PymcRltReport

Fiona Zhu

2025-11-13

```
library(rlist)
library(ez)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.1     v stringr   1.5.2
## v ggplot2   4.0.0     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.1.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(latex2exp)
library(lsr)
library(rticles)
library(DescTools)
library(heplots)
```

```
## Loading required package: broom
```

```
library(lme4)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(rstatix)
```

```
##
## Attaching package: 'rstatix'
##
## The following object is masked from 'package:stats':
##
##     filter
```

```
library(boot)
library(ggpubr)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(ggplot2)
library(readr)
source('mytheme.R')
```

Setting

```
modelname = "Experimentwise" # "FreeParameters"
filepath = paste0("../data/", modelname,"/")
#
order_exp <- c("Encoding", "Reproduction", "Baseline", "Both", "Both_gap")
exp_labels <- c(
  "Encoding"     = "Exp. 1\nEncoding",
  "Reproduction" = "Exp. 2\nReproduction",
  "Baseline"     = "Exp. 3\nBaseline",
  "Both"         = "Exp. 4\nBoth",
  "Both_gap"     = "Exp. 5\nBoth (with gap)"
)
Exp.labs.2lines <- c("Exp. 1\nEncoding", "Exp. 2\nReproduction", "Exp. 3\nBaseline", "Exp. 4\nBoth", "E
```

Load data

```
mdat_all <- read_csv(paste0(filepath, modelname, "_mdat_all.csv"))
```

```
## New names:
## Rows: 1440 Columns: 15
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (2): WMSize, Exp dbl (13): ...1, Unnamed: 0, NSub, curDur, repDur_mean,
## repDur_std, mPred, sd...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
mdat_all$Exp <- factor(mdat_all$Exp, levels = order_exp)
mdat_all$Gap <- factor(mdat_all$Gap, labels = c("short", "long"))
mdat_all$WMSize <- factor(mdat_all$WMSize, labels = c("low", "medium",  "high"))
```

```
# Here the predicted RP was based on function predict_single_subject
PredRPList <- read_csv(paste0(filepath, modelname, "_PredRPList.csv"))
```

```
## New names:
## Rows: 67680 Columns: 7
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Exp dbl (6): ...1, curDur, Gap, WMSize, mu_r, NSub
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

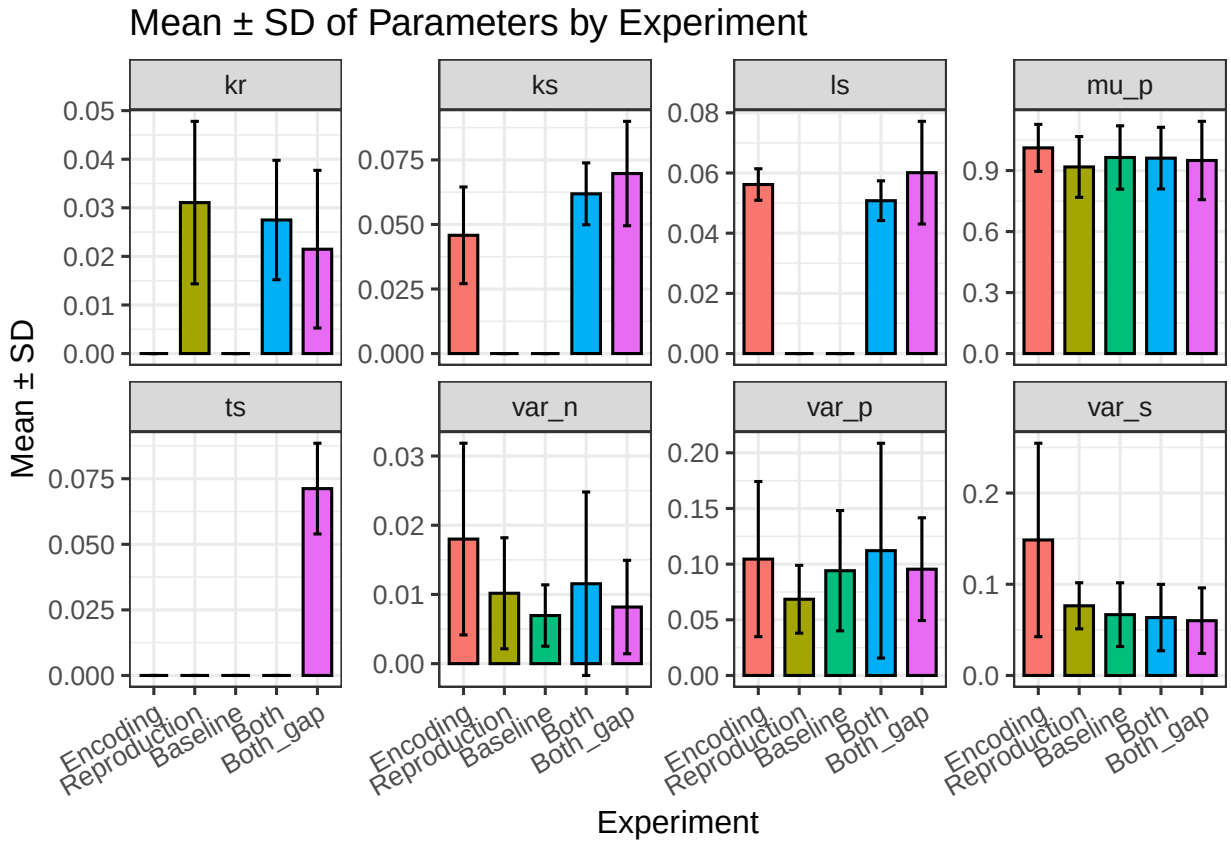
```
PredRPList$WMSize <- factor(PredRPList$WMSize, labels = c("low", "medium",  "high"))
PredRPList$Gap <- factor(PredRPList$Gap, labels = c("short", "long"))

PredRPList$Exp <- factor(PredRPList$Exp, levels = order_exp)
```

Estimated parameters

```
#
ParaList <-  read_csv(paste0(filepath, modelname, "_ParaList.csv"))
```

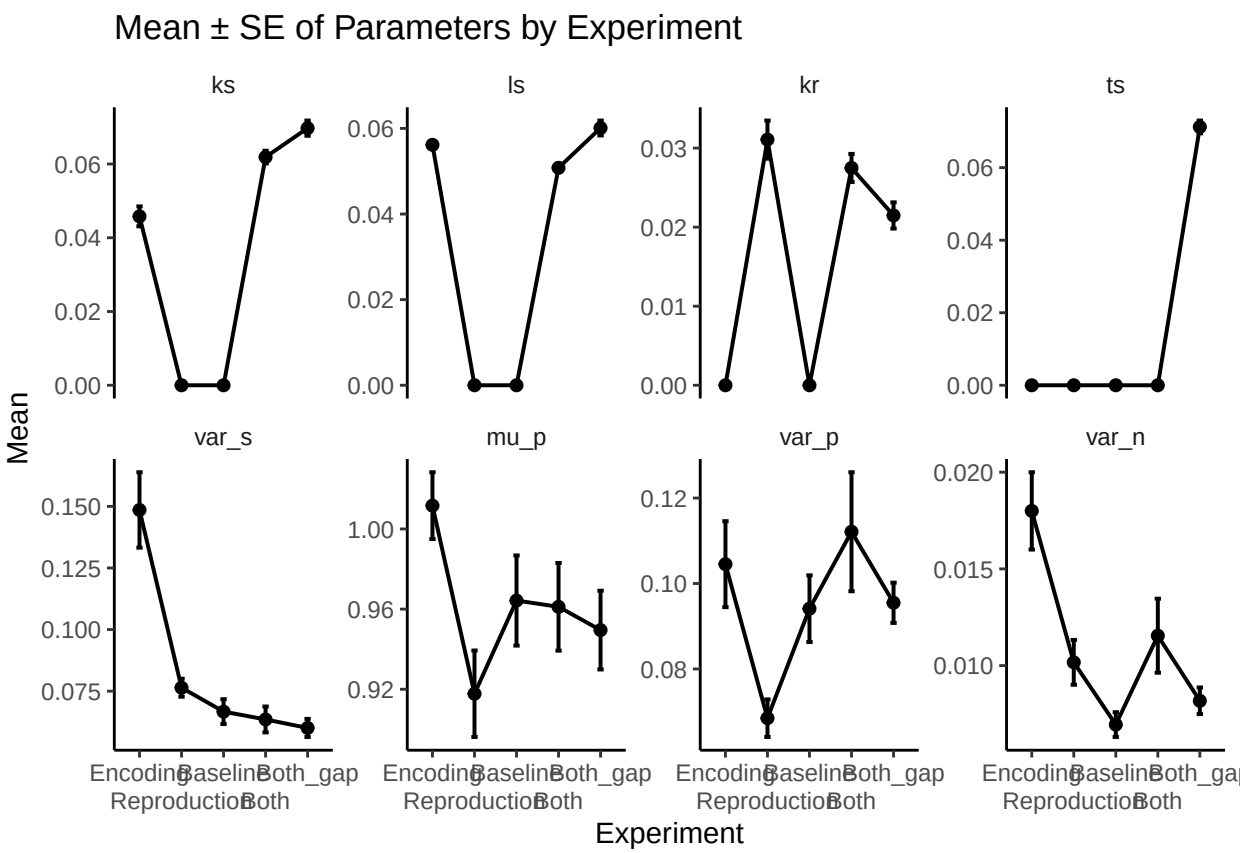
```
## New names:
## Rows: 288 Columns: 17
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Exp dbl (16): ...1, ks, ls, kr, ts, var_s, mu_log, sigma_log, var_n, mu_p,
## var_p...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

```
ParaList$Gap <- factor(ParaList$Gap, labels = c("short", "long"))
ParaList$Exp <- factor(ParaList$Exp, levels = order_exp)
ParaList$WMSize <- factor(ParaList$WMSize, labels = c("low", "medium",  "high"))
```

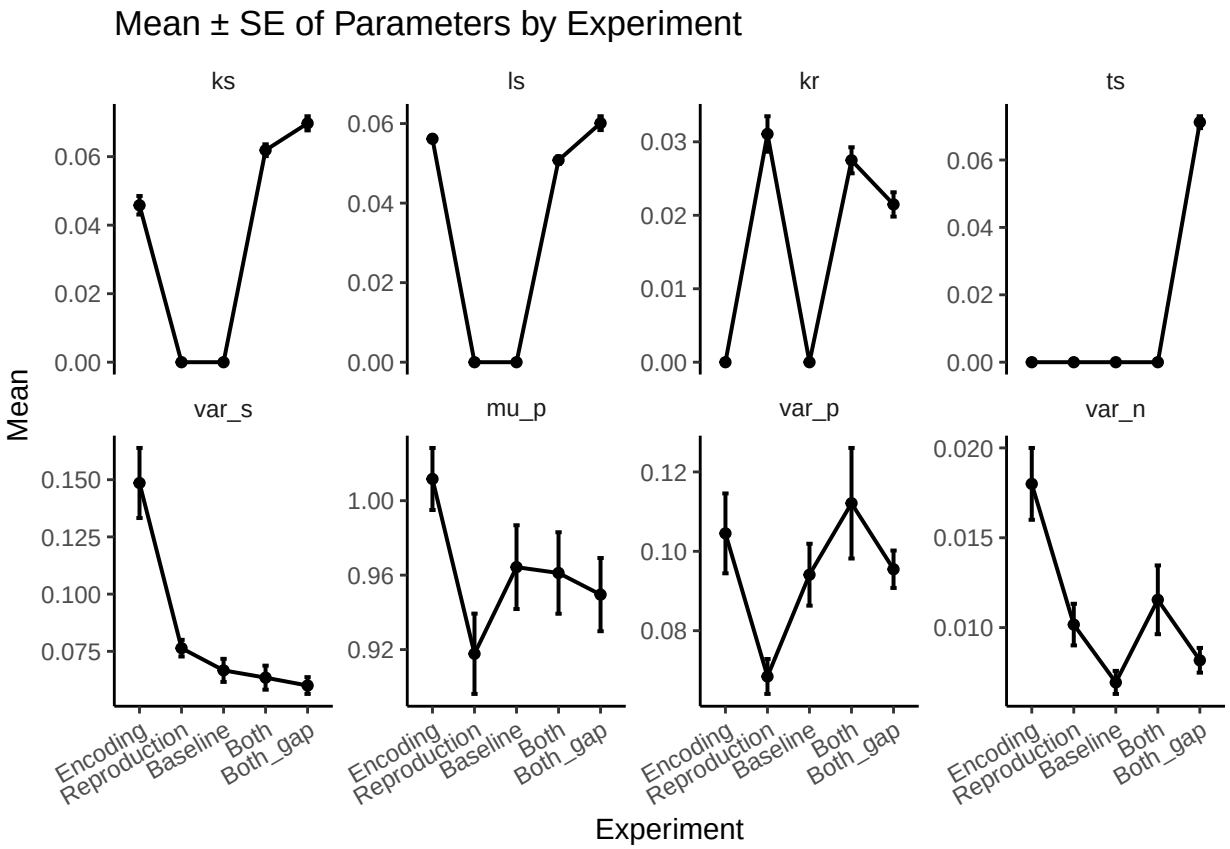


Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```r
# ---   param_labels  named list of expressions
param_labels <- list(
  ks    = expression(k[s]),
  ls    = expression(l[s]),
  kr    = expression(k[r]),
  ts    = expression(t[s]),
  var_s = expression(var[s]),
  mu_p  = expression(mu[p]),
  var_p = expression(var[p]),
  var_n = expression(var[n])
)


#
params <- c("ks", "ls", "kr", "ts", "var_s", "mu_p", "var_p", "var_n")

#          SE
df_summary <- ParaList %>%
  pivot_longer(cols = all_of(params), names_to = "Parameter", values_to = "Value") %>%
  group_by(Exp, Parameter) %>%
  summarise(
    Mean = mean(Value, na.rm = TRUE),
    n = sum(!is.na(Value)),
    SD = sd(Value, na.rm = TRUE),
    SE = SD / sqrt(n),
    .groups = "drop"
  )

#   Parameter     param_labels
df_summary$Parameter <- factor(df_summary$Parameter, levels = names(param_labels))

#
ggplot(df_summary, aes(x = Exp, y = Mean, group = 1)) +
  geom_line(color = "black", linewidth = 0.7) +
  geom_point(size = 1.8, color = "black") +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                width = 0.15, linewidth = 0.7) +
  facet_wrap(~ Parameter, scales = "free_y", ncol = 4,
             labeller = labeller(Parameter = as_labeller(param_labels))) +
  scale_x_discrete(guide = guide_axis(n.dodge = 2)) +   #   x
  theme_bw(base_size = 12) +
  theme(
    axis.text.x = element_text(angle = 30, hjust = 1),
    legend.position = "none",
    strip.text = element_text(size = 11)
  ) +
  labs(
    title = "Mean ± SE of Parameters by Experiment",
    x = "Experiment",
    y = "Mean"
  ) +
  theme_new
```




```
#
ggplot(df_summary, aes(x = Exp, y = Mean, group = Parameter)) +
  geom_line(color = "black", linewidth = 0.7) +
  geom_point(size = 1.5, color = "black") +
  geom_errorbar(aes(ymin = Mean - SE, ymax = Mean + SE),
                width = 0.15, linewidth = 0.7) +
  facet_wrap(~ Parameter, scales = "free_y", ncol = 4,
             labeller = labeller(Parameter = as_labeller(param_labels, label_parsed))) +
  theme_bw(base_size = 12) +
  theme(
    axis.text.x = element_text(angle = 30, hjust = 1),
    legend.position = "none",
    strip.text = element_text(size = 11)
  ) +
  labs(
    title ="Mean ± SE of Parameters by Experiment",
    x = "Experiment",
    y = "Mean"
  ) +
  theme_new+scale_x_discrete(guide = guide_axis(n.dodge = 1)) +
theme(axis.text.x = element_text(angle = 30, hjust = 1))
```



predicted bias

```
# Here the predicted RP was based on function predict_newdata_from_posterior

newdat_all <- read_csv(paste0(filepath, modelname, "_newdat_all.csv")) %>%
  filter(!(Exp != "Both_gap" & Gap == 2.5))
```

```
## New names:
## Rows: 67680 Columns: 12
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (1): Exp dbl (11): ...1, NSub, xnew, WMSize, Gap, mPred, sdPred, CI_low,
## CI_high, pre...
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
```

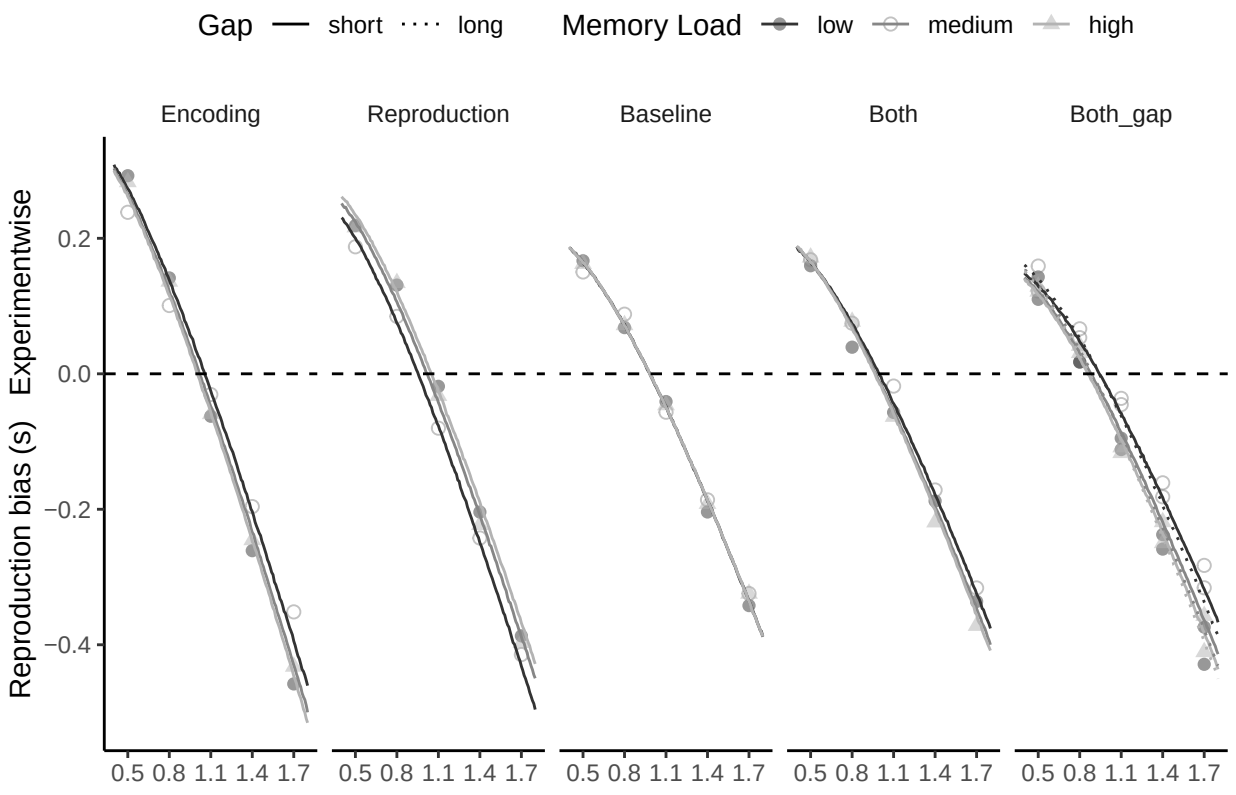
```
newdat_all$cv = newdat_all$sdPred/newdat_all$mPred
newdat_all$Exp <- factor(newdat_all$Exp, levels = order_exp)
newdat_all$Gap <- factor(newdat_all$Gap, labels = c("short", "long"))
newdat_all$WMSize <- factor(newdat_all$WMSize, labels = c("low", "medium",  "high"))
newdat_all <- newdat_all %>%
  rename(curDur = xnew)
```

```
RP_bias_bw <- ggplot(data = mdat_all%>%
                       dplyr::group_by(Exp, curDur, WMSize, Gap) %>%
                       dplyr::summarize(m_repDur_mean = mean(repDur_mean),  n= n(), se_repDur_mean = s
                     aes(x = curDur, y = m_repDur_mean - curDur,
                         group = interaction(Gap, WMSize),
```

```
                          color = as.factor(WMSize),
                          shape = as.factor(WMSize))) +
  geom_point(size = 2, alpha = 0.5) +
  geom_line(data = newdat_all %>% dplyr::group_by(Exp, curDur, WMSize, Gap) %>% dplyr::summarize(mmPred
  geom_hline(yintercept = 0, linetype = "dashed")+ scale_x_continuous(breaks=c(0.5, 0.8, 1.1, 1.4, 1.7))

  facet_grid(cols = vars(Exp)) +
  scale_color_grey(start = 0.2, end = 0.7) +
  scale_shape_manual(values = c(16, 1, 17, 2, 15, 0)) +
  scale_linetype_manual(values = c("solid", "dotted")) +
  labs(x = " ",
       y = paste0("Reproduction bias (s)   ", modelname),
       shape = "Memory Load",
       linetype = "Gap",
       color = "Memory Load") +
  theme_new +
  theme(legend.position = "top")
```

```
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
```

```
RP_bias_bw
```




```
ggsave(paste0(getwd(), "/figures/RP_bias_bw_",modelname,".png"), RP_bias_bw, width = 8, height = 4)
```

predicted CV

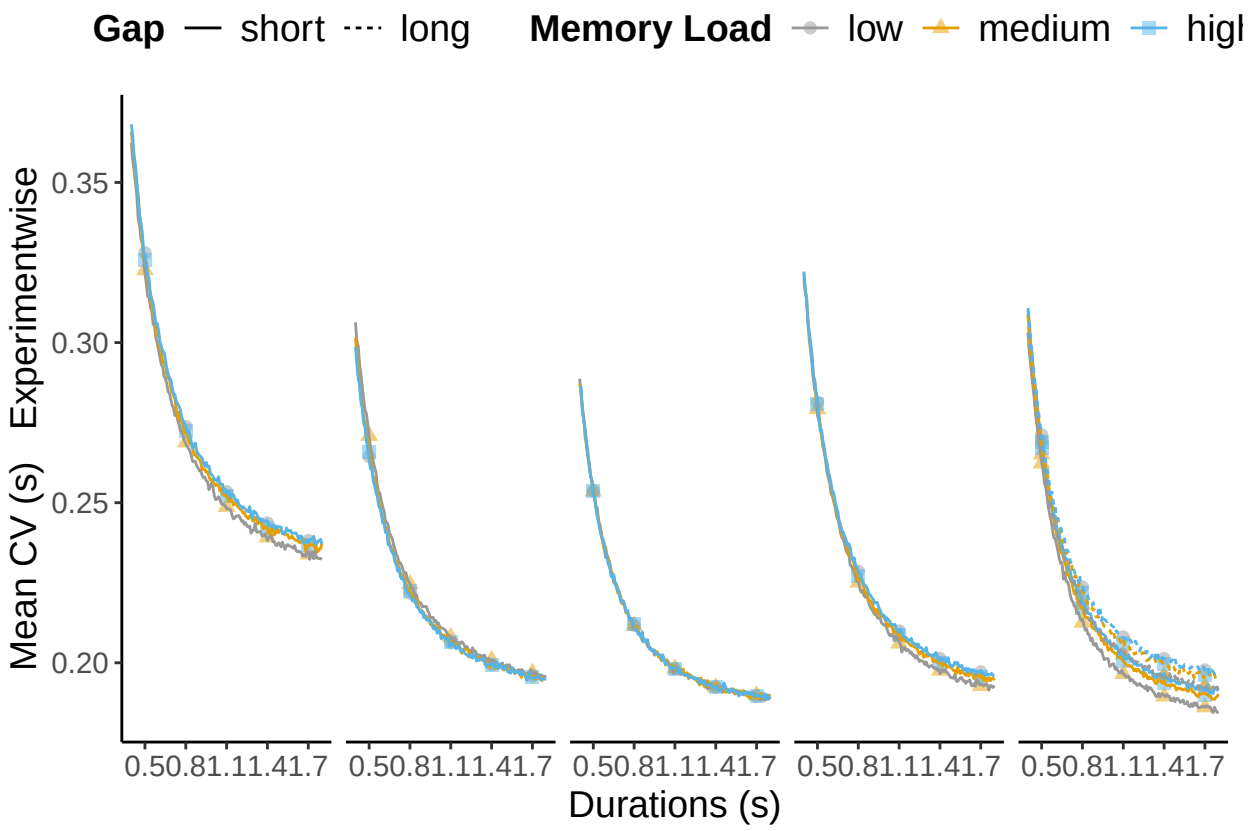
```
cv_newY <- newdat_all %>%
  group_by(Exp, curDur, WMSize, Gap) %>%
  summarize(m_cv = mean(cv)) %>%
  arrange(Exp, WMSize, Gap, curDur)
```

```
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
```

```
RP_CV <- ggplot(data= mdat_all%>%
  dplyr::group_by(Exp, curDur, WMSize, Gap) %>%
  dplyr::summarize(m_predCV = mean(predCV)),
                aes(x=curDur, y= m_predCV, group = interaction(Gap, WMSize),
                    color = as.factor(WMSize), shape = as.factor(WMSize))) +
  geom_point(size=2, alpha = 0.5)+
  geom_line(data = cv_newY, aes(x=curDur, y=m_cv, group = interaction(WMSize, Gap),
                linetype = Gap, color=WMSize)) +
  scale_x_continuous(breaks=c(0.5, 0.8, 1.1, 1.4, 1.7)) +
  facet_grid(~Exp) +
  labs(x="Durations (s)", y=paste0("Mean CV (s)", "   ", modelname), shape="Memory Load", linetype = "Ga
       color = "Memory Load") + theme_new + colorSet3 +
  theme(legend.position = "top",
        legend.title = element_text(size = 14, face = "bold"),
        legend.text = element_text(size = 14),
        axis.title = element_text(size = 14),
        axis.text = element_text(size = 11),
        strip.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5))
```

```
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
```

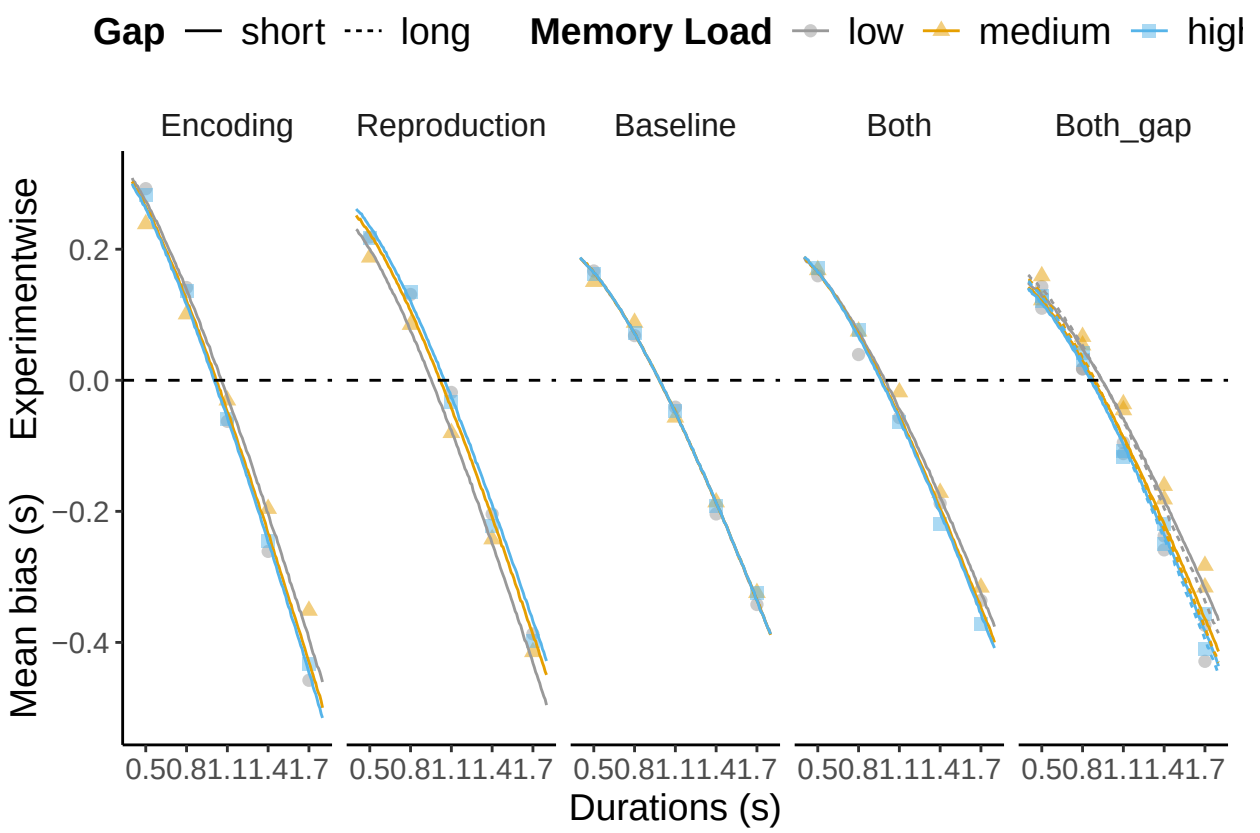
```
RP_CV
```




```
# plot color plots for Appendix
RP_bias <- ggplot(data = mdat_all%>%
                        dplyr::group_by(Exp, curDur, WMSize, Gap) %>%dplyr::summarize(m_repDur = mean(
                      aes(x = curDur, y = m_repDur - curDur, group = interaction(Gap, WMSize),
                          color=as.factor(WMSize), shape = as.factor(WMSize))) +
  geom_point(size=2, alpha = 0.5)+
  geom_line(data= newdat_all %>% dplyr::group_by(Exp, curDur, WMSize, Gap) %>% dplyr::summarize(mmPred =
                linetype = Gap, color=WMSize)) +
  scale_x_continuous(breaks=c(0.5, 0.8, 1.1, 1.4, 1.7)) +
  geom_hline(yintercept = 0, linetype='dashed')+
  facet_grid(cols = vars(Exp), labeller = labeller(Exp = Exp.labs.2lines)) +
  labs(x="Durations (s)", y=paste0("Mean bias (s)     ", modelname), shape ="Memory Load", linetype = "
       color = "Memory Load")+
  theme_new + colorSet3 + theme(legend.position = "top",
        legend.title = element_text(size = 14, face = "bold"),
        legend.text = element_text(size = 14),
        axis.title = element_text(size = 14),
        axis.text = element_text(size = 11),
        strip.text.x = element_text(size = 12),
        plot.title = element_text(hjust = 0.5))
```

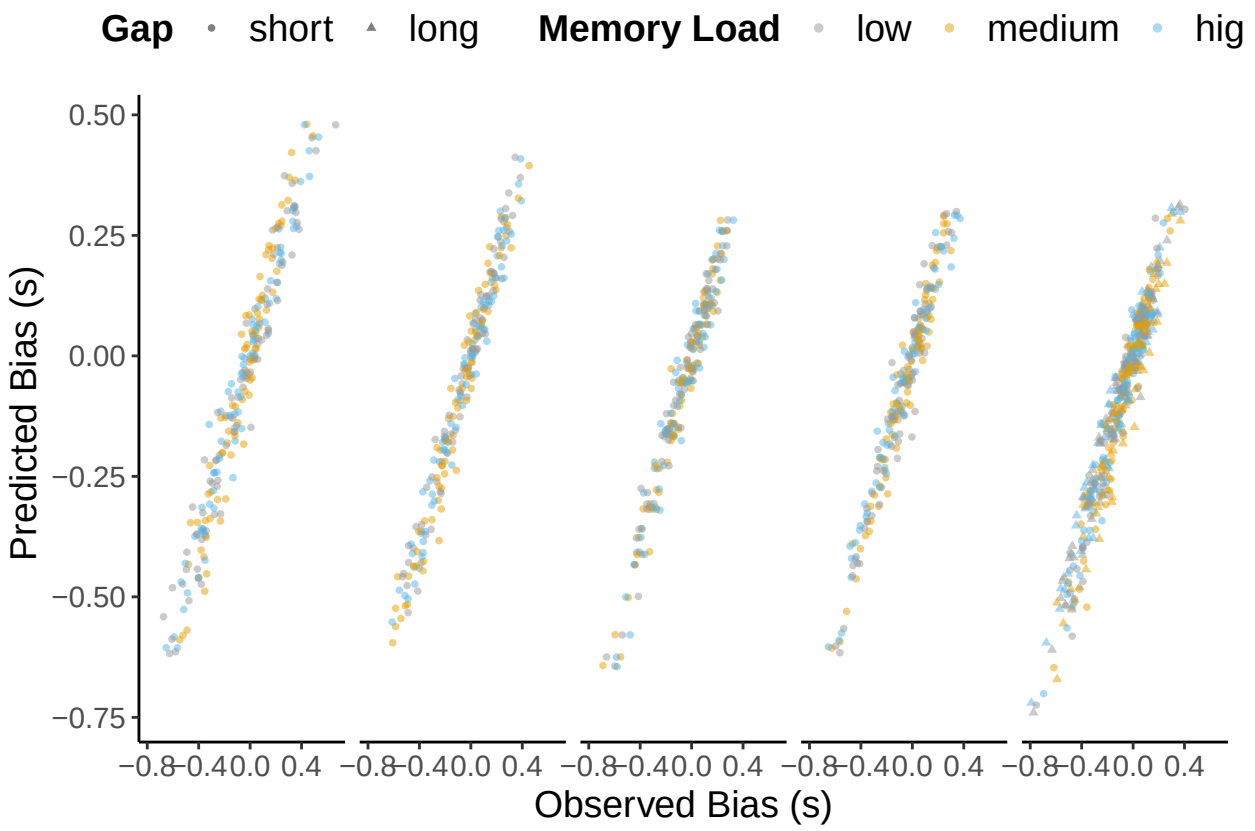
```
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
## `summarise()` has grouped output by 'Exp', 'curDur', 'WMSize'. You can override
## using the `.groups` argument.
```

```
RP_bias
```

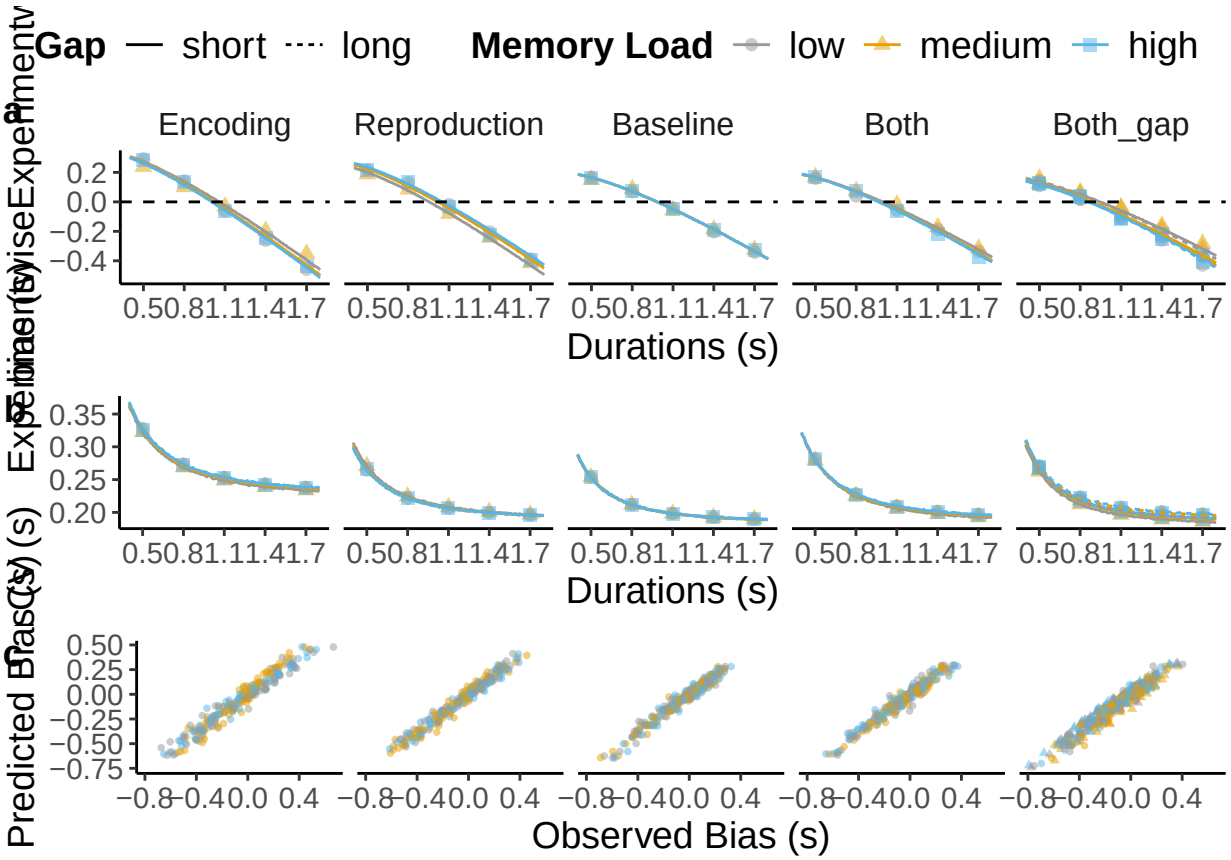


correlations of bias

```
# plot correlation coefficients r of observed and predicted bias
plt_corr_obs_pred <- ggplot(data = mdat_all,
                        aes(x = repErr, y = predErr, color = as.factor(WMSize),
                            shape = as.factor(Gap))) +
  geom_point(size=1, alpha = 0.5) +
  facet_grid(~Exp) +
  labs(x="Observed Bias (s)", y=" Predicted Bias (s)", shape="Gap",
       color = "Memory Load") + theme_new + colorSet3 +
  theme(legend.position = "top",
        legend.title = element_text(size = 14, face = "bold"),
        legend.text = element_text(size = 14),
        axis.title = element_text(size = 14),
        axis.text = element_text(size = 11),
        strip.text.x = element_blank(),
        plot.title = element_text(hjust = 0.5))
plt_corr_obs_pred
```




```
## combine predicted bias and cv
fig_bias_cv <- ggarrange(RP_bias, RP_CV, plt_corr_obs_pred, common.legend = TRUE,
                         ncol=1, nrow=3, heights = c(1.1, 0.9, 0.9),
                         labels = c("a", "b", "c"))
ggsave(paste0(getwd(), "/figures/fig_bias_cv_pred_", modelname,".png"),
       fig_bias_cv, width = 9, height = 9)
fig_bias_cv
```



weight of prior

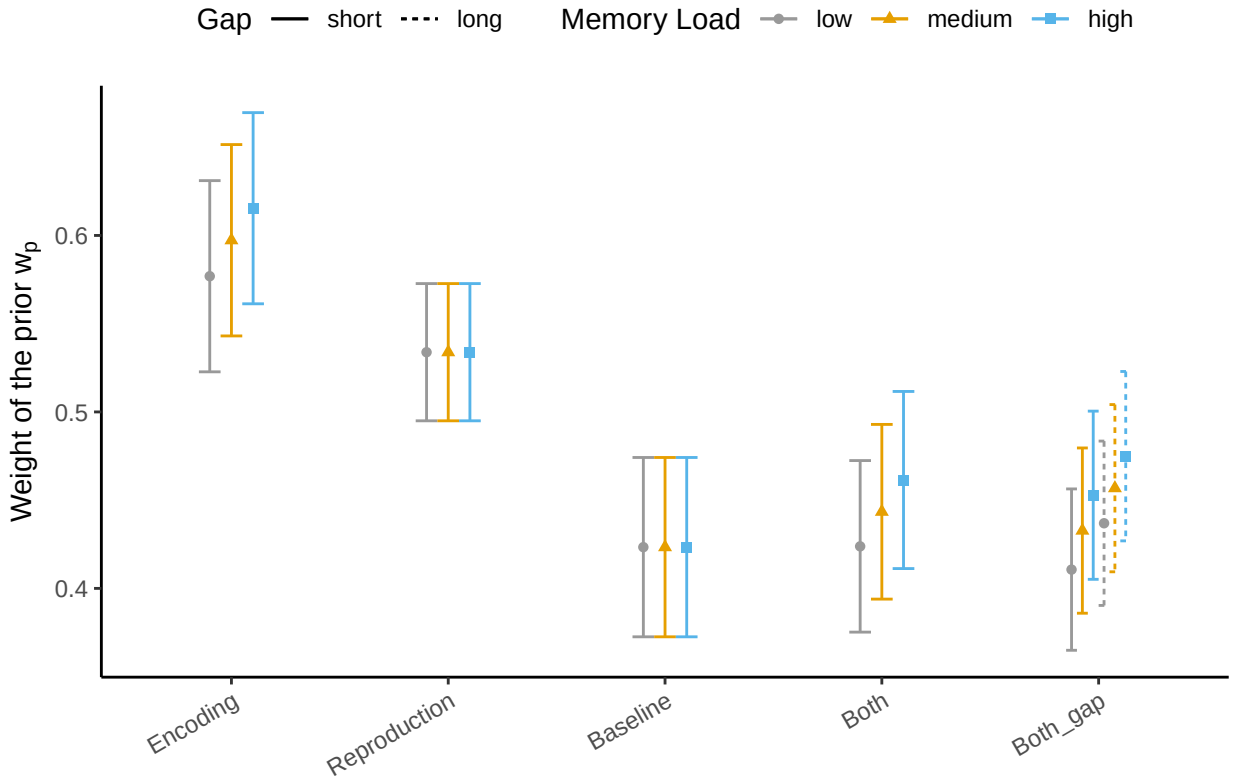
```
plt_wp_Gap <- ggplot(
  data = ParaList %>%
    dplyr::group_by(Exp, WMSize, Gap) %>%
    dplyr::summarise(
      m_wp = mean(w_p),
      n = n(),
      se_wp = sd(w_p) / sqrt(n - 1)
    ),
  aes(
    x = Exp,
    y = m_wp,
    ymin = m_wp - se_wp,
    ymax = m_wp + se_wp,
    group = interaction(Exp, WMSize, Gap),
    color = factor(WMSize),          # FIX 1
    shape = factor(WMSize),          # FIX 2
    linetype = factor(Gap)           # FIX 3
  )
) +
  geom_line(position = position_dodge(width = 0.3)) +
  geom_point(position = position_dodge(width = 0.3)) +
  geom_errorbar(width = .3, position = position_dodge(width = .3)) +
  colorSet5 +
  labs(
```

```
    x = "",
    y = TeX("Weight of the prior $w_p$"),
    color = "Memory Load",
    shape = "Memory Load",
    linetype = "Gap"
  ) +
  theme_new+scale_x_discrete(guide = guide_axis(n.dodge = 1)) +
theme(axis.text.x = element_text(angle = 30, hjust = 1))+theme(legend.position = "top",
      legend.direction = "horizontal")
```

```
## `summarise()` has grouped output by 'Exp', 'WMSize'. You can override using the
## `.groups` argument.
```

```
plt_wp_Gap
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```




```
ggsave(paste0(getwd(), "/figures/plt_wp_",modelname,".png"), plt_wp_Gap, width = 7, height = 4)
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```

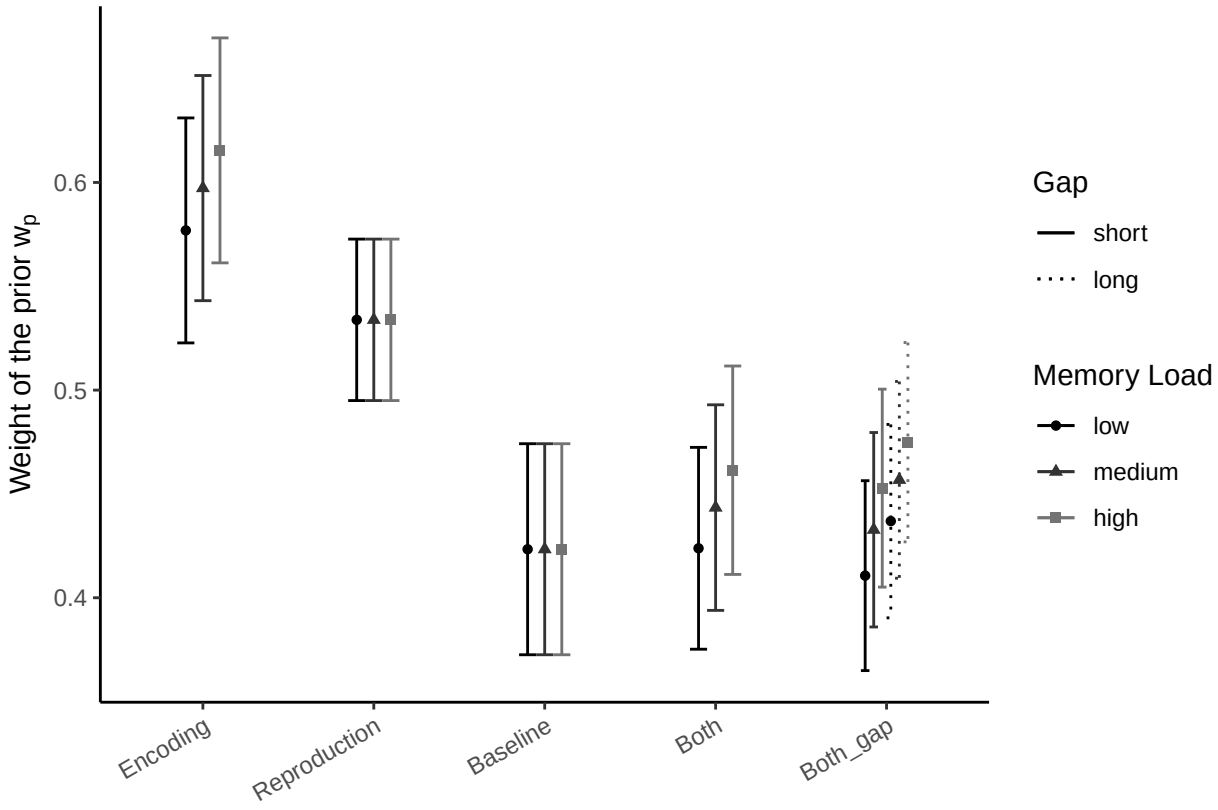
```
# ---     ---
colorSet5_bw <- list(
  scale_color_manual(values = c("black", "grey20", "grey45")),
  scale_linetype_manual(values = c("solid", "dotted"))
)

plt_wp_Gap_wb <- ggplot(
  data = ParaList %>%
```

```
    dplyr::group_by(Exp, WMSize, Gap) %>%
    dplyr::summarise(
      m_wp = mean(w_p),
      n = n(),
      se_wp = sd(w_p) / sqrt(n - 1),
      .groups = "drop"
    ),
  aes(
    x = Exp,
    y = m_wp,
    ymin = m_wp - se_wp,
    ymax = m_wp + se_wp,
    group = interaction(Exp, WMSize, Gap),
    color = factor(WMSize),
    shape = factor(WMSize),
    linetype = factor(Gap)
  )
) +
  geom_line(position = position_dodge(width = 0.3)) +
  geom_point(position = position_dodge(width = 0.3)) +
  geom_errorbar(width = .3, position = position_dodge(width = .3)) +
  colorSet5_bw +        #
  scale_x_discrete(labels = exp_labels) +
  labs(
    x = "",
    y = TeX("Weight of the prior $w_p$"),
    color = "Memory Load",
    shape = "Memory Load",
    linetype = "Gap"
  ) +
  theme_new+ theme(legend.position="top")+
  theme_new+scale_x_discrete(guide = guide_axis(n.dodge = 1)) +
theme(axis.text.x = element_text(angle = 30, hjust = 1))
```

```
## Scale for x is already present.
## Adding another scale for x, which will replace the existing scale.
```

```
plt_wp_Gap_wb
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```



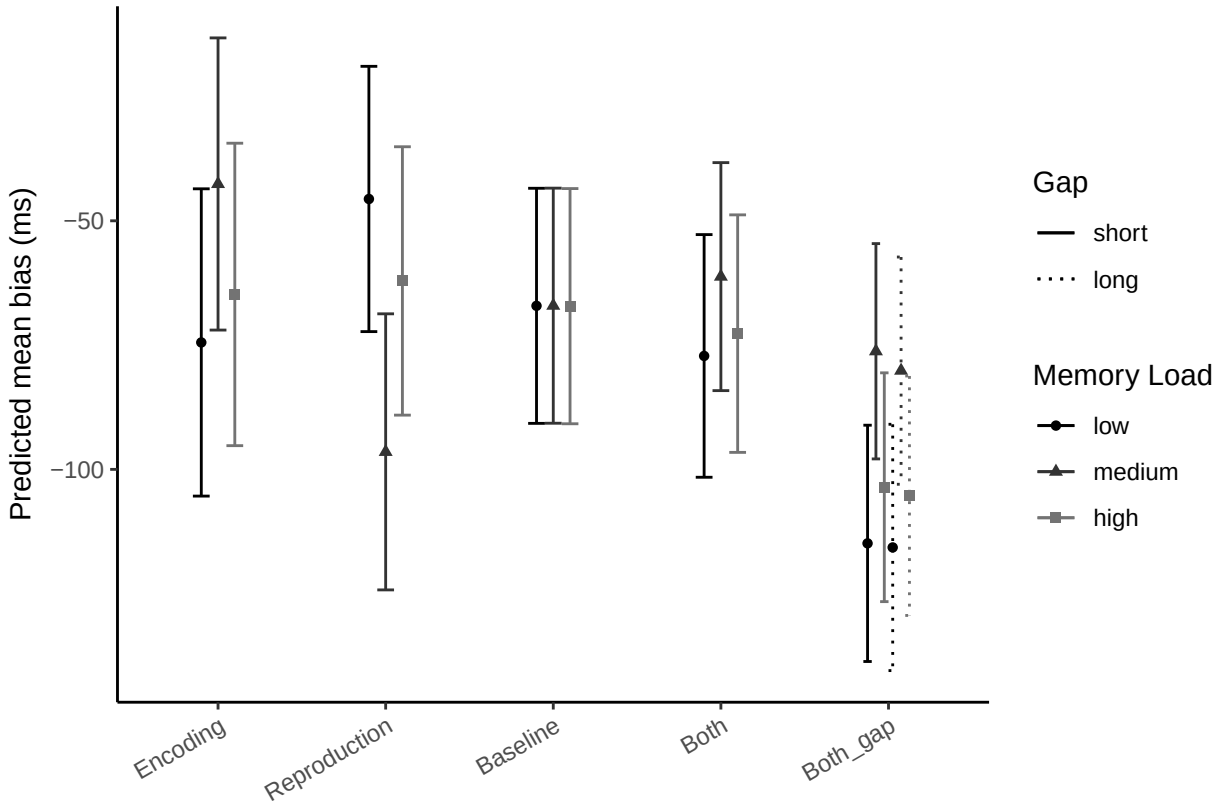
predicted mean bias

```
plt_pred_Bias_bw <-ggplot(data = mdat_all %>%dplyr::group_by(Exp, WMSize, Gap) %>% dplyr::summarise(mmP
  geom_line(stat = "identity", position = position_dodge(width = 0.3))+
  geom_point(stat = "identity",position = position_dodge(width = 0.3))+
  geom_errorbar(width=.3,  position = position_dodge(width = .3)) +
  scale_x_discrete(labels = exp_labels) +
  colorSet5_bw+
  labs(x = "", y = TeX("Predicted mean bias (ms)"), color = 'Memory Load', shape = 'Memory Load', linet
  theme_new+scale_x_discrete(guide = guide_axis(n.dodge = 1)) +
theme(axis.text.x = element_text(angle = 30, hjust = 1))
```

```
## `summarise()` has grouped output by 'Exp', 'WMSize'. You can override using the
## `.groups` argument.
## Scale for x is already present. Adding another scale for x, which will replace
## the existing scale.
```

```
plt_pred_Bias_bw
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```

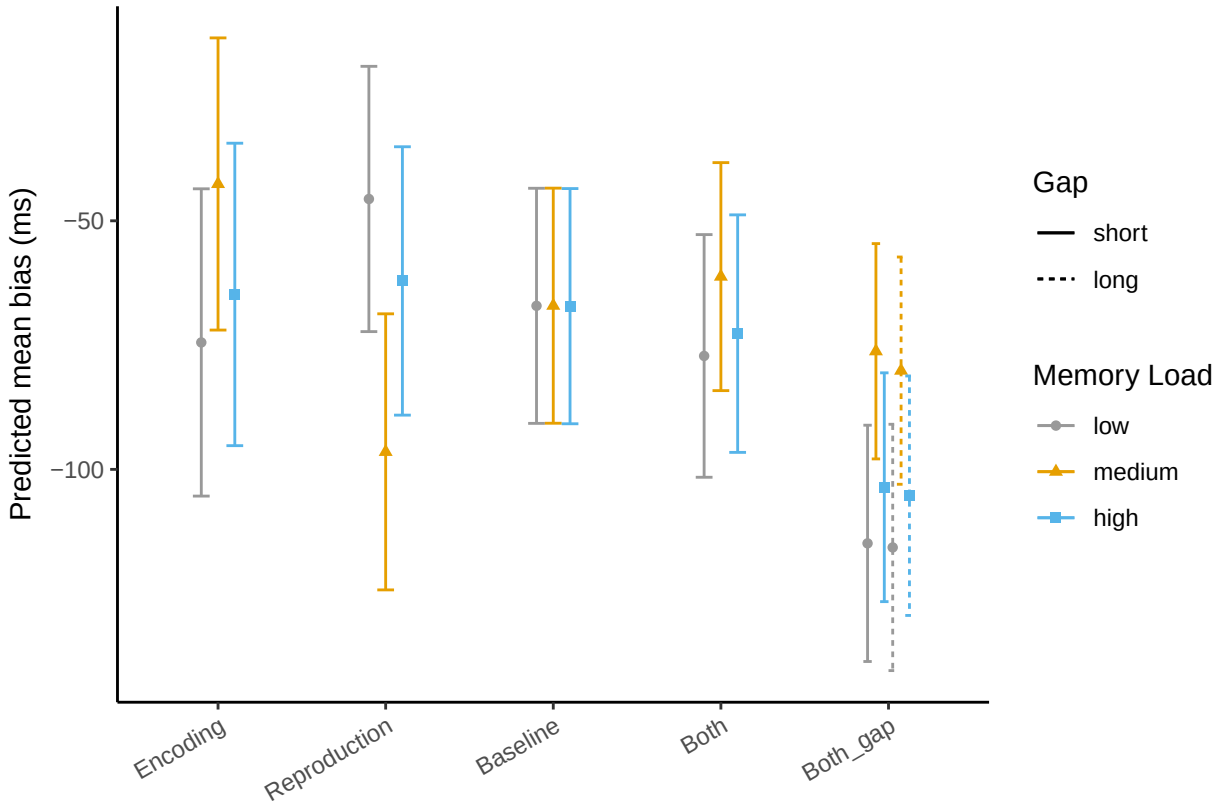



```
plt_pred_Bias <-ggplot(data = mdat_all %>%dplyr::group_by(Exp, WMSize, Gap) %>% dplyr::summarise(mmPred
  geom_line(stat = "identity", position = position_dodge(width = 0.3))+
  geom_point(stat = "identity",position = position_dodge(width = 0.3))+
  geom_errorbar(width=.3,  position = position_dodge(width = .3)) +
  scale_x_discrete(labels = exp_labels) +
  colorSet5+
  labs(x = "", y = TeX("Predicted mean bias (ms)"), color = 'Memory Load', shape = 'Memory Load', linety
  theme_new+scale_x_discrete(guide = guide_axis(n.dodge = 1)) +
theme(axis.text.x = element_text(angle = 30, hjust = 1))
```

```
## `summarise()` has grouped output by 'Exp', 'WMSize'. You can override using the
## `.groups` argument.
## Scale for x is already present. Adding another scale for x, which will replace
## the existing scale.
```

```
plt_pred_Bias
```

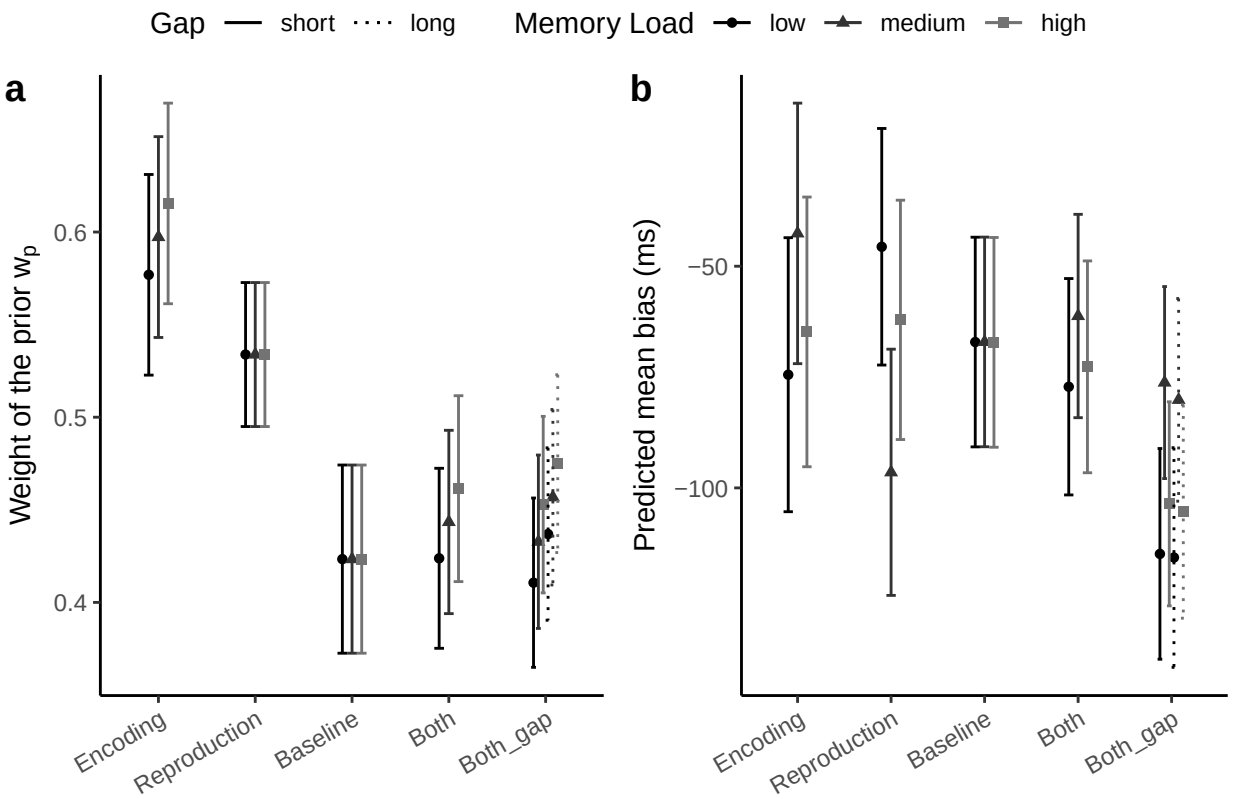
```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```




```
fig7_wb<-ggarrange(plt_wp_Gap_wb, plt_pred_Bias_bw, common.legend = TRUE, ncol=2, nrow=1,  labels = c("
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```

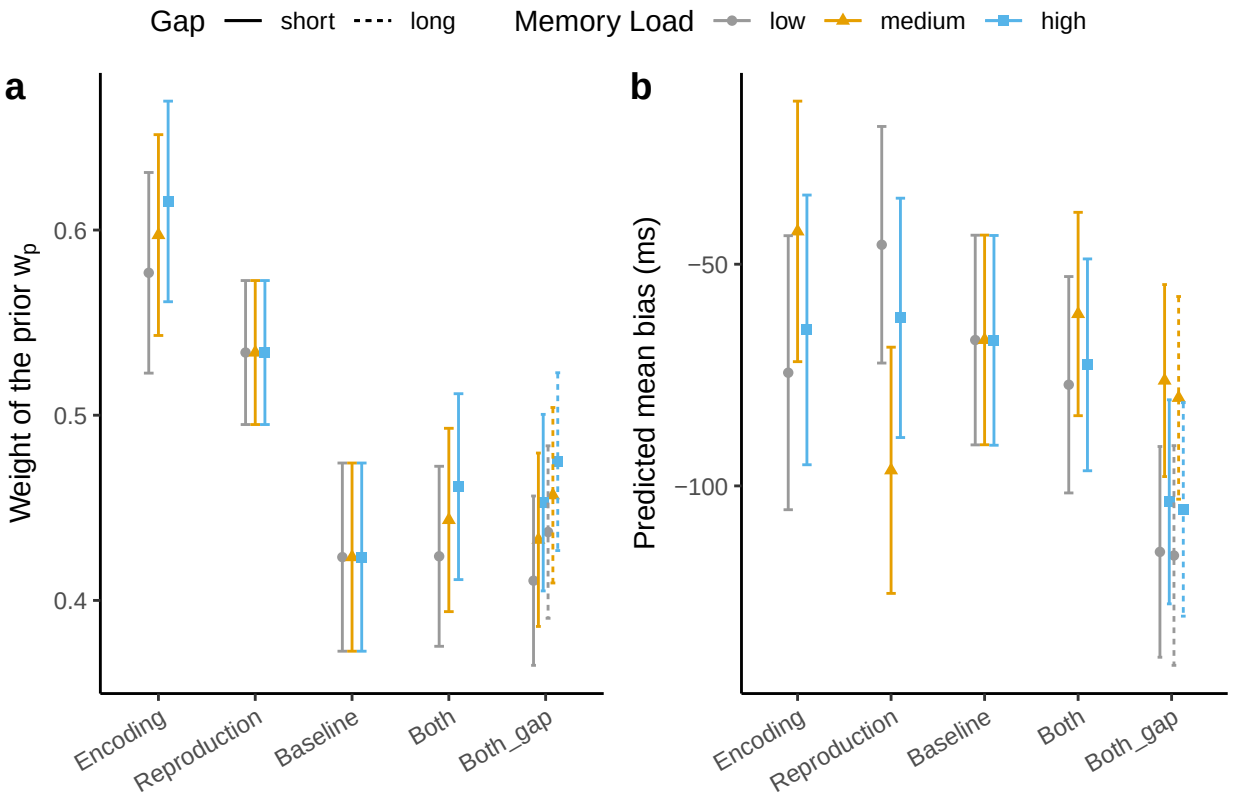
```
ggsave(paste0(getwd(), "/figures/fig7_wb_",modelname,".png"), fig7_wb, width = 7, height = 4)
fig7_wb
```




```
fig7<-ggarrange(plt_wp_Gap, plt_pred_Bias, common.legend = TRUE, ncol=2, nrow=1,  labels = c("a", "b"))
```

```
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
## `geom_line()`: Each group consists of only one observation.
## i Do you need to adjust the group aesthetic?
```

```
ggsave(paste0(getwd(), "/figures/fig7_",modelname,".png"), fig7, width = 7, height = 4)
fig7
```


Gap
short
long
Memory Load
low
medium
high
a
Weight of the prior w_p
0.6
0.5
0.4
Encoding
Reproduction
Baseline
Both
Both_gap
b
Predicted mean bias (ms)
−50
−100
Encoding
Reproduction
Baseline
Both
Both_gap