Reproducibility - HW 2

Wendy Shi

2023-11-22

Loading in Data

```r
# load in libraries
package_list <- c("tidyverse", "vroom", "dplyr", "patchwork")

# load in packages
for (package_name in package_list) {
  if (!requireNamespace(package_name, quietly = TRUE)) {
    install.packages(package_name)
  }
  library(package_name, character.only = TRUE)
}

# load in data
df <- vroom("ddh.csv")
head(df)
```

```
## # A tibble: 6 x 24
##      ID cold_ischemia r_height r_weight r_bmi r_age r_caucasian r_genderf
##   <dbl>         <dbl>    <dbl>    <dbl> <dbl> <dbl>       <dbl>     <dbl>
## 1     1           433     183      85.5  25.5    60           1         0
## 2     2           617     180.     73.8  22.7    36           0         0
## 3     3           282     165.     63.9  23.4    52           0         1
## 4     4           349     192.    116.   31.5    53           1         0
## 5     5           310     158.     55.8  22.5    52           1         1
## 6     6           468     187.     71.8  20.6    60           1         0
## # i 16 more variables: r_meld_assign <dbl>, r_meld_calc <dbl>, d_age <dbl>,
## #   d_cod <dbl>, d_caucasian <dbl>, d_genderf <dbl>, donorrisk <dbl>,
## #   sodium <dbl>, steatosis <dbl>, sbp_lt_90 <dbl>, hemo_instability <dbl>,
## #   glucount <dbl>, glutwa <dbl>, glurange <dbl>, glusd <dbl>, delayed_fn <dbl>
```

Reproduction of the first box-plot

```r
# faceted by IGF and LGD (initial graft function and liver graft dysfunction)
# Identified by variable "delayed_fn", "1" = yes.
# Ploted vertically on a logrithmic scale

# creating a boxplot for the TWA values
twa<-ggplot(df, aes(x = factor(delayed_fn), y = glutwa, fill = factor(delayed_fn))) +
  # error bars function to create horizontal lines at ends of whiskers
  stat_boxplot(geom ='errorbar', coef = Inf, size = 0.2) +

  # overlaying the boxplots over the error bars, extending whiskers with coef = Inf
  geom_boxplot(coef = Inf, size = 0.2) +

  # manually setting scale breaks and limits
  scale_y_continuous(limits=c(50, 500),
                     breaks=c(50, 100, 200, 300, 400, 500),

  # putting in log scale
                     trans='log10') +

  # coloring gray and white
  scale_fill_manual(values=c("0"="white", "1"="gray")) +

  # labeling IGF vs LGD
  scale_x_discrete(labels=c("0"="IGF", "1"="LGD")) +

  # labelinga and getting rid of legend
  labs(title="", x= "TWA", y="") +
  theme(legend.position="none")

# creating boxplot for range values of glucose measurements
range <- ggplot(df, aes(x = factor(delayed_fn), y=glurange, fill=factor(delayed_fn))) +
  stat_boxplot(geom ='errorbar', coef = Inf, size = 0.2) +
  geom_boxplot(coef = Inf, size = 0.2) +
  scale_y_continuous(limits=c(1, 1000),
                     breaks=c(1, 10, 100, 1000),
                     trans='log10') +
  scale_fill_manual(values=c("0"="white", "1"="gray")) +
  scale_x_discrete(labels=c("0"="IGF", "1"="LGD")) +
  labs(title="", x= "Range", y="") +
  theme(legend.position="none")

# Creating a plot for standard deviation values of glucose measurements
sd <- ggplot(df, aes(x = factor(delayed_fn), y=glusd, fill=factor(delayed_fn))) +
  stat_boxplot(geom ='errorbar', coef = Inf, size = 0.2) +
  geom_boxplot(coef = Inf, size = 0.2) +
  scale_y_continuous(limits=c(1, 300),
                     breaks=c(1, 10, 100, 200, 300),
                     trans='log10') +
    scale_fill_manual(values=c("0"="white", "1"="gray")) +
    scale_x_discrete(labels=c("0"="IGF", "1"="LGD")) +
    labs(title="", x= "SD", y="")+
    theme(legend.position="none")

# putting all plots together
twa + range + sd + plot_annotation(title = "Donor Glucose Measures mg/dl")
```

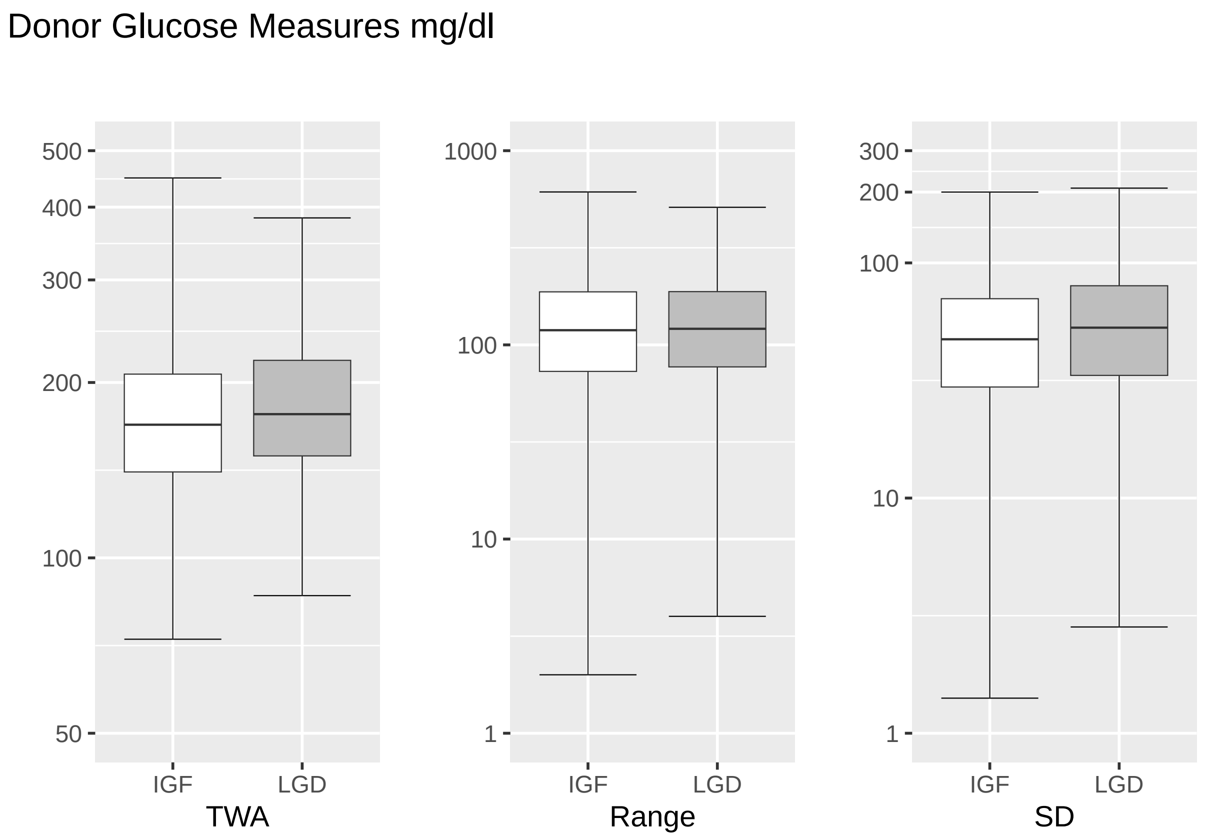

Figure 1: Boxplots of time-weighted average of donor glucose measurements, donor glucose range, and donor glucose standard deviation for 427 recipients experiencing initial graft function (IGF in white) and for 145 recipients experiencing liver graft dysfunction (LGD, in gray).

Create models to recreate Table 2

Primary Exposure Models

```r
# create a multivariable logistic regression model where the response is liver
# graft function "delayed_fn" interested predictor variable is twa of donor
# glucose measurements "glutwa". Controling for other predictors such as age,
# cause of death, and hemodynamic instability "d_age", "d_cod", "r_meld_calc",
# "hemo_instability"

model <- glm(delayed_fn ~ log(glutwa), data = df, family = "binomial")

model_a <- glm(delayed_fn ~ log(glutwa) + d_age + as.factor(d_cod) +
                 hemo_instability , data = df, family = "binomial")
```

```r
# odds ratio extraction, CI creation. Testing

coefficients <- coef(model)
standard_errors <- summary(model)$coefficients[, "Std. Error"]

# Calculate the variance of log-relative_doubling_odds_ratio using the delta method
var <- (log(2))^2 * standard_errors["log(glutwa)"]^2

# Calculate the standard error of the coefficient
std_err <- sqrt(var)

# getting the raw coefficient
coef_logTWA <- coefficients["log(glutwa)"]

# getting the coefficient for interpretation associated with doubling of TWA
coef_estimate <- exp(coef_logTWA*log(2))

# creating confidence interval estimates associated with this coefficient
lower_CI <- coef_estimate * exp(-1.96 * std_err["log(glutwa)"])
upper_CI <- coef_estimate * exp(1.96 * std_err["log(glutwa)"])



# Conducting the Wald Z test
wald_z <- coef_logTWA / standard_errors["log(glutwa)"]

# Calculate p-values
p_value <- 2 * (1 - pnorm(abs(wald_z)))


# print out values
print(paste0("Odds ratio and 95% CI in primary exposure unadjusted model (TWA) is: ",
             round(coef_estimate[["log(glutwa)"]], 2),
             " (", round(lower_CI[["log(glutwa)"]], 2), ", ",
             round(upper_CI[["log(glutwa)"]], 2),")" ))
```

```
## [1] "Odds ratio and 95% CI in primary exposure unadjusted model (TWA) is: 1.48 (0.94, 2.33)"
```

```
print(paste0("The p value is: ",
             round(p_value[["log(glutwa)"]], 2)))
```

```
## [1] "The p value is: 0.09"
```

```
# Paper is doing a profile likelihood instead of a wald test
```

```
# same things for adjusted model
coefficients_a <- coef(model_a)
standard_errors_a <- summary(model_a)$coefficients[, "Std. Error"]
var_a <- (log(2))^2 * standard_errors_a["log(glutwa)"]^2
std_err_a <- sqrt(var_a)
coef_logTWA_a <- coefficients_a["log(glutwa)"]
coef_estimate_a <- exp(coef_logTWA_a*log(2))
lower_CI_a <- coef_estimate_a * exp(-1.96 * std_err_a["log(glutwa)"])
upper_CI_a <- coef_estimate_a * exp(1.96 * std_err_a["log(glutwa)"])

# Conducting the Wald Z test
wald_z_a <- coef_logTWA_a / standard_errors_a["log(glutwa)"]
p_value_a <- 2 * (1 - pnorm(abs(wald_z_a)))


# print out values
print(paste0("Odds ratio and 95% CI in primary exposure adjusted model (TWA) is: ",
            round(coef_estimate_a[["log(glutwa)"]], 2),
            " (", round(lower_CI_a[["log(glutwa)"]], 2), ", ",
            round(upper_CI_a[["log(glutwa)"]], 2),")" ))
```

```
## [1] "Odds ratio and 95% CI in primary exposure adjusted model (TWA) is: 1.46 (0.91, 2.35)"
```

```
print(paste0("The p value is: ",
             round(p_value_a[["log(glutwa)"]], 2)))
```

```
## [1] "The p value is: 0.11"
```

Secondary Exposure Models

```
# create a multivariable logistic regression model where the response is liver
# graft function "delayed_fn"
# Interested predictor variable is now Range of donor glucose measurements. "glurange"
# Controling for other predictors such as age, cause of death, calculated model
# for end stage liver disease score, and hemodynamic instability "d_age", "d_cod",
# "r_meld_calc", "hemo_instability"

model <- glm(delayed_fn ~ log(glurange), data = df, family = "binomial")
model_a <- glm(delayed_fn ~ log(glurange) + d_age + as.factor(d_cod) +
                 hemo_instability , data = df, family = "binomial")
```

```
# odds ratio extraction, CI creation. Testing
```

```r
coefficients <- coef(model)
standard_errors <- summary(model)$coefficients[, "Std. Error"]
var <- (log(2))^2 * standard_errors["log(glurange)"]^2
std_err <- sqrt(var)
coef_logRange <- coefficients["log(glurange)"]
coef_estimate <- exp(coef_logRange*log(2))

critical_value_z <- qnorm(0.0125)
lower_CI <- coef_estimate* exp(critical_value_z* std_err["log(glurange)"])
upper_CI<- coef_estimate* exp(-critical_value_z * std_err["log(glurange)"])

# Conducting the Wald Z test
wald_z <- coef_logRange / standard_errors["log(glurange)"]
p_value<- 2 * (1 - pnorm(abs(wald_z)))


# printing out
print(paste0("Odds ratio and 95% CI in secondary exposure unadjusted model (Range) is: ",
            round(coef_estimate[["log(glurange)"]], 2),
            " (", round(lower_CI[["log(glurange)"]], 2), ", ",
            round(upper_CI[["log(glurange)"]], 2),")" ))
```

Range

```
## [1] "Odds ratio and 95% CI in secondary exposure unadjusted model (Range) is: 1.03 (0.86, 1.24)"
```

```r
print(paste0("The p value is: ",
             round(p_value[["log(glurange)"]], 2)))
```

```
## [1] "The p value is: 0.69"
```

```r
# same things for adjusted model
coefficients_a <- coef(model_a)
standard_errors_a <- summary(model_a)$coefficients[, "Std. Error"]
var_a <- (log(2))^2 * standard_errors_a["log(glurange)"]^2
std_err_a <- sqrt(var_a)
coef_logRange_a <- coefficients_a["log(glurange)"]
coef_estimate_a <- exp(coef_logRange_a*log(2))

lower_CI_a <- coef_estimate_a * exp(critical_value_z * std_err_a["log(glurange)"])
upper_CI_a <- coef_estimate_a * exp(-critical_value_z * std_err_a["log(glurange)"])

# Conducting the Wald Z test
wald_z_a <- coef_logRange_a / standard_errors_a["log(glurange)"]
p_value_a <- 2 * (1 - pnorm(abs(wald_z_a)))


# print out values
print(paste0("Odds ratio and 95% CI in Secondary exposure adjusted model Range is: ",
            round(coef_estimate_a[["log(glurange)"]], 2),
            " (", round(lower_CI_a[["log(glurange)"]], 2), ", ",
            round(upper_CI_a[["log(glurange)"]], 2),")" ))
```

```
## [1] "Odds ratio and 95% CI in Secondary exposure adjusted model Range is: 1.06 (0.87, 1.28)"
```

```r
print(paste0("The p value is: ",
             round(p_value_a[["log(glurange)"]], 2)))
```

```
## [1] "The p value is: 0.53"
```

```r
# create a multivariable logistic regression model where the response is liver
# graft function "delayed_fn"
# interested predictor variable is now standard deviation of donor glucose
# measurements. "glusd"
# controling for other predictors such as age, cause of death, calculated model
# for end stage liver disease score, and hemodynamic instability "d_age",
# "d_cod", "hemo_instability"

model <- glm(delayed_fn ~ log(glusd), data = df, family = "binomial")
model_a <- glm(delayed_fn ~ log(glusd) + d_age + as.factor(d_cod) +
                 hemo_instability , data = df, family = "binomial")
```

```r
# odds ratio extraction, CI creation. Testing
coefficients <- coef(model)
standard_errors <- summary(model)$coefficients[, "Std. Error"]
var <- (log(2))^2 * standard_errors["log(glusd)"]^2
std_err <- sqrt(var)
coef_logSd <- coefficients["log(glusd)"]
coef_estimate <- exp(coef_logSd*log(2))

critical_value_z <- qnorm(0.0125)
lower_CI <- coef_estimate* exp(critical_value_z* std_err["log(glusd)"])
upper_CI<- coef_estimate* exp(-critical_value_z * std_err["log(glusd)"])

# Conducting the Wald Z test
wald_z <- coef_logSd / standard_errors["log(glusd)"]
p_value<- 2 * (1 - pnorm(abs(wald_z)))


# printing out
print(paste0("Odds ratio and 95% CI in unadjusted model standard deviation is: ",
            round(coef_estimate[["log(glusd)"]], 2),
            " (", round(lower_CI[["log(glusd)"]], 2), ", ",
            round(upper_CI[["log(glusd)"]], 2),")" ))
```

Standard Deviation

```
## [1] "Odds ratio and 95% CI in unadjusted model standard deviation is: 1.12 (0.91, 1.38)"
```

```r
print(paste0("The p value is: ",
             round(p_value[["log(glusd)"]], 2)))
```

```
## [1] "The p value is: 0.24"
```

```r
# same things for adjusted model
coefficients_a <- coef(model_a)
```

```r
standard_errors_a <- summary(model_a)$coefficients[, "Std. Error"]
var_a <- (log(2))^2 * standard_errors_a["log(glusd)"]^2
std_err_a <- sqrt(var_a)
coef_logSd_a <- coefficients_a["log(glusd)"]
coef_estimate_a <- exp(coef_logSd_a*log(2))

lower_CI_a <- coef_estimate_a * exp(critical_value_z * std_err_a["log(glusd)"])
upper_CI_a <- coef_estimate_a * exp(-critical_value_z * std_err_a["log(glusd)"])

# Conducting the Wald Z test
wald_z_a <- coef_logSd_a / standard_errors_a["log(glusd)"]
p_value_a <- 2 * (1 - pnorm(abs(wald_z_a)))


# print out values
print(paste0("Odds ratio and 95% CI in adjusted model standard deviation is: ",
            round(coef_estimate_a[["log(glusd)"]], 2),
            " (", round(lower_CI_a[["log(glusd)"]], 2), ", ",
            round(upper_CI_a[["log(glusd)"]], 2),")" ))
```

```
## [1] "Odds ratio and 95% CI in adjusted model standard deviation is: 1.13 (0.91, 1.41)"
```

```r
print(paste0("The p value is: ",
             round(p_value_a[["log(glusd)"]], 2)))
```

```
## [1] "The p value is: 0.2"
```

The results were produced in about 6-8 hours. The part that actually took the most work was figuring out how to visually present the box plots without outliers and with horizontal lines at the ends of the whiskers, which I understand is not the most important in terms of reproducibility of results, but I still found worth it to spend a little bit extra time on making them visually complete. The actual models were easier to recreate, however, the odds ratios actually took a few more trials for me to understand. I don't think this is a common way of manipulating the coefficients / interpreting them that we have learned about before in class, and it took some googling and asking chatGPT.

I was able to mostly reproduce these results, the confidence intervals were sometimes off by about 1/100ths from the values stated in the table from the paper. From asking the professor, I believe the reason for these small discrepancies is because in the paper, they are doing a profile likelihood instead of a wald test to creat the CIs. There were also some actual big discrepancies with regard to the final log odds estimated in with the Secondary Exposures : Standard Deviation, between my results and the table in the paper. My results yielded the log odds : 1.12 and 1.13 for unadjusted and adjusted respectively, and in the table they were 1.03 for both. However, in another part of the paper, they do offer in writing, a very close log odds ratio of 1.14 " standard deviation of donor glucose measurements (P = .13, odds ratio corresponding to a relative doubling in TWA of donor glucose measurements of 1.14 [0.92, 1.42]) " . I believe that this may be simply a misprint in the table.

Overall I found it much easier than expected to reproduce these results, I was surprised that even though I wasn't completely familiar with the methods used, it was very easy to learn about and replicate the coding that was used.