Lucas Lyons STA238 Final Project Appendix

2022-03-20

Citation for data set: https://CRAN.R-project.org/package=Stat2Data

Appendix 1

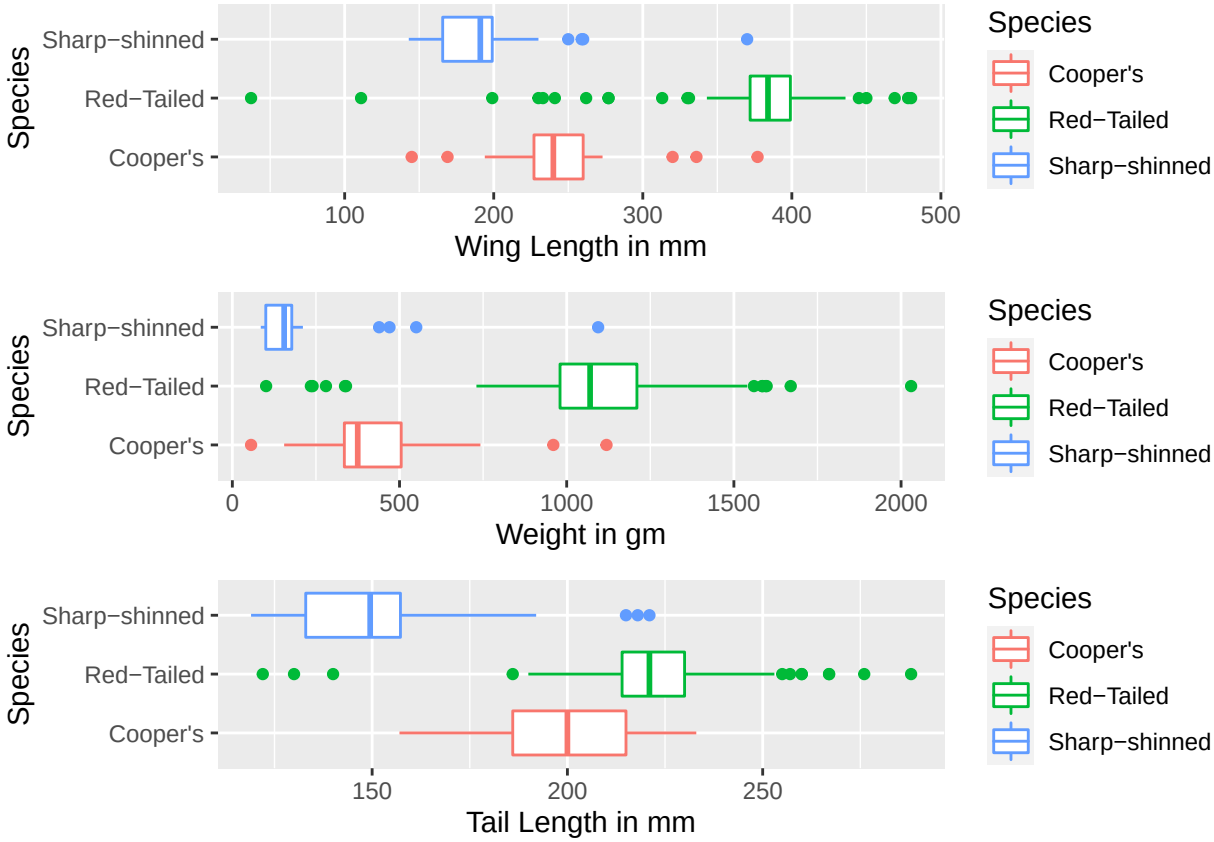
```
#Count missing data
count <- Hawks %>%
  gather(data, value, Age:Tail) %>%
  filter(is.na(value))
length(count)
```

```
## [1] 14
```

```
# Load in some variables
hawks <- Hawks %>%
  select(c("Species", "Age", "Sex", "Wing", "Weight", "Tail"))
# Remove missing data
hawks <- na.omit(hawks)
# Label Species
hawks <- hawks %>%
  mutate(Species = recode(Species,
                          RT = "Red-Tailed",
                          SS = "Sharp-shinned",
                          CH = "Cooper's"))
```

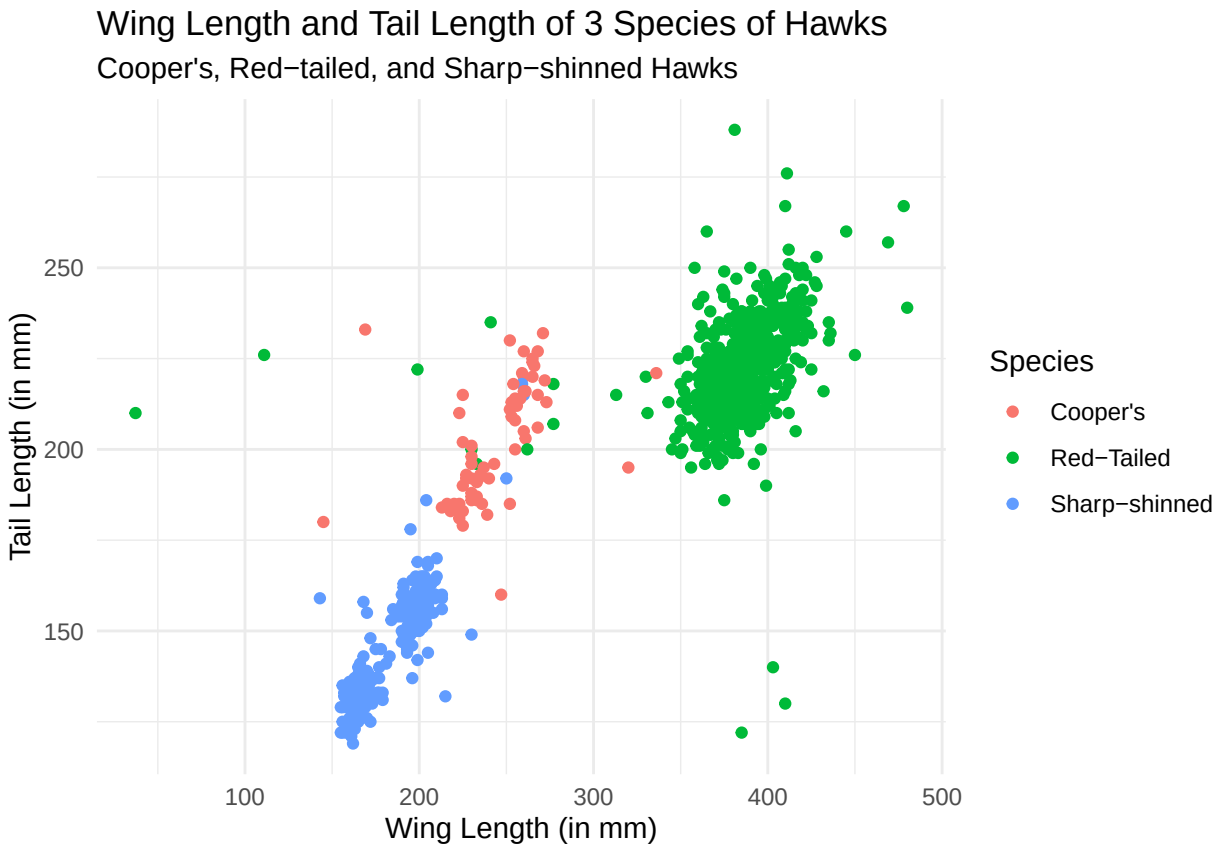
Appendix 2

```
#make box plots
p1 <- ggplot(hawks, aes(x=Species, y=Wing, color=Species)) +
  geom_boxplot() +
  ylab("Wing Length in mm") +
  coord_flip() #flip x and y
p2 <- ggplot(hawks, aes(x=Species, y=Weight, color=Species)) +
  geom_boxplot() +
  ylab("Weight in gm") +
  coord_flip()
p3 <- ggplot(hawks, aes(x=Species, y=Tail, color=Species)) +
  geom_boxplot() +
  ylab("Tail Length in mm") +
  coord_flip()

#arrange!
grid.arrange(p1, p2, p3)
```



Appendix 3

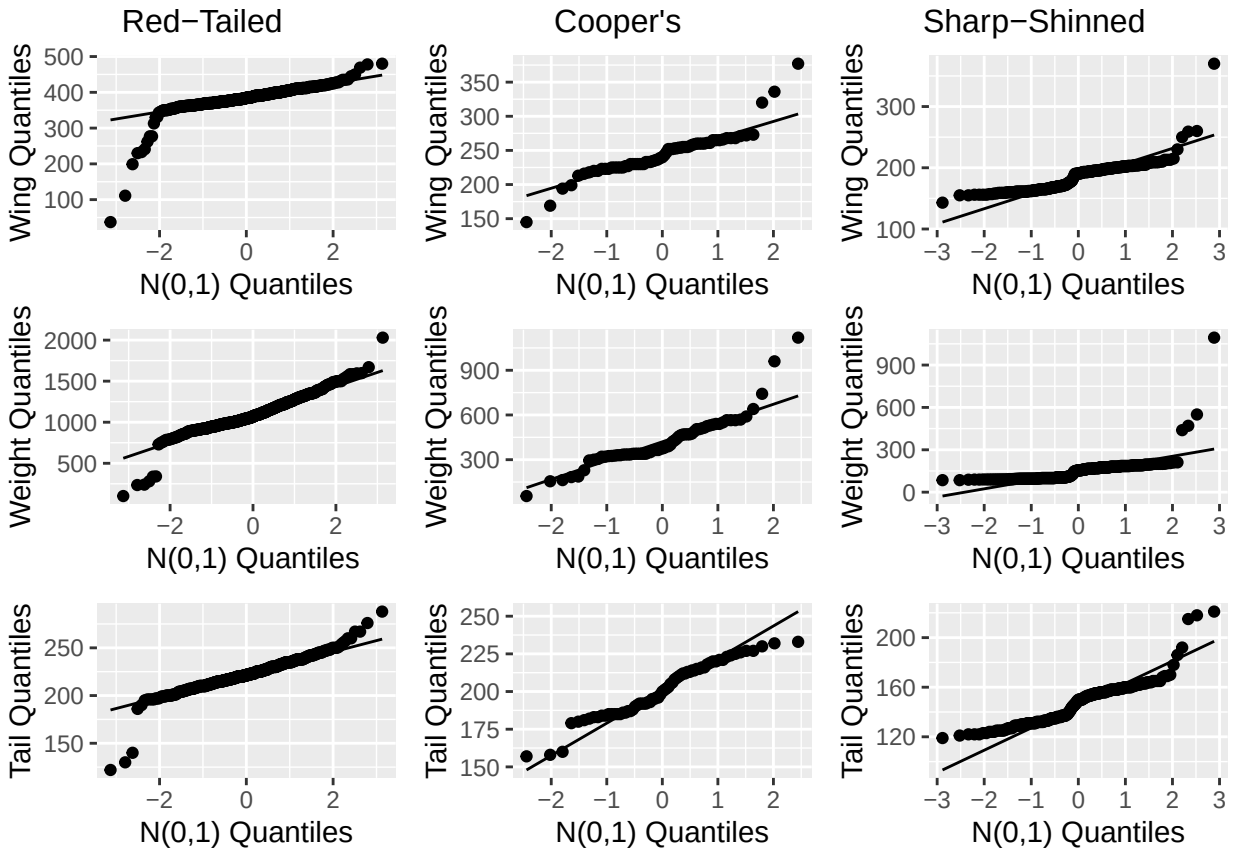
```
hawks %>%
  ggplot(aes(x = Wing, y = Tail, color = Species)) +
  geom_point() +
  labs(title = "Wing Length and Tail Length of 3 Species of Hawks",
       subtitle = "Cooper's, Red-tailed, and Sharp-shinned Hawks") +
  ylab("Tail Length (in mm)") +
  xlab("Wing Length (in mm)") +
  theme_minimal()
```



Appendix 4

```
#Subset our data by species
red <- hawks %>%
  subset(Species == "Red-Tailed")
sharp <- hawks %>%
  subset(Species == "Sharp-shinned")
coop <- hawks %>%
  subset(Species == "Cooper's")

#Assess whether our data is normally distributed via qq plot

#Create list of variable names
```

```r
params <- names(hawks)

#Create empty lists for qq plots
plots.red <- list()
plots.coop <- list()
plots.sharp <- list()

#Loops making qq plots for wing, tail, and weight for each species
for(i in 4:6){
  plot <- red %>%
    ggplot(aes_string(sample=params[i])) +
    geom_qq() +
    geom_qq_line() +
    xlab("N(0,1) Quantiles") +
    ylab(paste(params[i], "Quantiles"))
  plots.red[[i-3]] <- plot}

for(i in 4:6){
  plot <- coop %>%
    ggplot(aes_string(sample=params[i])) +
    geom_qq() +
    geom_qq_line() +
    xlab("N(0,1) Quantiles") +
    ylab(paste(params[i], "Quantiles"))
  plots.coop[[i-3]] <- plot}

for(i in 4:6){
  plot <- sharp %>%
    ggplot(aes_string(sample=params[i])) +
    geom_qq() +
    geom_qq_line() +
    xlab("N(0,1) Quantiles") +
    ylab(paste(params[i], "Quantiles"))
  plots.sharp[[i-3]] <- plot}

#Arrange our plots in a nice format
grid.arrange(arrangeGrob(grobs = plots.red, top = "Red-Tailed"),
             arrangeGrob(grobs = plots.coop, top = "Cooper's"),
             arrangeGrob(grobs = plots.sharp, top = "Sharp-Shinned"), ncol=3)
```




```
#Set parameters
len <- c(length(red$Wing), length(coop$Wing), length(sharp$Wing))
B <- 2000
set.seed(1001755451)

#Create empty matrices
boot.wing.red <- matrix(ncol = len[1], nrow = B)
boot.weight.red <- matrix(ncol = len[1], nrow = B)
boot.tail.red <- matrix(ncol = len[1], nrow = B)

boot.wing.coop <- matrix(ncol = len[2], nrow = B)
boot.weight.coop <- matrix(ncol = len[2], nrow = B)
boot.tail.coop <- matrix(ncol = len[2], nrow = B)

boot.wing.sharp <- matrix(ncol = len[3], nrow = B)
boot.weight.sharp <- matrix(ncol = len[3], nrow = B)
boot.tail.sharp <- matrix(ncol = len[3], nrow = B)

#Bootstrap!
for(i in 1:B){
  for(x in 1:len[1]){
    boot.wing.red[i,x] <- sample(red$Wing, 1, replace = TRUE)
  }
}

for(i in 1:B){
```

```
  for(x in 1:len[1]){
    boot.weight.red[i,x] <- sample(red$Weight, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[1]){
    boot.tail.red[i,x] <- sample(red$Tail, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[2]){
    boot.wing.coop[i,x] <- sample(coop$Wing, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[2]){
    boot.weight.coop[i,x] <- sample(coop$Weight, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[2]){
    boot.tail.coop[i,x] <- sample(coop$Tail, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[3]){
    boot.wing.sharp[i,x] <- sample(sharp$Wing, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[3]){
    boot.weight.sharp[i,x] <- sample(sharp$Weight, 1, replace = TRUE)
  }
}

for(i in 1:B){
  for(x in 1:len[3]){
    boot.tail.sharp[i,x] <- sample(sharp$Tail, 1, replace = TRUE)
  }
}
```

```
#Generate parameters
mean.wing.red <- mean(red$Wing)
sd.wing.red <- sd(red$Wing)
mean.weight.red <- mean(red$Weight)
sd.weight.red <- sd(red$Weight)
mean.tail.red <- mean(red$Tail)
```

```
sd.tail.red <- sd(red$Tail)
mean.wing.coop <- mean(coop$Wing)
sd.wing.coop <- sd(coop$Wing)
mean.weight.coop <- mean(coop$Weight)
sd.weight.coop <- sd(coop$Weight)
mean.tail.coop <- mean(coop$Tail)
sd.tail.coop <- sd(coop$Tail)
mean.wing.sharp <- mean(sharp$Wing)
sd.wing.sharp <- sd(sharp$Wing)
mean.weight.sharp <- mean(sharp$Weight)
sd.weight.sharp <- sd(sharp$Weight)
mean.tail.sharp <- mean(sharp$Tail)
sd.tail.sharp <- sd(sharp$Tail)

#Create empty vectors
bs.means.redwing <- c()
bs.means.redweight <- c()
bs.means.redtail <- c()
bs.means.coopwing <- c()
bs.means.coopweight <- c()
bs.means.cooptail <- c()
bs.means.sharpwing <- c()
bs.means.sharpweight <- c()
bs.means.sharptail <- c()

#Create empty list for confidence intervals
ci <- list()

# Find studentized means and plug in result to
# confidence interval list
for(i in 1:B){
  boot.mean <- mean(boot.wing.red[i,])
  boot.sd <- sd(boot.wing.red[i,])
  bs.means.redwing[i] <- (boot.mean - mean.wing.red)/(boot.sd/sqrt(len[1]))
}
crit <- quantile(bs.means.redwing, probs=c(0.995, 0.005))
ci[[1]] <- mean.wing.red-crit*sd.wing.red/sqrt(len[1])

for(i in 1:B){
  boot.mean <- mean(boot.weight.red[i,])
  boot.sd <- sd(boot.weight.red[i,])
  bs.means.redweight[i] <- (boot.mean - mean.weight.red)/(boot.sd/sqrt(len[1]))
}
crit <- quantile(bs.means.redweight, probs=c(0.995, 0.005))
ci[[2]] <- mean.weight.red-crit*sd.weight.red/sqrt(len[1])

for(i in 1:B){
  boot.mean <- mean(boot.tail.red[i,])
  boot.sd <- sd(boot.tail.red[i,])
  bs.means.redtail[i] <- (boot.mean - mean.tail.red)/(boot.sd/sqrt(len[1]))
}
crit <- quantile(bs.means.redtail, probs=c(0.995, 0.005))
ci[[3]] <- mean.tail.red-crit*sd.tail.red/sqrt(len[1])
```

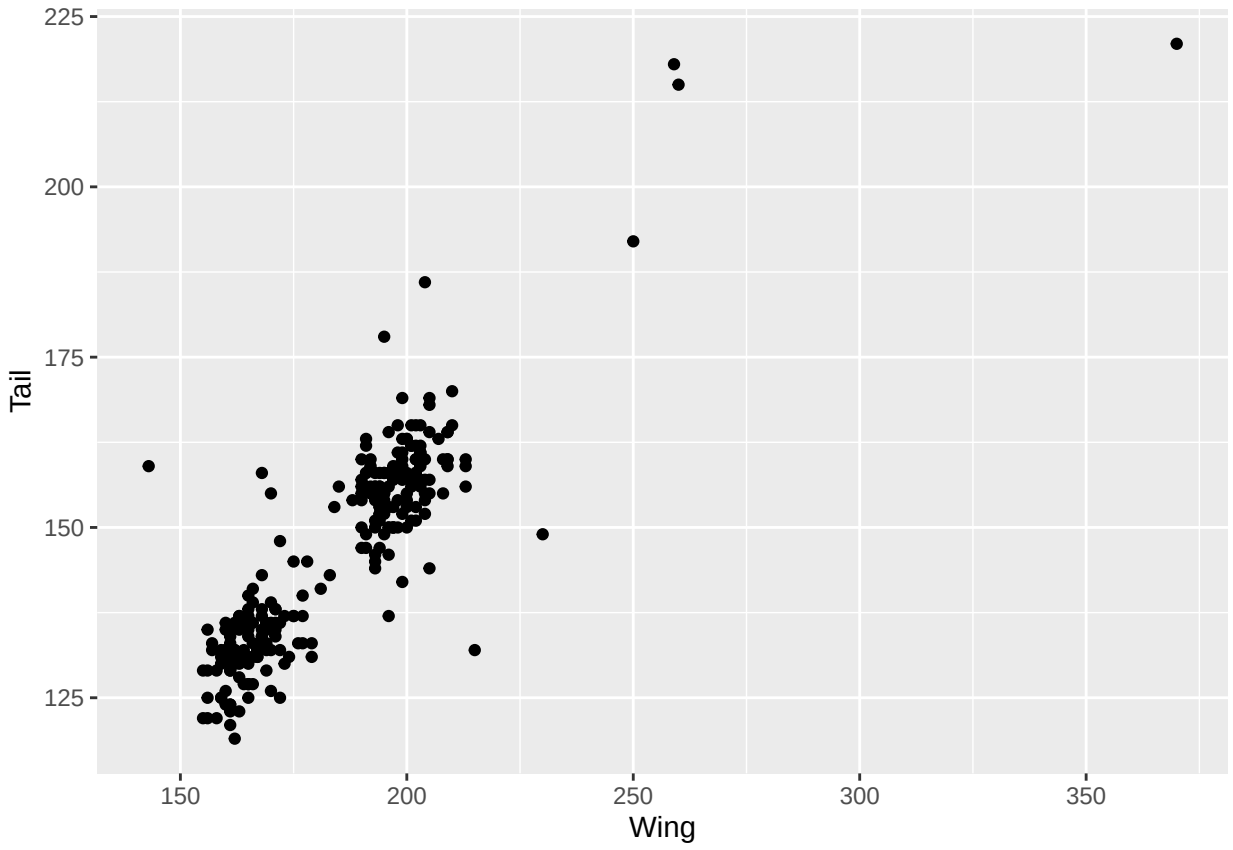
```
for(i in 1:B){
  boot.mean <- mean(boot.wing.coop[i,])
  boot.sd <- sd(boot.wing.coop[i,])
  bs.means.coopwing[i] <- (boot.mean - mean.wing.coop)/(boot.sd/sqrt(len[2]))
}
crit <- quantile(bs.means.coopwing, probs=c(0.995, 0.005))
ci[[4]] <- mean.wing.coop-crit*sd.wing.coop/sqrt(len[2])

for(i in 1:B){
  boot.mean <- mean(boot.weight.coop[i,])
  boot.sd <- sd(boot.weight.coop[i,])
  bs.means.coopweight[i] <- (boot.mean - mean.weight.coop)/(boot.sd/sqrt(len[2]))
}
crit <- quantile(bs.means.coopweight, probs=c(0.995, 0.005))
ci[[5]] <- mean.weight.coop-crit*sd.weight.coop/sqrt(len[2])

for(i in 1:B){
  boot.mean <- mean(boot.tail.coop[i,])
  boot.sd <- sd(boot.tail.coop[i,])
  bs.means.cooptail[i] <- (boot.mean - mean.tail.coop)/(boot.sd/sqrt(len[2]))
}
crit <- quantile(bs.means.cooptail, probs=c(0.995, 0.005))
ci[[6]] <- mean.tail.coop-crit*sd.tail.coop/sqrt(len[2])

for(i in 1:B){
  boot.mean <- mean(boot.wing.sharp[i,])
  boot.sd <- sd(boot.wing.sharp[i,])
  bs.means.sharpwing[i] <- (boot.mean - mean.wing.sharp)/(boot.sd/sqrt(len[3]))
}
crit <- quantile(bs.means.sharpwing, probs=c(0.995, 0.005))
ci[[7]] <- mean.wing.sharp-crit*sd.wing.sharp/sqrt(len[3])

for(i in 1:B){
  boot.mean <- mean(boot.weight.sharp[i,])
  boot.sd <- sd(boot.weight.sharp[i,])
  bs.means.sharpweight[i] <- (boot.mean - mean.weight.sharp)/(boot.sd/sqrt(len[3]))
}
crit <- quantile(bs.means.sharpweight, probs=c(0.995, 0.005))
ci[[8]] <- mean.weight.sharp-crit*sd.weight.sharp/sqrt(len[3])

for(i in 1:B){
  boot.mean <- mean(boot.tail.sharp[i,])
  boot.sd <- sd(boot.tail.sharp[i,])
  bs.means.sharptail[i] <- (boot.mean - mean.tail.sharp)/(boot.sd/sqrt(len[3]))
}
crit <- quantile(bs.means.sharptail, probs=c(0.995, 0.005))
ci[[9]] <- mean.tail.sharp-crit*sd.tail.sharp/sqrt(len[3])
ci
```

```
## [[1]]
##    99.5%     0.5%
## 379.2818 386.4965
##
```

```
## [[2]]
##    99.5%     0.5%
## 1073.321 1115.160
##
## [[3]]
##    99.5%     0.5%
## 220.5654 223.6654
##
## [[4]]
##    99.5%     0.5%
## 234.2050 255.2423
##
## [[5]]
##    99.5%     0.5%
## 374.1991 479.8345
##
## [[6]]
##    99.5%     0.5%
## 195.1662 206.5352
##
## [[7]]
##    99.5%     0.5%
## 181.7455 188.9006
##
## [[8]]
##    99.5%     0.5%
## 138.3456 168.1399
##
## [[9]]
##    99.5%     0.5%
## 144.2415 149.2940
```

Appendix 5

```
#Check for linear relation
sharp %>%
  ggplot(aes(x=Wing,y=Tail)) +
  geom_point()
```

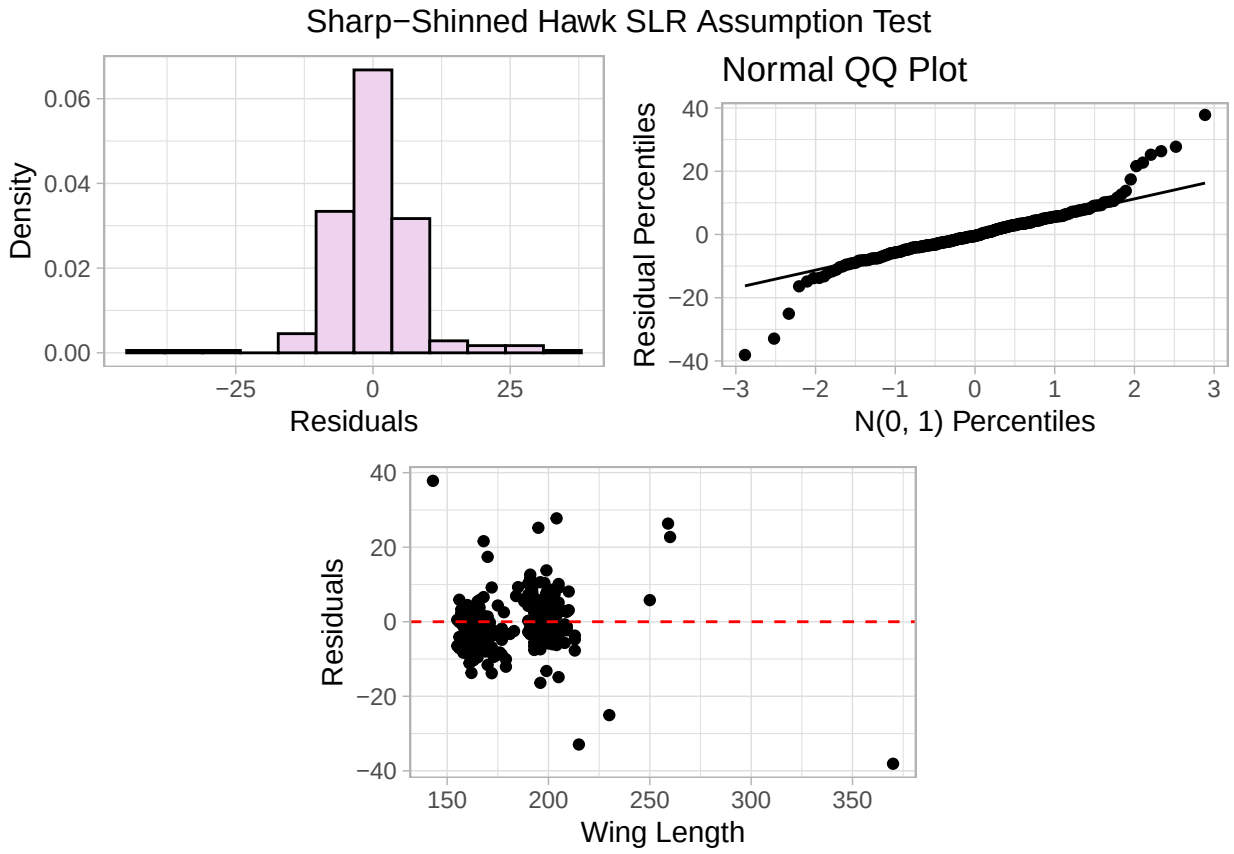



```
#Create linear model
model.sharp <- lm(sharp$Tail ~ sharp$Wing)
summary(model.sharp)
```

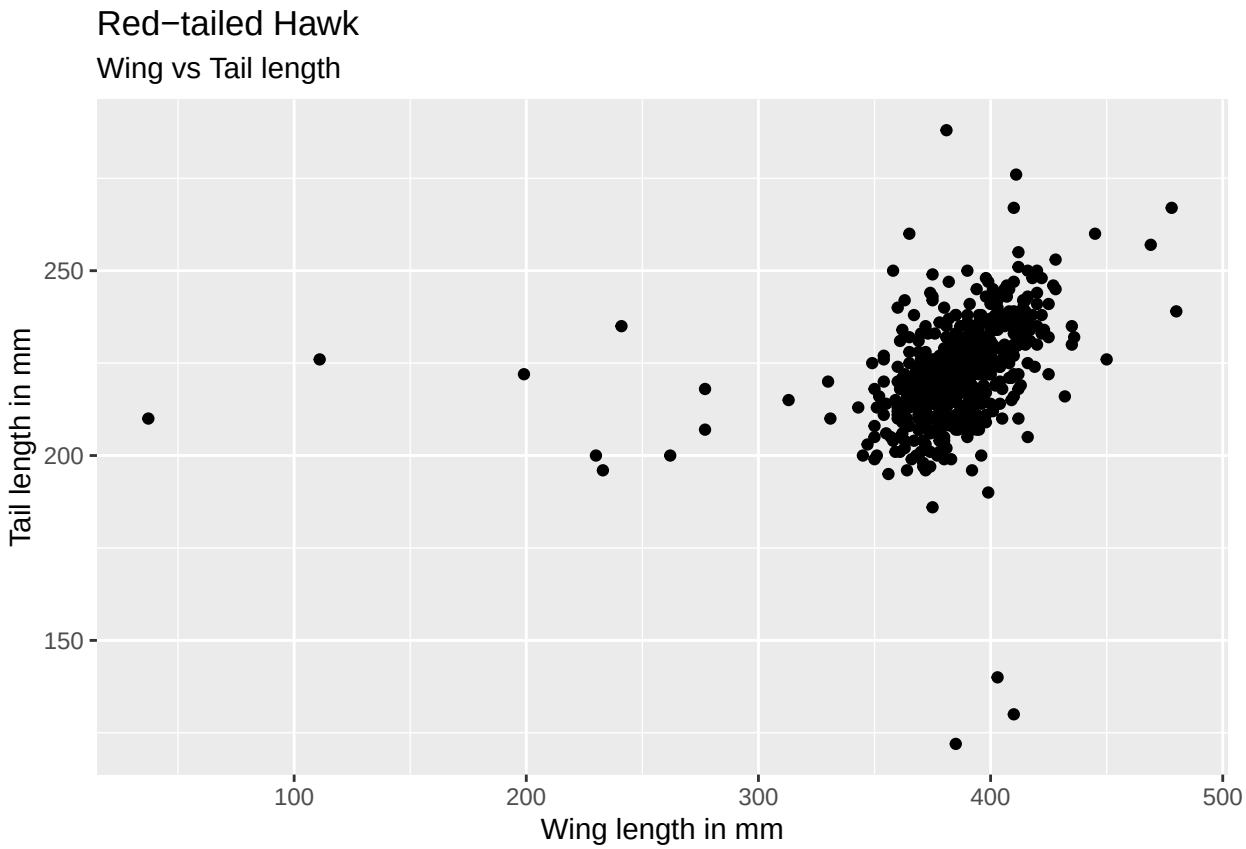
```
##
## Call:
## lm(formula = sharp$Tail ~ sharp$Wing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -38.105  -3.807  -0.479   3.791  37.803
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 34.32042    4.01270   8.553 1.15e-15 ***
## sharp$Wing   0.60753    0.02155  28.191  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 7.752 on 254 degrees of freedom
## Multiple R-squared:  0.7578, Adjusted R-squared:  0.7568
## F-statistic: 794.7 on 1 and 254 DF,  p-value: < 2.2e-16
```

```
#Create new entry in data set for residuals
sharp$res <- model.sharp$residuals

#Histogram for normalcy test
hist <- ggplot(sharp, aes(x=res, y=..density..))+
  geom_histogram(bins = 12,
                 fill='thistle2',
                 colour='black')+
  theme_light()+
  labs(x='Residuals',
       y='Density')

#qq plot for normalcy test
qq <- ggplot(sharp, aes(sample=res))+
  geom_qq()+
  geom_qq_line()+
  theme_light()+
  labs(x='N(0, 1) Percentiles',
       y='Residual Percentiles',
       title='Normal QQ Plot')

#Scatter plot for residual independence and distribution test
scatter <- ggplot(sharp, aes(x=Wing, y=res))+
  geom_point()+
  geom_hline(yintercept=0, colour='red', lty=2)+
  theme_light()+
  labs(x='Wing Length',
       y='Residuals')

#Arrange nicely!
grid.arrange(hist, qq, scatter,
             layout_matrix=rbind(c(1, 1, 2, 2),
                                 c(NA, 3, 3, NA)),
             top = "Sharp-Shinned Hawk SLR Assumption Test")
```




```
#Check for linear relation
red %>%
  ggplot(aes(x=Wing,y=Tail)) +
  geom_point() +
  xlab("Wing length in mm") +
  ylab("Tail length in mm") +
  ggtitle("Red-tailed Hawk", sub="Wing vs Tail length")
```

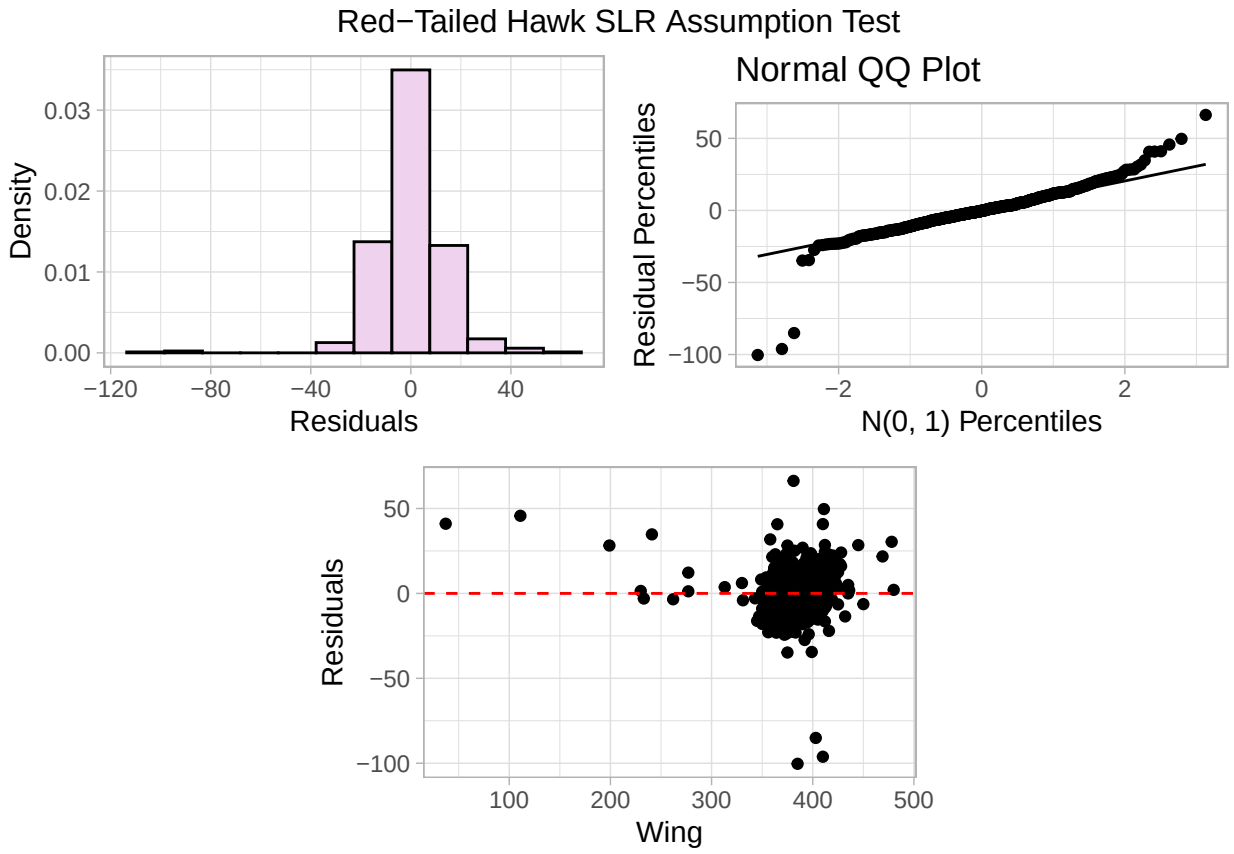



```
#Create linear model
model.red <- lm(red$Tail ~ red$Wing)
summary(model.red)
```

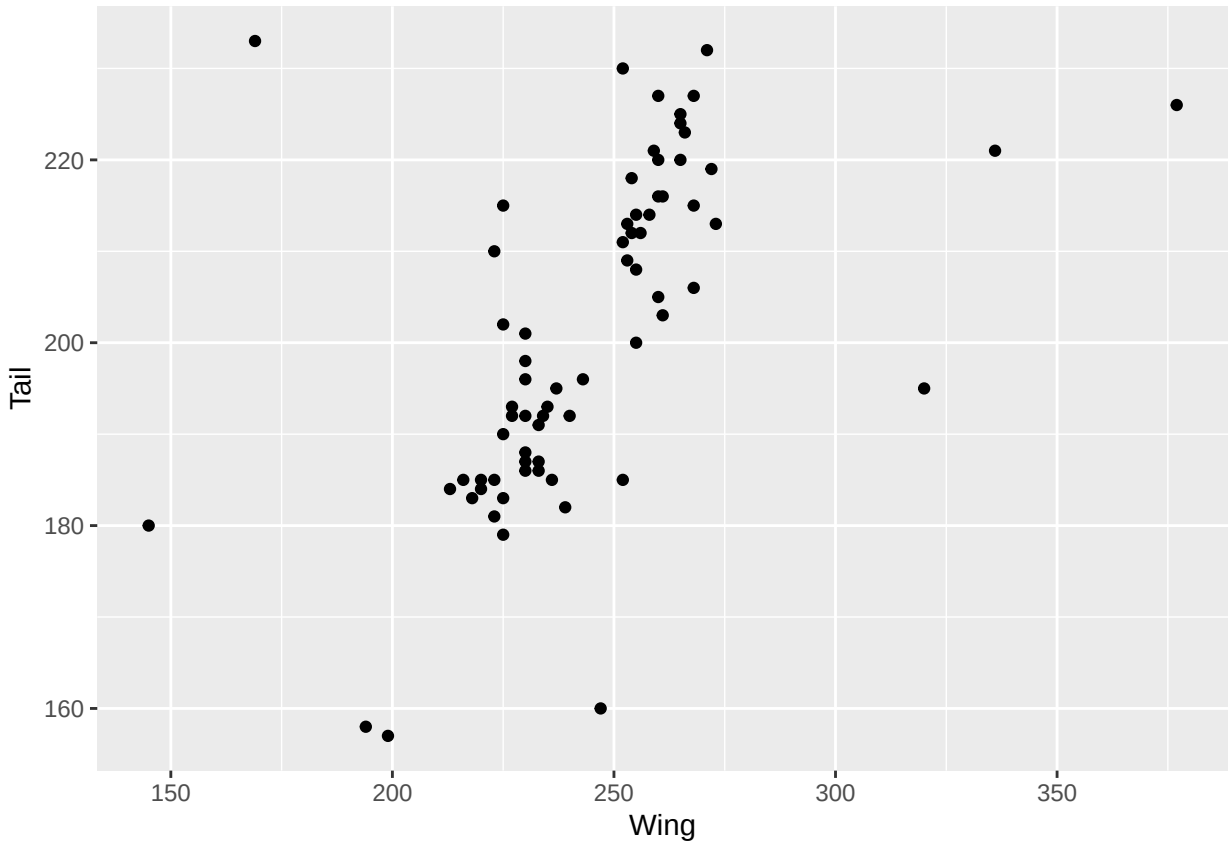
```
##
## Call:
## lm(formula = red$Tail ~ red$Wing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -100.343   -6.830   -0.301    6.928   66.270
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 163.26036    7.10724   22.97  < 2e-16 ***
## red$Wing      0.15346    0.01847    8.31 7.04e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.75 on 570 degrees of freedom
## Multiple R-squared:  0.1081, Adjusted R-squared:  0.1065
## F-statistic: 69.05 on 1 and 570 DF,  p-value: 7.041e-16
```

```
#Create new entry in data set for residuals
red$res <- model.red$residuals
```

```r
#Histogram for normalcy test
hist2 <- ggplot(red, aes(x=res, y=..density..))+
  geom_histogram(bins = 12,
                 fill='thistle2',
                 colour='black')+
  theme_light()+
  labs(x='Residuals',
       y='Density')

#qq plot for normalcy test
qq2 <- ggplot(red, aes(sample=res))+
  geom_qq()+
  geom_qq_line()+
  theme_light()+
  labs(x='N(0, 1) Percentiles',
       y='Residual Percentiles',
       title='Normal QQ Plot')

#Scatter plot for residual independence and distribution test
scatter2 <- ggplot(red, aes(x=Wing, y=res))+
  geom_point()+
  geom_hline(yintercept=0, colour='red', lty=2)+
  theme_light()+
  labs(x='Wing',
       y='Residuals')

#Arrange nicely!
grid.arrange(hist2, qq2, scatter2,
             layout_matrix=rbind(c(1, 1, 2, 2),
                                 c(NA, 3, 3, NA)),
             top = "Red-Tailed Hawk SLR Assumption Test")
```




```
#Check for linear relation
coop %>%
  ggplot(aes(x=Wing,y=Tail)) +
  geom_point()
```

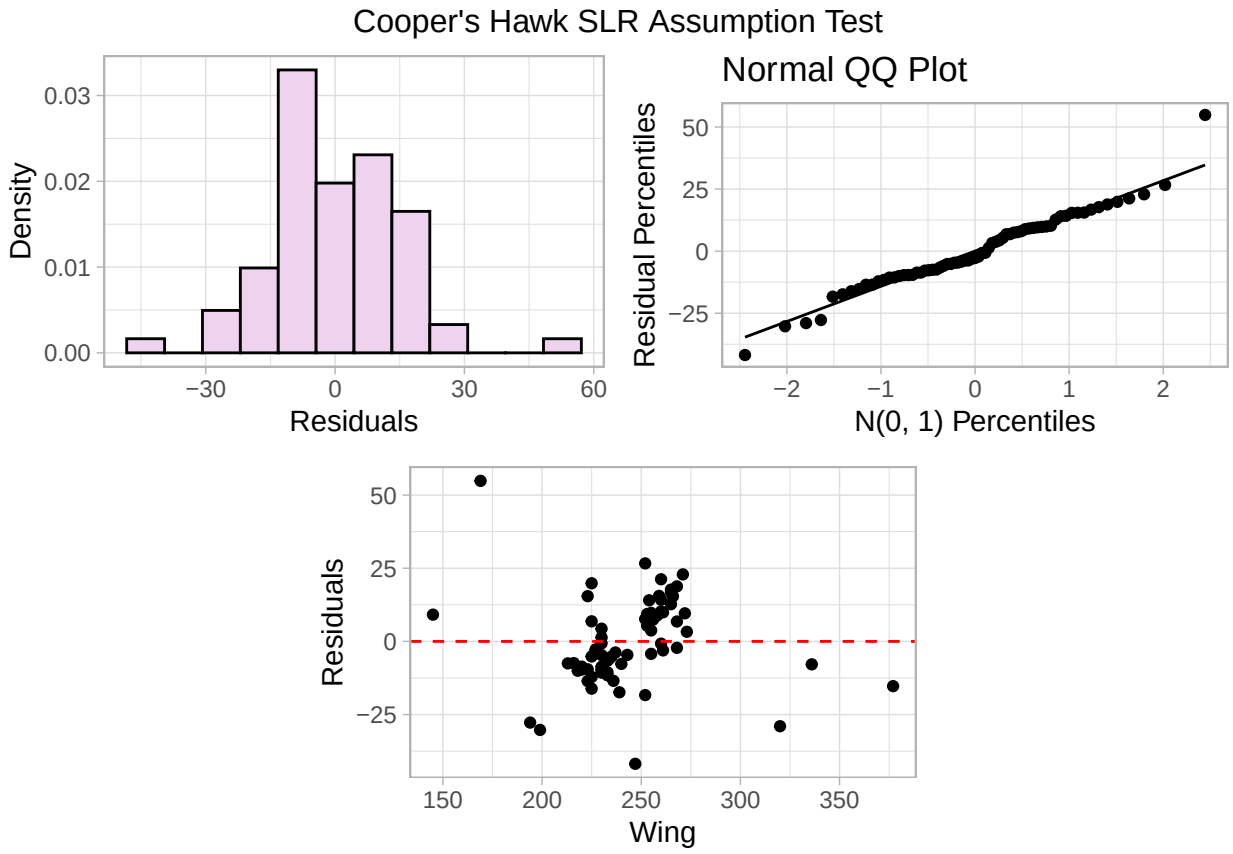



```
#Create linear model
model.coop <- lm(coop$Tail ~ coop$Wing)
summary(model.coop)
```

```
##
## Call:
## lm(formula = coop$Tail ~ coop$Wing)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -41.823  -9.538  -2.752   9.588  54.854
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.84625   14.17062   8.951 4.68e-13 ***
## coop$Wing     0.30355    0.05755   5.274 1.54e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.25 on 67 degrees of freedom
## Multiple R-squared:  0.2934, Adjusted R-squared:  0.2828
## F-statistic: 27.82 on 1 and 67 DF,  p-value: 1.539e-06
```

```
#Create new entry in data set for residuals
coop$res <- model.coop$residuals
```

```r
#Histogram for normalcy test
hist3 <- ggplot(coop, aes(x=res, y=..density..))+
  geom_histogram(bins = 12,
                 fill='thistle2',
                 colour='black')+
  theme_light()+
  labs(x='Residuals',
       y='Density')

#qq plot for normalcy test
qq3 <- ggplot(coop, aes(sample=res))+
  geom_qq()+
  geom_qq_line()+
  theme_light()+
  labs(x='N(0, 1) Percentiles',
       y='Residual Percentiles',
       title='Normal QQ Plot')

#Scatter plot for residual independence and distribution test
scatter3 <- ggplot(coop, aes(x=Wing, y=res))+
  geom_point()+
  geom_hline(yintercept=0, colour='red', lty=2)+
  theme_light()+
  labs(x='Wing',
       y='Residuals')

#Arrange nicely!
grid.arrange(hist3, qq3, scatter3,
             layout_matrix=rbind(c(1, 1, 2, 2),
                                 c(NA, 3, 3, NA)),
             top = "Cooper's Hawk SLR Assumption Test")
```


Cooper's Hawk SLR Assumption Test
Density
0.03
0.02
0.01
0.00
−30
0
30
60
Residuals
Normal QQ Plot
Residual Percentiles
50
25
0
−25
−2
−1
0
1
2
N(0, 1) Percentiles
Residuals
50
25
0
−25
150
200
250
300
350
Wing

Appendix 6

```
#make confidence intervals from linear models
confint(model.sharp)
```

```
##                  2.5 %    97.5 %
## (Intercept) 26.4180233 42.222821
## sharp$Wing   0.5650864  0.649968
```

```
confint(model.red)
```

```
##                  2.5 %      97.5 %
## (Intercept) 149.300785 177.2199354
## red$Wing      0.117189   0.1897359
```

```
confint(model.coop)
```

```
##                  2.5 %     97.5 %
## (Intercept) 98.5615665 155.130938
## coop$Wing    0.1886747   0.418426
```