Supplement 3

Primary analysis: frequency of pain

Peter Kamerman

Last knitted: 27 June 2019

Contents

This script addresses the primary analysis of the study, namely, what is the frequency of pain in the last week in individuals attending an urban HIV voluntary counselling and testing centre, and does the frequency of pain differ between individuals infected by HIV and those who are not.

Inspect data

Data structure

```
dim(data)
```

```
## [1] 540   3
```

```
head(data)
```

```
## # A tibble: 6 x 3
##   PID   pain  HIV_positive
##   <chr> <chr> <chr>
## 1 001   no    no
## 2 003   yes   no
## 3 004   yes   no
## 4 005   yes   no
## 5 006   yes   no
## 6 007   no    no
```

```
tail(data)
```

```
## # A tibble: 6 x 3
##   PID   pain  HIV_positive
##   <chr> <chr> <chr>
## 1 536   no    yes
## 2 537   no    yes
## 3 538   no    no
## 4 539   yes   yes
## 5 540   no    no
## 6 541   no    yes
```

```
glimpse(data)
```

```
## Observations: 540
## Variables: 3
## $ PID          <chr> "001", "003", "004", "005", "006", "007", "008", ...
## $ pain         <chr> "no", "yes", "yes", "yes", "yes", "no", "yes", "y...
## $ HIV_positive <chr> "no", "no", "no", "no", "no", "no", "no", "no", "...
```

```
# HIV status data
dim(data[data$HIV_positive == 'yes' & !is.na(data$HIV_positive), ])
```

```
## [1] 70  3
```

```
dim(data[data$HIV_positive == 'no' & !is.na(data$HIV_positive), ])
```

```
## [1] 465   3
```

```
dim(data[is.na(data$HIV_positive), ])
```

```
## [1] 5 3
```

```
dim(data[!is.na(data$HIV_positive), ])
```

```
## [1] 535   3
```

```
# Remove rows with missing HIV test results (n = 5)
data %<>%
    filter(!is.na(HIV_positive))

# By pain status
dim(data[data$pain == 'yes' & !is.na(data$pain), ])
```

```
## [1] 316   3
```

```
dim(data[data$pain == 'no' & !is.na(data$pain), ])
```

```
## [1] 219   3
```

```
dim(data[is.na(data$pain), ])
```

```
## [1] 0 3
```

```
dim(data[!is.na(data$pain), ])
```

```
## [1] 535   3
```

Data summaries

```
# Convert character to factor
data_fct <- data %>%
    mutate(HIV_positive = factor(HIV_positive),
           pain = factor(pain)) %>%
    select(-PID)

# Full cohort
data_fct %>%
    skim() %>%
    kable(caption = 'Full cohort')
```

```
## Skim summary statistics
##  n obs: 535
##  n variables: 2
##
## Variable type: factor
##
##    variable      missing    complete     n     n_unique          top_counts
## --------------  ---------  ----------  -----  ----------  --------------------------
##  HIV_positive       0          535      535       2        no: 465, yes: 70, NA: 0
##      pain           0          535      535       2        yes: 316, no: 219, NA: 0
```

```
# By HIV status
data_fct %>%
    group_by(pain) %>%
    skim() %>%
    kable(caption = 'By HIV status')
```

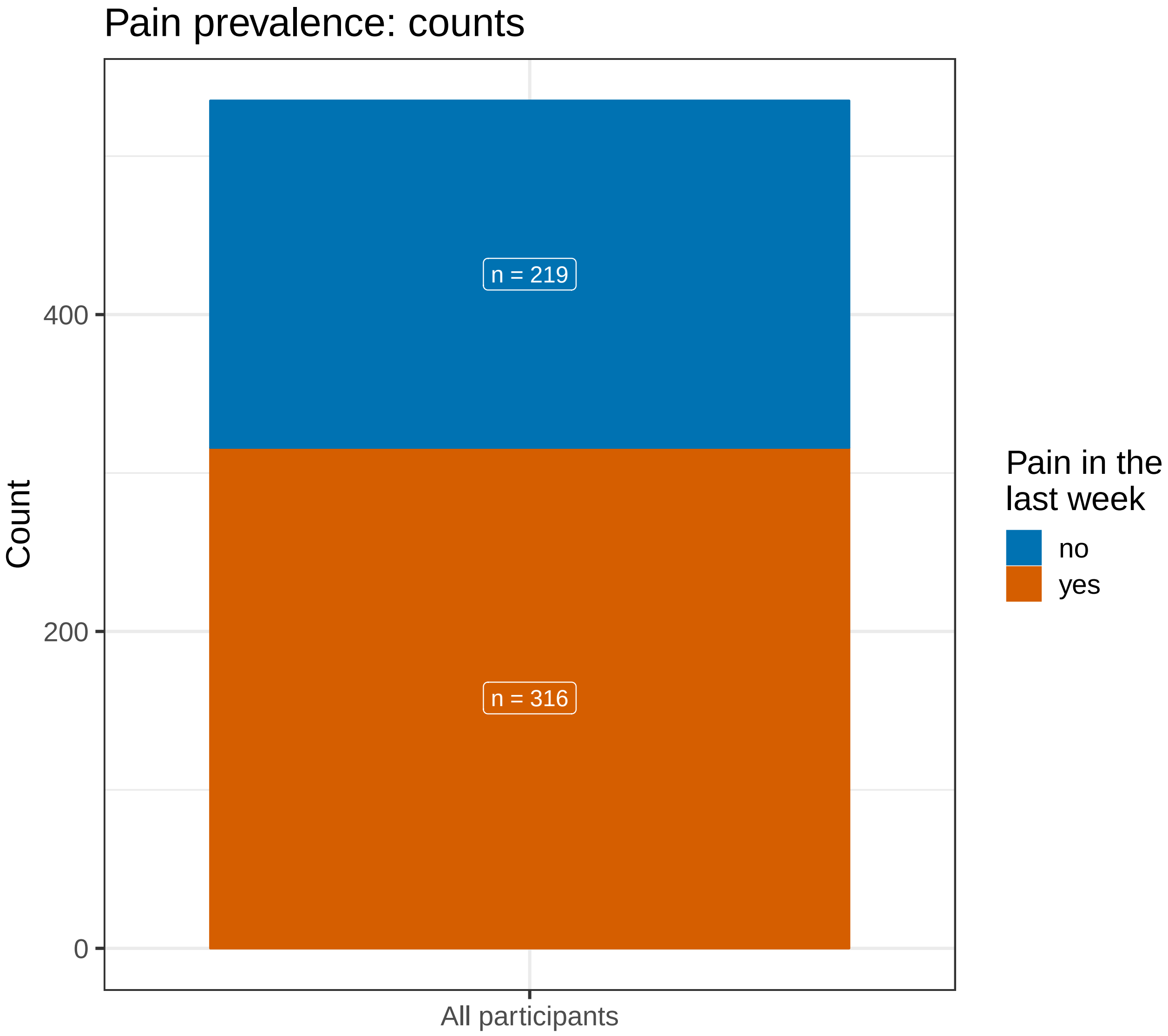
```
## Skim summary statistics
##  n obs: 535
##  n variables: 2
##
## Variable type: factor
##
##  pain      variable      missing    complete     n     n_unique          top_counts
## ------  --------------  ---------  ----------  -----  ----------  --------------------------
##   no     HIV_positive       0          219      219       2        no: 184, yes: 35, NA: 0
##  yes     HIV_positive       0          316      316       2        no: 281, yes: 35, NA: 0
```

Primary questions

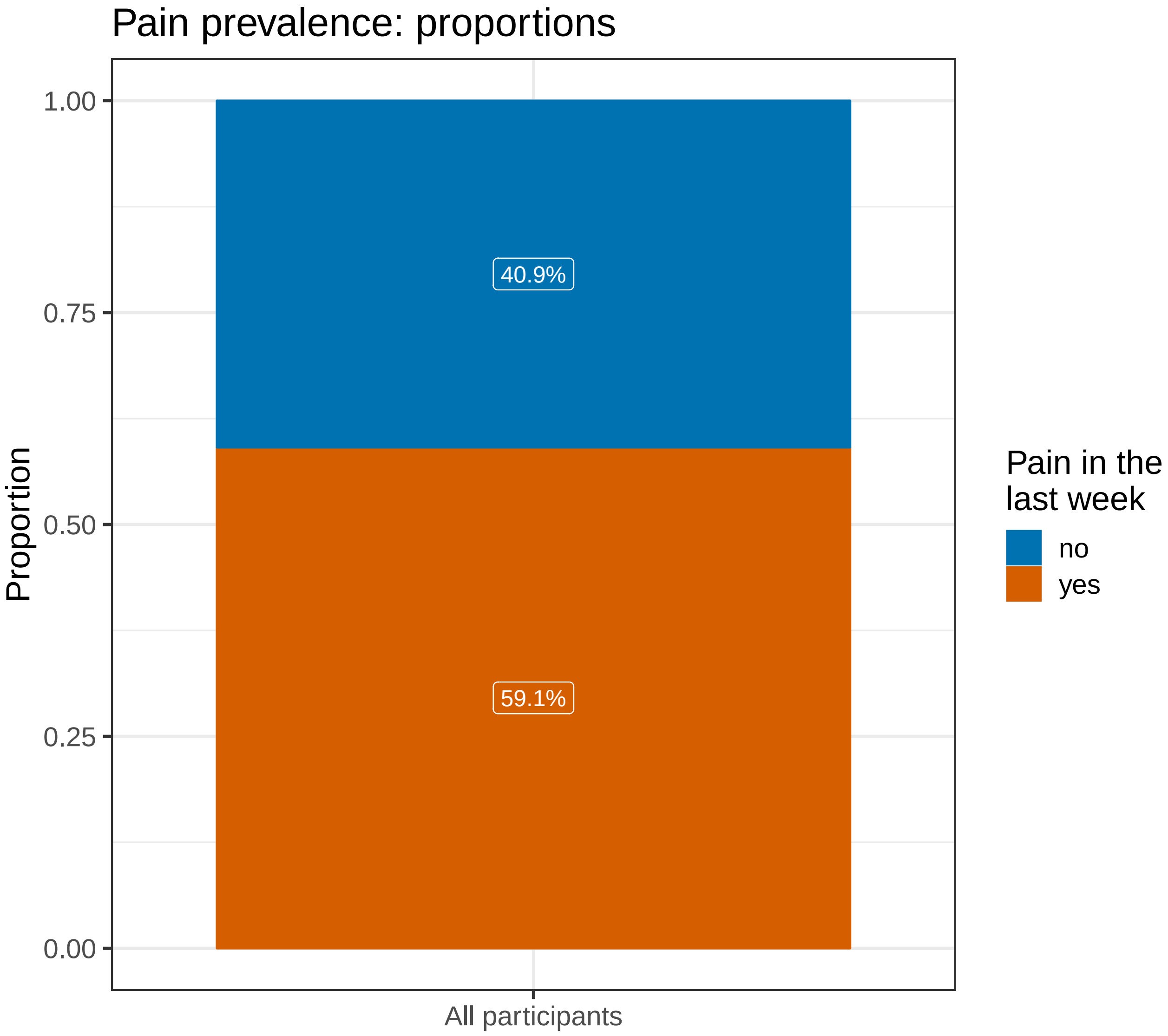
Frequency of pain

Exploratory plots

```
# Prepare data
plot_total <- data %>%
    group_by(pain) %>%
    summarise(count = n()) %>%
    mutate(prop = round(count/sum(count) * 100, 1)) %>%
    ungroup()

# Count plot
ggplot(data = plot_total) +
    aes(x = 'All participants',
        y = count,
        fill = pain,
        colour = pain) +
    geom_col() +
    geom_label(aes(label = paste0('n = ', count)),
               position = position_stack(vjust = 0.5),
               colour = 'white',
               show.legend = FALSE) +
    labs(title = 'Pain prevalence: counts',
         y = 'Count') +
    scale_fill_manual(name = 'Pain in the\nlast week',
                      values = pal) +
    scale_colour_manual(name = 'Pain in the\nlast week',
                        values = pal) +
    theme(axis.title.x = element_blank())
```



```
# Proportion plot
ggplot(data = plot_total) +
    aes(x = 'All participants',
        y = prop,
        fill = pain,
        colour = pain) +
    geom_col(position = position_fill()) +
    geom_label(aes(label = paste0(prop, '%')),
               position = position_fill(vjust = 0.5),
               colour = 'white',
               show.legend = FALSE) +
    labs(title = 'Pain prevalence: proportions',
         y = 'Proportion') +
    scale_fill_manual(name = 'Pain in the\nlast week',
                      values = pal) +
    scale_colour_manual(name = 'Pain in the\nlast week',
                        values = pal) +
    theme(axis.title.x = element_blank())
```

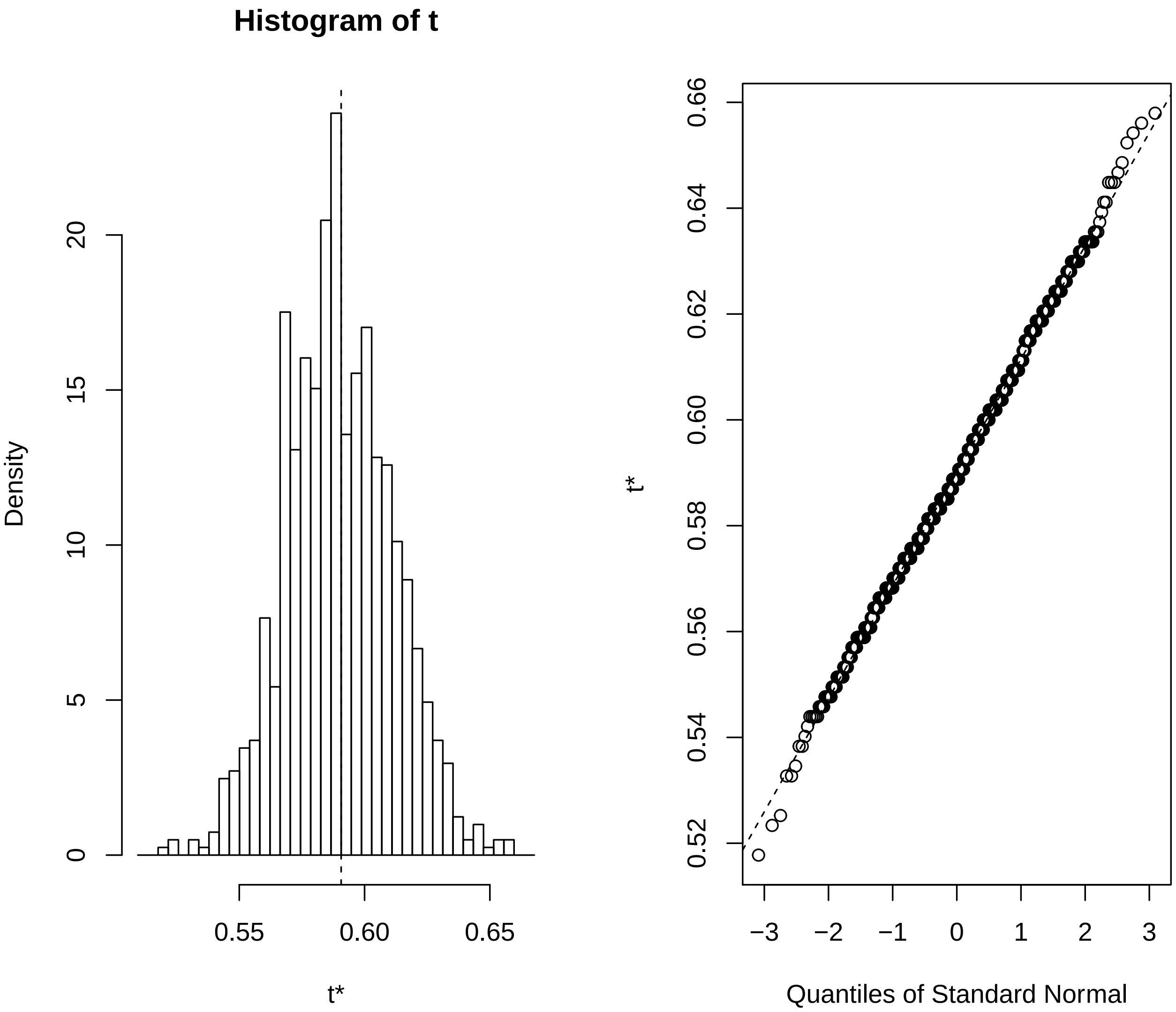


95% CI of pain frequency

```r
# Define boot function
freq <- function(d, i){
    data <- d[i, ]
    mean(data$pain == 'yes')
}

# Generate bootstraps
set.seed(2019)
(boot_all <- boot(data = data,
                  statistic = freq,
                  R = 999,
                  stype = 'i'))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = freq, R = 999, stype = "i")
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 0.5906542 -0.0005556959  0.02138734
```

```r
# Plot the distribution
plot(boot_all)
```



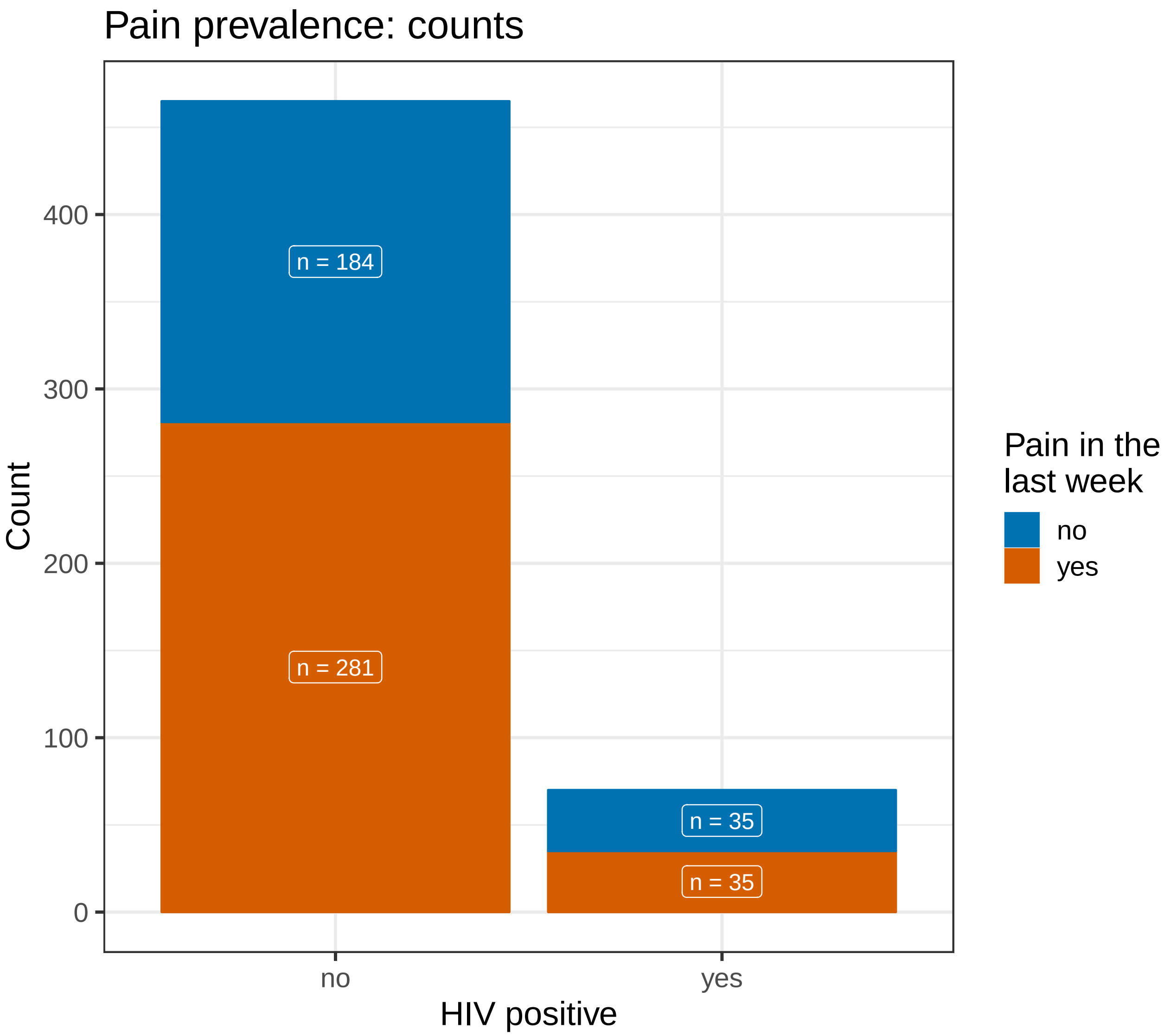
```
# Generate bias accelerated interval
(ci_all <- boot.ci(boot_all,
                   type = 'bca'))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_all, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.5495,  0.6336 )
## Calculations and Intervals on Original Scale
```

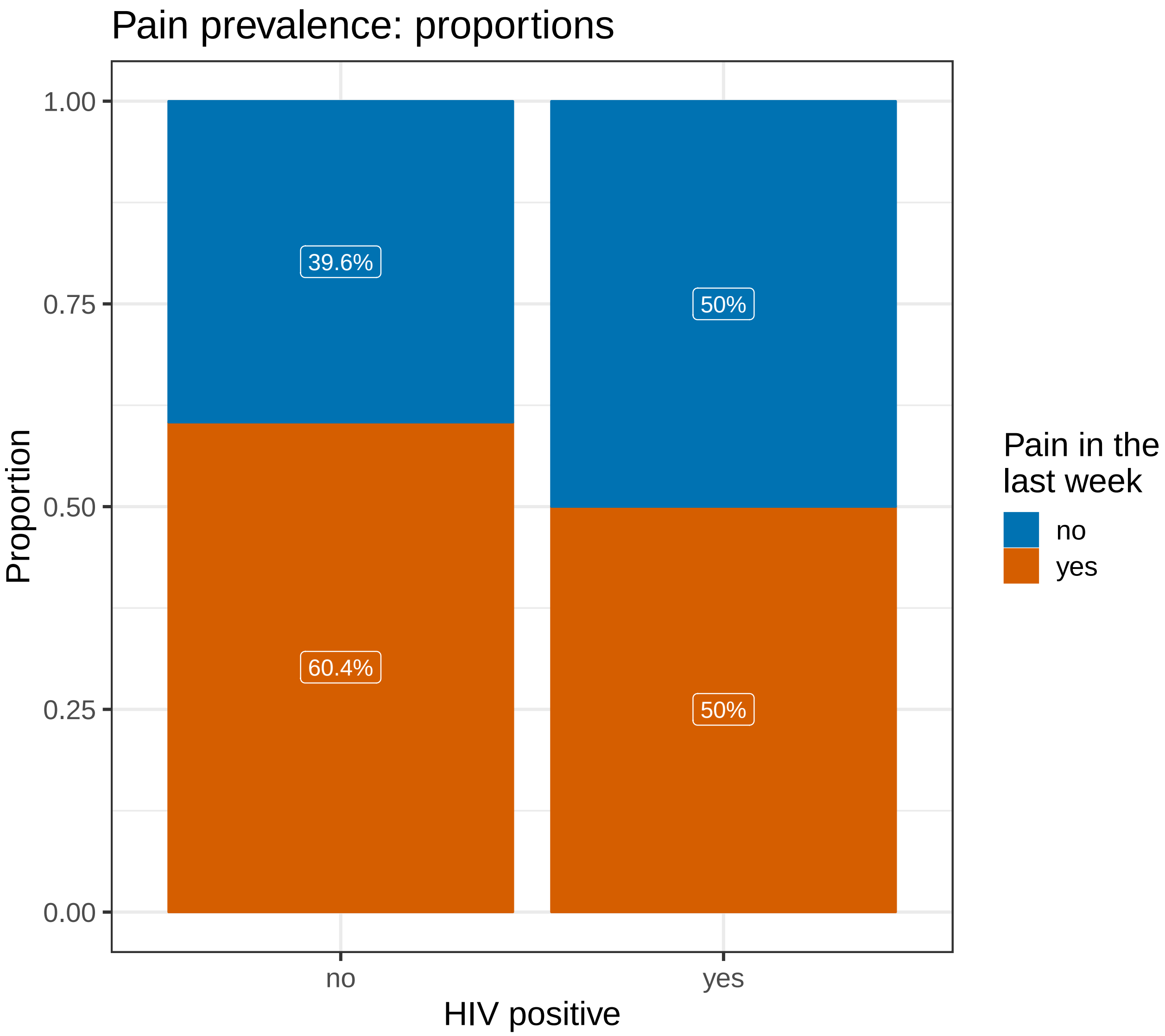
Frequency of pain by HIV status

Exploratory plot

```
# Prepare data
plot_data <- data %>%
    group_by(HIV_positive, pain) %>%
    summarise(count = n()) %>%
    mutate(prop = round(count/sum(count) * 100, 1)) %>%
    ungroup()

# Count plot
ggplot(data = plot_data) +
    aes(x = HIV_positive,
        y = count,
        fill = pain,
        colour = pain) +
    geom_col() +
    geom_label(aes(label = paste0('n = ', count)),
               position = position_stack(vjust = 0.5),
               colour = 'white',
               show.legend = FALSE) +
    labs(title = 'Pain prevalence: counts',
         y = 'Count',
         x = 'HIV positive') +
    scale_fill_manual(name = 'Pain in the\nlast week',
                      values = pal) +
    scale_colour_manual(name = 'Pain in the\nlast week',
                        values = pal)
```



```
# Proportion plot
ggplot(data = plot_data) +
    aes(x = HIV_positive,
        y = prop,
        fill = pain,
        colour = pain) +
    geom_col(position = position_fill()) +
    geom_label(aes(label = paste0(prop, '%')),
               position = position_fill(vjust = 0.5),
               colour = 'white',
               show.legend = FALSE) +
    labs(title = 'Pain prevalence: proportions',
         y = 'Proportion',
         x = 'HIV positive') +
    scale_fill_manual(name = 'Pain in the\nlast week',
                      values = pal) +
    scale_colour_manual(name = 'Pain in the\nlast week',
                        values = pal)
```



95% CI of pain frequency

```
# Define boot function
## HIV+
freq_hiv <- function(d, i){
    data <- d[i, ]
    hiv <- data[data$HIV_positive == 'yes', ]
    pain <- mean(hiv$pain == 'yes')
    pain
}
## HIV-
freq_nohiv <- function(d, i){
    data <- d[i, ]
    hiv <- data[data$HIV_positive == 'no', ]
    pain <- mean(hiv$pain == 'yes')
    pain
}
```

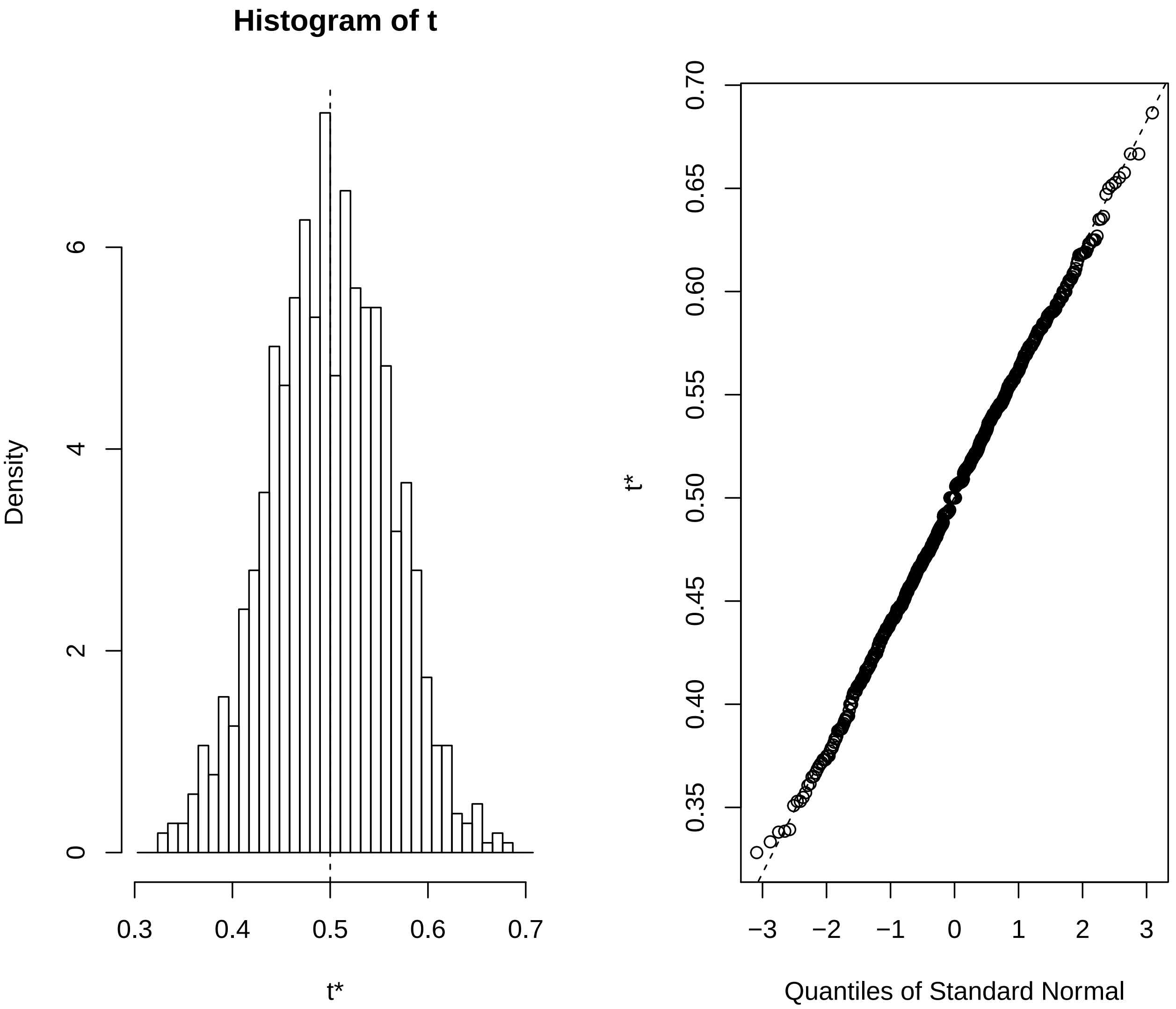
```
# Generate bootstraps
## HIV+
set.seed(2019)
(boot_hiv <- boot(data = data,
                  statistic = freq_hiv,
                  R = 999,
                  stype = 'i'))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = freq_hiv, R = 999, stype = "i")
##
##
## Bootstrap Statistics :
##     original       bias    std. error
## t1*      0.5 0.0003686687  0.06073863
```

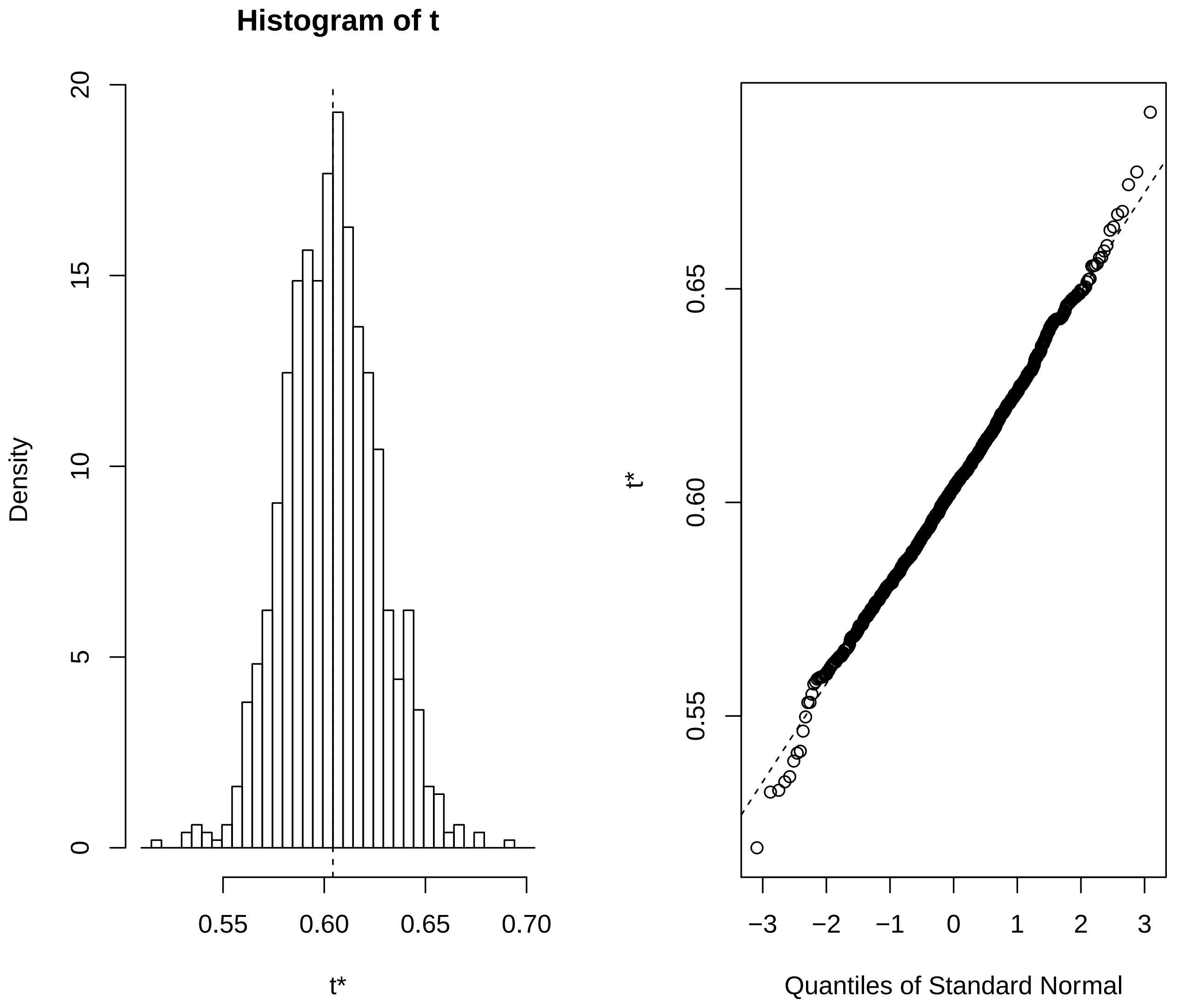
```
## HIV-
set.seed(2019)
(boot_nohiv <- boot(data = data,
                    statistic = freq_nohiv,
                    R = 999,
                    stype = 'i'))
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = freq_nohiv, R = 999, stype = "i")
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.6043011 -0.000775804  0.02298169
```

```
# Plot the distribution
## HIV+
plot(boot_hiv)
```



```
## HIV-
plot(boot_nohiv)
```



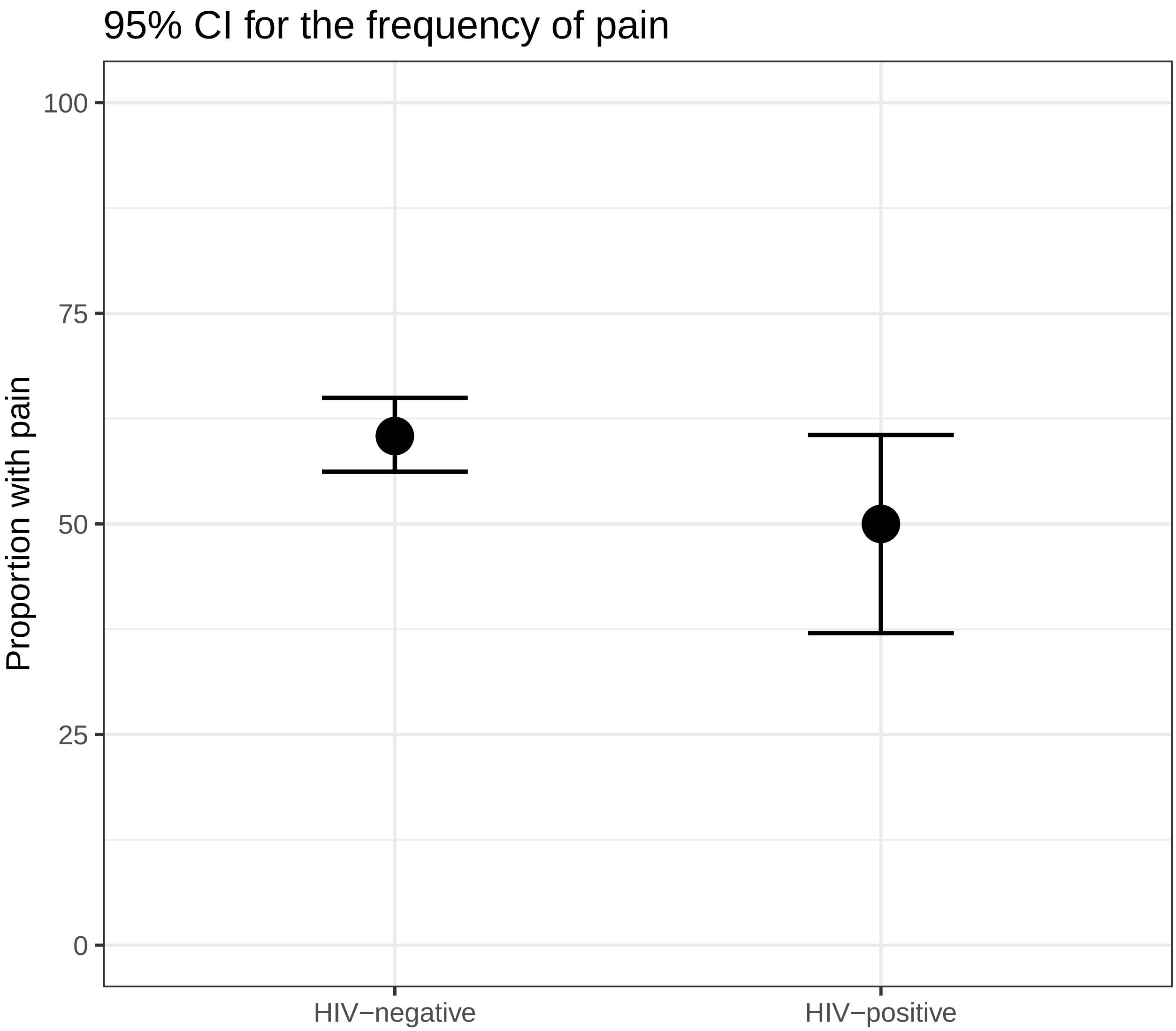
```
# Generate bias accelerated interval
## HIV+
(ci_hiv <- boot.ci(boot_hiv,
                   type = 'bca'))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_hiv, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.3705,  0.6056 )
## Calculations and Intervals on Original Scale
```

```
## HIV-
(ci_nohiv <- boot.ci(boot_nohiv,
                     type = 'bca'))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_nohiv, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.5620,  0.6497 )
## Calculations and Intervals on Original Scale
```

```
# Generate df for plotting
df_hiv <- tibble(group = c('HIV-negative',
                           'HIV-positive'),
                 t = c(100 * ci_nohiv$t0,
                       100* ci_hiv$t0),
                 lower_ci = c(100 * ci_nohiv$bca[[4]],
                              100 * ci_hiv$bca[[4]]),
                 upper_ci = c(100 * ci_nohiv$bca[[5]],
                              100* ci_hiv$bca[[5]]))


# Plot
ggplot(data = df_hiv) +
    aes(x = group,
        y = t) +
    geom_errorbar(aes(ymin = lower_ci,
                      ymax = upper_ci),
                  width = 0.3,
                  size = 1) +
    geom_point(size = 8) +
    scale_y_continuous(limits = c(0, 100)) +
    labs(title = '95% CI for the frequency of pain',
         y = 'Proportion with pain') +
    theme(axis.title.x = element_blank())
```



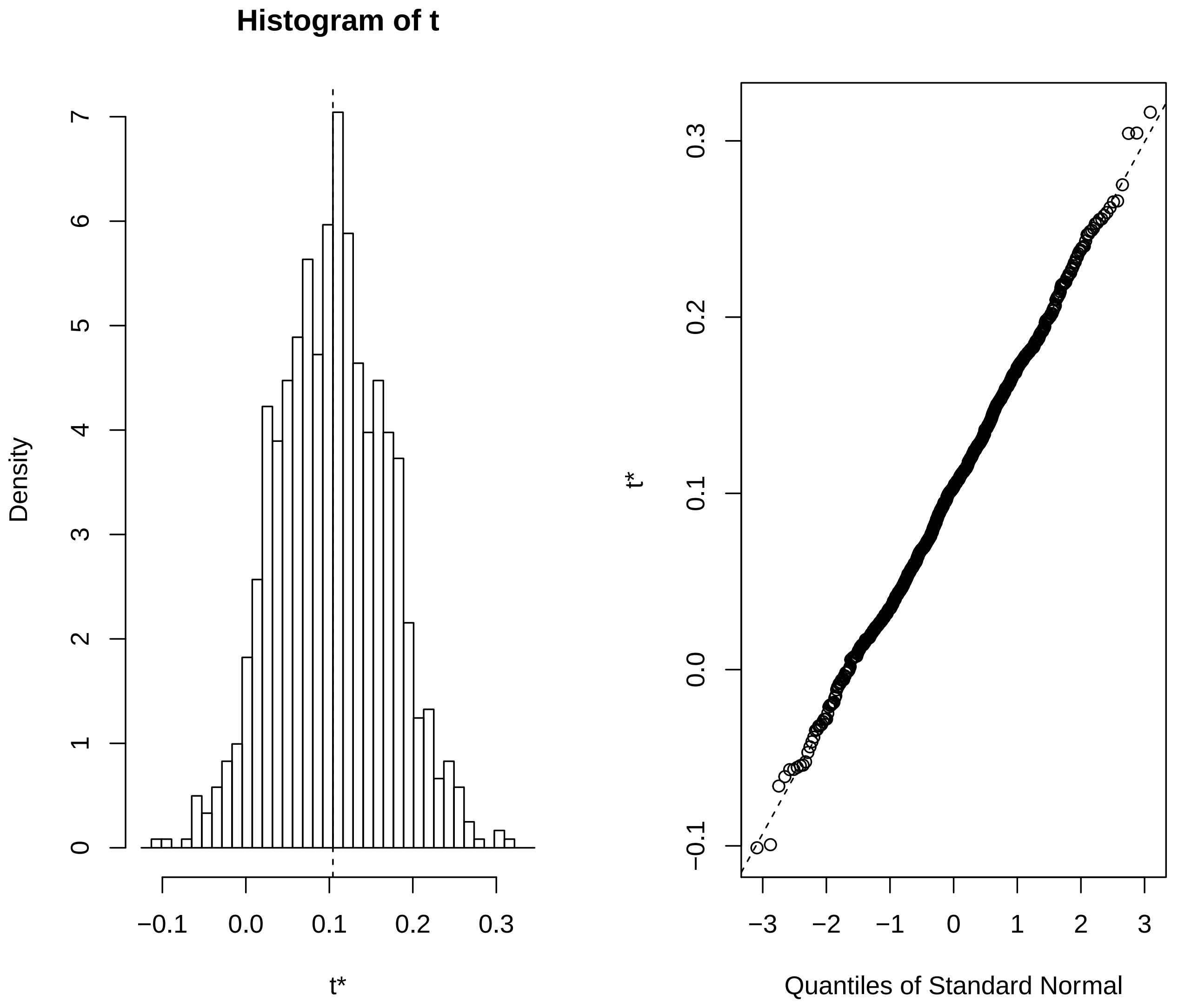
```
# Publication plot
pp <- ggplot(data = df_hiv) +
    aes(x = group,
        y = t) +
    geom_errorbar(aes(ymin = lower_ci,
                      ymax = upper_ci),
                  width = 0.3,
                  size = 1) +
    geom_point(size = 8) +
    scale_y_continuous(limits = c(0, 100)) +
    scale_x_discrete(labels = c('Negative', 'Positive')) +
    labs(y = 'Frequency of pain (%)',
         x = 'HIV test result') +
    theme_bw(base_size = 20) +
    theme(panel.grid = element_blank(),
          axis.text = element_text(colour = '#000000'))

ggsave(filename = 'figures/figure1.png',
       plot = pp,
       width = 7,
       height = 6)
```

95% CI for the confidence interval of the difference in proportion with pain

```
# Define boot function
freq_delta <- function(d, i){
    data <- d[i, ]

    hiv <- data[data$HIV_positive == 'yes', ]
    hivpain <- mean(hiv$pain == 'yes')

    nohiv <- data[data$HIV_positive == 'no', ]
    nohivpain <- mean(nohiv$pain == 'yes')

    nohivpain - hivpain
}

# Generate bootstraps
set.seed(2019)
(boot_delta <- boot(data = data,
                    statistic = freq_delta,
                    R = 999,
                    stype = 'i'))
```

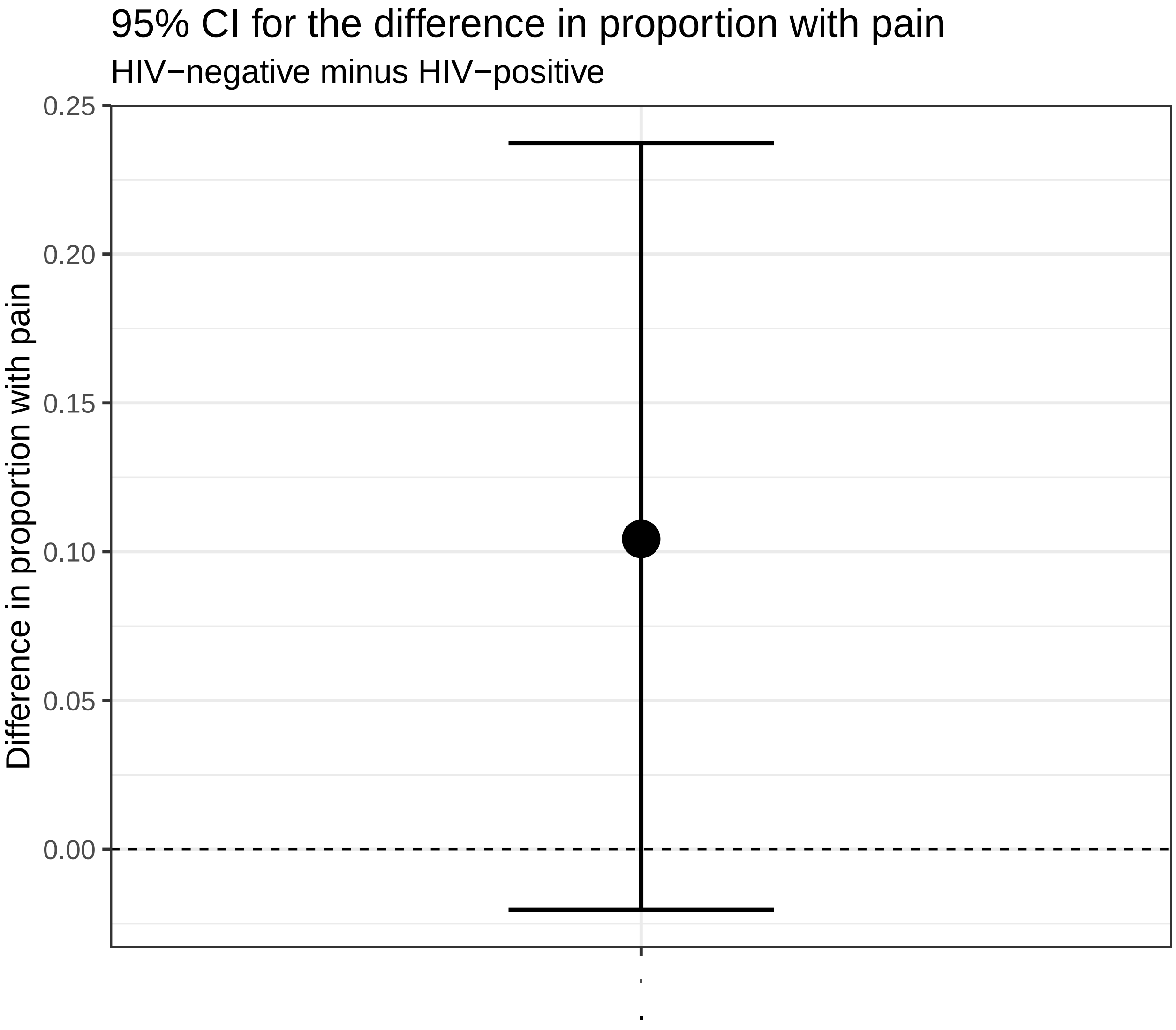
```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = data, statistic = freq_delta, R = 999, stype = "i")
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.1043011 -0.001144473  0.06540082
```

```
# Plot the distribution
plot(boot_delta)
```



```
# Generate bias accelerated interval
(ci_delta <- boot.ci(boot_delta,
                     type = 'bca'))

## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 999 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot_delta, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   (-0.0202,  0.2373 )
## Calculations and Intervals on Original Scale

# Generate df for plotting
df_delta <- tibble(t = ci_delta$t0,
                   lower_ci = ci_delta$bca[[4]],
                   upper_ci = ci_delta$bca[[5]])


# Plot
```

```r
ggplot(data = df_delta) +
    aes(x = '.') +
    geom_errorbar(aes(ymin = lower_ci,
                      ymax = upper_ci),
                  width = 0.3,
                  size = 1) +
    geom_point(aes(y = t),
               size = 8) +
    geom_hline(yintercept = 0,
               linetype = 2) +
    labs(title = '95% CI for the difference in proportion with pain',
         subtitle = 'HIV-negative minus HIV-positive',
         y = 'Difference in proportion with pain')
```



Session information

```
sessionInfo()
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.5
## 
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
## 
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base     
## 
## other attached packages:
##  [1] car_3.0-2          carData_3.0-2      sjPlot_2.6.3      
##  [4] DataExplorer_0.8.0 boot_1.3-22        skimr_1.0.5       
##  [7] forcats_0.4.0      stringr_1.4.0      dplyr_0.8.0.1     
## [10] purrr_0.3.2        readr_1.3.1        tidyr_0.8.3       
## [13] tibble_2.1.1       ggplot2_3.1.1      tidyverse_1.2.1   
## [16] magrittr_1.5      
## 
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-139      lubridate_1.7.4   insight_0.2.0    
##  [4] httr_1.4.0        tools_3.6.0       TMB_1.7.15       
##  [7] backports_1.1.4   utf8_1.1.4        R6_2.4.0         
## [10] sjlabelled_1.0.17 lazyeval_0.2.2    colorspace_1.4-1 
## [13] withr_2.1.2.9000  tidyselect_0.2.5  gridExtra_2.3    
## [16] mnormt_1.5-5      emmeans_1.3.4     curl_3.3         
## [19] compiler_3.6.0    performance_0.1.0 cli_1.1.0        
## [22] rvest_0.3.3       xml2_1.2.0        sandwich_2.5-1   
## [25] labeling_0.3      bayestestR_0.1.0  scales_1.0.0     
## [28] mvtnorm_1.0-10    psych_1.8.12      digest_0.6.18    
## [31] foreign_0.8-71    minqa_1.2.4       rmarkdown_1.12   
## [34] rio_0.5.16        pkgconfig_2.0.2   htmltools_0.3.6  
## [37] lme4_1.1-21       highr_0.8         htmlwidgets_1.3  
## [40] rlang_0.3.4       readxl_1.3.1      rstudioapi_0.10  
## [43] generics_0.0.2    zoo_1.8-5         jsonlite_1.6     
## [46] zip_2.0.1         Matrix_1.2-17     fansi_0.4.0      
## [49] Rcpp_1.0.1        munsell_0.5.0     abind_1.4-5      
## [52] stringi_1.4.3     multcomp_1.4-10   yaml_2.2.0       
## [55] MASS_7.3-51.4     plyr_1.8.4        grid_3.6.0       
## [58] parallel_3.6.0    sjmisc_2.7.9      crayon_1.3.4     
## [61] lattice_0.20-38   ggeffects_0.9.0   haven_2.1.0      
## [64] splines_3.6.0     sjstats_0.17.4    hms_0.4.2        
## [67] knitr_1.22        pillar_1.3.1      igraph_1.2.4.1   
## [70] estimability_1.3  codetools_0.2-16  glue_1.3.1       
## [73] evaluate_0.13     data.table_1.12.2 modelr_0.1.4     
## [76] nloptr_1.2.1      networkD3_0.4     cellranger_1.1.0 
## [79] gtable_0.3.0      assertthat_0.2.1  openxlsx_4.1.0   
## [82] xfun_0.6          xtable_1.8-4      broom_0.5.2
## [85] coda_0.19-2       survival_2.44-1.1 glmmTMB_0.2.3
## [88] TH.data_1.0-10
```