**#read data**

mydata <- read.csv("~/Desktop/ecgi.csv")

**#test all data for normality using histograms, qq plot and wilk Shapiro test**

**# Get names of numeric columns**

numeric\_columns <- names(mydata)[sapply(mydata, is.numeric)]

**# Loop over columns and perform tests and plots**

for (colname in numeric\_columns) {

**# Create filename for output**

filename <- paste(colname, "normality test results.csv", sep = "\_")

**# Perform normality test**

shapiro.test\_res <- shapiro.test(mydata[[colname]])

**# Create histogram and qq plot**

hist\_filename <- paste(colname, "histogram.png", sep = "\_")

qq\_filename <- paste(colname, "qqplot.png", sep = "\_")

png(hist\_filename)

hist(mydata[[colname]], breaks = "FD", col = "gray", main = colname)

dev.off()

png(qq\_filename)

qqnorm(mydata[[colname]], main = colname)

qqline(mydata[[colname]])

dev.off()

**# Create data frame of results**

results\_df <- data.frame(

variable = colname,

test = "Shapiro-Wilk",

statistic = shapiro.test\_res$statistic,

p.value = shapiro.test\_res$p.value

)

**# Write results to file**

write.csv(results\_df, file = filename, row.names = FALSE)

}

**#Split into groups based on age**

group\_a <- subset(mydata, group == "A")

group\_b <- subset(mydata, group == "B")

**#Create a summary table of means and SD of all variables according to age groups and perform an unpaired t-test for differences between the groups displaying a p value.**

**# Get the numerical variables in the data frame**

numeric\_vars <- sapply(mydata, is.numeric)

**# Create an empty data frame to store the results**

results <- data.frame(variable = character(), t\_value = numeric(), p\_value = numeric(), group\_a\_mean = numeric(), group\_a\_sd = numeric(), group\_b\_mean = numeric(), group\_b\_sd = numeric(), stringsAsFactors = FALSE)

**# Perform a two-sample t-test on all numerical variables in "group\_a" and "group\_b"**

for (var in names(mydata)[numeric\_vars]) {

cat("Variable:", var, "\n")

**# Check that both groups have at least one non-missing observation for this variable**

if (sum(!is.na(group\_a[[var]])) > 1 & sum(!is.na(group\_b[[var]])) > 1) {

**# Check that both groups have more than one unique value for this variable**

if (length(unique(group\_a[[var]], na.rm = TRUE)) > 1 & length(unique(group\_b[[var]], na.rm = TRUE)) > 1) {

**# Calculate the mean and standard deviation for each group**

group\_a\_mean <- mean(group\_a[[var]], na.rm = TRUE)

group\_a\_sd <- sd(group\_a[[var]], na.rm = TRUE)

group\_b\_mean <- mean(group\_b[[var]], na.rm = TRUE)

group\_b\_sd <- sd(group\_b[[var]], na.rm = TRUE)

**# Perform the t-test with missing values removed**

t\_test <- t.test(group\_a[[var]], group\_b[[var]], na.rm = TRUE)

**# Store the results in the data frame**

results <- rbind(results, data.frame(variable = var, t\_value = t\_test$statistic, p\_value = t\_test$p.value, group\_a\_mean = group\_a\_mean, group\_a\_sd = group\_a\_sd, group\_b\_mean = group\_b\_mean, group\_b\_sd = group\_b\_sd))

} else {

cat("One of the groups has essentially constant data for this variable.\n")

}

} else {

cat("One of the groups has no data or only one non-missing observation for this variable.\n")

}

cat("\n")

}

**# Save the results to a CSV file**

write.csv(results, "ttest\_results\_with\_means\_and\_sd.csv", row.names = FALSE)

**#Create a summary table of medians and IQ of all variables according to age groups and perform an wilcox signed rank for differences between the groups displaying a p value.**

**# Get the numerical variables in the data frame**

numeric\_vars <- sapply(mydata, is.numeric)

**# Create an empty data frame to store the results**

results <- data.frame(variable = character(), median\_a = numeric(), median\_b = numeric(), iqr\_A = numeric(), iqr\_b = numeric(), p\_value = numeric(), stringsAsFactors = FALSE)

**# Perform a two-sample Wilcoxon rank-sum test on all numerical variables in "group\_a" and "group\_b"**

for (var in names(mydata)[numeric\_vars]) {

cat("Variable:", var, "\n")

**# Check that both groups have at least one non-missing observation for this variable**

if (sum(!is.na(group\_a[[var]])) > 0 & sum(!is.na(group\_b[[var]])) > 0) {

**# Get the median and IQR for both groups**

group\_a\_median <- median(group\_a[[var]], na.rm = TRUE)

group\_b\_median <- median(group\_b[[var]], na.rm = TRUE)

group\_a\_iqr <- IQR(group\_a[[var]], na.rm = TRUE)

group\_b\_iqr <- IQR(group\_b[[var]], na.rm = TRUE)

**# Perform the two-sample Wilcoxon rank-sum test with missing values removed**

wilcox\_test <- wilcox.test(group\_a[[var]], group\_b[[var]], alternative = "two.sided", na.rm = TRUE)

**# Store the results in the data frame**

results <- rbind(results, data.frame(variable = var, median\_a = group\_a\_median, median\_b = group\_b\_median, iqr\_a = group\_a\_iqr, iqr\_b = group\_b\_iqr, p\_value = wilcox\_test$p.value))

} else {

cat("One of the groups has no data for this variable.\n")

}

cat("\n")

}

**# Save the results to a CSV file**

write.csv(results, "wilcox\_results.csv", row.names = FALSE)

**#Create box plots for a list of variables according to age group and save to desktop.**

**# Load ggplot2 package**

library(ggplot2)

**# List of variables**

variables <- c("rtb", "rtf", "arib", "arif", "lg\_at", "lg\_arib", "lg\_arif", "lg\_rtb", "lg\_rtf", "ld\_at", "ld\_arib", "ld\_arif", "at")

**# Function to create and save box plots for each variable**

create\_boxplot <- function(var) {

**# Combine group\_a and group\_b into a single data frame**

combined\_data <- rbind(group\_a, group\_b)

**# Create a new column "Group" to indicate which group each row belongs to**

combined\_data$Group <- factor(c(rep("Group A", nrow(group\_a)), rep("Group B", nrow(group\_b))), levels = c("Group B", "Group A"))

**# Create the box plot**

p <- ggplot(data = combined\_data, aes(x = Group, y = combined\_data[[var]])) +

geom\_boxplot(width = 0.5, fill = "lightgray", alpha = 0.5, color = "black", outlier.color = "black", outlier.shape = 16) +

labs(x = "", y = var) +

theme\_bw() +

theme(panel.grid.major = element\_blank(),

panel.grid.minor = element\_blank(),

panel.border = element\_blank(),

axis.text.x = element\_blank(),

axis.ticks.x = element\_blank(),

axis.line.y = element\_line(color = "black", size = 0.5),

text = element\_text(family = "Arial"),

axis.text.y = element\_text(size = 14, color = "black"))

**# Save the box plot to your desktop**

ggsave(paste0("~/Desktop/", var, "\_boxplot.png"), plot = p, width = 6, height = 4)

}

**# Loop through the variables and create and save box plots**

for (var in variables) {

create\_boxplot(var)

}

**#Create a correlation matrix table to include all the ECGI parameters as outcome variables against all other variables in datasheet (spearman’s/pearson’s).**

**# List of predictor variables**

predictor\_vars <- c("age", "sex", "height", "weight", "bsa", "hr", "mapse", "tapse", "la", "lai", "ra", "lvedv", "lvedvi", "lvesv", "lvesvi", "lvsv", "lvsvi", "lvco", "lvef", "lvmd", "lvmdi", "lvms", "lvmsi", "rvedv", "rvedvi", "rvesv", "rvesvi", "rvsv", "rvsvi", "rvco", "rvef", "t1", "ecv", "t2")

**# List of outcome variables**

outcome\_vars <- c("at", "rtb", "rtf", "arib", "arif", "lg\_at", "lg\_arib", "lg\_arif", "lg\_rtb", "lg\_rtf", "ld\_at", "ld\_arib", "ld\_arif", "ld\_rtb", "ld\_rtf", "amp\_qrs", "Amp\_tw", "frac")

**# Create an empty data frame to store results**

results <- data.frame()

**# Loop through each combination of predictor and outcome variables**

for (pred\_var in predictor\_vars) {

for (out\_var in outcome\_vars) {

**# Calculate Spearman's correlation**

test\_result <- cor.test(mydata[[pred\_var]], mydata[[out\_var]], method = "spearman", use = "pairwise.complete.obs")

**# Extract correlation coefficient and p-value**

rho <- test\_result$estimate

p\_value <- test\_result$p.value

**# Append the results to the data frame**

results <- rbind(results, data.frame(Predictor = pred\_var, Outcome = out\_var, Rho = rho, P\_value = p\_value))

}

}

**# Write the results to a CSV file**

write.csv(results, "spearman\_correlation\_results.csv", row.names = FALSE)

**#For significant variables creat a scatter plot with line of best fit, correlation and p value for a range of defined outcome and predictor variables. Save to desktop. Pearsons and spearman**

**# Load ggpubr package**

library(ggpubr)

**# List of predictor variables**

predictor\_vars <- c("ecv", "t2")

**# List of outcome variables**

outcome\_vars <- c("rtb", "rtf", "arib", "arif", "lg\_rtf", "ld\_rtb", "ld\_rtf")

**# Function to create and save scatter plots for each combination of predictor and outcome variables**

create\_scatterplot <- function(pred\_var, out\_var) {

plot <- ggscatter(mydata, x = pred\_var, y = out\_var,

cor.coef = TRUE, cor.method = "spearman",

cor.coef.size = 4, cor.coef.fontsize = 8, # Reduced font size

cor.coef.x = 0.95, cor.coef.y = 0.95, # Position of correlation coefficient and p-value

color = "black", size = 1) + # Reduced size of dots to 1

stat\_smooth(method = "lm", se = FALSE, linetype = "dashed", color = "black", size = 0.5) + # Customize regression line

labs(x = pred\_var, y = out\_var) +

theme(text = element\_text(family = "Arial")) # Set font family to Arial

**# Save the scatter plot to your desktop**

ggsave(paste0("~/Desktop/", out\_var, "\_vs\_", pred\_var, "\_scatterplot.png"), plot = plot, width = 6, height = 4)

}

# Loop through each combination of predictor and outcome variables and create and save scatter plots

for (pred\_var in predictor\_vars) {

for (out\_var in outcome\_vars) {

create\_scatterplot(pred\_var, out\_var)

}

}

**#Create a table of univariate linear regression analysis results. Outcome variables (ECGI) and predictor variables. Log transform those outcome variables with non normal distribution. Save to a csv file.**

**# Loop through each combination of predictor and outcome variables and create and save scatter plots**

for (pred\_var in predictor\_vars) {

for (out\_var in outcome\_vars) {

create\_scatterplot(pred\_var, out\_var)

}

}

**# Create an empty data frame to store the results**

results <- data.frame(outcome = character(), predictor = character(), beta = numeric(), conf.low = numeric(), conf.high = numeric(), p.value = numeric(), stringsAsFactors = FALSE)

**# Define the list of outcome variables**

outcome\_vars <- c("at", "rtb", "rtf", "arib", "arif", "lg\_at", "lg\_arib", "lg\_arif", "lg\_rtb", "lg\_rtf", "ld\_at", "ld\_arib", "ld\_arif", "ld\_rtb", "ld\_rtf", "amp\_qrs", "Amp\_tw", "frac")

**# Define the list of predictor variables**

predictor\_vars <- c("age", "sex", "bsa", "la", "lvedvi", "mapse", "lvef", "lvsvi", "lvmdi", "rvedvi", "rvef", "tapse", "t1", "ecv", "t2")

**# Log-transform specified outcome variables**

mydata$log\_ld\_at <- log(mydata$ld\_at)

mydata$log\_ld\_rtb <- log(mydata$ld\_rtb)

mydata$log\_ld\_rtf <- log(mydata$ld\_rtf)

mydata$log\_lg\_at <- log(mydata$lg\_at)

mydata$log\_lg\_rtb <- log(mydata$lg\_rtb)

mydata$log\_lg\_rtf <- log(mydata$lg\_rtf)

mydata$log\_at <- log(mydata$at)

**# Update the list of outcome variables with the transformed variables**

outcome\_vars <- c("log\_at", "rtb", "rtf", "arib", "arif", "log\_lg\_at", "lg\_arib", "lg\_arif", "log\_lg\_rtb", "log\_lg\_rtf", "log\_ld\_at", "ld\_arib", "ld\_arif", "log\_ld\_rtb", "log\_ld\_rtf", "amp\_qrs", "Amp\_tw", "frac")

**# Loop through all outcome variables**

for (outcome in outcome\_vars) {

# Loop through all predictor variables

for (predictor in predictor\_vars) {

model <- lm(as.formula(paste(outcome, "~", predictor)), data = mydata)

**# Extract the coefficients and confidence intervals from the model summary**

coef <- coef(summary(model))

beta <- coef[2, 1]

conf.low <- coef[2, 2]

conf.high <- coef[2, 3]

p.value <- coef[2, 4]

**# Add the results to the data frame**

results <- rbind(results, data.frame(outcome = outcome, predictor = predictor, beta = beta, conf.low = conf.low, conf.high = conf.high, p.value = p.value))

}

}

**# Write the results to a CSV file**

write.csv(results, "univariate\_results.csv", row.names = FALSE)

**#Perform linear regression analysis example**

model1 <- lm(arib ~ age + sex + bsa + lvef + t2, data = mydata)

summary(model1)

vif\_values <- vif(model1)

print(vif\_values)

**#Perform partial correlation coefficient analysis:**

if (!requireNamespace("ppcor", quietly = TRUE)) {

install.packages("ppcor")

}

# Load the ppcor package

library(ppcor)

# Calculate the partial correlation between rtf and ecv, controlling for age, sex, and bsa

partial\_cor\_result <- pcor(cbind(mydata\_clean$rtf, mydata\_clean$ecv, mydata\_clean$age, mydata\_clean$sex, mydata\_clean$bsa))

# Display the result

print(partial\_cor\_result)

# Access the partial correlation coefficient between rtf and ecv

rtf\_ecv\_partial\_cor <- partial\_cor\_result$estimate[1, 2]

# Display the partial correlation coefficient

print(rtf\_ecv\_partial\_cor)

# Compute the Fisher's Z transformation

z\_score <- 0.5 \* log((1 + rtf\_ecv\_partial\_cor) / (1 - rtf\_ecv\_partial\_cor))

# Determine the standard error

n <- partial\_cor\_result$n[1]

se <- 1 / sqrt(n - 3)

# Calculate the Z-score-based confidence interval

alpha <- 0.05

z\_crit <- qnorm(1 - alpha / 2)

z\_CI <- c(z\_score - z\_crit \* se, z\_score + z\_crit \* se)

# Transform the Z-score-based confidence interval back to the partial correlation scale

rtf\_ecv\_ci <- (exp(2 \* z\_CI) - 1) / (exp(2 \* z\_CI) + 1)

# Display the confidence interval

print(rtf\_ecv\_ci)