36-303 NHANES

Team 6: Divya Rao, Sage Betko, Kieran Ireland, Baekjoong Kang, Jamie Kim

2022-04-13

Load and merge data

```
# Merge data
demographics <- haven::read_xpt("DEMO_J.XPT")
sleep <- haven::read_xpt("SLQ_J.XPT")
merged <- merge(demographics, sleep, by="SEQN")

# Sleep questionnaire only given to age 16 and older
merged$over16 <- (merged$RIDAGEYR >= 16)
```

Rename sleep columns to be more descriptive

```
# Rename sleep columns to be more descriptive
sleep.cols.mapping <- matrix(c(
  "SLQ300", "sleep.time.weekday", "Usual sleep time on weekdays or workdays",
  "SLQ310", "wake.time.weekday", "Usual wake time on weekdays or workdays",
  "SLD012", "sleep.hours.weekday", "Sleep hours - weekdays or workdays",
  "SLQ320", "sleep.time.weekend", "Usual sleep time on weekends",
  "SLQ330", "wake.time.weekend", "Usual wake time on weekends",
  "SLD013", "sleep.hours.weekend", "Sleep hours - weekends",
  "SLQ030", "frequency.snore", "How often do you snore?",
  "SLQ040", "frequency.stop.breathing", "How often do you snort or stop breathing?",
  "SLQ050", "told.doctor.sleep.trouble", "Ever told doctor had trouble sleeping?",
  "SLQ120", "frequency.daytime.fatigue", "How often feel overly sleepy during day?"),
  ncol=3, byrow=T
)

sleep.cols.original <- sleep.cols.mapping[,1]
sleep.cols.renamed <- sleep.cols.mapping[,2]
sleep.cols.description <- sleep.cols.mapping[,3]

setnames(merged, old = sleep.cols.original, new = sleep.cols.renamed)
```

Look at unique values for all the sleep variables

```
# Look at unique values for all the sleep variables
for (sleep.col in sleep.cols.renamed) {
  print(sleep.col)
  print(sort(unique(merged[,sleep.col])))
}
```

```
## [1] "sleep.time.weekday"
```

```
##  [1] ""      "00:00" "00:15" "00:30" "01:00" "01:30" "02:00" "02:30" "03:00"
## [10] "03:30" "03:45" "04:00" "04:30" "05:00" "05:30" "06:00" "06:30" "07:00"
## [19] "07:30" "08:00" "08:30" "09:00" "09:30" "09:45" "10:00" "10:30" "11:00"
## [28] "12:00" "13:00" "13:30" "14:00" "15:00" "16:00" "17:00" "18:00" "18:30"
## [37] "19:00" "19:30" "20:00" "20:30" "20:45" "21:00" "21:10" "21:15" "21:30"
## [46] "21:40" "21:45" "21:50" "22:00" "22:10" "22:15" "22:30" "22:45" "23:00"
## [55] "23:15" "23:30" "23:40" "23:45" "77777" "99999"
## [1] "wake.time.weekday"
##   [1] ""      "00:00" "01:00" "01:30" "02:00" "02:30" "02:40" "02:50" "03:00"
##  [10] "03:05" "03:10" "03:15" "03:20" "03:30" "03:40" "03:45" "03:50" "04:00"
##  [19] "04:05" "04:10" "04:15" "04:20" "04:25" "04:30" "04:40" "04:45" "04:50"
##  [28] "05:00" "05:05" "05:10" "05:15" "05:20" "05:25" "05:30" "05:35" "05:40"
##  [37] "05:45" "05:50" "05:55" "06:00" "06:05" "06:10" "06:15" "06:20" "06:30"
##  [46] "06:35" "06:40" "06:45" "06:50" "06:55" "07:00" "07:05" "07:10" "07:15"
##  [55] "07:20" "07:30" "07:40" "07:45" "07:50" "07:55" "08:00" "08:15" "08:30"
##  [64] "08:40" "08:45" "09:00" "09:10" "09:30" "09:45" "10:00" "10:30" "10:50"
##  [73] "11:00" "11:30" "12:00" "12:30" "13:00" "13:30" "14:00" "14:15" "14:30"
##  [82] "14:35" "14:50" "15:00" "15:15" "15:30" "16:00" "16:30" "17:00" "17:30"
##  [91] "18:00" "19:00" "20:00" "20:30" "21:00" "21:30" "21:45" "21:50" "22:00"
## [100] "22:30" "23:00" "23:30" "77777" "99999"
## [1] "sleep.hours.weekday"
##  [1]  2.0  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0  8.5  9.0  9.5
## [16] 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5 14.0
## [1] "sleep.time.weekend"
##  [1] ""      "00:00" "00:15" "00:20" "00:30" "00:40" "01:00" "01:30" "02:00"
## [10] "02:30" "02:45" "03:00" "03:30" "04:00" "04:30" "05:00" "05:30" "06:00"
## [19] "06:30" "07:00" "07:30" "08:00" "08:30" "09:00" "10:00" "11:00" "12:00"
## [28] "13:30" "14:00" "15:00" "16:00" "17:00" "17:30" "18:00" "18:30" "19:00"
## [37] "19:30" "20:00" "20:10" "20:30" "20:45" "21:00" "21:10" "21:15" "21:30"
## [46] "21:40" "21:45" "22:00" "22:10" "22:15" "22:30" "22:45" "23:00" "23:15"
## [55] "23:20" "23:30" "23:35" "23:45" "23:50" "77777" "99999"
## [1] "wake.time.weekend"
##  [1] ""      "00:00" "01:00" "01:10" "01:30" "02:00" "02:30" "03:00" "03:05"
## [10] "03:30" "03:50" "04:00" "04:15" "04:20" "04:30" "04:45" "05:00" "05:15"
## [19] "05:20" "05:30" "05:35" "05:40" "05:45" "05:50" "06:00" "06:15" "06:30"
## [28] "06:45" "06:50" "07:00" "07:15" "07:30" "07:40" "07:45" "08:00" "08:05"
## [37] "08:15" "08:30" "08:45" "09:00" "09:15" "09:30" "09:45" "10:00" "10:30"
## [46] "11:00" "11:30" "12:00" "12:30" "13:00" "13:30" "14:00" "14:30" "14:50"
## [55] "15:00" "15:30" "16:00" "16:30" "17:00" "17:30" "18:00" "18:30" "20:00"
## [64] "20:30" "21:00" "23:00" "77777" "99999"
## [1] "sleep.hours.weekend"
##  [1]  2.0  3.0  3.5  4.0  4.5  5.0  5.5  6.0  6.5  7.0  7.5  8.0  8.5  9.0  9.5
## [16] 10.0 10.5 11.0 11.5 12.0 12.5 13.0 13.5 14.0
## [1] "frequency.snore"
## [1] 0 1 2 3 7 9
## [1] "frequency.stop.breathing"
## [1] 0 1 2 3 7 9
## [1] "told.doctor.sleep.trouble"
## [1] 1 2 9
## [1] "frequency.daytime.fatigue"
## [1] 0 1 2 3 4 9
```

Set missing value codes to NA

```
# Set missing value codes to NA
replace_with_na <- function(df, col, old.val) {
  for (v in old.val) {
    to_replace <- (df[, col] == v) | is.na(df[, col])
    df[to_replace, col] <- NA
  }
  return(df)
}

merged <- replace_with_na(merged, "sleep.time.weekday", c("", "77777", "99999"))
merged <- replace_with_na(merged, "wake.time.weekday",  c("", "77777", "99999"))
merged <- replace_with_na(merged, "sleep.time.weekend",  c("", "77777", "99999"))
merged <- replace_with_na(merged, "wake.time.weekend",  c("", "77777", "99999"))
merged <- replace_with_na(merged, "frequency.snore", c(7, 9))
merged <- replace_with_na(merged, "frequency.stop.breathing", c(7, 9))
merged <- replace_with_na(merged, "told.doctor.sleep.trouble", c(7, 9))
merged <- replace_with_na(merged, "frequency.daytime.fatigue", c(7, 9))
```

Create age groups

```
# Create age groups
merged[(merged$RIDAGEYR >= 16) & (merged$RIDAGEYR < 24), "age.group"] <- "16to24"
merged[(merged$RIDAGEYR >= 24) & (merged$RIDAGEYR < 65), "age.group"] <- "24to65"
merged[(merged$RIDAGEYR >= 65), "age.group"] <- "65plus"
```

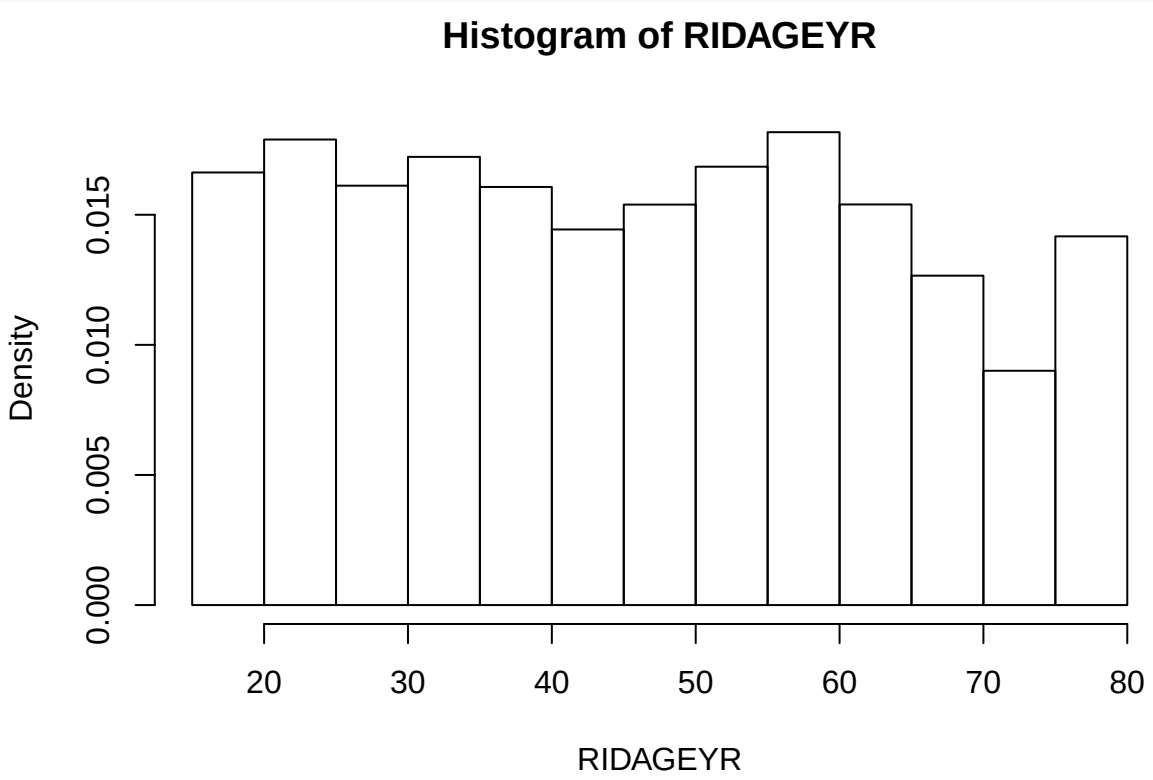
Convert sleep / wake time strings to numeric values

```
# Convert sleep / wake time strings to numeric values
sleep.t0 <- 12
wake.t0  <- 0

time.str.to.numeric.helper <- function(xi) {
  if (any(is.na(xi)) | length(xi) != 2) {
    return(NA)
  } else {
    return(as.numeric(xi[[1]]) + as.numeric(xi[[2]]) / 60)
  }
}

# Converts to hours with decimal precision
# E.g., "18:30" -> "18.5"
time.str.to.numeric <- function(x) {
  hours.minutes <- strsplit(x, ":", fixed=T)
  result <- as.numeric(lapply(hours.minutes, time.str.to.numeric.helper))
  result
}

# Effectively shifts the time to a new timezone where t0 is midnight
offset.clock <- function(x, t0) {
  result <- numeric(length(x))
  time.before.midnight <- (24 - t0) %% 24
```

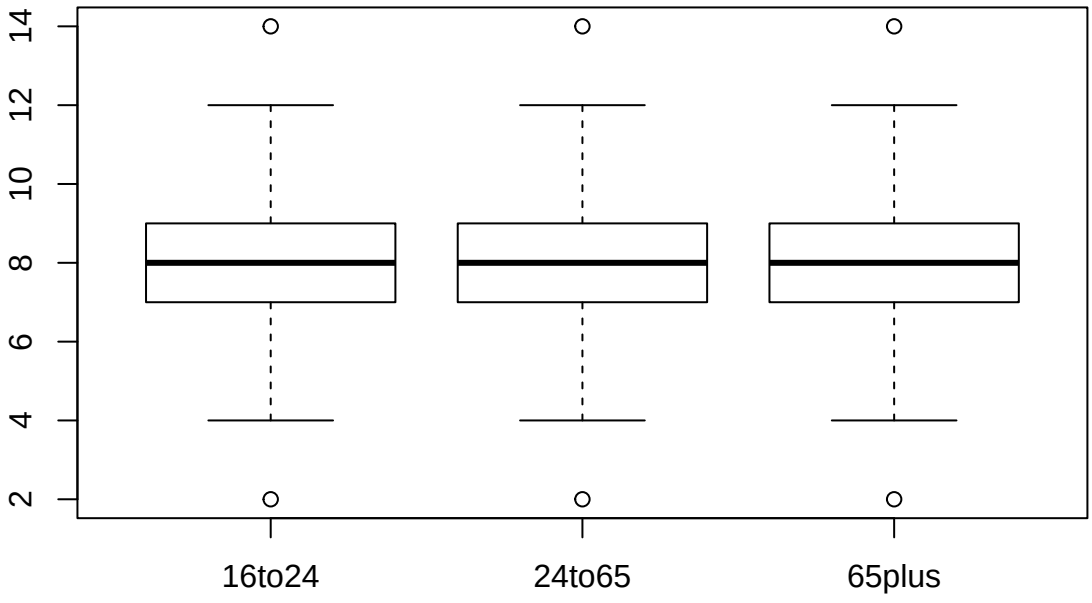
```
  before.midnight <- !is.na(x) & ((x >= t0) & (x < 24))
  after.midnight  <- !is.na(x) & !before.midnight
  result[before.midnight] <- x[before.midnight] - t0
  result[after.midnight] <- x[after.midnight] + time.before.midnight
  return(result)
}

# Convert time strings to numerics, e.g., "19:45" -> 19.75
merged$sleep.time.weekday.num <- time.str.to.numeric(merged$sleep.time.weekday)
merged$sleep.time.weekend.num <- time.str.to.numeric(merged$sleep.time.weekend)

merged$wake.time.weekday.num <- time.str.to.numeric(merged$wake.time.weekday)
merged$wake.time.weekend.num <- time.str.to.numeric(merged$wake.time.weekend)

# Offset sleep and wake times e.g., (sleep.time = 19.75, sleep.t0 = 19) -> 0.75
merged$sleep.time.weekday.offset <- offset.clock(merged$sleep.time.weekday.num, sleep.t0)
merged$sleep.time.weekend.offset <- offset.clock(merged$sleep.time.weekend.num, sleep.t0)

merged$wake.time.weekday.offset <- offset.clock(merged$wake.time.weekday.num, wake.t0)
merged$wake.time.weekend.offset <- offset.clock(merged$wake.time.weekend.num, wake.t0)
```

```
# Create survey design object
design <- svydesign(
  data=merged, id=~SDMVPSU, strata=~SDMVSTRA, weights=~WTMEC2YR, nest=TRUE)
design <- subset(design, over16)
```

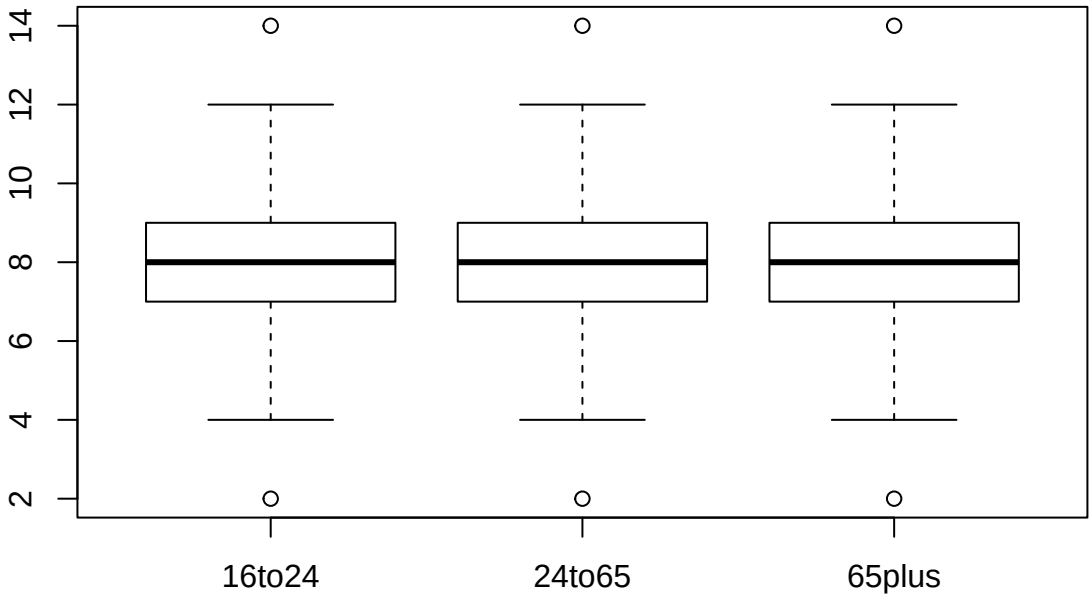
```
svyhist(~RIDAGEYR, design)
```



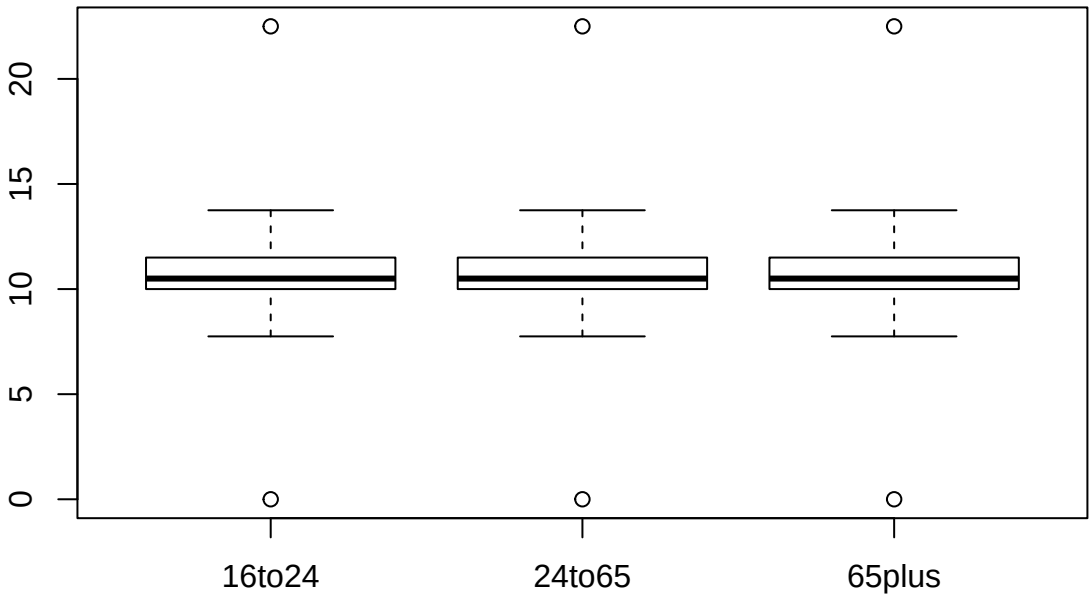

```
svyboxplot(sleep.hours.weekday ~ age.group, design,
           main="Sleep Hours Weekday")
```



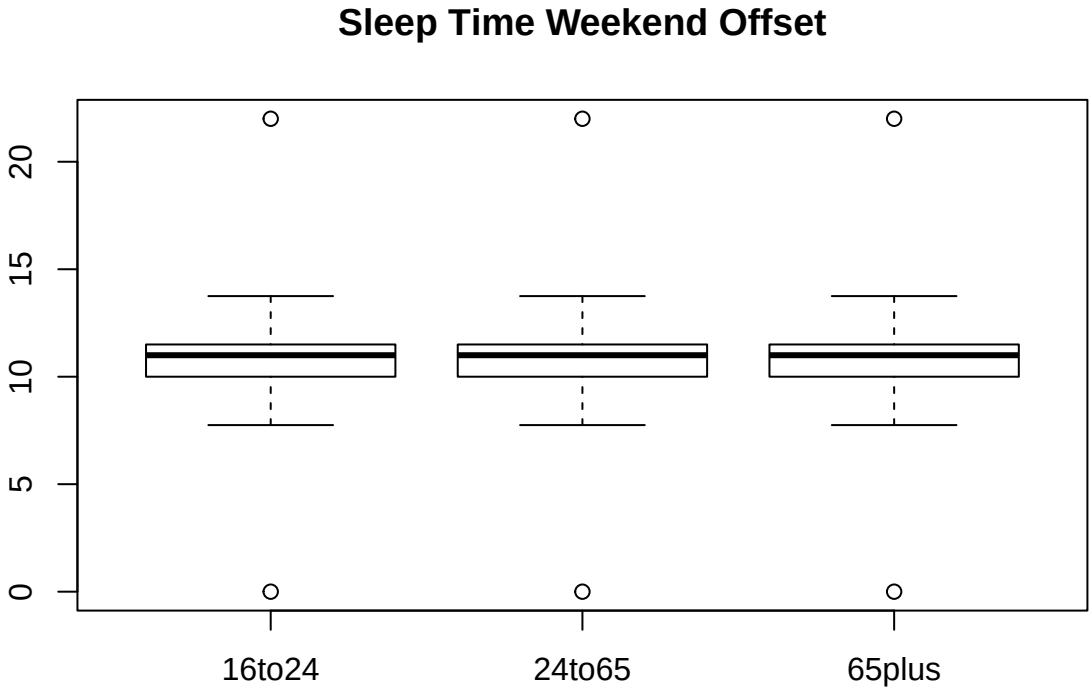

```
svyboxplot(sleep.hours.weekend ~ age.group, design,
           main="Sleep Hours Weekend")
```



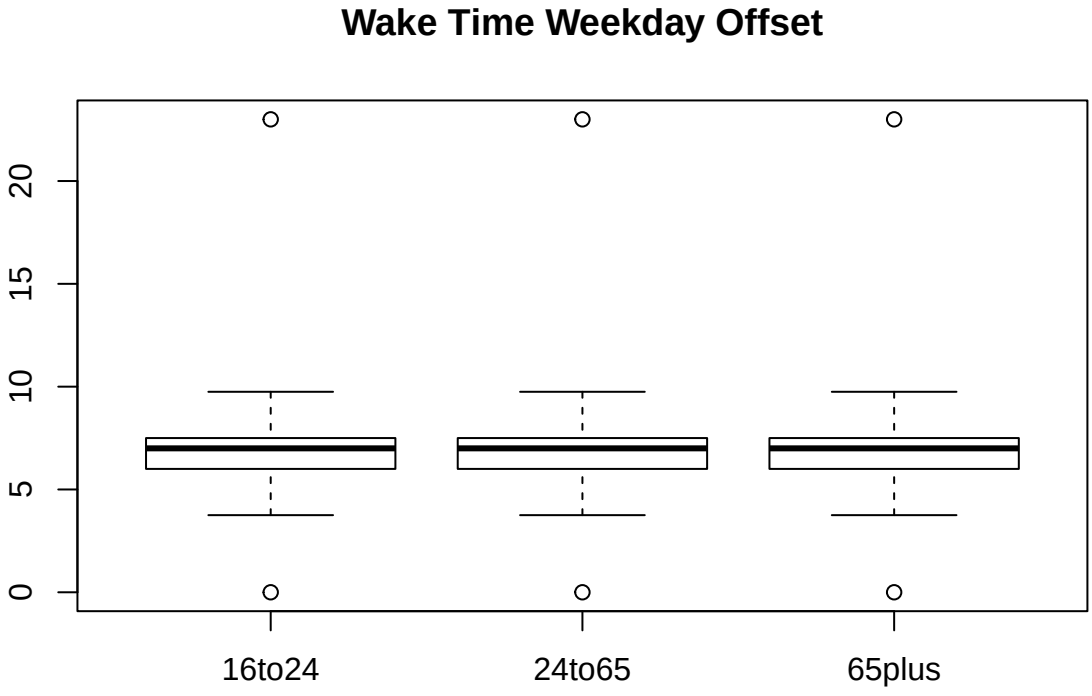

```
svyboxplot(sleep.time.weekday.offset ~ age.group, design,
           main="Sleep Time Weekday Offset")
```



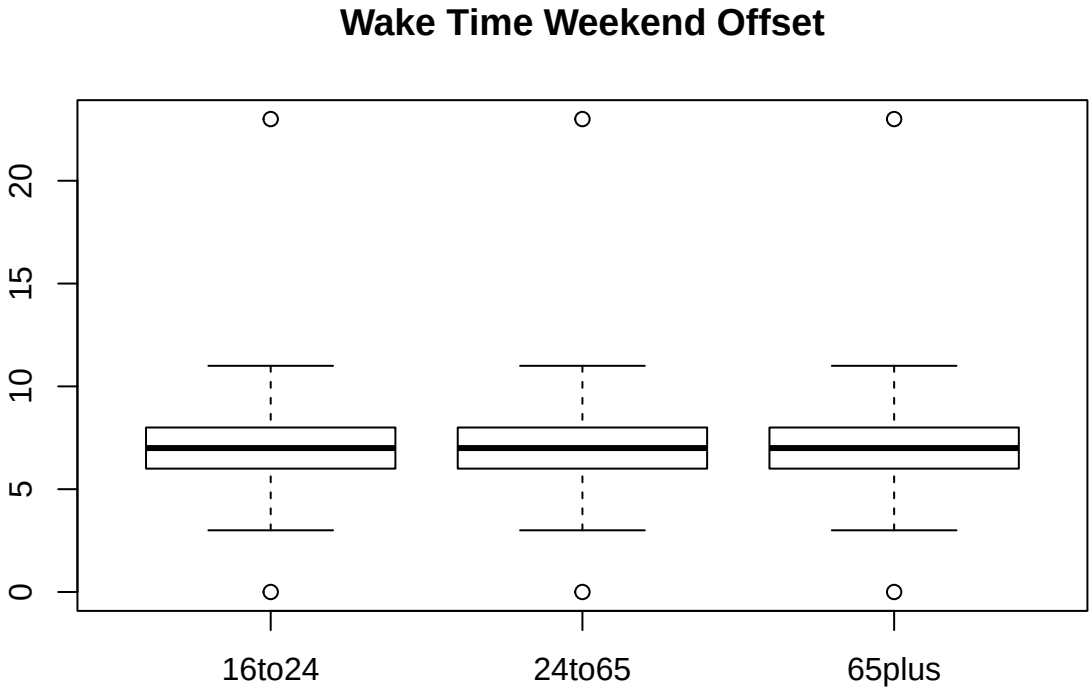

```
svyboxplot(sleep.time.weekend.offset ~ age.group, design,
           main="Sleep Time Weekend Offset")
```




```
svyboxplot(wake.time.weekday.offset ~ age.group, design,
           main="Wake Time Weekday Offset")
```



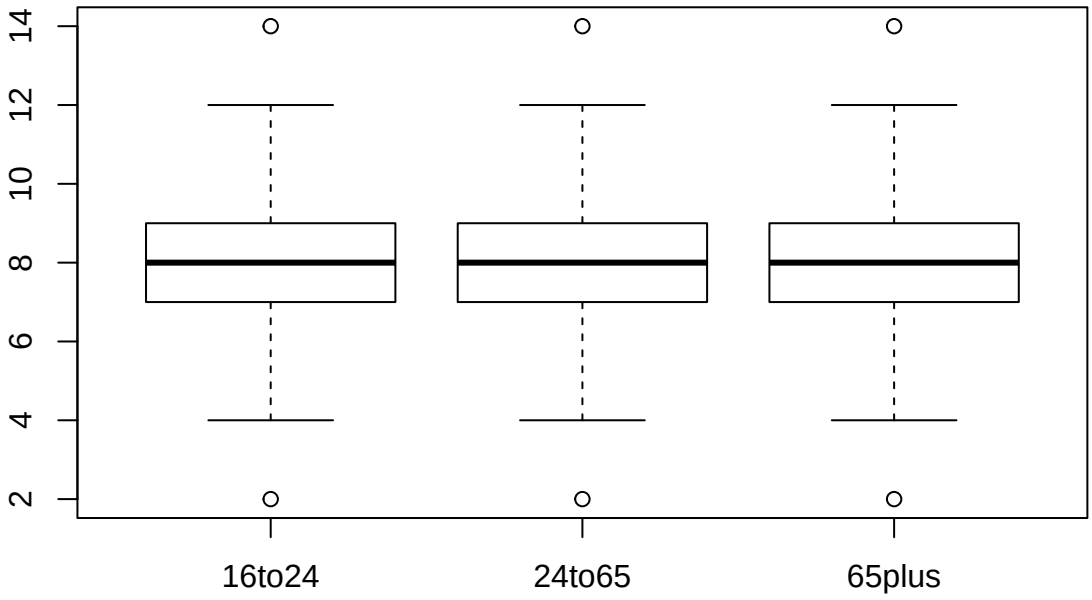

```
svyboxplot(wake.time.weekend.offset ~ age.group, design,
           main="Wake Time Weekend Offset")
```




```
svychisq(~sleep.hours.weekday + age.group, design)
```

```
##
##  Pearson's X^2: Rao & Scott adjustment
##
## data:  svychisq(~sleep.hours.weekday + age.group, design)
## F = 4.4847, ndf = 9.8976, ddf = 148.4640, p-value = 1.671e-05
```

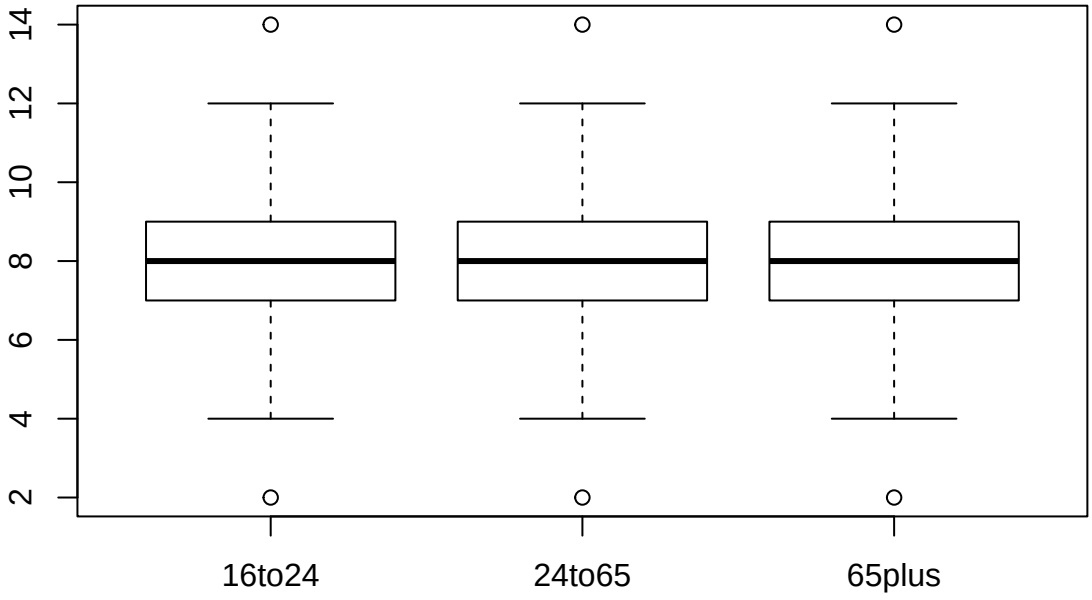
```
svyboxplot(sleep.hours.weekday ~ age.group, design)
```




```
svychisq(~sleep.hours.weekend + age.group, design)
```

```
##
##  Pearson's X^2: Rao & Scott adjustment
##
## data:  svychisq(~sleep.hours.weekend + age.group, design)
## F = 4.076, ndf = 9.3316, ddf = 139.9734, p-value = 9.347e-05
```

```
svyboxplot(sleep.hours.weekend ~ age.group, design)
```




```
summary(svyglm(sleep.hours.weekend ~ RIDAGEYR, design))
```

```
## Warning in summary.glm(g): observations with zero weight not used for
## calculating dispersion
```

```
## Warning in summary.glm(glm.object): observations with zero weight not used for
## calculating dispersion
```

```
##
## Call:
## svyglm(formula = sleep.hours.weekend ~ RIDAGEYR, design = design)
##
## Survey design:
## subset(design, over16)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.00186    0.09174  98.125  < 2e-16 ***
## RIDAGEYR    -0.01443    0.00147  -9.815 1.18e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.649458)
##
## Number of Fisher Scoring iterations: 2
```

```
summary(svyglm(sleep.hours.weekday ~ RIDAGEYR, design))
```

```
## Warning in summary.glm(g): observations with zero weight not used for
## calculating dispersion
```

```
## Warning in summary.glm(glm.object): observations with zero weight not used for
## calculating dispersion
```

```
##
## Call:
## svyglm(formula = sleep.hours.weekday ~ RIDAGEYR, design = design)
##
## Survey design:
## subset(design, over16)
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 7.5994853  0.0749173 101.438   <2e-16 ***
## RIDAGEYR    0.0002688  0.0015669   0.172    0.866
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 2.380138)
##
## Number of Fisher Scoring iterations: 2
```