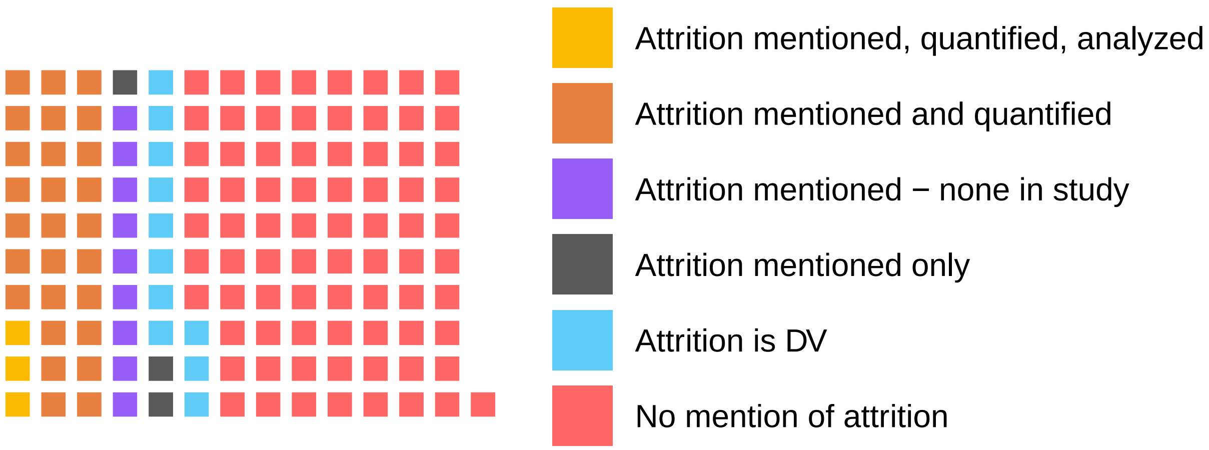
# Replication Code for Figures in `A Practical Guide to Dealing with Attrition in Political Science Experiments`

This version: September 2022

Code to replicate figures from the paper A Practical Guide to Dealing with Attrition in Political Science Experiments by Adeline Lo, Jonathan Renshon, and Lotem Bassan-Nygate.

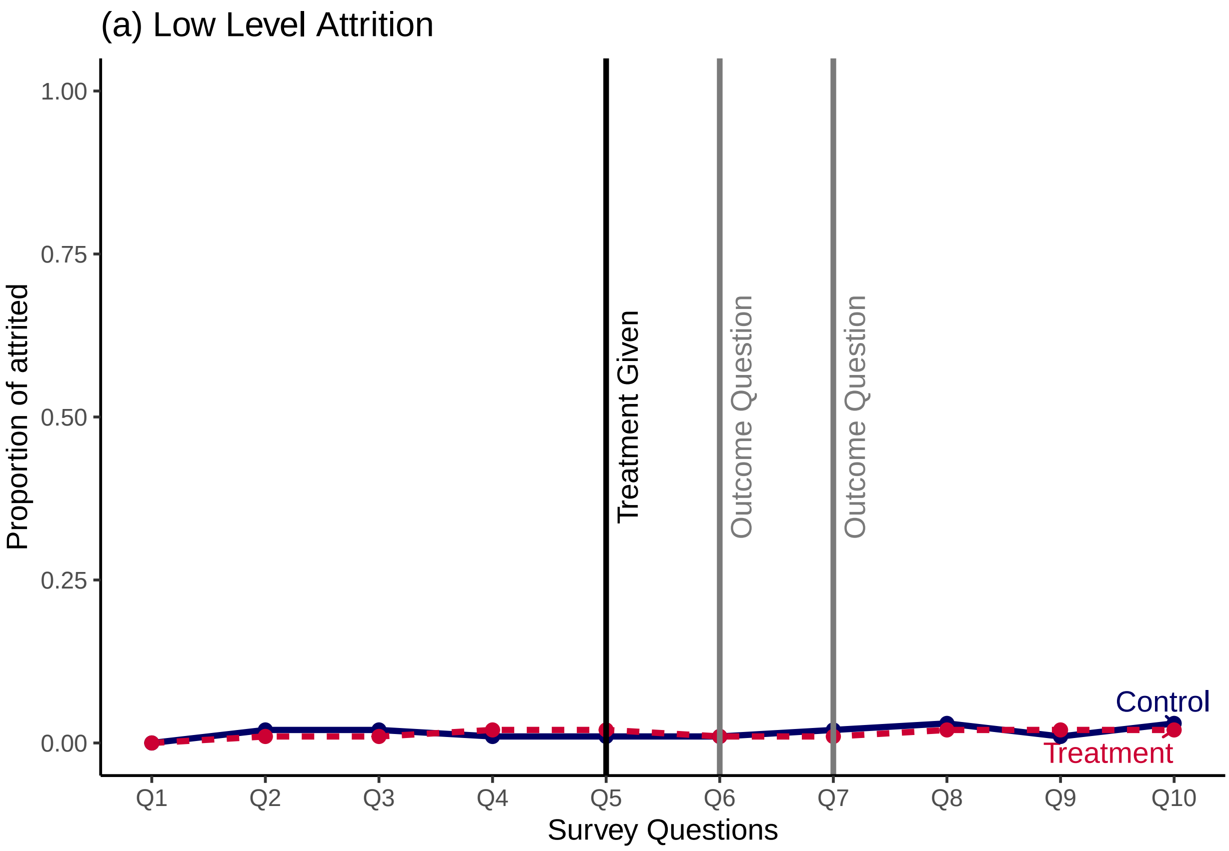
## Fiure 1: Experimental papers in fullJEPScorpus and their discussion of attrition

```
#Reading in CSV Data
attrition <- read_csv("lit_review.csv")


#Functions to remove "*" and change "Yes" to 1 and "No" to 0
remove_star <- function(x) {
  return(str_extract(x, "Yes|No"))
}

yesno_onezero <- function(x) {
  return(case_when(x == "Yes" ~ 1,
                   x == "No" ~ 0))
}

attrition <- attrition %>%
  mutate_at(c(7:14), remove_star) %>%
  mutate_at(c(7:14), yesno_onezero)


#Creating table of proportions
prop_att <- mean(attrition$Attrition)
prop_noatt <- mean(attrition$`0 Attrition`[attrition$Attrition == 1])
prop_attdv <- mean(attrition$`Response Rate DV`[attrition$Attrition == 1])
prop_quan <- mean(attrition$`Quantified Attrition`[attrition$Attrition == 1 & attrition$`0 Attrition` ==
prop_adj <- mean(attrition$`Sample Adjustments`[attrition$Attrition == 1 & attrition$`0 Attrition` == 0

attrition_summary <- as_tibble(data.frame(
  c("Measurement",
    "Proportion that mention attrition",
    "Proportion \"no attrition\"",
    "Proportion DV",
    "Proportion quantify",
    "Proportion adjust"),
  c("Value",
    prop_att,
    prop_noatt,
    prop_attdv,
    prop_quan,
    prop_adj)
))


#Creating variable for the waffle plot
count <- attrition %>%
  mutate(waffle = case_when(`Sample Adjustments` == 1 ~ "Attrition mentioned, quantified, analyzed",
                            Attrition == 1 & `0 Attrition` == 0 & `Response Rate DV` == 0 & `Sample Adju
                            Attrition == 1 & `0 Attrition` == 0 & `Response Rate DV` == 0 & `Sample Adju
                            `Response Rate DV` == 1 ~ "Attrition is DV",
                            `0 Attrition` == 1 ~ "Attrition mentioned - none in study",
                            Attrition == 0 ~ "No mention of attrition")) %>%
  group_by(waffle) %>%
  summarise(n = n())

#Reordering to make legend easier to read and plot look better
count <- count[c(5,3,2,4,1,6),]

#Creating waffle plot
case_counts <- count$n
names(case_counts) <- count$waffle

waffle(case_counts, colors = c(
  "#fcba03", #For Attrition mentioned, quantified, analyzed
  "#e8803f", #For Attrition mentioned and quantified
  "#965ef7", #For Attrition mentioned, none in study
  "#595959", #For Attrition mentioned only
  "#5eccf7", #For Attrition is DV
  "#ff6666"  #For No mention of attrition
  )) +
  theme(legend.key.size = unit(10, "mm"), legend.text = element_text(size = 12))
```

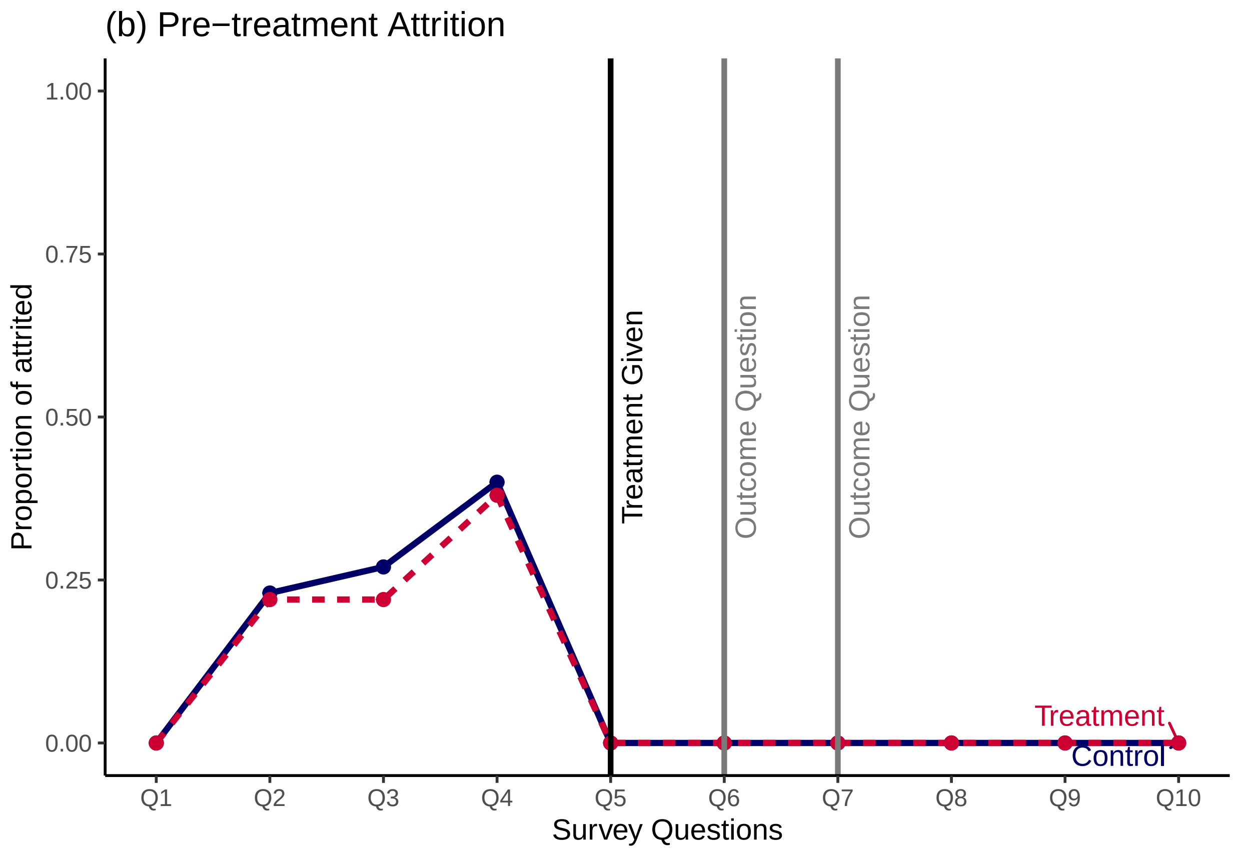


# Figure 3: Attrition timeline visualization

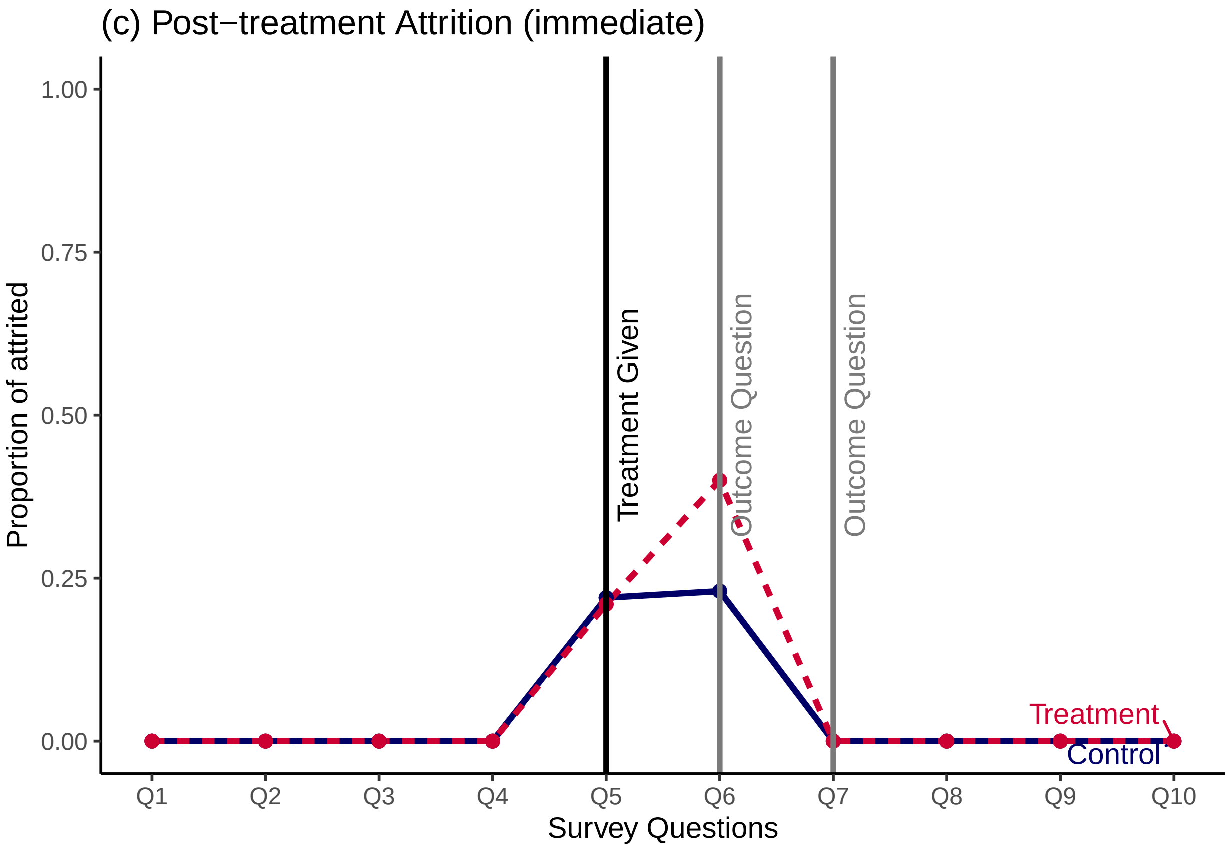
```
#plot_attrition() function.
plot_attrition <- function(data
                           ,treatment_a = NULL
                           ,treatment_q = NULL
                           ,outcome_q = NULL
                           ,mycolors= NULL
                           ,title = NULL)
  {
  #required packages
  require(ggplot2)
  require(viridis)
  require(Hmisc)
  require(dplyr)
  require(ggrepel)
  require(data.table)

# rename the condition variable
data_original<-data #save this original data for reference
data2 <- rename(data, cond_new = treatment_a) #create `cond_new` var based on conditions
data$cond_new<-data2$cond_new

#split the dataset into a list by conditions
data_split<-split(data, with(data, cond_new), drop = TRUE)

#for loop to account for attrition by condition
listofdfs <- list()
for (i in 1:length(data_split)) {
  #first remove the `cond_new` var we created before
    df<-as.data.frame(data_split[i])
    df[ncol(df)]<-NULL
  #for each missing value assign value 1, for complete response assign 0.
    df<- apply(df,2,function(x) {ifelse(is.na(x),1,0)})
  #apply "skip_to_attrite" to get rid of skippers
    df<-t(apply(df,1,skip_to_attrite))
  #sum the number of missing (minus skippers) per q
    df1<- data.frame(colSums(df))
  #rename this variable `missing`
    colnames(df1)<- "missing"
  #create variable `attrited`, rather than missing minus skippers
    df1$attrited<-c(df1[1,], df1[-1,] - df1[-nrow(df1),])
  #`prop_q` = attrited / n entering into the question
    df1$n_prev<- Lag(nrow(df) - as.numeric(df1$missing), +1)
    df1$n_prev[1] <- nrow(df)
    df1$prop_q <- round(df1$attrited/df1$n_prev,2)
  #`proportion`= attrited / starting N
    df1$proportion <- round(df1$attrited/nrow(df),2)
  #add variable for question name
    df1$questions<-colnames(data_original)
  #based on rownames per dataset, create `treatment` var
    df1$treatment<-rownames(df1)
    df1$treatment<-gsub("\\..*","",df1$treatment)
  #remove rownames
    rownames(df1) <- c()
  #save as a list
    listofdfs[[i]] <- df1
}

#merge all datasets in the list
data_combined<- rbindlist(listofdfs)

#create a vector for the unique values of the question names
question_names<-unique(data_combined$questions)
#change question var to factor and numeric for plotting
data_combined$questions <- factor(data_combined$questions,
                                  levels=question_names)
data_combined$questions2<-as.numeric(data_combined$questions)


#create indicators for Vlines
#treatment Vline
treatment_vars<-as.data.frame(match(treatment_q, question_names))
colnames(treatment_vars) <- "treatment_q"
treatment_vars$label<-"Treatment Given" #labels
treatment_vars$color<- "black" #color of vline
treatment_vars$ynum<- 0.5 #where the label appears on yaxis
treatment_vars$size<-1

#outcome Vline
DV<-as.data.frame(match(outcome_q, question_names))
colnames(DV) <- "outcome_q"
DV$label<-"Outcome Question"
DV$color<- "gray48"
DV$ynum<- 0.5
DV$size<-1

p <- data_combined %>%

    ggplot(aes(questions2,prop_q, group = treatment)) +

        #scale x axis from 1:10
       scale_x_continuous(breaks=unique(data_combined$questions2),
       labels=question_names) + #label questions with Q

      #create geomlines for `treatment` and `control`
      geom_line(data = data_combined, aes(questions2, prop_q,
                             color = treatment,
                             linetype=treatment),
                size = 1.1,
                             show.legend = FALSE) +

      #label `treatment` and `control`

      geom_text_repel(data = data_combined %>% filter(questions2 == length(question_names)),
                aes(label = treatment,
                    x = questions2,
                    y = prop_q,
                    color = treatment),
                min.segment.length = 0,
                    show.legend = FALSE)+

     #add a geom_point
     geom_point(size=2, aes(colour=factor(treatment),
                            fill = factor(treatment)), show.legend = FALSE)

    if(!is.null(mycolors)) {p <- p+ scale_colour_manual(values=mycolors)}

     #make treatment red and control blue

    #remove gray background
   p <- p + theme(panel.grid.major = element_blank(), panel.grid.minor =
          element_blank(), panel.background = element_blank(),
           axis.line = element_line(colour = "black")) +

    ylim (0, 1) +
    # add title and labels to axis
    labs(x = "Survey Questions", y = "Proportion of attrited") +
    ggtitle(title)

#add the vertical lines
a<-p +
    #DV vertical lines
    annotate(geom = "vline",
             x = c(DV$outcome_q),
             xintercept = c(DV$outcome_q),
             color = c(DV$color),
             size = c(DV$size)) +

    annotate(geom = "text",
             label = c(DV$label),
             x = c(DV$outcome_q),
             y = c(DV$ynum),
             color = c(DV$color),
             angle = 90,
             vjust = 1.5) +

   #treatments vertical lines
   annotate(geom = "vline",
             x = c(treatment_vars$treatment_q),
             xintercept = c(treatment_vars$treatment_q),
             color = c(treatment_vars$color),
             size = c(treatment_vars$size)) +

    annotate(geom = "text",
             label = c(treatment_vars$label),
             x = c(treatment_vars$treatment_q),
             y = c(treatment_vars$ynum),
             color = c(treatment_vars$color),
             angle = 90,
             vjust = 1.5)
print(a)
}



#Make toy plots for paper
require(ggpattern)

#(a) Low Level Attrition
#Attition post-treatment (throughout survey)
n <- 1000
df <- data.frame(
Q1 = sample(c("Treatment", "Control"), n, rep = TRUE), #we will assume conditions are assigned when ent
Q2 = sample(c(18:90), n, rep = TRUE), #age
Q3 = sample(c("m", "f"), n, rep = TRUE, prob = c(0.55, 0.45)), #sex
Q4 = sample(c(0,1), n, rep = TRUE))#other general pre-treatment questions
df$Q5 = df$Q1 #at Q5 respondents are presented with treatment (say, vignette)
df$Q6 = sample(c(0,1), n, rep = TRUE) #post treatment questions
df$Q7 = sample(c(0,1), n, rep = TRUE)
df$Q8 = sample(c(0,1), n, rep = TRUE)
df$Q9 = sample(c(0,1), n, rep = TRUE)
df$Q10 = sample(c(0,1), n, rep = TRUE)

df_a<-df

#Generate attrition post
invisible(
sapply(sample(1:nrow(df_a), 150),function(x) {
    a <- sample(1:10,1)
    df_a[x,a:ncol(df_a)] <<- NA
}
))



#generate plot (a)
a<-plot_attrition(data=df_a,
              treatment_a = "Q1",
              treatment_q = "Q5",
              outcome_q =  c("Q6", "Q7"),
              title = "(a) Low Level Attrition",
              mycolors = c("#000066","#CC0033"))
```



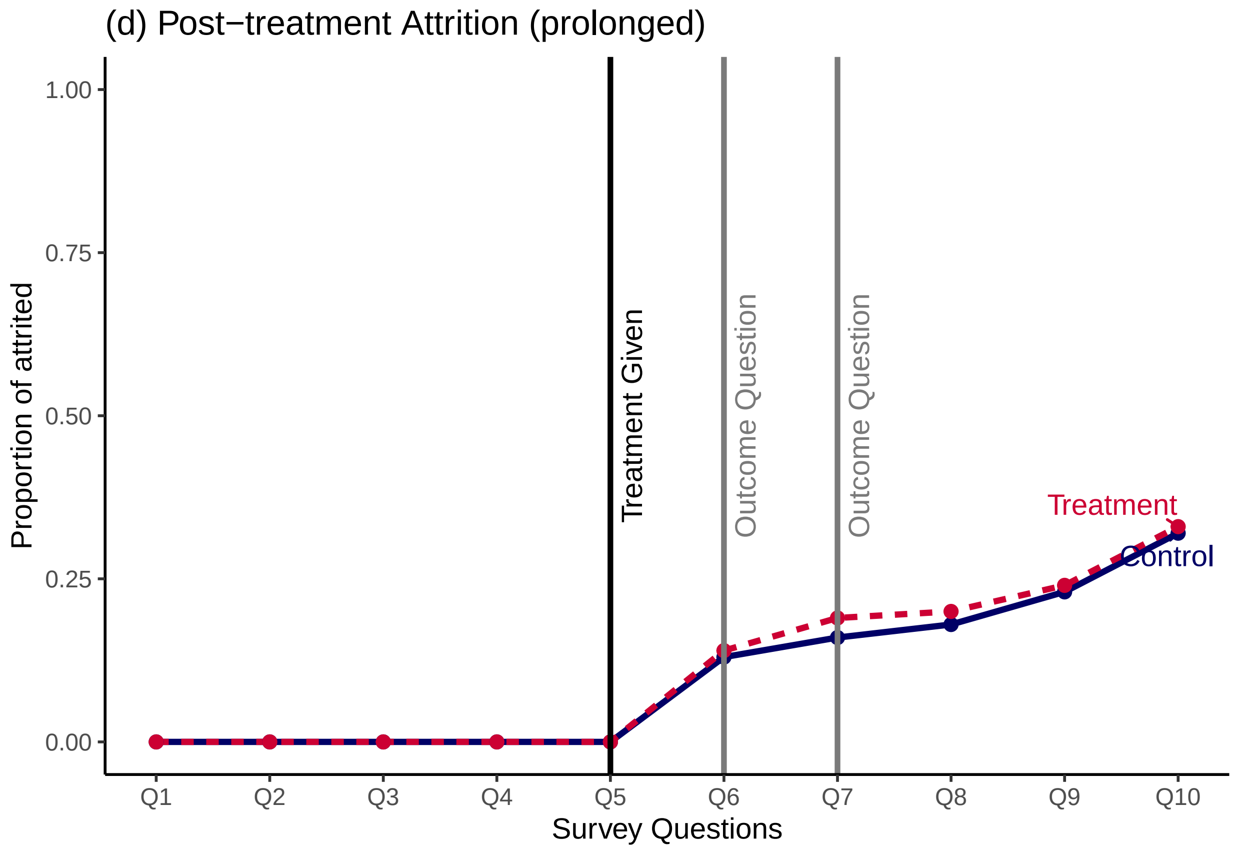
```
#(b) Pre-treatment Attrition
df_b<-df
#Generate attrition pre-treatment
invisible(
sapply(sample(1:nrow(df_b), 700),function(x) {
    a <- sample(1:4,1)
    df_b[x,a:ncol(df_b)] <<- NA
}
))


#generate plot (b)
b<-plot_attrition(data=df_b,
              treatment_a = "Q1",
              treatment_q = "Q5",
              outcome_q =  c("Q6", "Q7"),
              title = "(b) Pre-treatment Attrition",
              mycolors = c("#000066","#CC0033"))
```



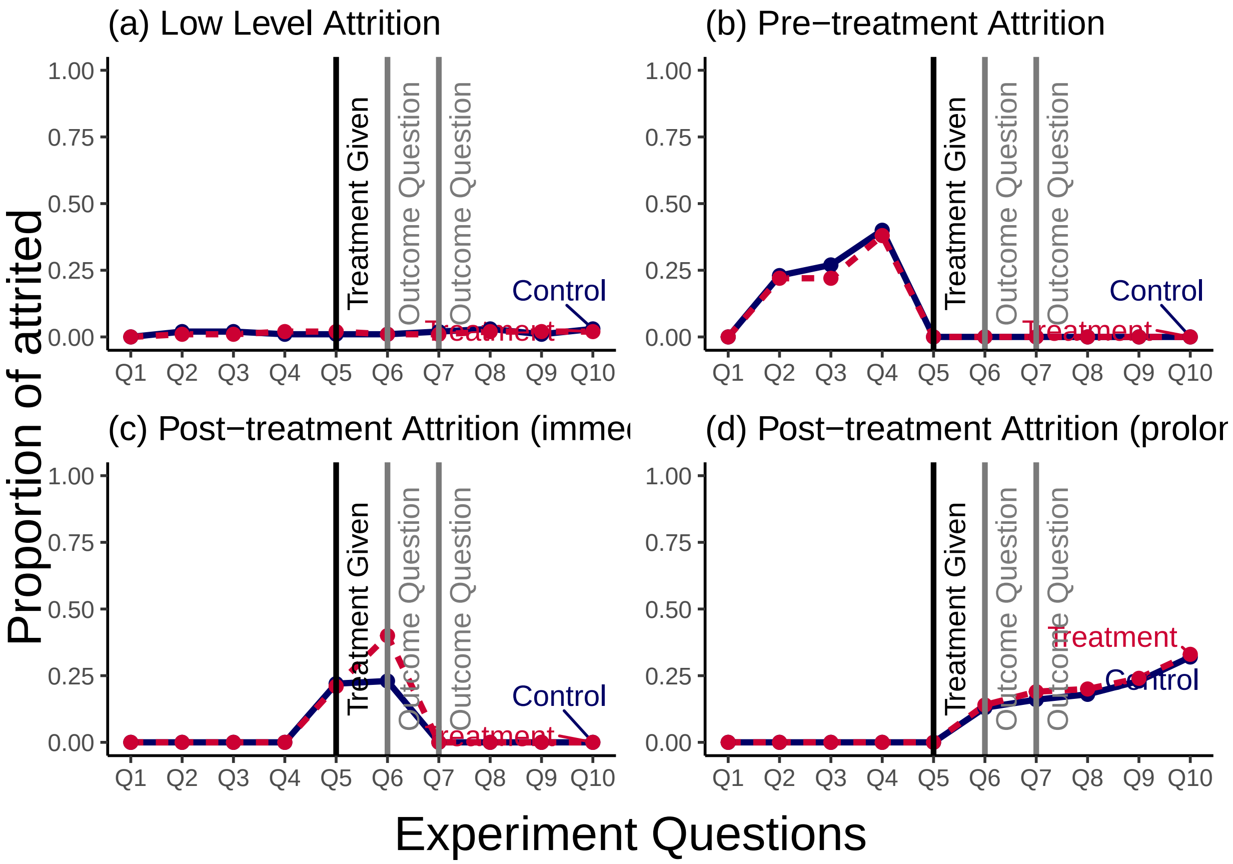
```
#(c) Post-treatment Attrition (immediate)
df_c<-df

#First, we generate some general attrition at treatment
invisible(
sapply(sample(1:nrow(df_c), 500, 0.8*nrow(df_c)),function(x) {
    a <- sample(5:6,1)
    df_c[x,a:ncol(df_c)] <<- NA
}
))

#second, we add some attrition that's correlated with the treatment
#specifically, we want to demonstrate attrition that happens at a certain time
#to do so, we add a running var that will demonstrate time
df_c$no<-rownames(df_c)
df_c$Q6<-ifelse(df_c$Q5=="Treatment"&(df_c$no>100&df_c$no<300), NA,df_c$Q6)
df_c$Q7<-ifelse(is.na(df_c$Q6),NA,df_c$Q7)
df_c$Q8<-ifelse(is.na(df_c$Q6),NA,df_c$Q8)
df_c$Q9<-ifelse(is.na(df_c$Q6),NA,df_c$Q9)
df_c$Q10<-ifelse(is.na(df_c$Q6),NA,df_c$Q10)

df_c$no<-NULL

#generate plot (c)
c<-plot_attrition(data=df_c,
              treatment_a = "Q1",
              treatment_q = "Q5",
              outcome_q =  c("Q6", "Q7"),
              title = "(c) Post-treatment Attrition (immediate)",
              mycolors = c("#000066","#CC0033"))
```



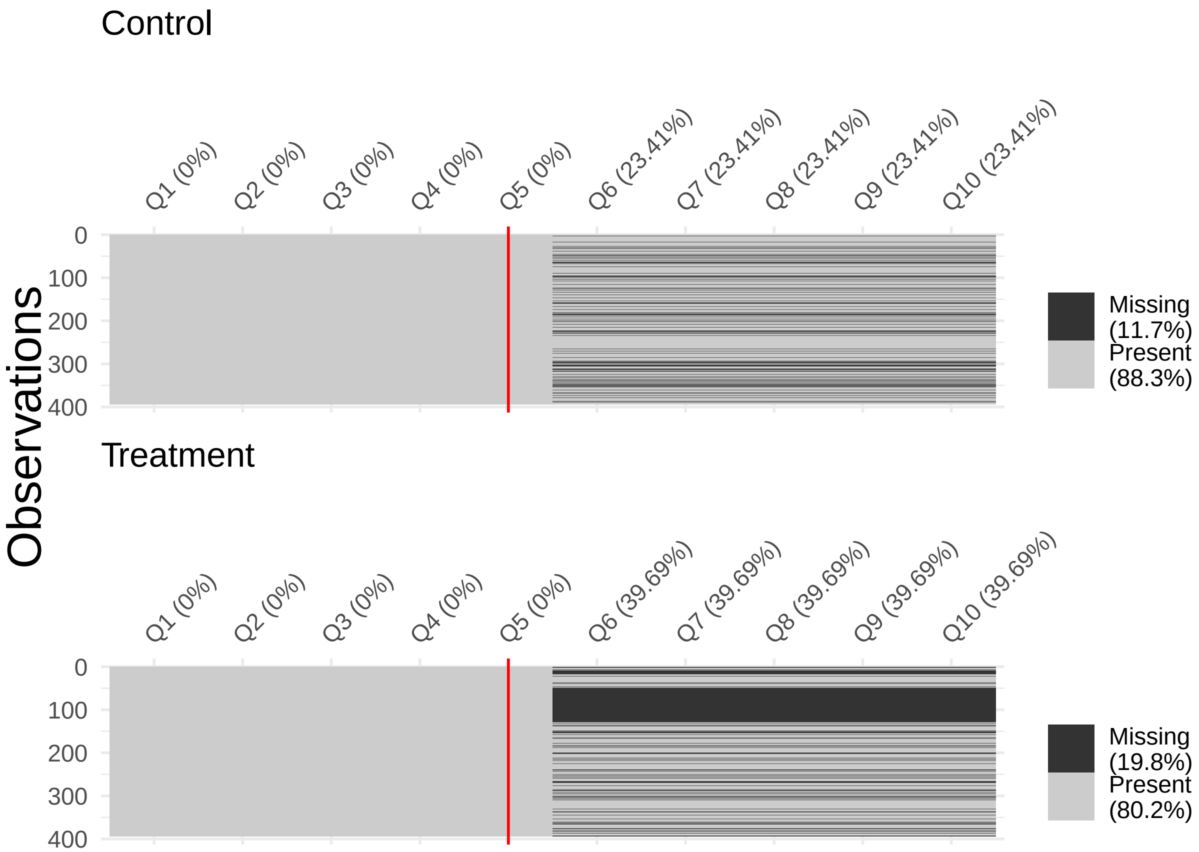
```
#(d) Post-treatment Attrition (prolonged)
df_d<-df
#Generate attrition at DV + after
invisible(
sapply(sample(1:nrow(df_d), 700),function(x) {
    a <- sample(6:10,1)
    df_d[x,a:ncol(df_d)] <<- NA
}
))

#generate plot (d)
d<-plot_attrition(data=df_d,
              treatment_a = "Q1",
              treatment_q = "Q5",
              outcome_q =  c("Q6", "Q7"),
              title = "(d) Post-treatment Attrition (prolonged)",
              mycolors = c("#000066","#CC0033"))
```



```
require(grid)

figure <- ggarrange(a + rremove("ylab") + rremove("xlab"), b + rremove("ylab")
                    + rremove("xlab"), c + rremove("ylab") + rremove("xlab"),
                    d + rremove("ylab") + rremove("xlab"), # remove axis labels from plots
                    labels = NULL,
                    ncol = 2, nrow = 2,
                    common.legend = TRUE, legend = "top",
                    align = "hv",
                    font.label = list(size = 10, color = "black", face = "bold",
                                      family = NULL, position = "top"))

annotate_figure(figure, left = textGrob("Proportion of attrited", rot = 90, vjust = 1, gp = gpar(cex =
                    bottom = textGrob("Experiment Questions", gp = gpar(cex = 1.5)))
```



```
vis_miss_treat(data=df_c,
               treatment = "Q5")
```


Control
Q1 (0%)
Q2 (0%)
Q3 (0%)
Q4 (0%)
Q5 (0%)
Q6 (23.41%)
Q7 (23.41%)
Q8 (23.41%)
Q9 (23.41%)
Q10 (23.41%)
0
100
200
300
400
Missing (11.7%)
Present (88.3%)
Observations
Treatment
Q1 (0%)
Q2 (0%)
Q3 (0%)
Q4 (0%)
Q5 (0%)
Q6 (39.69%)
Q7 (39.69%)
Q8 (39.69%)
Q9 (39.69%)
Q10 (39.69%)
0
100
200
300
400
Missing (19.8%)
Present (80.2%)