
correlation_coefficient <- sqrt(r_squared)  # Square root of R-squared gives
correlation coefficient

# -------------------------------
# Create text annotation for the plot
# -------------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# Determine annotation position for displaying text on the plot
text_x_pos <- min(data$Sensor1) + (max(data$Sensor1) - min(data$Sensor1)) * 0
```

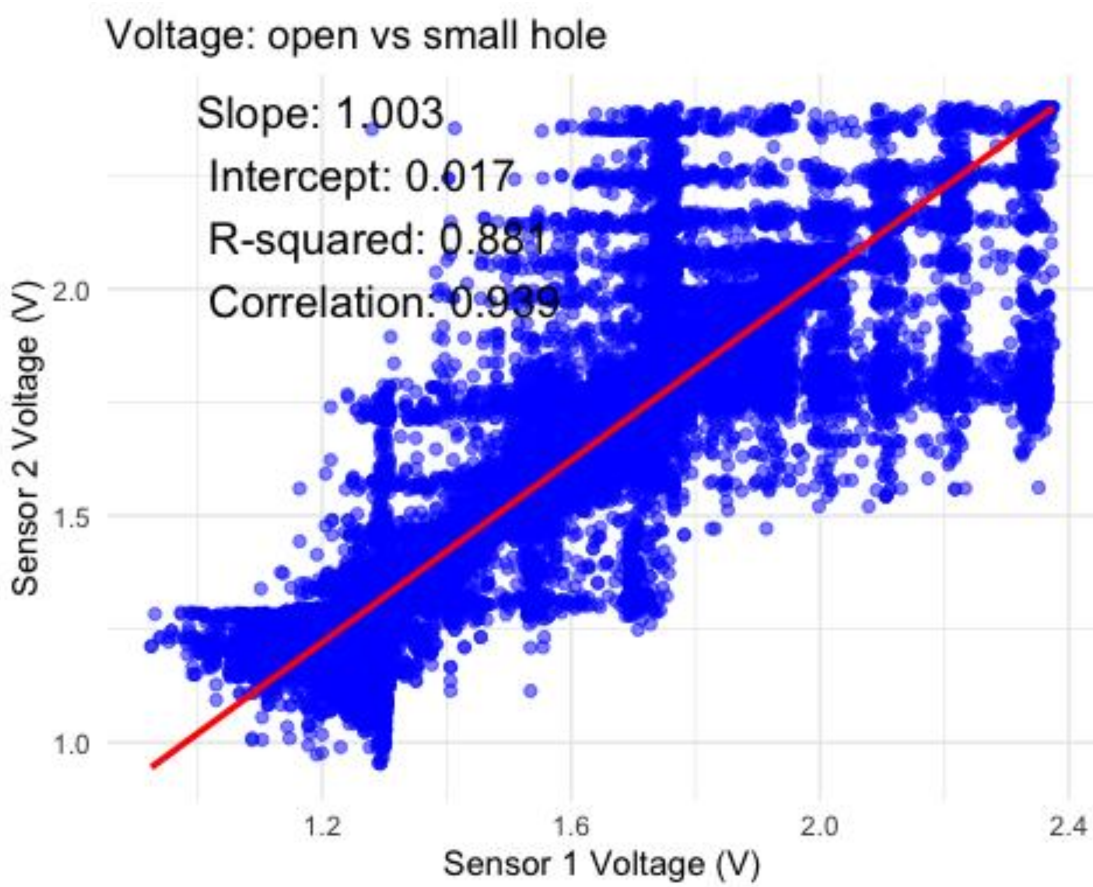
```
.05  # 5% from the left
text_y_pos <- min(data$Sensor2) + (max(data$Sensor2) - min(data$Sensor2)) * 0
.85  # 85% from the bottom

# ------------------------------
# Create a scatter plot of raw analog values with best-fit line
# ------------------------------
plot <- ggplot(data, aes(x = Sensor1, y = Sensor2)) +
  # Add scatter points in blue with 50% transparency
  geom_point(alpha = 0.5, color = "blue") +
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("Raw Analog Values: open vs small hole ") +
  # Set x-axis label
  xlab("Sensor 1 Raw Analog Value") +
  # Set y-axis label
  ylab("Sensor 2 Raw Analog Value") +
  # Use a minimal theme for a clean layout
  theme_minimal()

# ------------------------------
# Display the plot
# ------------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```




```r
# ------------------------------
# Save the plot as a JPEG file
# ------------------------------
ggsave("Sensor_Correlation_rawlongvalue.jpeg", plot = plot, width = 6, height
= 4, dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# ------------------------------
# Print regression results to the console
# ------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results ---\n")

##
## --- Regression Results ---

cat("Slope:", slope, "\n")

## Slope: 1.003421

cat("Intercept:", intercept, "\n")

## Intercept: 91.00957
```

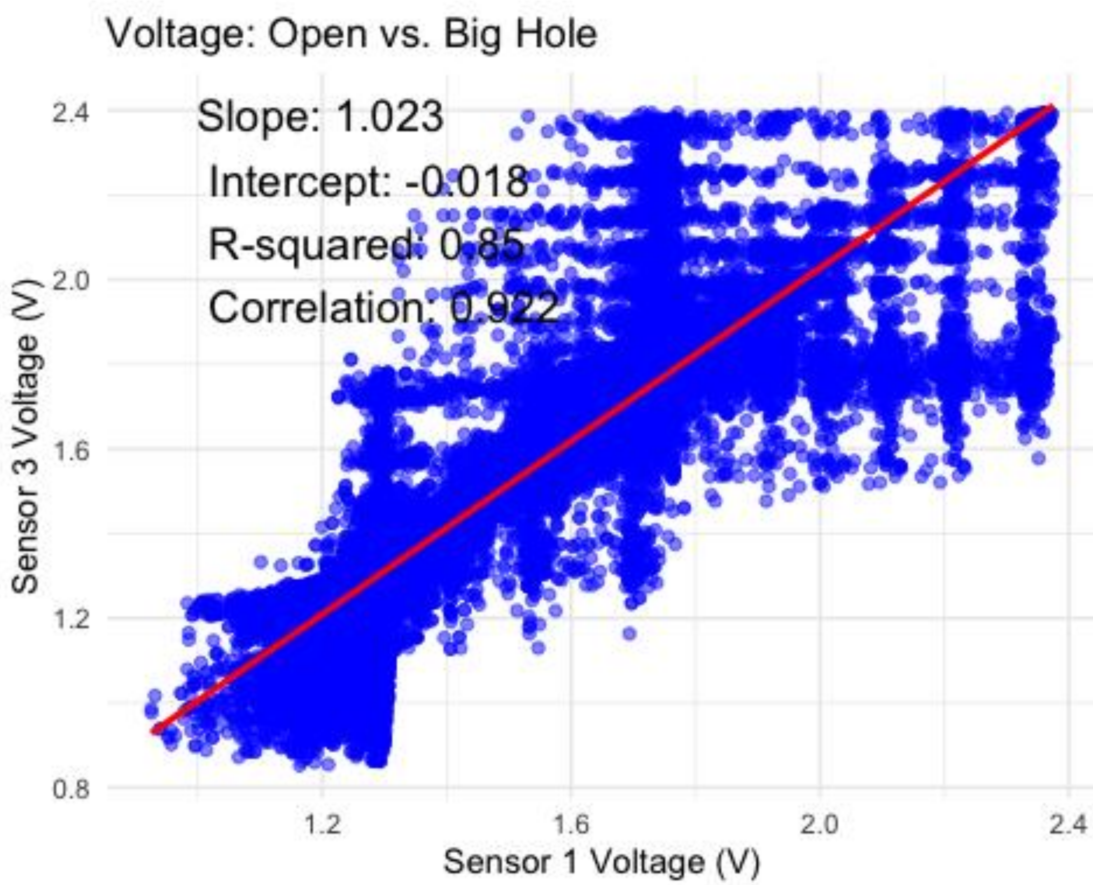
```r
cat("R-squared:", r_squared, "\n")

## R-squared: 0.8812448

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9387464

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 1 Voltage vs. Sensor 2 Voltage with Statistics
# ================================
# ------------------------------
# Compute Linear Regression
# ------------------------------
# Fit a linear regression model where:
# Sensor2 Voltage = slope * Sensor1 Voltage + intercept
model <- lm(Sensor2_Voltage ~ Sensor1_Voltage, data = data)

# Extract regression parameters
slope <- coef(model)[2]  # The slope of the regression line
intercept <- coef(model)[1]  # The y-intercept of the regression line
r_squared <- summary(model)$r.squared  # R-squared value, indicating model fi
t
correlation_coefficient <- sqrt(r_squared)  # Square root of R-squared gives
correlation coefficient

# ------------------------------
# Create text annotation for the plot
# ------------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# Determine annotation position for displaying text on the plot
text_x_pos <- min(data$Sensor1_Voltage) + (max(data$Sensor1_Voltage) - min(da
ta$Sensor1_Voltage)) * 0.05  # 5% from the left
text_y_pos <- min(data$Sensor2_Voltage) + (max(data$Sensor2_Voltage) - min(da
ta$Sensor2_Voltage)) * 0.85  # 85% from the bottom

# ==============================
# plot of Sensor 1 Voltage vs. Sensor 2 Voltage with best-fit line
# ==============================
plot <- ggplot(data, aes(x = Sensor1_Voltage, y = Sensor2_Voltage)) +
  # Add scatter points in blue with 50% transparency
  geom_point(alpha = 0.5, color = "blue") +
```

```
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("Voltage: open vs small hole") +
  # Set x-axis label
  xlab("Sensor 1 Voltage (V)") +
  # Set y-axis label
  ylab("Sensor 2 Voltage (V)") +
  # Use a minimal theme for a clean layout
  theme_minimal()

# ------------------------------
# Display the plot
# ------------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```



```r
# ------------------------------
# Save the plot as a JPEG file
# ------------------------------
ggsave("Sensor_Correlation_voltage.jpeg", plot = plot, width = 6, height = 4,
dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# ------------------------------
# Print regression results to the console
# ------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results ---\n")

##
## --- Regression Results ---

cat("Slope:", slope, "\n")

## Slope: 1.003421

cat("Intercept:", intercept, "\n")

## Intercept: 0.01705318

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8812448

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9387464

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 1 Voltage vs. Sensor 3 Voltage with Statistics
# ================================
# ------------------------------
# Compute Linear Regression for Sensor 1 vs. Sensor 3
# ------------------------------
# Fit a linear regression model where:
# Sensor3 Voltage = slope * Sensor1 Voltage + intercept
model <- lm(Sensor3_Voltage ~ Sensor1_Voltage, data = data)

# Extract regression parameters
slope <- coef(model)[2]  # The slope of the regression line
intercept <- coef(model)[1]  # The y-intercept of the regression line
r_squared <- summary(model)$r.squared  # R-squared value, indicating model fi
t
correlation_coefficient <- sqrt(r_squared)  # Square root of R-squared gives
correlation coefficient
```

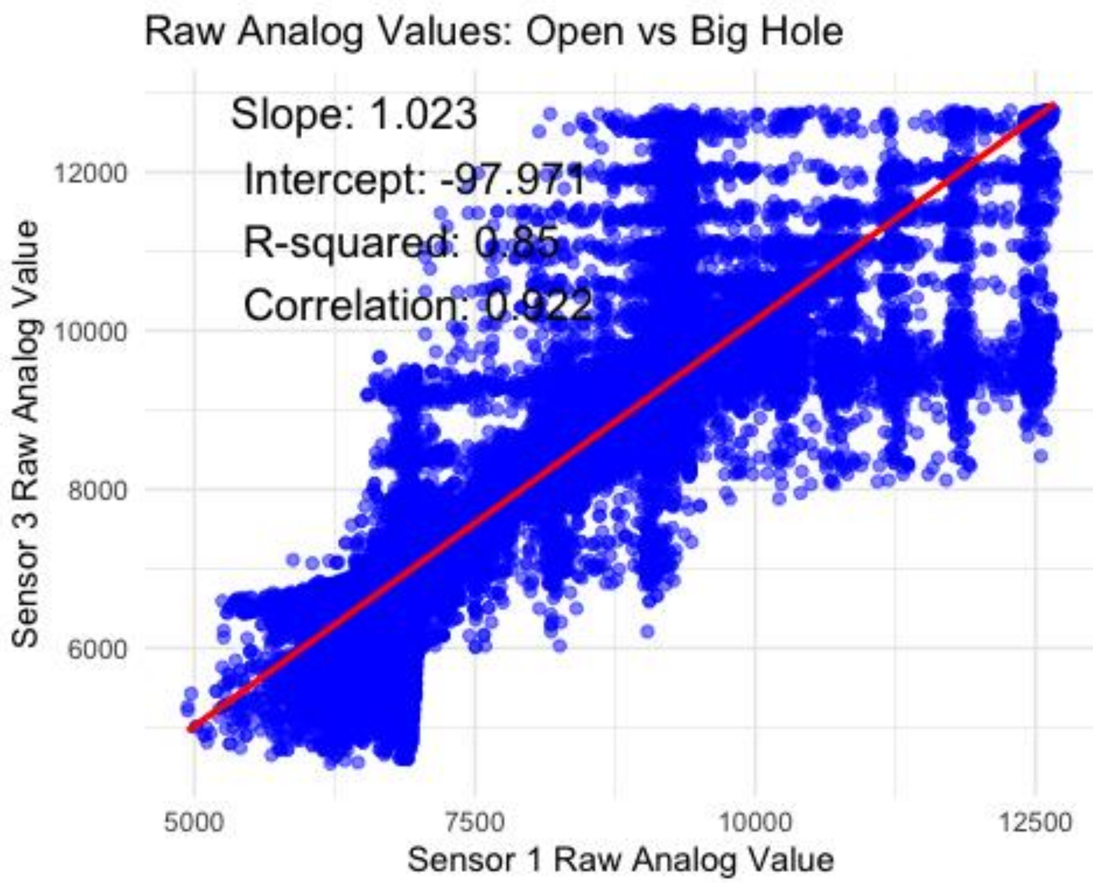
```r
# -----------------------------
# Create text annotation for the plot (Sensor 1 vs. Sensor 3)
# -----------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# Determine annotation position for displaying text on the plot
text_x_pos <- min(data$Sensor1_Voltage) + (max(data$Sensor1_Voltage) - min(da
ta$Sensor1_Voltage)) * 0.05  # 5% from the left
text_y_pos <- min(data$Sensor3_Voltage) + (max(data$Sensor3_Voltage) - min(da
ta$Sensor3_Voltage)) * 0.85  # 85% from the bottom

# ==============================
# Plot of Sensor 1 Voltage vs. Sensor 3 Voltage with best-fit line
# ==============================
plot <- ggplot(data, aes(x = Sensor1_Voltage, y = Sensor3_Voltage)) +
  # Add scatter points in blue with 50% transparency
  geom_point(alpha = 0.5, color = "blue") +
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("Voltage: Open vs. Big Hole") +
  # Set x-axis label
  xlab("Sensor 1 Voltage (V)") +
  # Set y-axis label
  ylab("Sensor 3 Voltage (V)") +
  # Use a minimal theme for a clean layout
  theme_minimal()

# -----------------------------
# Display the plot
# -----------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```




```r
# ------------------------------
# Save the plot as a JPEG file
# ------------------------------
ggsave("Sensor1_vs_Sensor3_Correlation.jpeg", plot = plot, width = 6, height
= 4, dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# ------------------------------
# Print regression results to the console
# ------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results (Sensor 1 vs. Sensor 3) ---\n")

##
## --- Regression Results (Sensor 1 vs. Sensor 3) ---

cat("Slope:", slope, "\n")

## Slope: 1.023012

cat("Intercept:", intercept, "\n")

## Intercept: -0.01835752
```

```r
cat("R-squared:", r_squared, "\n")

## R-squared: 0.8498029

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9218475

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)

# =================================
# Plot Raw Sensor 1 vs. Sensor 3 Analog Values with Statistics
# =================================
# -----------------------------
# Compute Linear Regression
# -----------------------------
# Fit a linear regression model where:
# Sensor3 = slope * Sensor1 + intercept
model <- lm(Sensor3 ~ Sensor1, data = data)

# Extract regression parameters
slope <- coef(model)[2]  # The slope of the regression line
intercept <- coef(model)[1]  # The y-intercept of the regression line
r_squared <- summary(model)$r.squared  # R-squared value, indicating model fi
t
correlation_coefficient <- sqrt(r_squared)  # Square root of R-squared gives
correlation coefficient

# -----------------------------
# Create text annotation for the plot
# -----------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# Determine annotation position for displaying text on the plot
text_x_pos <- min(data$Sensor1) + (max(data$Sensor1) - min(data$Sensor1)) * 0
.05  # 5% from the left
text_y_pos <- min(data$Sensor3) + (max(data$Sensor3) - min(data$Sensor3)) * 0
.85  # 85% from the bottom

# -----------------------------
# Create a scatter plot of raw analog values with best-fit line
# -----------------------------
plot <- ggplot(data, aes(x = Sensor1, y = Sensor3)) +
  # Add scatter points in blue with 50% transparency
```

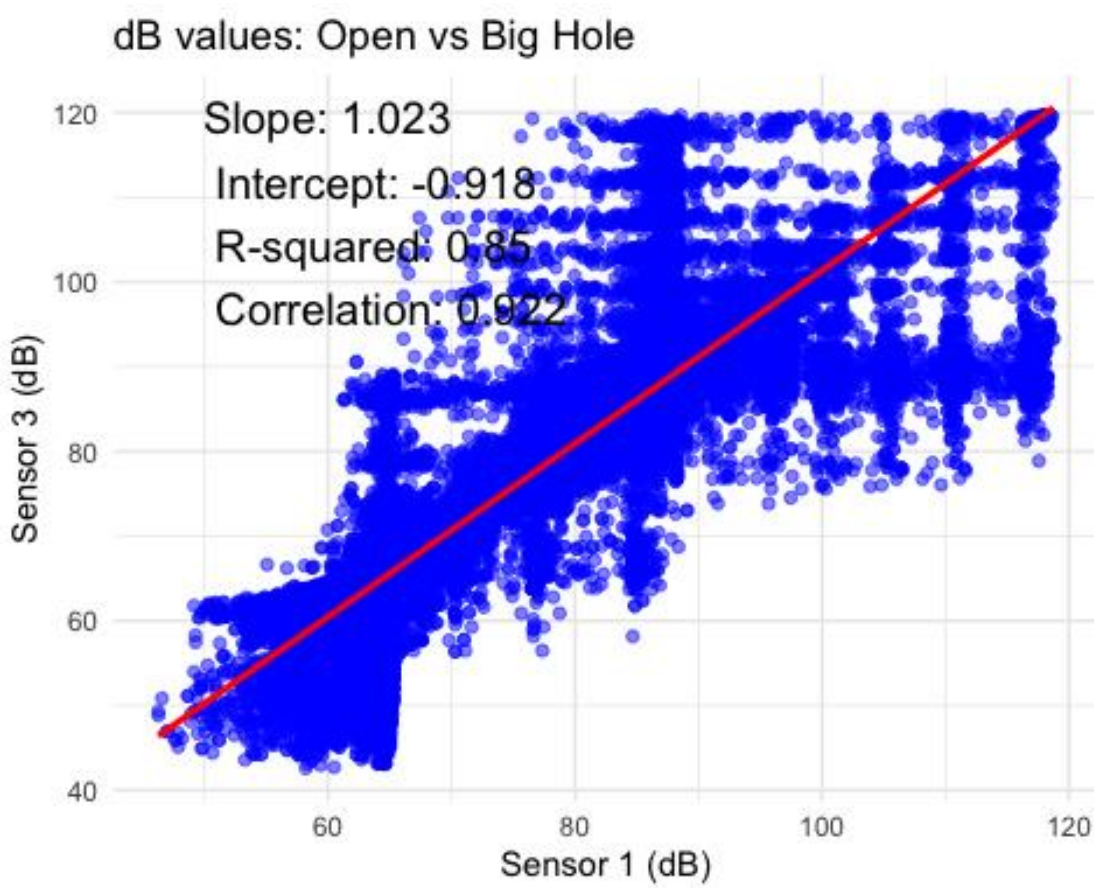
```r
  geom_point(alpha = 0.5, color = "blue") +
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("Raw Analog Values: Open vs Big Hole") +
  # Set x-axis label
  xlab("Sensor 1 Raw Analog Value") +
  # Set y-axis label
  ylab("Sensor 3 Raw Analog Value") +
  # Use a minimal theme for a clean layout
  theme_minimal()

# ------------------------------
# Display the plot
# ------------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```



```r
# ------------------------------
# Save the plot as a JPEG file
# ------------------------------
ggsave("Sensor1_vs_Sensor3_AnalogValues.jpeg", plot = plot, width = 6, height
= 4, dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# ------------------------------
# Print regression results to the console
# ------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results (Sensor 1 vs. Sensor 3) ---\n")

##
## --- Regression Results (Sensor 1 vs. Sensor 3) ---

cat("Slope:", slope, "\n")

## Slope: 1.023012

cat("Intercept:", intercept, "\n")

## Intercept: -97.97054

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8498029

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9218475

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 1 vs. Sensor 3 dB Values with Statistics
# ================================
# ------------------------------
# Compute linear regression
# ------------------------------
# Fit a linear regression model where:
# Sensor3_dB = slope * Sensor1_dB + intercept
model <- lm(Sensor3_dB ~ Sensor1_dB, data = data)

# Display the regression summary in the console
summary(model)

##
## Call:
## lm(formula = Sensor3_dB ~ Sensor1_dB, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -40.471  -0.758   0.484   1.416  41.897
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.9178759  0.0691792  -13.27   <2e-16 ***
## Sensor1_dB   1.0230117  0.0008806 1161.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.759 on 238508 degrees of freedom
## Multiple R-squared:  0.8498, Adjusted R-squared:  0.8498
## F-statistic: 1.349e+06 on 1 and 238508 DF,  p-value: < 2.2e-16

# Extract regression parameters
# Extract the slope (coefficient for Sensor1_dB)
slope <- coef(model)[2]
# Extract the y-intercept
intercept <- coef(model)[1]
# Compute the R-squared value, indicating model fit
r_squared <- summary(model)$r.squared
# Compute correlation coefficient as square root of R-squared
correlation_coefficient <- sqrt(r_squared)

# ------------------------------
# Create text annotation for the plot
# ------------------------------
# Format the regression parameters as a text string to display on the plot
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# ------------------------------
# Determine annotation position on the plot
# ------------------------------
# Set the x-position for annotation as 5% from the left side
text_x_pos <- min(data$Sensor1_dB) + (max(data$Sensor1_dB) - min(data$Sensor1
_dB)) * 0.05
# Set the y-position for annotation as 85% from the bottom
text_y_pos <- min(data$Sensor3_dB) + (max(data$Sensor3_dB) - min(data$Sensor3
_dB)) * 0.85

# ------------------------------
# Create a scatter plot with best-fit line
# ------------------------------
# Generate a scatter plot using ggplot2
plot <- ggplot(data, aes(x = Sensor1_dB, y = Sensor3_dB)) +
  # Add scatter points in blue with 50% transparency
```

```
  geom_point(alpha = 0.5, color = "blue") +
  # Add a linear regression line (best-fit line) in red
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  # Annotate the plot with regression parameters
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  # Set the plot title
  ggtitle("dB values: Open vs Big Hole") +
  # Set x-axis label
  xlab("Sensor 1 (dB)") +
  # Set y-axis label
  ylab("Sensor 3 (dB)") +
  # Use a minimal theme for cleaner visualization
  theme_minimal()

# ------------------------------
# Display the plot
# ------------------------------
# Print the scatter plot with best-fit line
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```



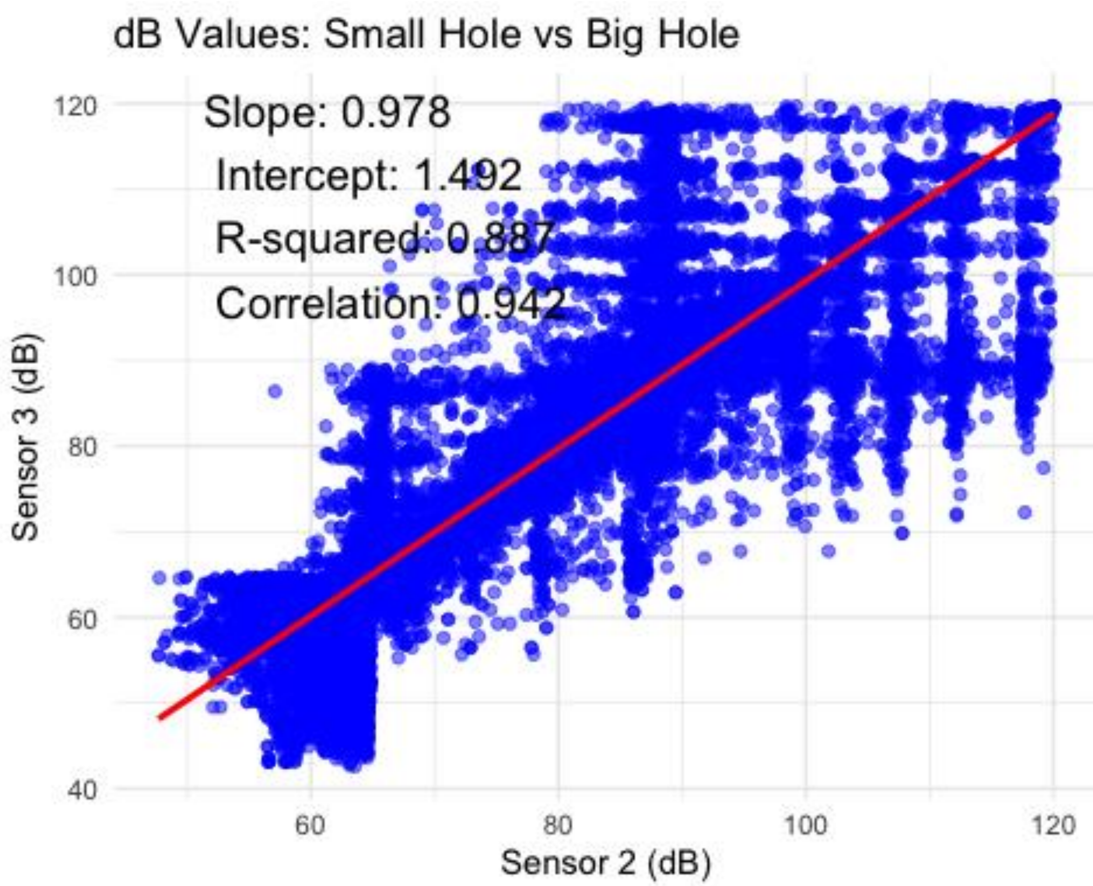
```r
# ------------------------------
# Save the plot as a JPEG file
# ------------------------------
ggsave("Sensor1_vs_Sensor3_Correlation_dB.jpeg", plot = plot, width = 6, heig
ht = 4, dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# ------------------------------
# Print regression results to the console
# ------------------------------
# Display regression results with explanatory text
cat("\n--- Regression Results (Sensor 1 vs. Sensor 3) ---\n")

##
## --- Regression Results (Sensor 1 vs. Sensor 3) ---

cat("Slope:", slope, "\n")

## Slope: 1.023012

cat("Intercept:", intercept, "\n")

## Intercept: -0.9178759

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8498029

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9218475

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 2 vs. Sensor 3 dB Values with Statistics
# ================================
# ------------------------------
# Compute linear regression
# ------------------------------
# Fit a linear regression model where:
# Sensor3_dB = slope * Sensor2_dB + intercept
model <- lm(Sensor3_dB ~ Sensor2_dB, data = data)

# Display the regression summary in the console
summary(model)

##
## Call:
## lm(formula = Sensor3_dB ~ Sensor2_dB, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -44.337  -0.653   0.190   0.931  39.254
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.4923130  0.0569346   26.21   <2e-16 ***
## Sensor2_dB  0.9779539  0.0007137 1370.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.122 on 238508 degrees of freedom
## Multiple R-squared:  0.8873, Adjusted R-squared:  0.8873
## F-statistic: 1.878e+06 on 1 and 238508 DF,  p-value: < 2.2e-16
```

```r
# Extract regression parameters
slope <- coef(model)[2]  # The slope of the regression line
intercept <- coef(model)[1]  # The y-intercept
r_squared <- summary(model)$r.squared  # R-squared value
correlation_coefficient <- sqrt(r_squared)  # Square root of R-squared

# ------------------------------
# Create text annotation for the plot
# ------------------------------
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# ------------------------------
# Determine annotation position
# ------------------------------
text_x_pos <- min(data$Sensor2_dB) + (max(data$Sensor2_dB) - min(data$Sensor2
_dB)) * 0.05
text_y_pos <- min(data$Sensor3_dB) + (max(data$Sensor3_dB) - min(data$Sensor3
_dB)) * 0.85

# ------------------------------
# Create scatter plot with best-fit line
# ------------------------------
plot <- ggplot(data, aes(x = Sensor2_dB, y = Sensor3_dB)) +
  geom_point(alpha = 0.5, color = "blue") +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  ggtitle("dB Values: Small Hole vs Big Hole") +
  xlab("Sensor 2 (dB)") +
  ylab("Sensor 3 (dB)") +
  theme_minimal()
```

```r
# ------------------------------
# Display and save the plot
# ------------------------------
print(plot)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
ggsave("Sensor2_vs_Sensor3_Correlation_dB.jpeg", plot = plot, width = 6, heig
ht = 4, dpi = 300)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```r
# ------------------------------
# Print regression results to the console
# ------------------------------
cat("\n--- Regression Results (Sensor 2 vs. Sensor 3) ---\n")
```

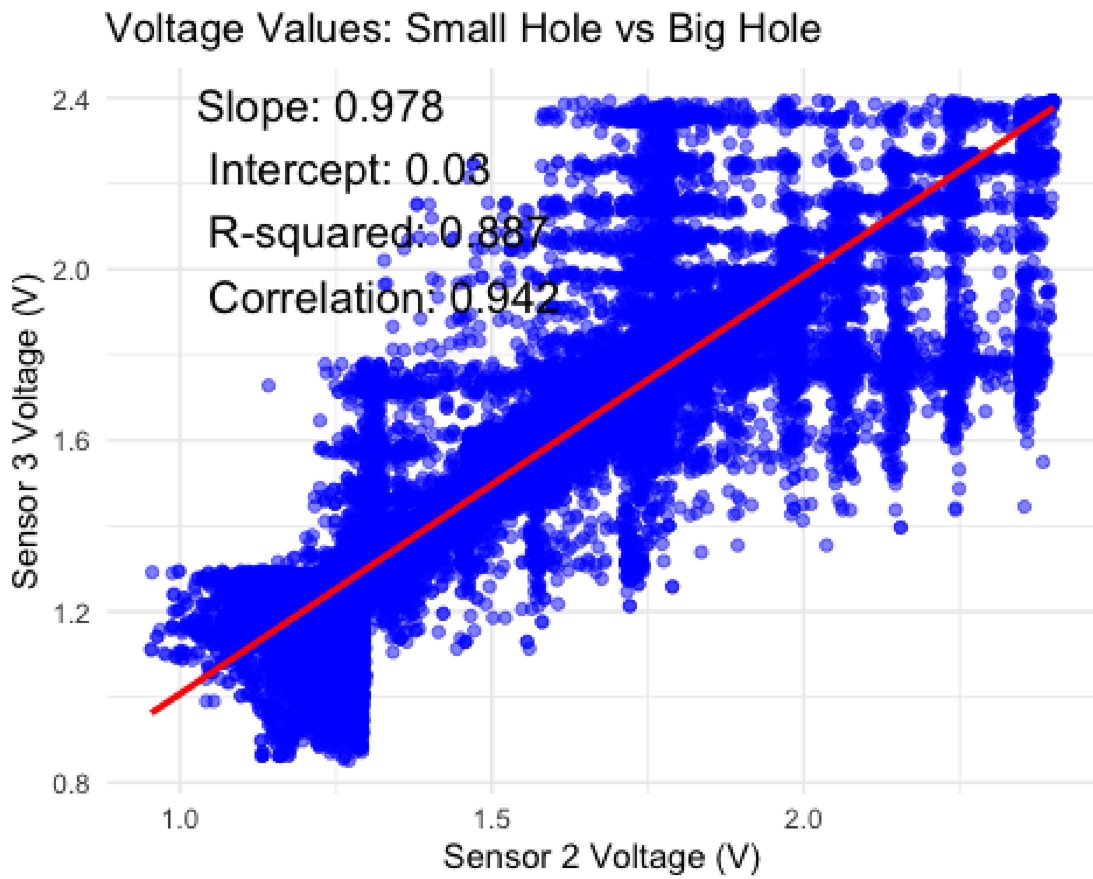
```
##
## --- Regression Results (Sensor 2 vs. Sensor 3) ---
```

```r
cat("Slope:", slope, "\n")
```

```
## Slope: 0.9779539
```

```r
cat("Intercept:", intercept, "\n")

## Intercept: 1.492313

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8872857

cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9419584

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 2 vs. Sensor 3 Voltage Values with Statistics
# ================================
# ------------------------------
# Compute linear regression
# ------------------------------
# Fit a linear regression model where:
# Sensor3_Voltage = slope * Sensor2_Voltage + intercept
model <- lm(Sensor3_Voltage ~ Sensor2_Voltage, data = data)

# Display the regression summary in the console
summary(model)

##
## Call:
## lm(formula = Sensor3_Voltage ~ Sensor2_Voltage, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.88674 -0.01307  0.00380  0.01862  0.78509
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.0298463  0.0011387   26.21   <2e-16 ***
## Sensor2_Voltage 0.9779539  0.0007137 1370.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08245 on 238508 degrees of freedom
## Multiple R-squared:  0.8873, Adjusted R-squared:  0.8873
## F-statistic: 1.878e+06 on 1 and 238508 DF,  p-value: < 2.2e-16

# Extract regression parameters
slope <- coef(model)[2]
intercept <- coef(model)[1]
r_squared <- summary(model)$r.squared
correlation_coefficient <- sqrt(r_squared)
```

```r
# ------------------------------
# Create text annotation for the plot
# ------------------------------
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
  "Correlation:", round(correlation_coefficient, 3)
)

# ------------------------------
# Determine annotation position
# ------------------------------
text_x_pos <- min(data$Sensor2_Voltage) + (max(data$Sensor2_Voltage) - min(da
ta$Sensor2_Voltage)) * 0.05
text_y_pos <- min(data$Sensor3_Voltage) + (max(data$Sensor3_Voltage) - min(da
ta$Sensor3_Voltage)) * 0.85

# ------------------------------
# Create scatter plot with best-fit line
# ------------------------------
plot <- ggplot(data, aes(x = Sensor2_Voltage, y = Sensor3_Voltage)) +
  geom_point(alpha = 0.5, color = "blue") +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  ggtitle("Voltage Values: Small Hole vs Big Hole") +
  xlab("Sensor 2 Voltage (V)") +
  ylab("Sensor 3 Voltage (V)") +
  theme_minimal()

# ------------------------------
# Display and save the plot
# ------------------------------
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```




```r
ggsave("Sensor2_vs_Sensor3_Correlation_Voltage.jpeg", plot = plot, width = 6,
height = 4, dpi = 300)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```r
# ------------------------------
# Print regression results to the console
# ------------------------------
cat("\n--- Regression Results (Sensor 2 vs. Sensor 3) ---\n")
```

```
##
## --- Regression Results (Sensor 2 vs. Sensor 3) ---
```

```r
cat("Slope:", slope, "\n")
```

```
## Slope: 0.9779539
```

```r
cat("Intercept:", intercept, "\n")
```

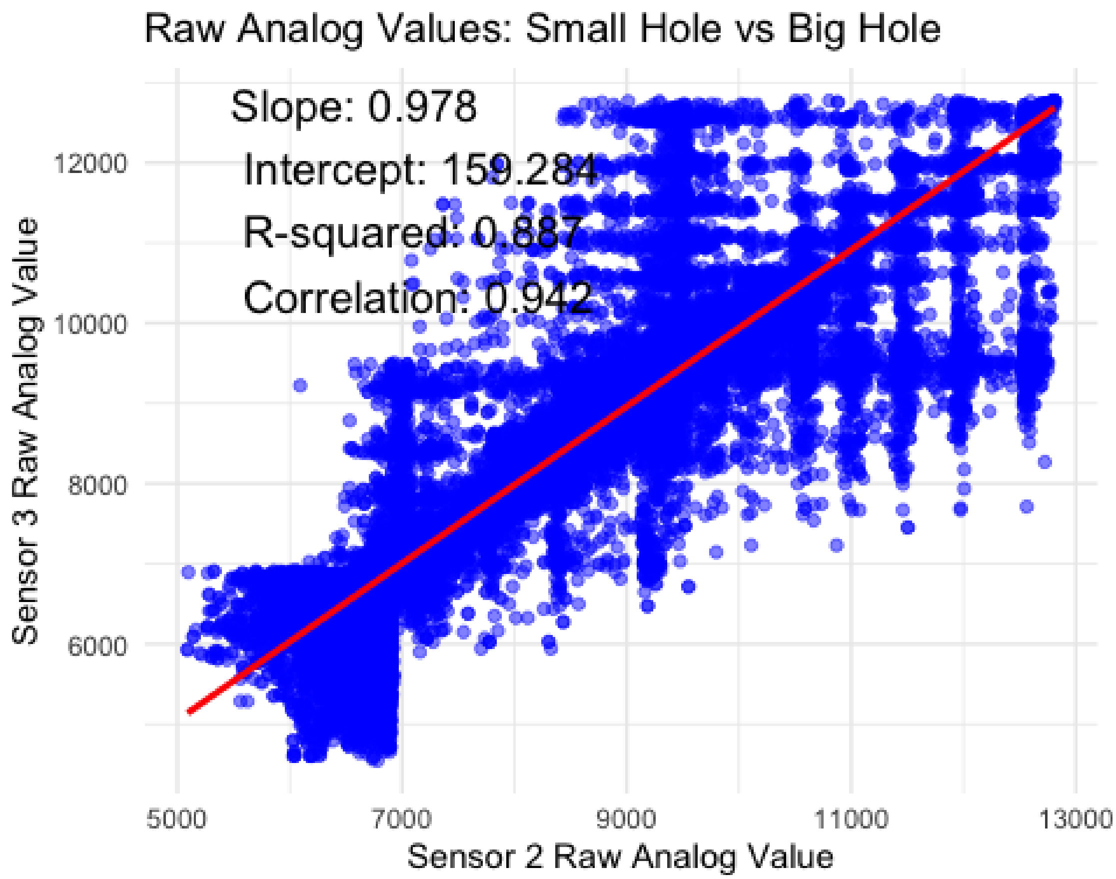
```
## Intercept: 0.02984626
```

```r
cat("R-squared:", r_squared, "\n")
```

```
## R-squared: 0.8872857
```

```
cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9419584

knitr::opts_chunk$set(echo = TRUE, message = TRUE, warning = FALSE)
# ================================
# Plot Sensor 2 vs. Sensor 3 Raw Analog Values with Statistics
# ================================
# ------------------------------
# Compute linear regression
# ------------------------------
# Fit a linear regression model where:
# Sensor3 = slope * Sensor2 + intercept
model <- lm(Sensor3 ~ Sensor2, data = data)

# Display the regression summary in the console
summary(model)

##
## Call:
## lm(formula = Sensor3 ~ Sensor2, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4732.3   -69.7    20.3    99.3  4189.9
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.593e+02  6.077e+00   26.21   <2e-16 ***
## Sensor2     9.780e-01  7.137e-04 1370.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 440 on 238508 degrees of freedom
## Multiple R-squared:  0.8873, Adjusted R-squared:  0.8873
## F-statistic: 1.878e+06 on 1 and 238508 DF,  p-value: < 2.2e-16

# Extract regression parameters
slope <- coef(model)[2]
intercept <- coef(model)[1]
r_squared <- summary(model)$r.squared
correlation_coefficient <- sqrt(r_squared)

# ------------------------------
# Create text annotation for the plot
# ------------------------------
regression_text <- paste(
  "Slope:", round(slope, 3), "\n",
  "Intercept:", round(intercept, 3), "\n",
  "R-squared:", round(r_squared, 3), "\n",
```

```r
  "Correlation:", round(correlation_coefficient, 3)
)

# ------------------------------
# Determine annotation position
# ------------------------------
text_x_pos <- min(data$Sensor2) + (max(data$Sensor2) - min(data$Sensor2)) * 0
.05
text_y_pos <- min(data$Sensor3) + (max(data$Sensor3) - min(data$Sensor3)) * 0
.85

# ------------------------------
# Create scatter plot with best-fit line
# ------------------------------
plot <- ggplot(data, aes(x = Sensor2, y = Sensor3)) +
  geom_point(alpha = 0.5, color = "blue") +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  annotate("text", x = text_x_pos, y = text_y_pos, label = regression_text, s
ize = 5, hjust = 0, color = "black") +
  ggtitle("Raw Analog Values: Small Hole vs Big Hole") +
  xlab("Sensor 2 Raw Analog Value") +
  ylab("Sensor 3 Raw Analog Value") +
  theme_minimal()

# ------------------------------
# Display and save the plot
# ------------------------------
print(plot)

## `geom_smooth()` using formula = 'y ~ x'
```




```
ggsave("Sensor2_vs_Sensor3_Correlation_Raw.jpeg", plot = plot, width = 6, hei
ght = 4, dpi = 300)

## `geom_smooth()` using formula = 'y ~ x'

# -----------------------------
# Print regression results to the console
# -----------------------------
cat("\n--- Regression Results (Sensor 2 vs. Sensor 3) ---\n")

##
## --- Regression Results (Sensor 2 vs. Sensor 3) ---

cat("Slope:", slope, "\n")

## Slope: 0.9779539

cat("Intercept:", intercept, "\n")

## Intercept: 159.2838

cat("R-squared:", r_squared, "\n")

## R-squared: 0.8872857
```

```r
cat("Correlation Coefficient:", correlation_coefficient, "\n")

## Correlation Coefficient: 0.9419584
```