MBIO600_Final
Rmd_Fertitta_Tramonte_Stark-Kinimaka_Snyder

Jake Snyder

2023-12-11

Key: HMS = Hawaiian Monk Seal

Data Wrangling

Below, the HMS call frequency dataset from Lehua Rock is uploaded and filtered. From this dataset, three subsets are derived: "calls_by_hour_day," "average_calls_per_hour," and "df.summary." "calls_by_hour_day" is utilized to plot Figure 3 and conduct a Kruskal-Wallis test, manual changepoint detection, and bcp() changepoint detection. "average_calls_per_hour" is utilized to plot Figure 3. "df.summary" is utilized to plot Figure 1 and 2.

```
# Upload HMS call frequency data from Lehua Rock
LehuaRock <- read.csv("/Users/gails/Desktop/MonkSealProject/LehuaRock_Analysis_Updated.csv")

# Sort call type to only include HMS calls
LehuaRock <- LehuaRock %>%
  filter(Call %in% c("Croak", "Groan", "Growl", "Moan", "Whoop"))

# Add "Hour" variable to LehuaRock dataset
LehuaRock$Hour <- substr(LehuaRock$Start.time, 1, 2)
LehuaRock$Hour <- gsub(":", "", LehuaRock$Hour)
LehuaRock$Hour <- as.numeric(LehuaRock$Hour)

# Extract dataframe of number of calls per hour for each day
calls_by_hour_day <- table(LehuaRock$Hour, LehuaRock$Date, dnn = c('hour', 'day'))
calls_by_hour_day <- as.data.frame(calls_by_hour_day)
calls_by_hour_day$hour <- as.character(calls_by_hour_day$hour)
calls_by_hour_day$hour <- as.numeric(calls_by_hour_day$hour)

# Define zeros (indicating call frequency before and after deployment) as NA
calls_by_hour_day$Freq[calls_by_hour_day$day == '5/10/2021'
                       & calls_by_hour_day$hour < 12] <- NA
calls_by_hour_day$Freq[calls_by_hour_day$day == '5/16/2021'
                       & calls_by_hour_day$hour > 15] <- NA

# Filter out NA values
calls_by_hour_day <- calls_by_hour_day %>%
  filter(Freq != "NA")

# Add day vs night label to calls_by_hour_day
```

```r
is_day <- calls_by_hour_day$hour > 5 & calls_by_hour_day$hour < 19
calls_by_hour_day$day_night <- 'dayvsnight'
calls_by_hour_day$day_night[is_day] <- 'day'
calls_by_hour_day$day_night[!is_day] <- 'night'
head(calls_by_hour_day,10)
```

```
##    hour       day Freq day_night
## 1    12 5/10/2021    5       day
## 2    13 5/10/2021   24       day
## 3    14 5/10/2021   32       day
## 4    15 5/10/2021   32       day
## 5    16 5/10/2021   20       day
## 6    17 5/10/2021   28       day
## 7    18 5/10/2021    0       day
## 8    19 5/10/2021   31     night
## 9    20 5/10/2021   24     night
## 10   21 5/10/2021   20     night
```

```r
# Calculate average number of calls per hour across all days
average_calls_per_hour <- aggregate(Freq ~ hour, mean, data=calls_by_hour_day)
head(average_calls_per_hour,10)
```

```
##    hour     Freq
## 1     0 50.00000
## 2     1 34.16667
## 3     2 42.50000
## 4     3 44.83333
## 5     4 28.00000
## 6     5 33.83333
## 7     6 34.50000
## 8     7 39.50000
## 9     8 29.00000
## 10    9 35.00000
```

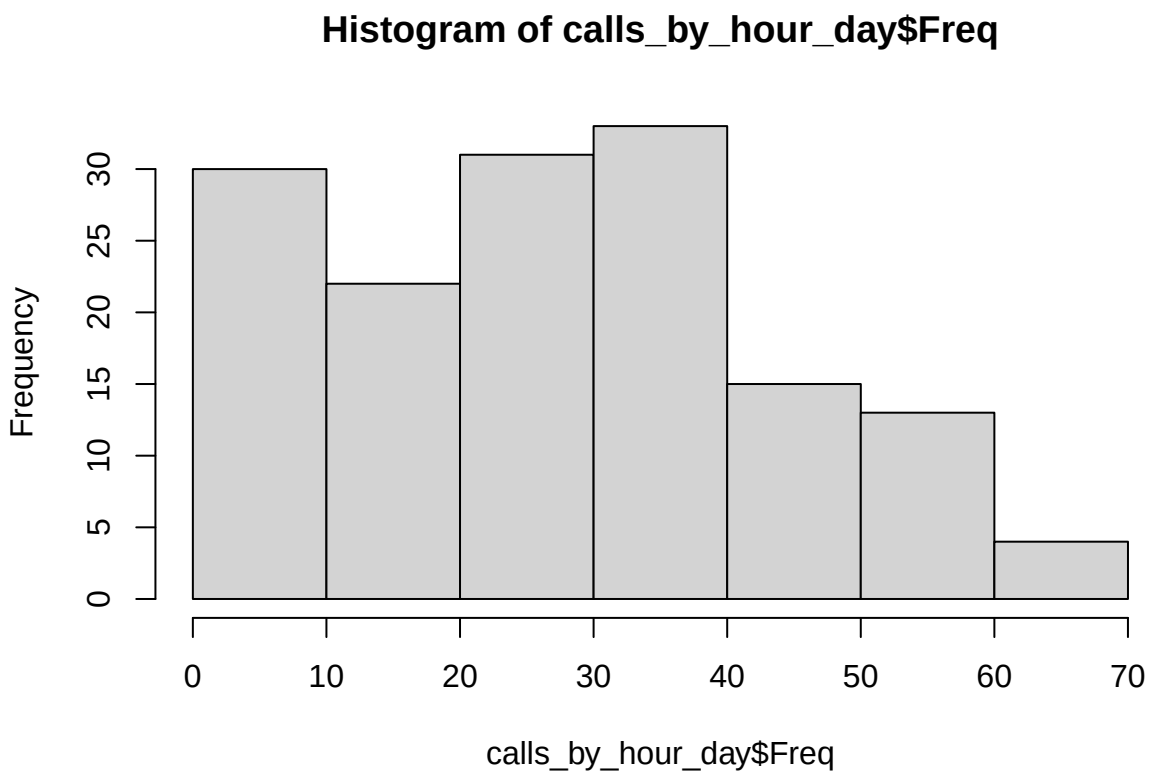
```r
# Create dataframe that includes sd of frequency of calls per hour
df.summary <- calls_by_hour_day %>%
  group_by(hour) %>%
  summarise(
    sd = sd(Freq, na.rm = TRUE),
    Freq = mean(Freq))
head(df.summary,10)
```

```
## # A tibble: 10 x 3
##     hour    sd  Freq
##    <dbl> <dbl> <dbl>
##  1     0 15.1   50
##  2     1  9.89  34.2
##  3     2 25.6   42.5
##  4     3 10.7   44.8
##  5     4 17.0   28
##  6     5  9.06  33.8
##  7     6 14.0   34.5
##  8     7 15.1   39.5
##  9     8 17.7   29
## 10     9  8.60  35
```

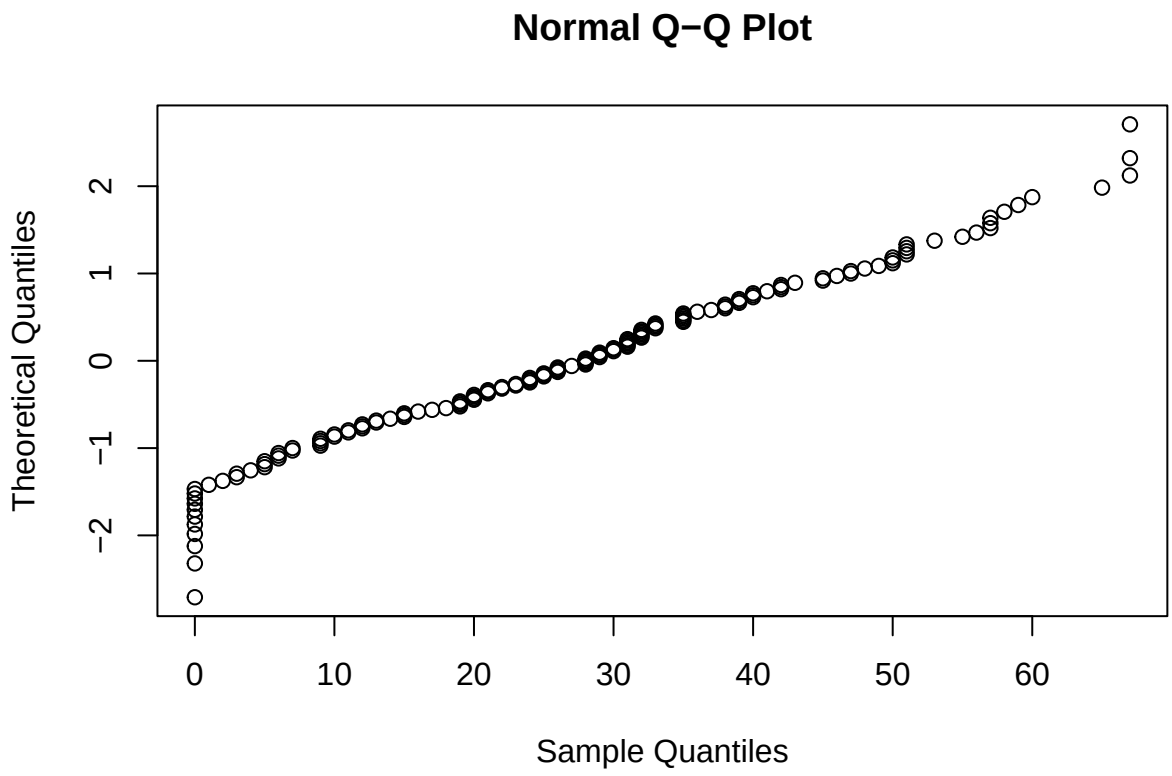
Analysis and Modeling

Below, a Shapiro-Wilk test for normality is applied to HMS call frequencies and yields a p-value < 0.05, both before and after a sqrt transformation. Thus, the null assuming normality is rejected, and a non-parametric Kruskal-Wallis test is conducted. The Kruskal-Wallis test yields a chi-squared value of 5.6244 and a p-value of 0.01771.

```
# Plot histogram of calls
hist(calls_by_hour_day$Freq)
```



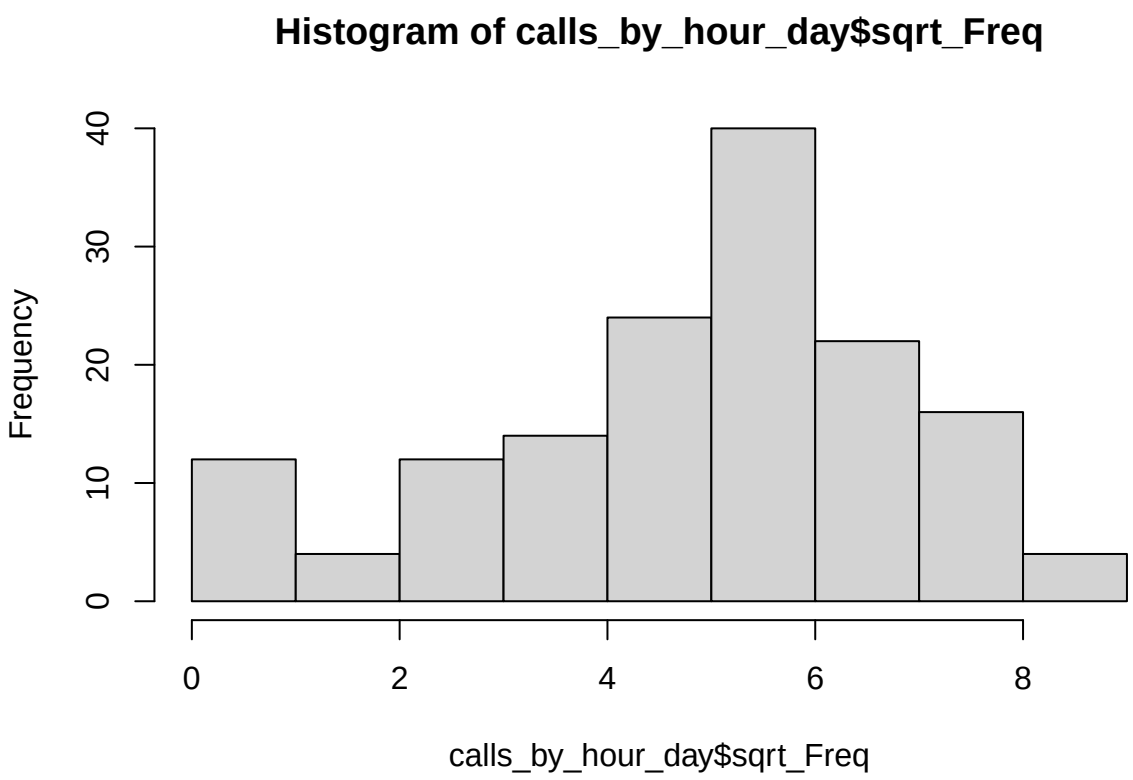

```
# Plot Q-Q plot for calls
qqnorm(calls_by_hour_day$Freq, datax = T)
```



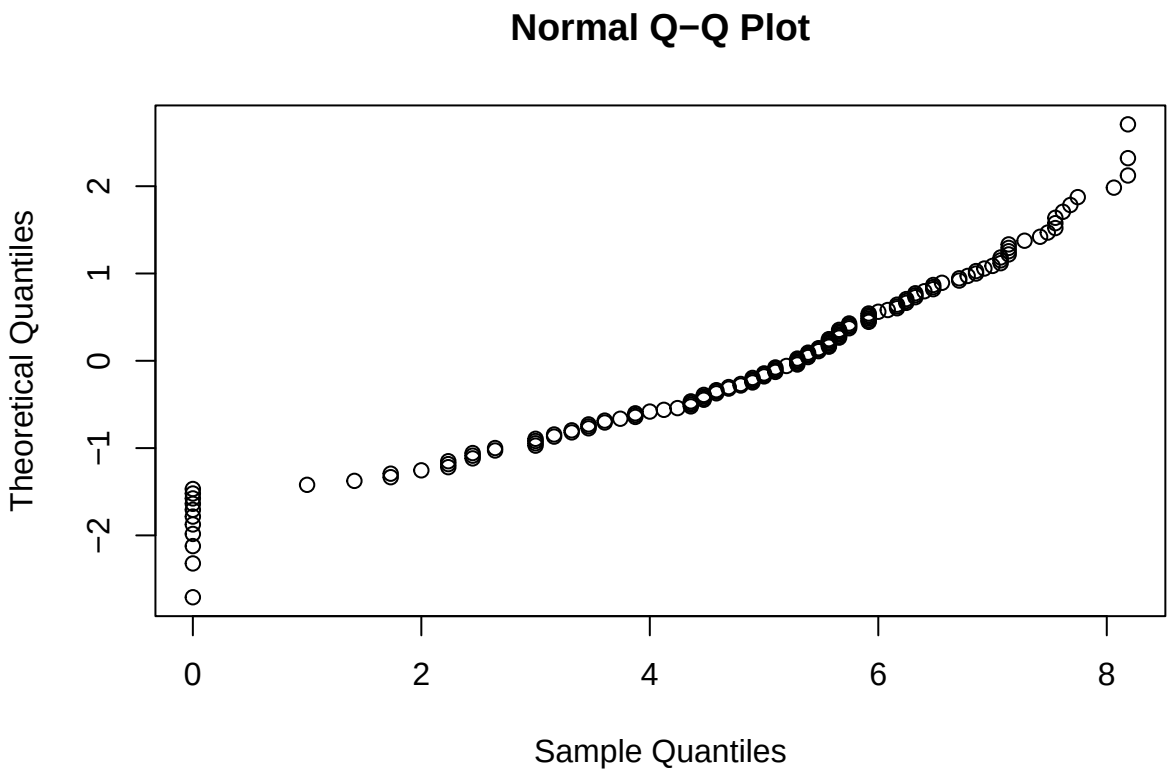

```
# Conduct Shapiro-Wilk normality test on non-transformed call frequency data
shapiro.test(calls_by_hour_day$Freq)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  calls_by_hour_day$Freq
## W = 0.97207, p-value = 0.004092
```

```
# sqrt transform call frequencies
calls_by_hour_day$sqrt_Freq <- sqrt(calls_by_hour_day$Freq)
# Plot histogram of transformed data
hist(calls_by_hour_day$sqrt_Freq)
```




```
# Plot Q-Q plot of transformed data
qqnorm(calls_by_hour_day$sqrt_Freq, datax = T)
```




```
# Conduct Shapiro-Wilk normality test on transformed call frequency data
shapiro.test(calls_by_hour_day$sqrt_Freq)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  calls_by_hour_day$sqrt_Freq
## W = 0.93074, p-value = 1.292e-06
```

```
# Conduct Kruskal-Wallis test
kruskal.test(calls_by_hour_day$Freq~calls_by_hour_day$day_night)
```

```
##
##  Kruskal-Wallis rank sum test
##
## data:  calls_by_hour_day$Freq by calls_by_hour_day$day_night
## Kruskal-Wallis chi-squared = 5.6244, df = 1, p-value = 0.01771
```

Below, the hour intervals during which the largest, significant magnitude of change in call frequency occurs are calculated. We can assume that these intervals contain a changepoint in call frequency. The intervals identified are from hours 20-23 (m = 9.444444, p-value = 0.0017497048), hours 9-12 (m = -8.666667, p-value = 0.0003129298), hours 7-10 (m = -8.444444, p-value = 0.0070150348), hours 12-14 (m = 7.285714, p-value = 0.0095862180), and hours 7-12 (m = -6.100000, p-value = 0.0009744568). The interval from hours 9-12 is most significant and has the second-largest slope, suggesting a significant change in call frequency and a potential changepoint.

```
HourIntFunc <- function(t0,t2,t3,t4,t5){

df <- filter(calls_by_hour_day, hour == t0 | hour == t2 |
               hour == t3 | hour == t4 | hour == t5)

w.df <- filter(df, hour == t0 | hour == t2)
x.df <- filter(df, hour == t0 | hour == t3)
y.df <- filter(df, hour == t0 | hour == t4)
z.df <- filter(df, hour == t0 | hour == t5)

w.lm <- lm(data = w.df, Freq ~ hour)
x.lm <- lm(data = x.df, Freq ~ hour)
y.lm <- lm(data = y.df, Freq ~ hour)
z.lm <- lm(data = z.df, Freq ~ hour)

w.sum <- summary(w.lm)
x.sum <- summary(x.lm)
y.sum <- summary(y.lm)
z.sum <- summary(z.lm)

w.coef <- w.sum$coefficients[2,c(1,4)]
x.coef <- x.sum$coefficients[2,c(1,4)]
y.coef <- y.sum$coefficients[2,c(1,4)]
z.coef <- z.sum$coefficients[2,c(1,4)]

coef.df <- rbind(w.coef, x.coef, y.coef, z.coef)

strt <- c(t0,t0,t0,t0)
strt <- data.frame(strt)

end <- c(t2,t3,t4,t5)
end <- data.frame(end)

HourInt <- cbind(strt, end, coef.df)

HourInt

}

x0 <-  HourIntFunc(0,2,3,4,5)
x1 <-  HourIntFunc(1,3,4,5,6)
x2 <-  HourIntFunc(2,4,5,6,7)
x3 <-  HourIntFunc(3,5,6,7,8)
x4 <-  HourIntFunc(4,6,7,8,9)
x5 <-  HourIntFunc(5,7,8,9,10)
```

```r
x6 <-  HourIntFunc(6,8,9,10,11)
x7 <-  HourIntFunc(7,9,10,11,12)
x8 <-  HourIntFunc(8,10,11,12,13)
x9 <-  HourIntFunc(9,11,12,13,14)
x10 <- HourIntFunc(10,12,13,14,15)
x11 <- HourIntFunc(11,13,14,15,16)
x12 <- HourIntFunc(12,14,15,16,17)
x13 <- HourIntFunc(13,15,16,17,18)
x14 <- HourIntFunc(14,16,17,18,19)
x15 <- HourIntFunc(15,17,18,19,20)
x16 <- HourIntFunc(16,18,19,20,21)
x17 <- HourIntFunc(17,19,20,21,22)
x18 <- HourIntFunc(18,20,21,22,23)
x19 <- HourIntFunc(19,21,22,23,0)
x20 <- HourIntFunc(20,22,23,0,1)
x21 <- HourIntFunc(21,23,0,1,2)
x22 <- HourIntFunc(22,0,1,2,3)
x23 <- HourIntFunc(23,1,2,3,4)

HourInt.lm.df <- rbind(x0,x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,
      x11,x12,x13,x14,x15,x16,x17,x18,
      x19,x20,x21,x22,x23)

colnames(HourInt.lm.df) <- c("start", "end", "slope", "p-value")

# Find 10 largest slopes in magnitude
HourInt.lm.df$abs.slope <- abs(HourInt.lm.df$slope)
top10.m <- head(HourInt.lm.df[order(-HourInt.lm.df$abs.slope),], 10)
top10.m
```

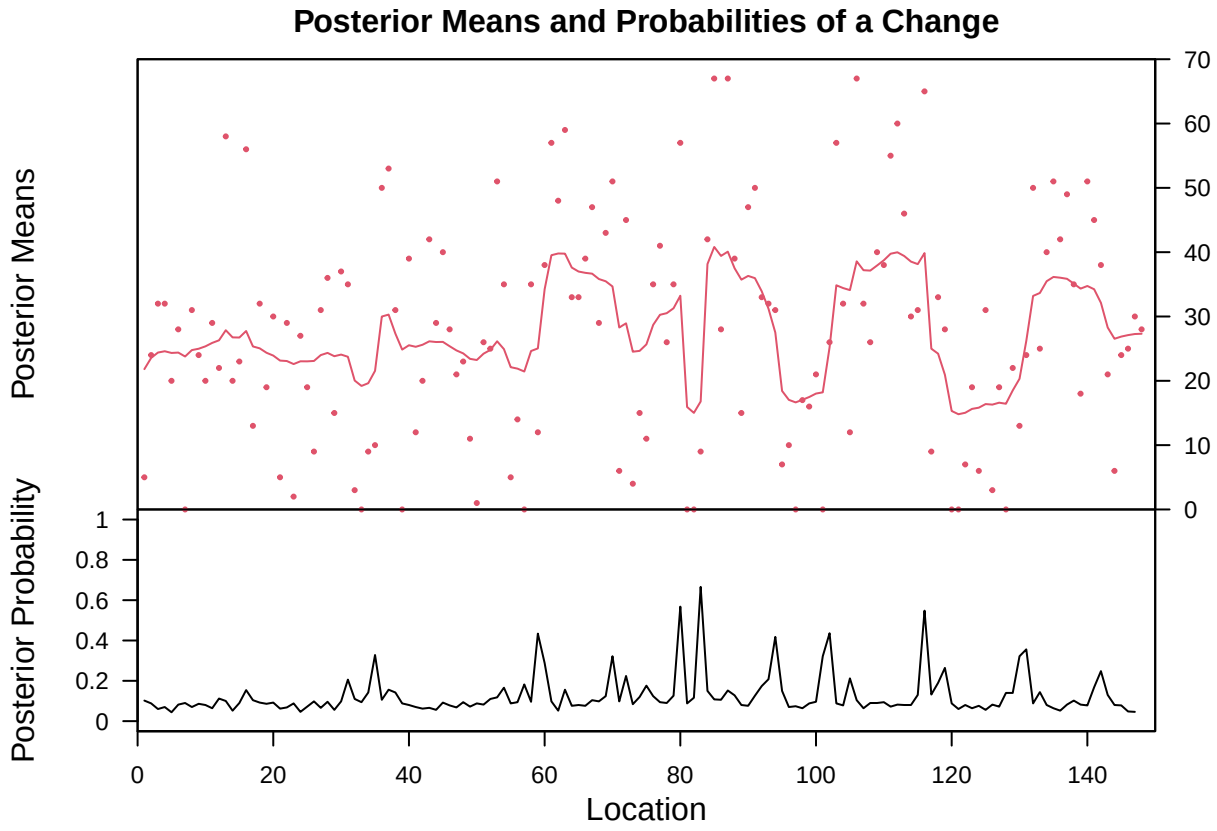
```
##          start end     slope      p-value abs.slope
## x.coef20    20  23  9.444444 0.0017497048  9.444444
## x.coef9      9  12 -8.666667 0.0003129298  8.666667
## x.coef7      7  10 -8.444444 0.0070150348  8.444444
## w.coef9      9  11 -8.250000 0.0552293329  8.250000
## w.coef18    18  20 -7.750000 0.1477436563  7.750000
## w.coef8      8  10 -7.416667 0.1072906610  7.416667
## w.coef12    12  14  7.285714 0.0095862180  7.285714
## w.coef2      2   4 -7.250000 0.2749111246  7.250000
## w.coef21    21  23  7.000000 0.2308312163  7.000000
## z.coef7      7  12 -6.100000 0.0009744568  6.100000
```

```r
# Identify which slopes are significant (p-value , 0.05)
sig.m <- top10.m[which(top10.m$`p-value` < 0.05),]
sig.m
```

```
##          start end     slope      p-value abs.slope
## x.coef20    20  23  9.444444 0.0017497048  9.444444
## x.coef9      9  12 -8.666667 0.0003129298  8.666667
## x.coef7      7  10 -8.444444 0.0070150348  8.444444
## w.coef12    12  14  7.285714 0.0095862180  7.285714
## z.coef7      7  12 -6.100000 0.0009744568  6.100000
```

Below, the hours at which there is a high posterior probability of change in call frequency (i.e. changepoints) are identified using Bayesian change point analysis via the bcp() function. The hours identified are 7, 19, and 22.

```
# Plot posterior probability of change at hours
x <- calls_by_hour_day$Freq
bcp_x <- bcp(x, return.mcmc = TRUE)
plot(bcp_x)
```




```
# Identify posterior probabilities above 0.25, 0.5, and 0.75
PostProbFunc <- function(data){
x <- data$Freq
bcp_x <- bcp(x, return.mcmc = TRUE)
bcp_sum <- as.data.frame(summary(bcp_x))

bcp_sum$id <- 1:length(x)
sel <- bcp_sum[which(bcp_x$posterior.prob > 0.25), ]
loc <- time(x)[sel$id]
prob25 <- cbind(".25", loc)
prob25 <- data.frame(prob25)
colnames(prob25) <- c("threshold", "loc")

bcp_sum$id <- 1:length(x)
(sel <- bcp_sum[which(bcp_x$posterior.prob > 0.5), ])
loc <- time(x)[sel$id]
prob50 <- cbind("0.5", loc)
```

```r
prob50 <- data.frame(prob50)
colnames(prob50) <- c("threshold", "loc")

rbind(prob25, prob50)

}

PostProbFunc(calls_by_hour_day)
```

```
##
## Bayesian Change Point (bcp) summary:
##
##
## Probability of a change in mean and posterior means:
##
##     Probability    X1
## 1         0.134 21.50
## 2         0.060 23.82
## 3         0.062 24.38
## 4         0.050 24.40
## 5         0.054 24.34
## 6         0.072 24.34
## 7         0.078 23.98
## 8         0.064 24.68
## 9         0.072 24.90
## 10        0.090 25.24
## 11        0.094 25.86
## 12        0.090 26.35
## 13        0.088 27.63
## 14        0.070 26.63
## 15        0.056 26.51
## 16        0.124 27.13
## 17        0.100 25.48
## 18        0.084 25.16
## 19        0.076 24.52
## 20        0.112 24.19
## 21        0.102 23.18
## 22        0.068 23.10
## 23        0.072 22.74
## 24        0.056 23.04
## 25        0.084 23.05
## 26        0.100 23.03
## 27        0.082 23.81
## 28        0.082 24.18
## 29        0.062 23.81
## 30        0.084 24.03
## 31        0.172 23.52
## 32        0.086 20.60
## 33        0.078 19.81
## 34        0.118 20.17
## 35        0.356 21.40
## 36        0.090 30.35
## 37        0.170 30.68
## 38        0.162 27.43
```

```
## 39        0.100 24.47
## 40        0.050 25.60
## 41        0.076 25.32
## 42        0.092 25.67
## 43        0.064 26.31
## 44        0.062 26.17
## 45        0.100 26.18
## 46        0.098 25.48
## 47        0.078 24.72
## 48        0.088 24.26
## 49        0.068 23.47
## 50        0.098 23.18
## 51        0.090 24.26
## 52        0.096 25.11
## 53        0.098 26.38
## 54        0.188 25.51
## 55        0.104 22.26
## 56        0.104 21.94
## 57        0.216 21.33
## 58        0.126 25.27
## 59        0.360 26.03
## 60        0.284 33.48
## 61        0.090 38.77
## 62        0.074 39.13
## 63        0.114 39.02
## 64        0.094 37.43
## 65        0.066 36.82
## 66        0.078 36.70
## 67        0.100 36.37
## 68        0.082 35.39
## 69        0.072 35.07
## 70        0.332 34.65
## 71        0.104 28.21
## 72        0.200 28.61
## 73        0.094 24.97
## 74        0.078 25.13
## 75        0.198 25.52
## 76        0.130 28.96
## 77        0.072 30.35
## 78        0.086 30.53
## 79        0.126 31.42
## 80        0.562 33.39
## 81        0.098 16.28
## 82        0.176 15.64
## 83        0.556 19.03
## 84        0.188 36.68
## 85        0.094 40.08
## 86        0.080 38.86
## 87        0.130 39.47
## 88        0.122 37.39
## 89        0.076 35.94
## 90        0.052 36.29
## 91        0.130 36.14
## 92        0.158 34.36
```

```
## 93        0.248 31.84
## 94        0.428 27.34
## 95        0.100 18.03
## 96        0.092 16.97
## 97        0.080 16.38
## 98        0.074 17.01
## 99        0.096 17.48
## 100       0.092 18.12
## 101       0.358 18.32
## 102       0.372 26.04
## 103       0.100 34.58
## 104       0.086 34.00
## 105       0.232 33.60
## 106       0.094 38.28
## 107       0.056 37.22
## 108       0.086 37.26
## 109       0.088 38.13
## 110       0.108 38.80
## 111       0.080 39.85
## 112       0.092 40.22
## 113       0.104 39.65
## 114       0.072 38.36
## 115       0.118 38.25
## 116       0.612 40.26
## 117       0.122 23.76
## 118       0.138 22.74
## 119       0.238 20.58
## 120       0.092 15.60
## 121       0.074 14.99
## 122       0.092 15.38
## 123       0.074 15.95
## 124       0.064 16.03
## 125       0.074 16.45
## 126       0.068 16.17
## 127       0.050 16.40
## 128       0.136 16.49
## 129       0.126 18.32
## 130       0.348 19.83
## 131       0.344 26.47
## 132       0.088 33.59
## 133       0.110 34.10
## 134       0.068 35.44
## 135       0.072 36.05
## 136       0.084 35.96
## 137       0.094 35.82
## 138       0.098 34.87
## 139       0.092 33.93
## 140       0.088 34.66
## 141       0.174 33.97
## 142       0.232 31.93
## 143       0.122 28.19
## 144       0.068 26.76
## 145       0.082 27.09
## 146       0.046 27.41
```

```
## 147       0.058 27.54
## 148          NA 27.55
```

```
##    threshold loc
## 1        .25  35
## 2        .25  59
## 3        .25  60
## 4        .25  70
## 5        .25  80
## 6        .25  83
## 7        .25  94
## 8        .25 101
## 9        .25 102
## 10       .25 116
## 11       .25 130
## 12       .25 131
## 13       0.5  80
## 14       0.5  83
## 15       0.5 116
```

```
# Derive hour values that correspond with "loc" (location) values
calls_by_hour_day[c(80,83,116),]
```

```
##     hour       day Freq day_night sqrt_Freq
## 80    19 5/13/2021   57     night  7.549834
## 83    22 5/13/2021    9     night  3.000000
## 116    7 5/15/2021   65       day  8.062258
```

Visualization and Communication of Results

Figure 1 plots the call frequencies per hour averaged across all trial days (5/10/21-5/16/21). Grey areas indicate calls occuring during nightime. Light areas indicate calls occurring during daytime.

```
AvgCallF_daynight <- df.summary %>%
  ggplot(aes(x = hour, y = Freq)) +
  geom_rect(aes(xmin = -Inf, xmax = 6, ymin = -Inf, ymax = Inf),
            fill = "lightgray", alpha = 0.4) +
  geom_rect(aes(xmin = 19, xmax = Inf, ymin = -Inf, ymax = Inf),
            fill = "lightgray", alpha = 0.4) +
  geom_line(size = 1.2, color = "black") +
  geom_point(size = 3, shape = 21, fill = "black") +
  geom_errorbar(aes(ymin = Freq-sd, ymax = Freq+sd),
            width = 0.2, color = "black") +
  theme_bw() +
  theme(
    plot.title = element_text(hjust = 0.5, face="bold", size=14, color="black"),
    axis.title.x = element_text(face="bold", size=12, color="black"),
    axis.title.y = element_text(face="bold", size=12, color="black"),
    panel.grid.major = element_line(linetype = "solid", color = "grey"),
    panel.grid.minor = element_blank(),
    panel.border = element_blank(),
    panel.background = element_blank(),
    plot.margin = margin(20, 20, 20, 20)) +
```

```
  labs(title = "Average Calls Per Hour", y = "Average Number of Calls", x = "Hour")
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
AvgCallF_daynight
```

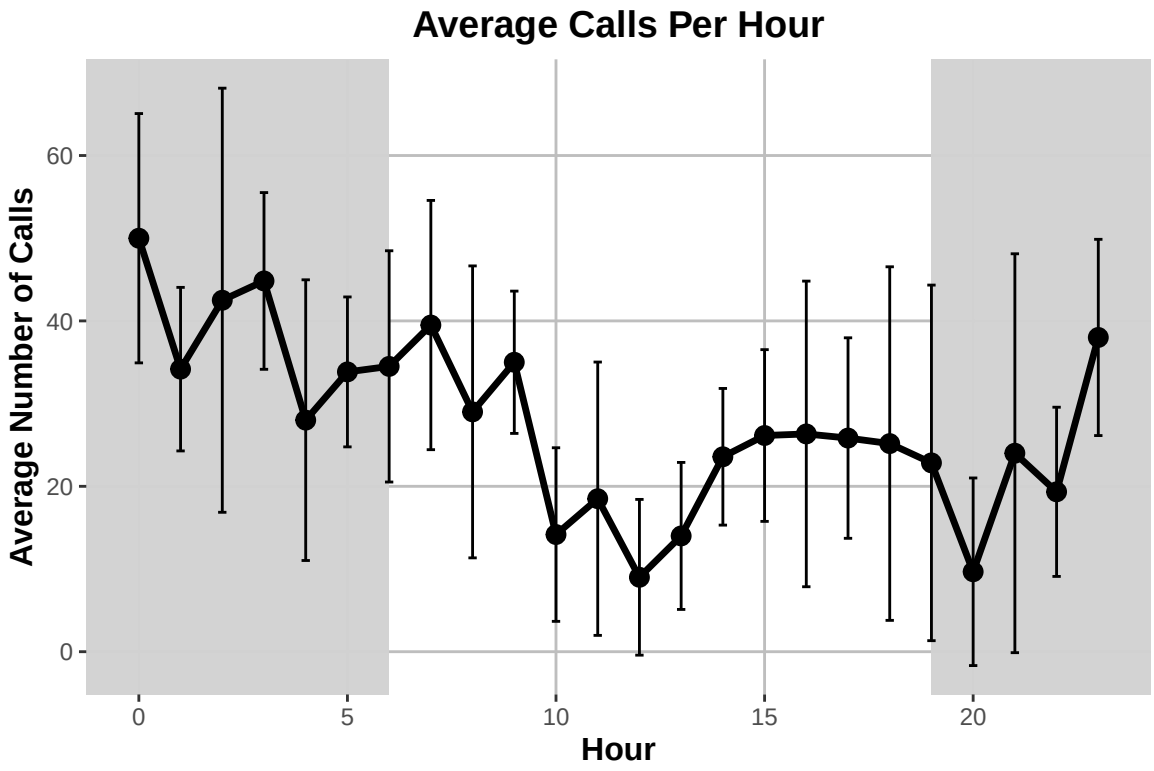


Figure 2 plots the call frequencies per hour averaged across all trial days (5/10/21-5/16/21). Grey areas indicate calls occuring during nightime. Light areas indicate calls occurring during daytime. Hour intervals during Which the largest magnitude change in call frequency occurrs are indicated in purple (hours 20-23), green (hours 9-12), orange (hours 7-10), blue (hours 12-14), and pink (hours 7-12). Red dots indicate the changepoints derived via the bcp() function (hours 7, 19, and 22).

```
segment_strtend <- data.frame(x1 = 20, x2 = 23, x3 = 9, x4 = 12, x5 = 7, x6 = 10, x7 = 12, x8 = 14, x9 =

AvgCallsPerHour_chngpt <- ggplot(df.summary, aes(x = hour, y = Freq)) +
    geom_rect(aes(xmin = -Inf, xmax = 6, ymin = -Inf, ymax = Inf),
        fill = "grey", alpha = 0.05) +
    geom_rect(aes(xmin = 19, xmax = Inf, ymin = -Inf, ymax = Inf),
        fill = "grey", alpha = 0.05) +
    geom_line(size = 1.2, color = "black") +
    geom_point(size = 3, shape = 21, fill = "black") +
```

```
    geom_errorbar(aes(ymin = Freq-sd, ymax = Freq+sd), width = 0.4) +
    labs(title = "Average Calls Per Hour", y = "Average Number of Calls",
        x = "Hour") +
    # manually-derived changepoint hour ranges defined
    # & color-coded using geom_segment()
    geom_segment(aes(x = x1, y = y1, xend = x2, yend = y2),
        colour = "purple", linewidth = 1.5, data = segment_strtend) +
    geom_segment(aes(x = x3, y = y3, xend = x4, yend = y4),
        colour = "darkgreen", linewidth = 1.5, data = segment_strtend) +
    geom_segment(aes(x = x5, y = y5, xend = x6, yend = y6),
        colour = "orange", linewidth = 1.5, data = segment_strtend) +
    geom_segment(aes(x = x7, y = y7, xend = x8, yend = y8),
        colour = "blue", linewidth = 1.5, data = segment_strtend) +
    geom_segment(aes(x = x9, y = y9, xend = x10, yend = y10),
        colour = "hotpink", linewidth = 1.5, data = segment_strtend) +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5,
            face="bold", size=14, color="black"),
          axis.title.x = element_text(face="bold", size=12, color="black"),
          axis.title.y = element_text(face="bold", size=12, color="black"),
          panel.grid.major = element_line(linetype = "solid",
            color = "darkgrey", linewidth = .01),
          panel.grid.minor = element_blank(),
          panel.border = element_blank(),
          panel.background = element_blank(),
          plot.margin = margin(10, 10, 10, 10)) +
   geom_point(data = df.summary[c(20,23,8),], aes(x = hour, y = Freq),
        colour="red", size = 5) # bcp()-derived changepoints in red

AvgCallsPerHour_chngpt
```

Figure 3 plots the call frequencies per hour with each day (5/10/21-5/16/21) of data collection plotted seperately.

```
CallsPerHour_alldays <-
    ggplot(data = calls_by_hour_day, mapping= aes(x = hour, y = Freq)) +
    geom_rect(aes(xmin = -Inf, xmax = 6, ymin = -Inf, ymax = Inf),
        fill = "grey", alpha = 0.05) +
    geom_rect(aes(xmin = 19, xmax = Inf, ymin = -Inf, ymax = Inf),
        fill = "grey", alpha = 0.05) +
    geom_line(aes(color = day), linetype= "longdash", alpha = .4, linewidth = .8) +
    scale_color_brewer(palette = "Dark2") +
    geom_point(data = average_calls_per_hour, aes(x = hour, y = Freq),
            size = 3, shape = 21, fill = "black") +
    geom_line(data = average_calls_per_hour, aes(x=hour, y=Freq,
            color = "average"), linewidth = 1, color = "black") +
    labs(title = "Calls Per Hour", y = "Number of Calls", x = "Hour") +
    theme(legend.position = "bottom") +
    theme_bw() +
    theme(plot.title = element_text(hjust = 0.5,
            face="bold", size=14, color="black"),
            axis.title.x = element_text(face="bold", size=12, color="black"),
            axis.title.y = element_text(face="bold", size=12, color="black"),
```

```
            panel.grid.major = element_line(linetype = "solid",
            color = "darkgrey", linewidth = .01),
            panel.grid.minor = element_blank(),
            panel.border = element_blank(),
            panel.background = element_blank(),
            plot.margin = margin(10, 10, 10, 10))

CallsPerHour_alldays
```