Script 3a

Pain progression: summary data

Peter Kamerman

22 January 2020

Contents

Analysis notes

Definitions of missingness

Data were regarded as **missing** when *pain in the last week* data were not present for one or more of weeks 0, 12, 24, 36, 48. Data also were classified as **missing** when there were inconsistencies in the data across the variables collected within a week.

Definition of data inconsistencies

Pain was defined as *pain in the last week* being 'Yes', and *pain at its worst* being > 0. These two measurements were then the "gatekeeper" measurements, such that the two measurements both had to be positive ('Yes' and '> 0', respectively) in order for there to be any entries for *site of pain* and *site of worst pain*. Were the data were inconsistent (e.g., when there was no *pain in the last week* and *pain at its worst* $= 0$, but there were entries for *site of pain* and *site of worst pain*), then the *site of pain* and *site of worst pain* entries were marked as **inconsistent**.

Data also were considered **inconsistent** when *pain in the last week* = 'Yes', but *site of worst pain* = 'None'.

Lastly, data were considered **inconsistent** when *site of worst pain* was not listed as one of the pain locations for a given measurement week.

For analysis purposes, missing data in the *site of pain* columns were changed to **'No'** (pain not present in the site). This approach was conservative, but we believed that the approach would have the least effect on the outcome, while still retaining as many participants as possible.

Import data

```
df <- read_rds('data-cleaned/data-ADVANCE.rds') %>%
    select(ranid, interval_name, pain_in_the_last_week,
           any_missing)
```

Quick look

```
head(df)
```

```
## # A tibble: 6 x 4
##   ranid   interval_name pain_in_the_last_week any_missing
##   <chr>   <ord>         <chr>                 <chr>
## 1 01-0001 0 weeks       No                    No
## 2 01-0001 12 weeks      No                    No
## 3 01-0001 24 weeks      No                    No
## 4 01-0001 36 weeks      No                    No
## 5 01-0001 48 weeks      No                    No
## 6 01-0002 0 weeks       No                    No
```

```
glimpse(df)
```

```
## Observations: 5,265
## Variables: 4
## $ ranid                 <chr> "01-0001", "01-0001", "01-0001", "01-0001"…
## $ interval_name         <ord> 0 weeks, 12 weeks, 24 weeks, 36 weeks, 48 …
## $ pain_in_the_last_week <chr> "No", "No", "No", "No", "No", "No", "Yes",…
## $ any_missing           <chr> "No", "No", "No", "No", "No", "No", "No", …
```

Basic clean

```
# Clean and process data
df %<>%
    filter(any_missing == 'No') %>%
    select(-any_missing) %>%
    # Add enrolment marker
    pivot_wider(names_from = interval_name,
                values_from = pain_in_the_last_week) %>%
    mutate(pain_at_enrolment = ifelse(`0 weeks` == 'Yes',
                                      yes = 'Yes',
                                      no = 'No'),
           pain_at_enrolment = factor(pain_at_enrolment)) %>%
    pivot_longer(cols = ends_with(' weeks'),
                 names_to = 'interval_name',
                 values_to = 'pain_in_the_last_week') %>%
    # Add numeric time
    mutate(time_weeks = str_extract(interval_name,
                                    pattern = '[0-9]?[0-9]'),
           time_weeks = as.numeric(time_weeks))
```

Quick tabulation

Analysis data set for the period 0 to 48 weeks

```
# Tabulate data
xtabs(~interval_name, data = df)
```

```
## interval_name
##  0 weeks 12 weeks 24 weeks 36 weeks 48 weeks
##      787      787      787      787      787
```

Analysis

Prepare data

Add a dummy frequency column for sankey diagram.

```
# Add dummy freq
df_sankey <- df %>%
    mutate(freq = 1)
```

Numeric summary: All

```
# Tabulate data
tab_prop1 <- as.data.frame(xtabs(~time_weeks + pain_in_the_last_week, data = df)) %>%
    pivot_wider(names_from = pain_in_the_last_week,
                values_from = Freq,
                names_prefix = 'pain_') %>%
    mutate(total_cases = rowSums(.[2:3])) %>%
    mutate(point_estimate = pain_Yes / total_cases) %>%
    select(time_weeks, total_cases, everything()) %>%
    mutate(time_weeks = as.numeric(as.character(time_weeks)))

# Calculate 95%CI
# Boot function
booted <- function(d, i, prop_factor = 'Yes'){
    data <- d[i, ]
    data2 <- data$pain_in_the_last_week
    prop <- mean(data2 == prop_factor)
    prop
}

# Set the seed
set.seed(2019)

# Bootstrap values
tab_prop2 <- df %>%
    group_by(time_weeks) %>%
    nest() %>%
    mutate(data_boot = map(.x = data,
                           ~ boot(data = .x,
                                  R = 5000,
                                  statistic = booted,
                                  stype = 'i'))) %>%
    mutate(data_ci = map(.x = data_boot,
                         ~ boot.ci(.x,
                                   type = 'basic'))) %>%
    mutate(data_point = map(.x = data_boot,
                            ~ .x$t0),
           data_ci.lower = map(.x = data_ci,
                               ~ .x$basic[[4]]),
           data_ci.upper = map(.x = data_ci,
                               ~ .x$basic[[5]]))

# Extract bootstrapped data
tab_prop2 %<>%
    select(time_weeks, data_point, data_ci.lower, data_ci.upper) %>%
    unnest(cols = c('data_point', 'data_ci.lower', 'data_ci.upper')) %>%
    rename(point_estimate = data_point,
           lower_CI = data_ci.lower,
           upper_CI = data_ci.upper)

# Join tab_prop1 and tab_prop2
tab_prop3 <- tab_prop1 %>%
    left_join(tab_prop2) %>%
    mutate(time_weeks = factor(time_weeks))
```

```
## Joining, by = c("time_weeks", "point_estimate")
```

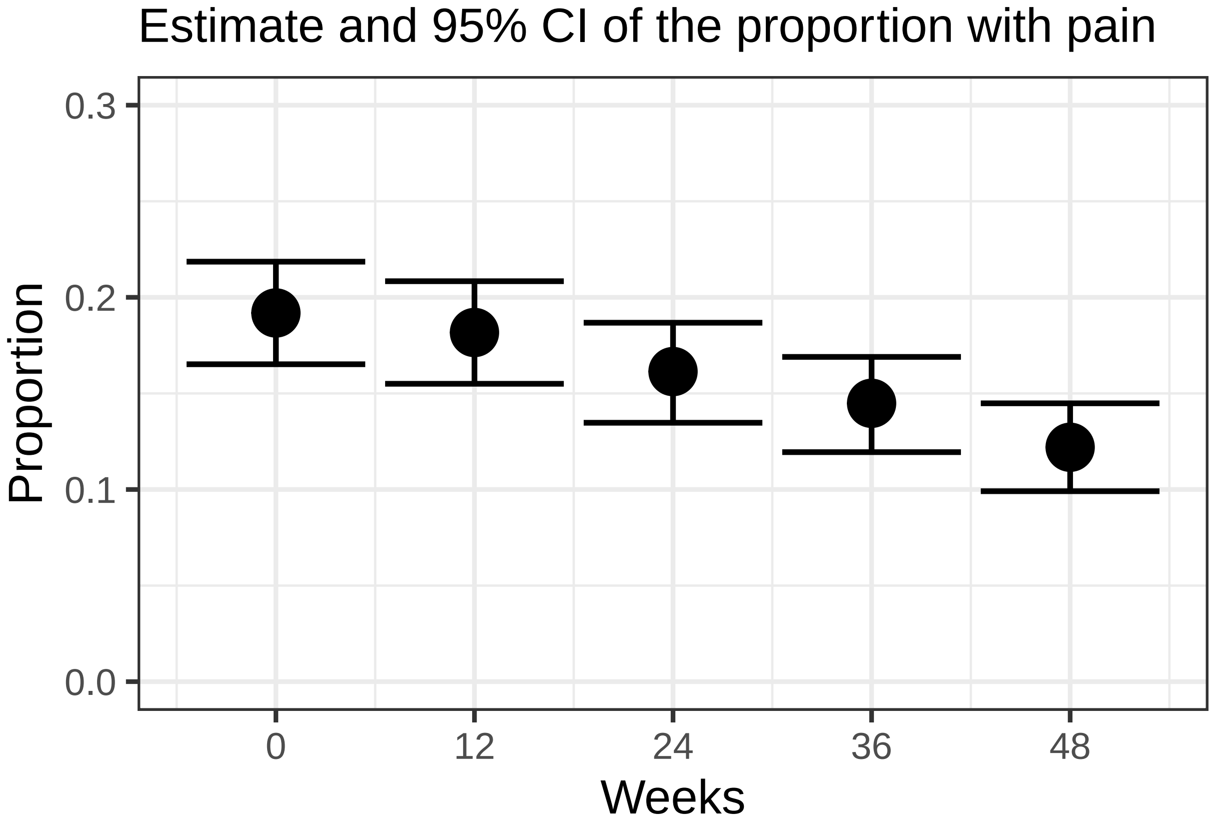
```
# Tabulate
knitr::kable(tab_prop3,
             caption = 'Estimate and 95% CI of the proportion with pain by week')
```

Table 1: Estimate and 95% CI of the proportion with pain by week

time_weeks	total_cases	pain_No	pain_Yes	point_estimate	lower_CI	upper_CI
0	787	636	151	0.1918679	0.1651842	0.2185515
12	787	644	143	0.1817027	0.1550191	0.2083863
24	787	660	127	0.1613723	0.1347206	0.1867853
36	787	673	114	0.1448539	0.1194409	0.1689962
48	787	691	96	0.1219822	0.0991105	0.1448539

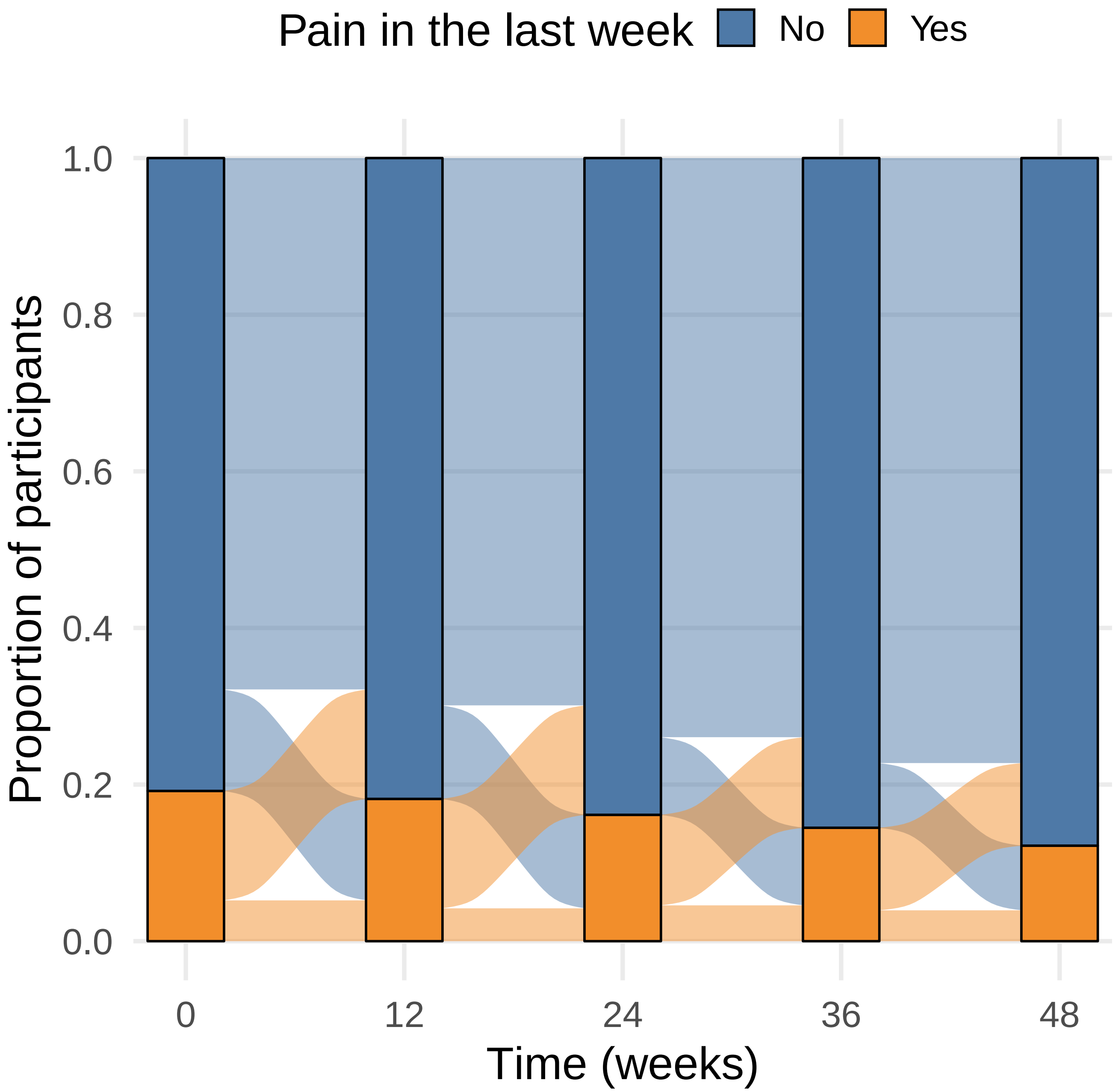
Point plot with 95% CI: All

```
# Plot bootstrapped data
ggplot(data = tab_prop3) +
    aes(x = as.numeric(as.character(time_weeks)),
        y = point_estimate,
        ymin = lower_CI,
        ymax = upper_CI) +
    geom_errorbar(size = 1) +
    geom_point(size = 8) +
    scale_y_continuous(limits = c(0, 0.3)) +
    scale_x_continuous(breaks = c(0, 12, 24, 36, 48)) +
    labs(subtitle = 'Estimate and 95% CI of the proportion with pain',
         x = 'Weeks',
         y = 'Proportion')
```



Sankey plot 1: All

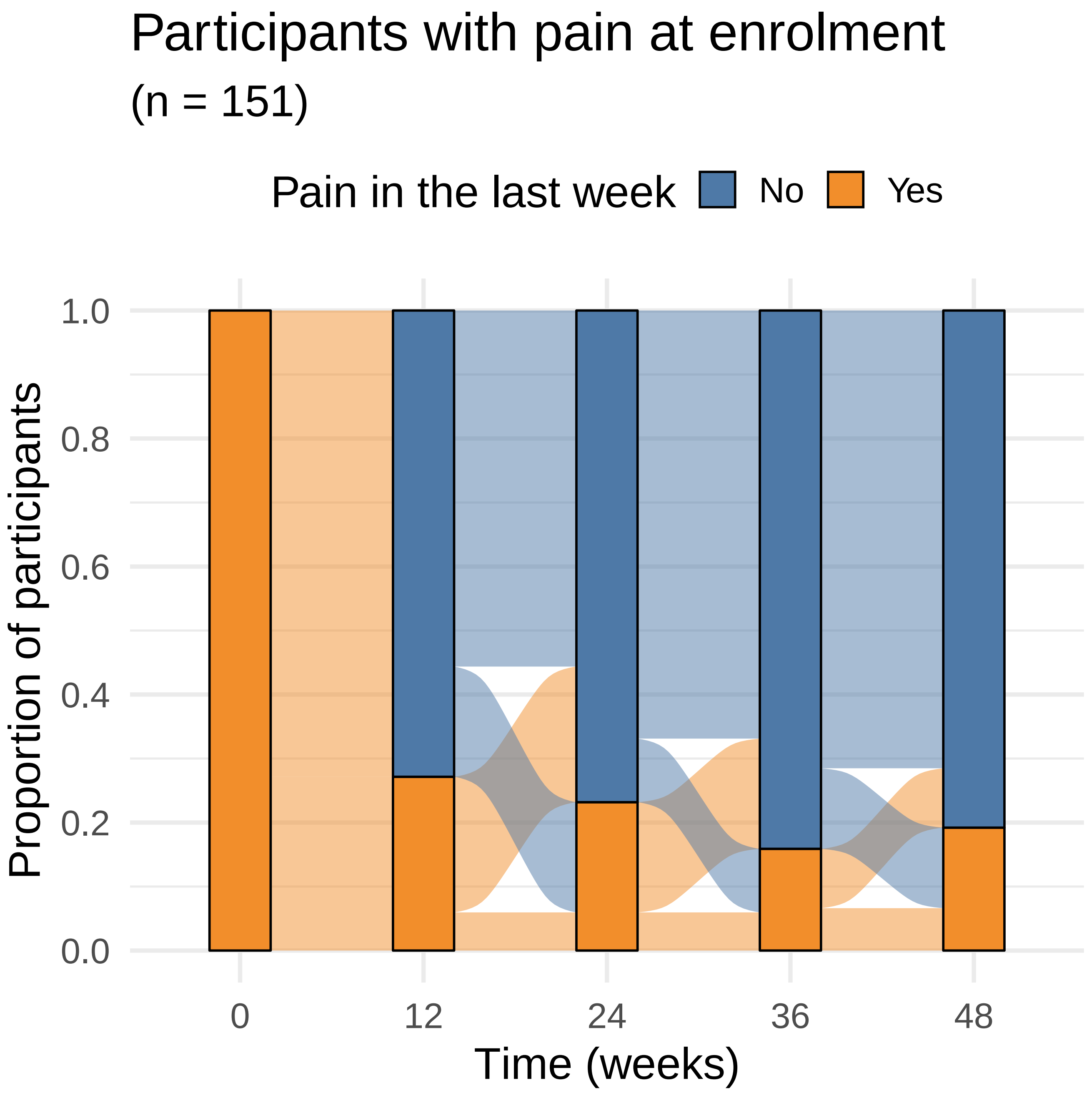
```
# Breaks
n <- length(unique(df$ranid))

# Generate plot
sankey_plot1 <- ggplot(data = df_sankey) +
    aes(x = factor(time_weeks),
        stratum = pain_in_the_last_week,
        alluvium = ranid,
        y = freq,
        fill = pain_in_the_last_week,
        label = pain_in_the_last_week) +
    geom_flow() +
    geom_stratum(width = 0.35) +
    scale_fill_tableau(name = 'Pain in the last week') +
    scale_y_continuous(breaks = c(n * 0, n * 0.2, n * 0.4, n * 0.6, n * 0.8, n),
                       labels = c('0.0', '0.2', '0.4', '0.6', '0.8', '1.0')) +
    scale_x_discrete(expand = c(0.06, 0)) +
    labs(x = 'Time (weeks)',
         y = 'Proportion of participants') +
    theme_minimal(base_size = 20) +
    theme(legend.position = 'top',
          panel.grid.minor = element_blank()); sankey_plot1
```



```
# Generate sankey plot for potential publication plot
sankey_plot2 <- ggplot(data = df_sankey) +
    aes(x = factor(time_weeks),
        stratum = pain_in_the_last_week,
        alluvium = ranid,
        y = freq,
        fill = pain_in_the_last_week,
        label = pain_in_the_last_week) +
    geom_flow() +
    geom_stratum(width = 0.35) +
    scale_fill_tableau(name = 'Pain in the last week') +
    scale_y_continuous(breaks = c(n * 0, n * 0.2, n * 0.4, n * 0.6, n * 0.8, n),
                       labels = c('0.0', '0.2', '0.4', '0.6', '0.8', '1.0')) +
    scale_x_discrete(expand = c(0.06, 0)) +
    labs(x = 'Time (weeks)',
         y = 'Proportion of participants') +
    theme_minimal(base_size = 20) +
    theme(legend.position = 'top',
          panel.grid.minor = element_blank(),
          axis.text.x = element_blank(),
          axis.title.x = element_blank())

# Generate participant-level plot for potential publication plot
plot_individual <- df %>%
    group_by(interval_name) %>%
    mutate(id = row_number()) %>%
    ggplot(.) +
    aes(x = factor(time_weeks),
        y = id) +
    geom_tile(aes(fill = pain_in_the_last_week),
              width = 0.35) +
    scale_fill_tableau() +
    labs(x = 'Time (weeks)',
         y = 'Participant number') +
    scale_x_discrete(expand = c(0.06, 0)) +
    theme_minimal(base_size = 20) +
    theme(legend.position = 'none',
          panel.grid.minor = element_blank())

# Patchwork plot
plot_patchwork <- sankey_plot2 + plot_individual +
    plot_layout(ncol = 1) + plot_annotation(tag_levels = 'A')

# Save plots
ggsave(filename = 'figures/figure-2.png', sankey_plot1,
       width = 10.2, height = 7.72)
```

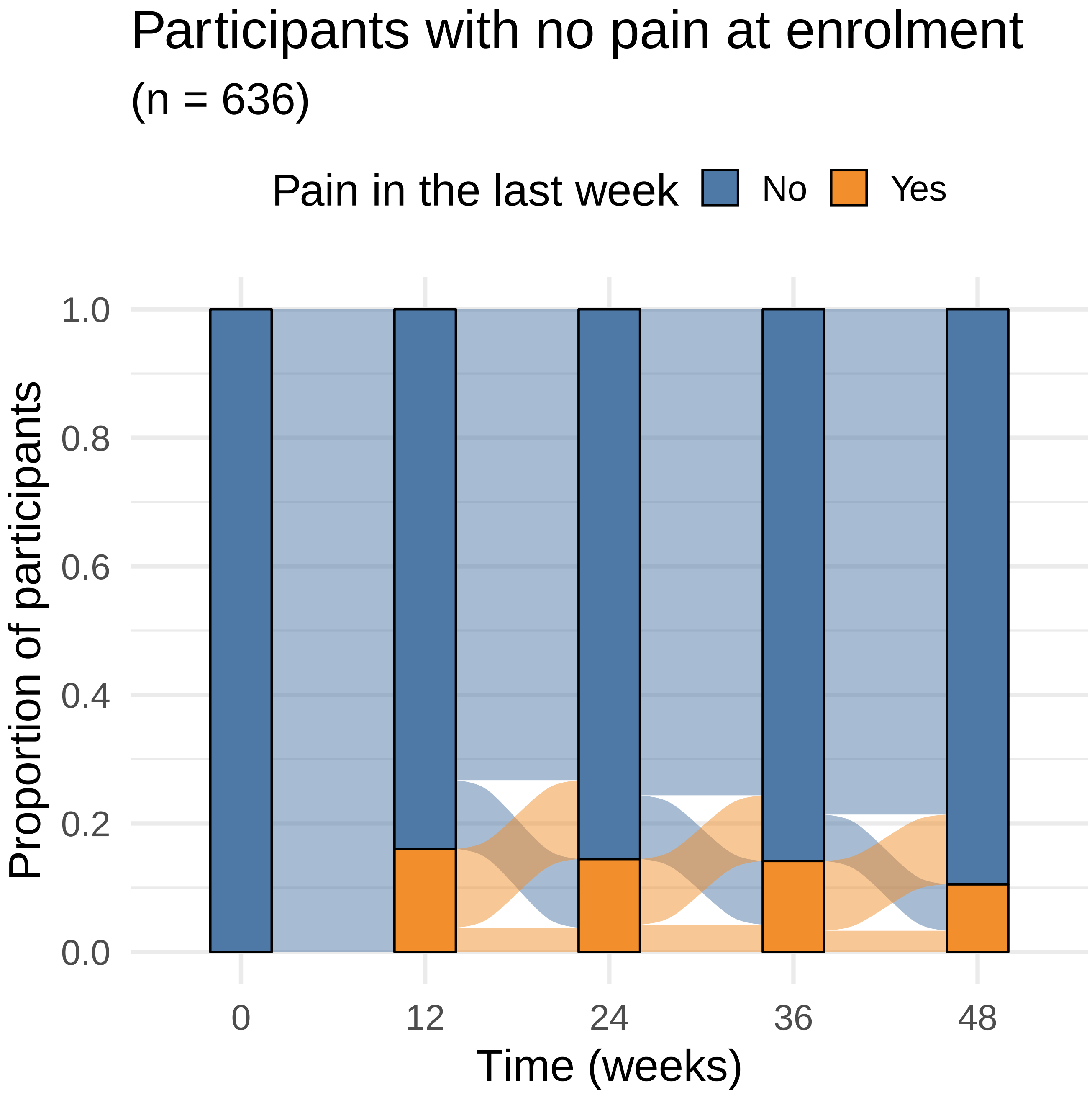
Sankey plot 2: Pain at enrolment

```
# Breaks
n2 <- length(unique(df$ranid[df$pain_at_enrolment == 'Yes']))

# Generate plot
sankey_plot2 <- df_sankey %>%
    select(-interval_name) %>%
    filter(pain_at_enrolment == 'Yes') %>%
    ggplot(data = .) +
    aes(x = factor(time_weeks),
        stratum = pain_in_the_last_week,
        alluvium = ranid,
        y = freq,
        fill = pain_in_the_last_week,
        label = pain_in_the_last_week) +
    geom_flow() +
    geom_stratum() +
    scale_fill_tableau(name = 'Pain in the last week') +
    scale_y_continuous(breaks = c(n2 * 0, n2 * 0.2, n2 * 0.4, n2 * 0.6, n2 * 0.8, n2),
                       labels = c('0.0', '0.2', '0.4', '0.6', '0.8', '1.0')) +
    labs(title = 'Participants with pain at enrolment',
         subtitle = str_glue("(n = {length(unique(df$ranid[df$pain_at_enrolment == 'Yes']))})"),
         x = 'Time (weeks)',
         y = 'Proportion of participants') +
    theme_minimal(base_size = 20) +
    theme(legend.position = 'top'); sankey_plot2
```



Sankey plot 3: No pain at enrolment

```
# Breaks
n3 <- length(unique(df$ranid[df$pain_at_enrolment == 'No']))

# Generate plot
sankey_plot3 <- df_sankey %>%
    select(-interval_name) %>%
    filter(pain_at_enrolment == 'No') %>%
    ggplot(data = .) +
    aes(x = factor(time_weeks),
        stratum = pain_in_the_last_week,
        alluvium = ranid,
        y = freq,
        fill = pain_in_the_last_week,
        label = pain_in_the_last_week) +
    geom_flow() +
    geom_stratum() +
    scale_fill_tableau(name = 'Pain in the last week') +
    scale_y_continuous(breaks = c(n3 * 0, n3 * 0.2, n3 * 0.4, n3 * 0.6, n3 * 0.8, n3),
                       labels = c('0.0', '0.2', '0.4', '0.6', '0.8', '1.0')) +
    labs(title = 'Participants with no pain at enrolment',
         subtitle = str_glue("(n = {length(unique(df$ranid[df$pain_at_enrolment == 'No']))})"),
         x = 'Time (weeks)',
         y = 'Proportion of participants') +
    theme_minimal(base_size = 20) +
    theme(legend.position = 'top'); sankey_plot3
```



Number of people transitioning between pain states at each time interval

Separate time intervals

```r
# Week 0 to 12
t0.12 <- df %>%
    filter(time_weeks %in% c(0, 12)) %>%
    select(-interval_name, -pain_at_enrolment)

# Week 12 to 24
t12.24 <- df %>%
    filter(time_weeks %in% c(12, 24)) %>%
    select(-interval_name, -pain_at_enrolment)

# Week 24 to 36
t24.36 <- df %>%
    filter(time_weeks %in% c(24, 36)) %>%
    select(-interval_name, -pain_at_enrolment)

# Week 36 to 48
t36.48 <- df %>%
    filter(time_weeks %in% c(36, 48)) %>%
    select(-interval_name, -pain_at_enrolment)
```

Calculate percent transitioning pain state

```r
# Week 0 to 12
t0.12 %>%
    pivot_wider(names_from = time_weeks,
                values_from = pain_in_the_last_week) %>%
    mutate_if(is.factor, as.character) %>%
    mutate(pain_transition = paste0(`0`, `12`)) %>%
    filter(!pain_transition %in% c('YesYes', 'NoNo')) %>%
    mutate(pain_transition = ifelse(pain_transition == 'YesNo',
                                    yes = 'Pain to no pain',
                                    no = 'No pain to pain')) %>%
    group_by(pain_transition) %>%
    summarise(count = n()) %>%
    mutate(prop = count / length(unique(df$ranid)),
           percent = 100 * round(prop, 3)) %>%
    knitr::kable(caption = 'Pain transitions: week 0 to 12')
```

Table 2: Pain transitions: week 0 to 12

pain_transition	count	prop	percent
No pain to pain	102	0.1296061	13
Pain to no pain	110	0.1397713	14

```r
# Week 12 to 24
t12.24 %>%
    pivot_wider(names_from = time_weeks,
                values_from = pain_in_the_last_week) %>%
    mutate_if(is.factor, as.character) %>%
    mutate(pain_transition = paste0(`12`, `24`)) %>%
    filter(!pain_transition %in% c('YesYes', 'NoNo')) %>%
    mutate(pain_transition = ifelse(pain_transition == 'YesNo',
                                    yes = 'Pain to no pain',
                                    no = 'No pain to pain')) %>%
    group_by(pain_transition) %>%
    summarise(count = n()) %>%
    mutate(prop = count / length(unique(df$ranid)),
           percent = 100 * round(prop, 3)) %>%
    knitr::kable(caption = 'Pain transitions: week 12 to 24')
```

Table 3: Pain transitions: week 12 to 24

pain_transition	count	prop	percent
No pain to pain	94	0.1194409	11.9
Pain to no pain	110	0.1397713	14.0

```
# Week 24 to 36
t24.36 %>%
    pivot_wider(names_from = time_weeks,
                values_from = pain_in_the_last_week) %>%
    mutate_if(is.factor, as.character) %>%
    mutate(pain_transition = paste0(`24`, `36`)) %>%
    filter(!pain_transition %in% c('YesYes', 'NoNo')) %>%
    mutate(pain_transition = ifelse(pain_transition == 'YesNo',
                                    yes = 'Pain to no pain',
                                    no = 'No pain to pain')) %>%
    group_by(pain_transition) %>%
    summarise(count = n()) %>%
    mutate(prop = count / length(unique(df$ranid)),
           percent = 100 * round(prop, 3)) %>%
    knitr::kable(caption = 'Pain transitions: week 24 to 36')
```

Table 4: Pain transitions: week 24 to 36

pain_transition	count	prop	percent
No pain to pain	78	0.0991105	9.9
Pain to no pain	91	0.1156290	11.6

```
# Week 36 to 48
t36.48 %>%
    pivot_wider(names_from = time_weeks,
                values_from = pain_in_the_last_week) %>%
    mutate_if(is.factor, as.character) %>%
    mutate(pain_transition = paste0(`36`, `48`)) %>%
    filter(!pain_transition %in% c('YesYes', 'NoNo')) %>%
    mutate(pain_transition = ifelse(pain_transition == 'YesNo',
                                    yes = 'Pain to no pain',
                                    no = 'No pain to pain')) %>%
    group_by(pain_transition) %>%
    summarise(count = n()) %>%
    mutate(prop = count / length(unique(df$ranid)),
           percent = 100 * round(prop, 3)) %>%
    knitr::kable(caption = 'Pain transitions: week 36 to 38')
```

Table 5: Pain transitions: week 36 to 38

pain_transition	count	prop	percent
No pain to pain	65	0.0825921	8.3
Pain to no pain	83	0.1054638	10.5

Session information

```r
sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.6
##
## 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] patchwork_0.0.1   boot_1.3-23       ggthemes_4.2.0
##  [4] ggalluvial_0.10.0 magrittr_1.5      forcats_0.4.0
##  [7] stringr_1.4.0     dplyr_0.8.3       purrr_0.3.3
## [10] readr_1.3.1       tidyr_1.0.0       tibble_2.1.3
## [13] ggplot2_3.2.1     tidyverse_1.2.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.5 xfun_0.10        haven_2.1.1      lattice_0.20-38
##  [5] colorspace_1.4-1 vctrs_0.2.0      generics_0.0.2   htmltools_0.4.0
##  [9] yaml_2.2.0       utf8_1.1.4       rlang_0.4.2      pillar_1.4.2
## [13] glue_1.3.1       withr_2.1.2      modelr_0.1.5     readxl_1.3.1
## [17] plyr_1.8.4       lifecycle_0.1.0  munsell_0.5.0    gtable_0.3.0
## [21] cellranger_1.1.0 rvest_0.3.4      evaluate_0.14    labeling_0.3
## [25] knitr_1.25       fansi_0.4.0      highr_0.8        broom_0.5.2
## [29] Rcpp_1.0.3       scales_1.0.0     backports_1.1.5  jsonlite_1.6
## [33] hms_0.5.1        digest_0.6.23    stringi_1.4.3    grid_3.6.1
## [37] cli_2.0.0        tools_3.6.1      lazyeval_0.2.2   crayon_1.3.4
## [41] pkgconfig_2.0.3  zeallot_0.1.0    xml2_1.2.2       lubridate_1.7.4
## [45] assertthat_0.2.1 rmarkdown_1.16   httr_1.4.1       rstudioapi_0.10
## [49] R6_2.4.1         nlme_3.1-141     compiler_3.6.1
```