

F03_mci_style_neu_plotting.R

2020-09-19

```
## MCI_STYLE_NEU PLOTTING SCRIPT ##

# Creates a bar plot, an ERP waveform, and scalp topographies for the N400 effect for the different semantic conditions
# (intuitive, violation, MCI) within each type of narrative context (normal, fairytale), separately for verb- and
# picture-related potentials.

## PREPARATION ## -----

# Load packages
library(tidyverse)    # Version 1.3.0
library(magrittr)     # Version 1.5
library(eeguana)      # Version 0.1.4.9000
library(cowplot)      # Version 1.0.0

# Load preprocessed data
a1 <- readRDS("EEG/export/a1.RDS")
avgs <- readRDS("EEG/export/avgs.RDS")

# Remove trials with errors or invalid RTs/ERPs
a1 %<>% filter(!error) %>% na.omit()

# Define ggplot theme
styling <- theme(panel.grid = element_blank(),
  panel.border = element_rect(colour = "black", size = 1),
  legend.position = "right",
  axis.ticks = element_line(colour = "black"),
  axis.title = element_text(color = "black", family = "Helvetica", size = 10),
  axis.text = element_text(color = "black", family = "Helvetica", size = 10),
  legend.title = element_text(color = "black", family = "Helvetica", size = 10, face = "bold"),
```

```

    legend.text = element_text(color = "black", family = "Helvetica", size = 10),
    strip.background = element_blank(),
    strip.text = element_text(color = "black", family = "Helvetica", size = 10))

# Rename some factor levels
a1 %<>% mutate(semantics = factor(semantics, levels = c("int", "vio", "mci"),
                                labels = c("Intuitive", "Violation", "MCI")),
              style = factor(style, levels = c("nor", "ftl"),
                              labels = c("Normal context", "Fairytale context")))
avgs %<>% mutate(semantics = factor(semantics, levels = c("int", "vio", "mci"),
                                labels = c("Intuitive", "Violation", "MCI")),
              style = factor(style, levels = c("nor", "ftl"),
                              labels = c("Normal context", "Fairytale context")))

# Define colours for conditions
condition_colors <- viridisLite::plasma(3, end = 0.9, direction = -1)[c(1, 2, 3)] %>%
  set_names(c("Intuitive", "Violation", "MCI"))

## BAR PLOTS ## -----

# Convert dependent variables to long format
a1_long <- a1 %>% pivot_longer(cols = c(N400_verb, N400_pict), names_to = "dv", values_to = "value",
                              names_transform = list(dv = factor))

# Compute summary statistics (means and confidence intervals) for verb-related and picture-related N400
summs <- map(c("N400_verb", "N400_pict"), function(dv){
  Rmisc::summarySEwithin(a1, measurevar = dv, withinvars = c("semantics", "style"), idvar = "participant",
                        na.rm = TRUE) %>%
    rename(value = !!dv) %>%
    mutate(dv = !!dv)
}) %>% set_names(c("N400_verb", "N400_pict"))

# Bar plots for verb-related and picture-related N400
bars <- map(c("N400_verb", "N400_pict"), function(what){
  if (what == "N400_verb"){
    ylims <- list(min = -1, max = 1.5, step = 0.5)
  } else {
    ylims <- list(min = -4, max = 1, step = 1)
  }

```

```

}
# Brackets and stars for statistical significance
significant <- list("N400_verb" = data.frame(style = as.factor("Normal context"),
      ymin = c(-0.1, -0.25, -0.15),
      ymax = c(-0.25, -0.25, -0.25),
      xmin = c(0.7, 0.7, 1.3),
      xmax = c(0.7, 1.3, 1.3),
      star = "*",
      ystar = -0.28),
    "N400_pict" = data.frame(style = as.factor("Normal context"),
      ymin = c(0.2, 0.5, 0.2),
      ymax = c(0.5, 0.5, 0.5),
      xmin = c(0.7, 0.7, 1.3),
      xmax = c(0.7, 1.3, 1.3),
      star = "**",
      ystar = 0.43))

# Actual plotting
ggplot(summs[names(summs) == what][[1]], aes(x = style, y = value, fill = semantics)) +
  geom_bar(stat = "identity", position = position_dodge(width = 0.9)) +
  geom_errorbar(aes(ymin = value - ci, ymax = value + ci), position = position_dodge(width = 0.9), width = 0.5) +
  geom_segment(data = significant[[what]], aes(x = xmin, y = ymin, xend = xmax, yend = ymax), inherit.aes = FALSE) +
  geom_label(data = significant[[what]], aes(x = style, y = ystar, label = star), inherit.aes = FALSE, size = 6,
    label.size = 0) +
  scale_fill_manual(values = condition_colors) +
  labs(fill = "Semantics") +
  coord_cartesian(ylim = c(ylims$min, ylims$max)) +
  scale_x_discrete(labels = c("Normal\ncontext", "Fairytale\ncontext")) +
  scale_y_continuous(name = "ROI amplitude (µV)", breaks = seq(ylims$min, ylims$max, ylims$step)) +
  geom_hline(yintercept = 0) +
  theme_bw() + styling + theme(axis.title.x = element_blank(), legend.position = "none")
}) %>% set_names(c("N400_verb", "N400_pict"))

## EXAMPLE TRIAL ## -----

# Example for one sentence with verbs in three conditions
stim <- ggplot() + theme_void() + theme(plot.background = element_rect(fill = "white", color = "white")) +
  coord_cartesian(xlim = c(0, 1.2), ylim = c(0, 1)) +
  geom_text(aes(x = 0.5, y = 0.5, label = "'The iron bed'"), size = 4.939, family = "Helvetica", hjust = 1) +

```

```

geom_segment(aes(x = 0.51, xend = 0.54, y = 0.5, yend = 0.8)) +
geom_segment(aes(x = 0.51, xend = 0.54, y = 0.5, yend = 0.5)) +
geom_segment(aes(x = 0.51, xend = 0.54, y = 0.5, yend = 0.2)) +
geom_text(aes(x = 0.55, y = 0.8, label = "creaked"), size = 4.939, family = "Helvetica", fontface = "bold",
           color = condition_colors[1], hjust = 0) +
geom_text(aes(x = 0.55, y = 0.5, label = "splintered"), size = 4.939, family = "Helvetica", fontface = "bold",
           color = condition_colors[2], hjust = 0) +
geom_text(aes(x = 0.55, y = 0.2, label = "walked"), size = 4.939, family = "Helvetica", fontface = "bold",
           color = condition_colors[3], hjust = 0) +
geom_text(aes(x = 0.698, y = 0.8, label = 'slightly'), size = 4.939, family = "Helvetica", hjust = 0) +
geom_text(aes(x = 0.734, y = 0.5, label = 'suddenly'), size = 4.939, family = "Helvetica", hjust = 0) +
geom_text(aes(x = 0.681, y = 0.2, label = 'around'), size = 4.939, family = "Helvetica", hjust = 0) +
draw_plot(get_legend(bars$N400_verb + theme(legend.position = c(0.45,0.5), legend.title = element_blank()))),
          x = 0.65, y = 0.5, vjust = 0.48)

## WAVEFORMS ## -----

# ERP waveforms for verb-related and picture-related N400
waves <- map(c("Verb-related", "Picture-related"), function(what){
  # Limits for y-axis
  ymin <- ifelse(what == "Verb-related", -1.5, -5)
  ymax <- ifelse(what == "Verb-related", 2.5, 3)
  step <- ifelse(what == "Verb-related", 1, 2)
  # Which time window to shade
  tmin <- ifelse(what == "Verb-related", 0.300, 0.150)
  tmax <- ifelse(what == "Verb-related", 0.500, 0.350)
  ROI <- ifelse(what == "Verb-related", "ROI_verb", "ROI_pict")
  # Significant area to highlight (MCI - intuitive in the normal context)
  highlight <- avgs %>%
    select(all_of(ROI)) %>% filter(between(as_time(.sample), !!tmin, !!tmax)) %>%
    group_by(semantics, style, type, .sample) %>% summarise_at(channel_names(.), mean, na.rm = TRUE)
  highlight <- data.frame(seq(tmin, tmax, 0.002),
    highlight %>% filter(type == what, semantics == "MCI", style == "Normal context") %>%
    signal_tbl %>% select(all_of(ROI)),
    highlight %>% filter(type == what, semantics == "Intuitive", style == "Normal context") %>%
    signal_tbl %>% select(all_of(ROI)))
  names(highlight) <- c(".time", "mci", "int")
  highlight$style <- as.factor("Normal context")

```

```

# Stars for significance levels
star <- data.frame("type" = as.factor(c("Verb-related", "Picture-related")), "style" = as.factor("Normal context"),
                  "stars" = c("*", "**"))
star <- subset(star, type == what)
# Actual plotting
avgs %>%
  filter(type == what) %>%
  select(all_of(ROI)) %>%
  ggplot(aes(x = .time, y = .value, color = semantics)) +
  geom_rect(aes(xmin = tmin, xmax = tmax, ymin = -Inf, ymax = Inf), fill = "gray90", inherit.aes = FALSE) +
  geom_ribbon(data = highlight, aes(x = .time, ymin = mci, ymax = int), fill = "#ffff33", inherit.aes = FALSE) +
  geom_text(data = star, aes(x = tmin+(tmax-tmin)/2, y = ymax-step/4, label = stars), inherit.aes = FALSE, size = 6) +
  geom_hline(yintercept = 0, linetype = "dotted") +
  geom_vline(xintercept = 0, linetype = "dotted") +
  stat_summary(fun = "mean", geom = "line") +
  scale_color_manual(values = condition_colors) +
  coord_cartesian(xlim = c(-0.2, 0.8), ylim = c(ymin-step/4, ymax+step/4), expand = FALSE) +
  scale_x_continuous(breaks = seq(-0.1, 0.7, 0.2), labels = seq(-100, 700, 200)) +
  scale_y_continuous(breaks = seq(ymin, ymax, step)) +
  xlab("Time (ms)") + ylab("ROI amplitude (µV)") +
  labs(color = NULL) +
  theme_bw() + styling + theme(legend.position = "none") +
  facet_grid(.~style)
}) %>% set_names(c("N400_verb", "N400_pict"))

```

```

## Adding missing grouping variables: .sample
## Adding missing grouping variables: .sample
## Adding missing grouping variables: .sample
## Adding missing grouping variables: .sample

```

```

## TOPOGRAPHIES ## -----

# Create scalp topographies for verb-related and picture-related N400
topos <- map(c("Verb-related", "Picture-related"), function(what){
  if(what == "Verb-related"){
    tmp <- avgs %>% filter(between(as_time(.sample), 0.300, 0.500), type == "Verb-related")
  } else{
    tmp <- avgs %>% filter(between(as_time(.sample), 0.150, 0.350), type == "Picture-related")}

```

```

tmp <- tmp %>%
  group_by(semantics, style) %>% summarise_at(channel_names(.), mean) %>%
  signal_tbl() %>% select(Fp1:A1) %>% t() %>% as.data.frame()
names(tmp) <- c("int.nor", "int.ftl", "vio.nor", "vio.ftl", "mci.nor", "mci.ftl")
tmp <- data.frame("diff.vio.nor" = tmp$vio.nor - tmp$int.nor,
                  "diff.vio.ftl" = tmp$vio.ftl - tmp$int.ftl,
                  "diff.mci.nor" = tmp$mci.nor - tmp$int.nor,
                  "diff.mci.ftl" = tmp$mci.ftl - tmp$int.ftl,
                  "electrode" = rownames(tmp))
if(what == "Verb-related"){
  els <- c("C1", "C2", "C3", "C4", "CZ", "CP1", "CP2", "CP3", "CP4", "CPZ", "P3", "P4")
} else {
  els <- c("FZ", "CZ")
}
topos <- map(1:4, function(x){
  p <- eegUtils::topoplot(data = tmp, quantity = colnames(tmp)[x], limits = c(-0.7, 0.7), r = 0.9,
                          palette = "plasma", contour = FALSE, highlights = els, scaling = 0.5)
  p$layers[[6]]$aes_params$size <- 0.1
  p$layers[[7]]$aes_params$colour <- "black"
  p <- p + theme(legend.position = "none", plot.title = element_text(hjust = 0.5, size = 10,
                                                                    family = "Helvetica"))})
}) %>% set_names(c("N400_verb", "N400_pict"))

```

Attempting to add standard electrode locations...

```

## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...
## Attempting to add standard electrode locations...

```

Create a colorbar

```

simdat1 <- data.frame(a = 1:10, b = 1:10, c = seq(-0.7, 0.7, length.out = 10))
colbar <- get_legend(ggplot(simdat1, aes(x = a, y = b, fill = c)) + geom_raster() + geom_line() +
                     scale_fill_viridis_c(option = "plasma",
                                           guide = guide_colorbar(ticks = FALSE, title.position = "left",

```

```

label.hjust = 1), breaks = c(-0.7, 0, 0.7)) +
  labs(fill = "Ampl.\n(µV)") +
  theme(legend.position = "right",
        legend.background = element_blank(),
        legend.key.height = unit(0.3, "cm"),
        legend.title = element_text(family = "Helvetica", size = 10, color = "black"),
        legend.text = element_text(family = "Helvetica", size = 10, color = "black"),
        legend.title.align = 0.5))

# Create one plot combining all four topographies (verb-related)
simdat2 <- data.frame("semantics" = factor(c("Violation - Intuitive", "MCI - Intuitive")),
                     "style" = factor(c("Normal\ncontext", "Fairytale\ncontext"),
                                       levels = c("Normal\ncontext", "Fairytale\ncontext")))
topos_verb <- ggplot(simdat2, aes(x = style, y = semantics)) +
  geom_point() +
  draw_plot(topos$N400_verb[[1]], x = 0.4, y = 1.5, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_verb[[2]], x = 1.5, y = 1.5, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_verb[[3]], x = 0.4, y = 0.4, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_verb[[4]], x = 1.5, y = 0.4, width = 1.1, height = 1.1) +
  annotate("text", x = 1.5, y = 0.53, label = "300-500 ms", size = 3.528, family = "Helvetica") +
  draw_plot(colbar, x = 0.95, y = 1) +
  styling +
  theme(panel.border = element_rect(colour = "black", size = 1, fill = alpha("white", 0)),
        panel.background = element_rect(fill = "white"),
        axis.title = element_blank(),
        axis.text.y = element_text(angle = 90, hjust = 0.5))

# Create one plot combining all four topographies (picture-related)
topos_pict <- ggplot(simdat2, aes(x = style, y = semantics)) +
  geom_point() +
  draw_plot(topos$N400_pict[[1]], x = 0.4, y = 1.5, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_pict[[2]], x = 1.5, y = 1.5, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_pict[[3]], x = 0.4, y = 0.4, width = 1.1, height = 1.1) +
  draw_plot(topos$N400_pict[[4]], x = 1.5, y = 0.4, width = 1.1, height = 1.1) +
  annotate("text", x = 1.5, y = 0.53, label = "150-350 ms", size = 3.528, family = "Helvetica") +
  draw_plot(colbar, x = 0.95, y = 1) +
  styling +
  theme(panel.border = element_rect(colour = "black", size = 1, fill = alpha("white", 0)),

```

```

    panel.background = element_rect(fill = "white"),
    axis.title = element_blank(),
    axis.text.y = element_text(angle = 90, hjust = 0.5))

## PUBLICATION-READY FIGURES ## -----

# Figure 1: Verb-Related N400 Effects
plot_grid(stim, waves$N400_verb,
  plot_grid(bars$N400_verb, topos_verb, nrow = 1, rel_widths = c(0.6, 1), labels = c("C", "D"),
    label_fontfamily = "Helvetica", label_y = 1.03),
  nrow = 3, rel_heights = c(0.2, 0.8, 1), labels = c("A", "B", NULL), label_fontfamily = "Helvetica") %>%
  ggsave(filename = "EEG/figures/N400_verb.pdf", width = 18, height = 22, units = "cm")

# Figure 2: Picture-Related N400 Effects
plot_grid(plot_grid(waves$N400_pict) +
  draw_plot(get_legend(bars$N400_pict +
    theme(legend.position = "right", legend.title = element_blank(),
      legend.background = element_blank()))), x = 0.41, y = -0.22),
  plot_grid(bars$N400_pict, topos_pict, nrow = 1, rel_widths = c(0.6, 1), labels = c("B", "C"),
    label_fontfamily = "Helvetica", label_y = 1.03),
  nrow = 2, rel_heights = c(0.8, 1), labels = c("A", NULL), label_fontfamily = "Helvetica") %>%
  ggsave(filename = "EEG/figures/N400_pict.pdf", width = 18, height = 19.8, units = "cm")

# System specs and package versions
sessionInfo()

## R version 4.0.2 (2020-06-22)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18362)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=German_Germany.1252 LC_CTYPE=German_Germany.1252 LC_MONETARY=German_Germany.1252 LC_NUMERIC=C
## [5] LC_TIME=German_Germany.1252
##
## attached base packages:
## [1] stats graphics grDevices datasets utils methods base

```



```
##
```

```
## other attached packages:
```

```
## [1] cowplot_1.0.0      eeguana_0.1.4.9000 magrittr_1.5      forcats_0.5.0      stringr_1.4.0      dplyr_1.0.0        purrr_0.3.4
```

```
## [8] readr_1.3.1        tidyr_1.1.0         tibble_3.0.3      ggplot2_3.3.2      tidyverse_1.3.0
```

```
##
```

```
## loaded via a namespace (and not attached):
```

```
## [1] nlme_3.1-148        matrixStats_0.56.0 fs_1.5.0           lubridate_1.7.9    httr_1.4.2         tools_4.0.2        backports_1.0.4
```

```
## [8] R6_2.4.1            lazyeval_0.2.2      DBI_1.1.0          mgcv_1.8-31        colorspace_1.4-1   withr_2.2.0        tidyselect_1.1.0
```

```
## [15] compiler_4.0.2      cli_2.0.2           rvest_0.3.6        eegUtils_0.5.0.9000 xml2_1.3.2         plotly_4.9.2.1     labeling_0.4.2
```

```
## [22] scales_1.1.1        digest_0.6.25       rmarkdown_2.3      R.utils_2.9.2      ini_0.3.1          pkgconfig_2.0.3    htmltools_0.5.0
```

```
## [29] highr_0.8           dbplyr_1.4.4        fastmap_1.0.1      htmlwidgets_1.5.1  Rmisc_1.5          rlang_0.4.7        readxl_1.3.1
```

```
## [36] rstudioapi_0.11     shiny_1.5.0         farver_2.0.3       generics_0.0.2     jsonlite_1.7.0     R.oo_1.23.0        R.matlab_3.5.1
```

```
## [43] Matrix_1.2-18       Rcpp_1.0.5          munsell_0.5.0      fansi_0.4.1        abind_1.4-5        lifecycle_0.2.0    R.methodsS3_1.0.1
```

```
## [50] yaml_2.2.1          stringi_1.4.6       MASS_7.3-51.6      plyr_1.8.6         grid_4.0.2         blob_1.2.1         parallel_4.0.3
```

```
## [57] listenv_0.8.0       promises_1.1.1      crayon_1.3.4       miniUI_0.1.1.1     lattice_0.20-41    haven_2.3.1        splines_4.0.1
```

```
## [64] hms_0.5.3           knitr_1.29          pillar_1.4.6       future.apply_1.6.0 codetools_0.2-16   reprex_0.3.0       glue_1.4.1
```

```
## [71] evaluate_0.14       data.table_1.13.0   renv_0.12.0        modelr_0.1.8       vctrs_0.3.2        httpuv_1.5.4       cellranger_1.1.0
```

```
## [78] gtable_0.3.0        future_1.18.0       assertthat_0.2.1   xfun_0.16          mime_0.9           xtable_1.8-4       broom_0.7.0
```

```
## [85] pracma_2.2.9        later_1.1.0.1       viridisLite_0.3.0  signal_0.7-6       globals_0.12.5     ellipsis_0.3.1
```