Results of Logarithmic encoding model

Xiuna Zhu

24/10/2022

load packages and functions

```
source('mytheme.R')
# model version
modelversion = 'gap_log_rstan'
rstanmodelPath = 'modelrlt'
modelPath = paste0(rstanmodelPath, '/models/', modelversion)
```

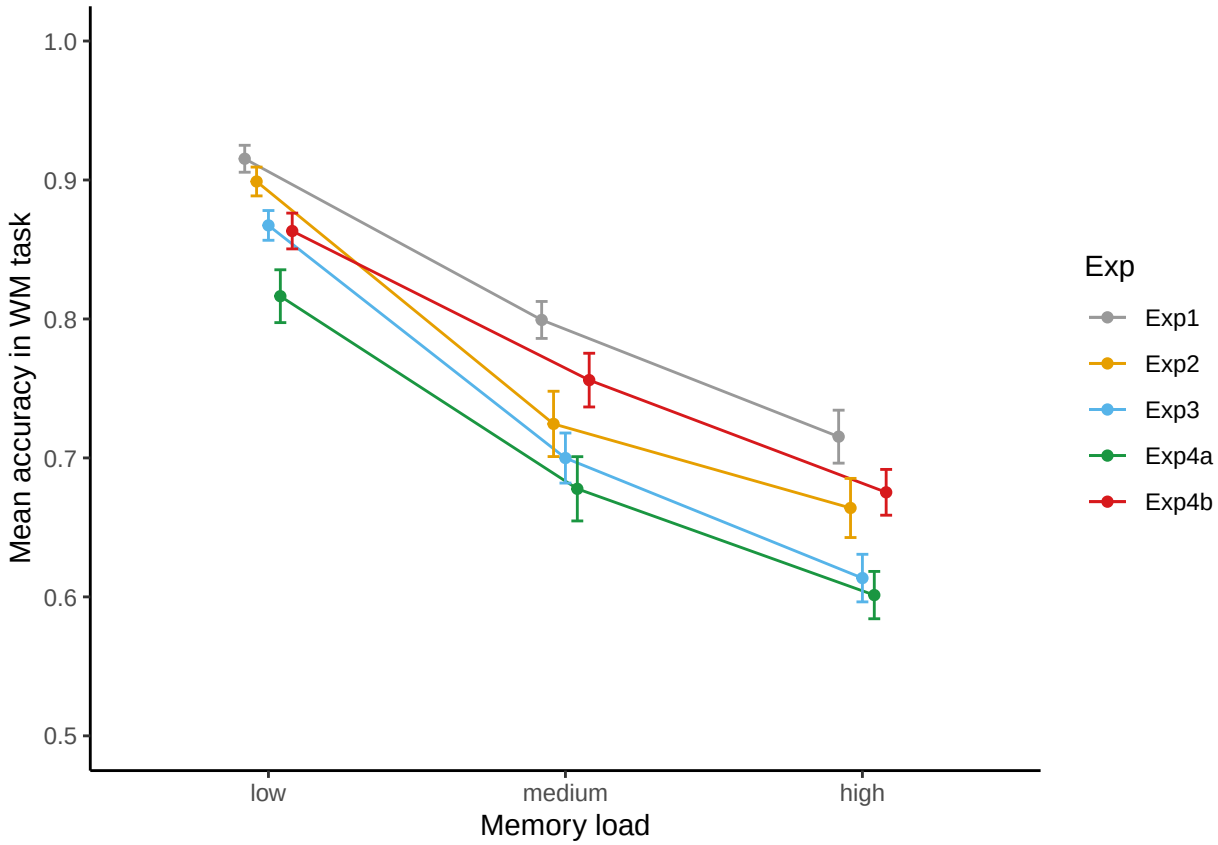
Merge the model Result data

1 load data

```
AllExpData = read.csv(paste0("../data/AllValidData.csv"))
dur <- sort(unique(AllExpData$curDur))

AllExpData$WMSize <- factor(AllExpData$WMSize, labels = c("low", "medium",  "high"))
# 1: 500ms,  2: 2500, 3 : 2000ms the mean reaction time of WM test
AllExpData$gap <- factor(AllExpData$gap,  labels = c('short','long', 'not sure'))
AllExpData[which(AllExpData$Exp == 'Exp4'),"Exp"] = "Exp4a"
AllExpData[which(AllExpData$Exp == 'Exp5'),"Exp"] = "Exp4b"
```

2 Corrct rate

```
WMCrr2 <- ggplot(meanForPlot, aes(WMSize, mean_WMCrr, ymin = mean_WMCrr - se_WMCrr, ymax = mean_WMCrr +
                                  group =Exp, color = Exp, fill = Exp))+
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  position = position_dodge(width = 0.2)) +
  coord_cartesian(ylim = c(0.5, 1)) +
  colorSet5+
  labs(x = "Memory load", y = "Mean accuracy in WM task") +
  theme_new
WMCrr2
```




```
ggsave(paste0(getwd(), "/figures/WMCrr2.png"), WMCrr2, width = 4, height = 4)
```

```
### generate WM correct rates
AllExpData$WMCrr <- AllExpData$TPresent == AllExpData$WMRP
m_wmp<- dplyr::group_by(AllExpData, Exp, WMSize, NSub) %>%
  dplyr::summarize(m_WMCrr = mean(WMCrr), n =n(), se_WMCrr = sd(WMCrr)/sqrt(n-1))
```

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

```
ezANOVA(data = WMCrr, dv=m_WMCrr, wid=NSub, within = .(WMSize), between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.

## Warning: Converting "Exp" to factor for ANOVA.

## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.

## $ANOVA
##       Effect DFn DFd          F            p p<.05        ges
## 2        Exp   4  75   8.816704 6.868286e-06     * 0.25356151
## 3     WMSize   2 150 526.088822 1.618810e-68     * 0.66068954
## 4 Exp:WMSize   8 150   2.434296 1.672504e-02     * 0.03478551
##
## $`Mauchly's Test for Sphericity`
##       Effect        W          p p<.05
## 3     WMSize 0.907248 0.02728116     *
## 4 Exp:WMSize 0.907248 0.02728116     *
```

```
##
## $`Sphericity Corrections`
##       Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 3     WMSize 0.9151207 6.137253e-63         * 0.9368512 2.288774e-64         *
## 4 Exp:WMSize 0.9151207 2.034718e-02         * 0.9368512 1.934824e-02         *
```

3 comparison the results between Exp.4a and Exp.4b

3.1 correct rate

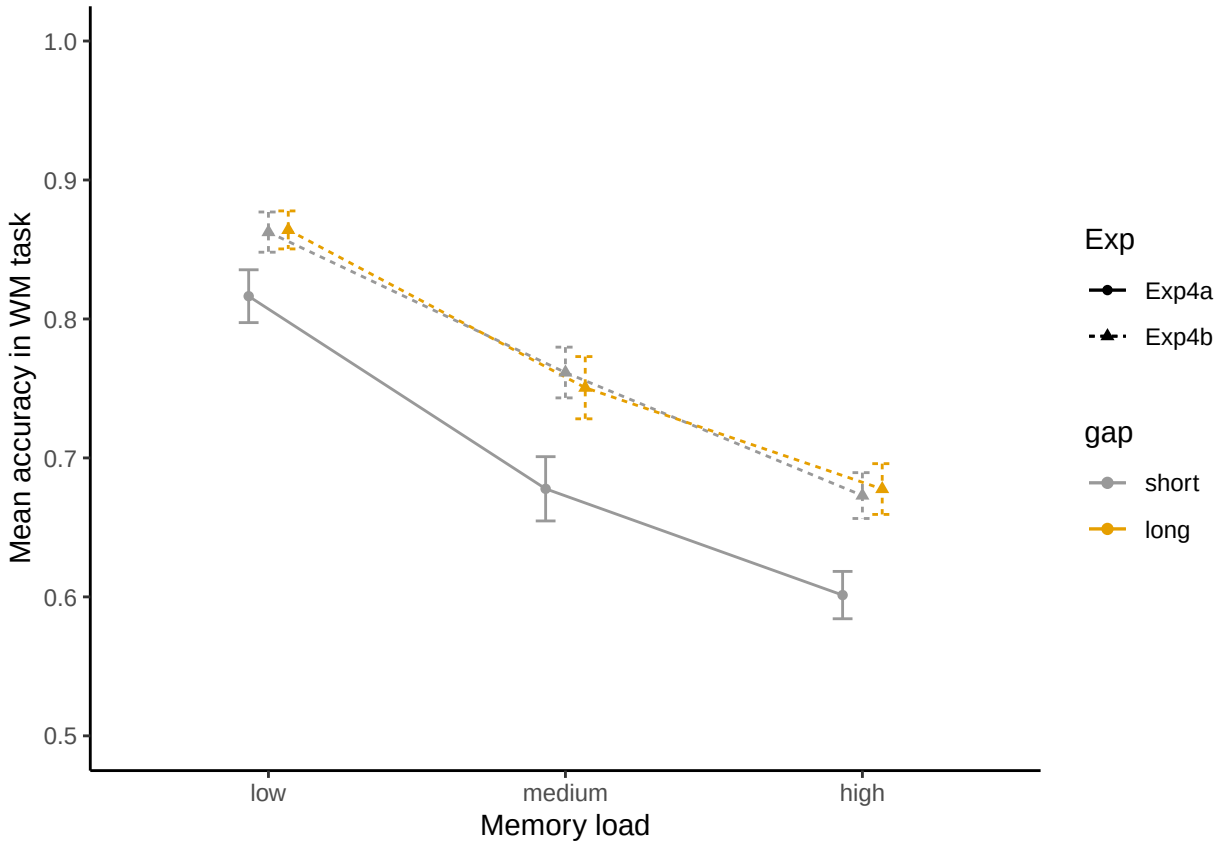
```
#plot WM correct rates
dplyr::group_by(AllExpData%>%filter(Exp %in% c('Exp4a', 'Exp4b')), Exp, WMSize, NSub, gap) %>%
  dplyr::summarize(m_WMCrr = mean(WMCrr), n = n(), se_WMCrr = sd(WMCrr)/sqrt(n-1)) -> correctrate_gap
```

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

```
write.csv(correctrate_gap, paste0(modelPath, '/rlt/correctrate_gap.csv'))



correctrate_gap%>%
  dplyr::group_by(Exp, WMSize, gap)%>%
  dplyr::summarize( n = n(),
            mean_WMCrr = mean(m_WMCrr), se_WMCrr = sd(m_WMCrr)/sqrt(n-1) ) -> meanForPlot_gap
```

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

```
WMCrr_gap <- ggplot(meanForPlot_gap, aes(WMSize, mean_WMCrr, ymin = mean_WMCrr - se_WMCrr, ymax = mean_W
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  position = position_dodge(width = 0.2)) +
  coord_cartesian(ylim = c(0.5, 1)) +
  colorSet5+
  labs(x = "Memory load", y = "Mean accuracy in WM task") +theme_new
WMCrr_gap
```




```
ggsave(paste0(getwd(), "/figures/WMCrr_gap.png"), WMCrr_gap, width = 4, height = 4)
```

3.2 observed RP

```
AllExpData%>% filter(Exp %in% c('Exp4a', 'Exp4b')) %>%
  dplyr::group_by(Exp, curDur, WMSize, NSub, gap) %>%
  dplyr::summarize(n = n(),
                   m_repDur = mean(repDur),
                   se_repDur = sd(repDur)/sqrt(n-1)) ->RP_obs_gap_bias
```

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

```
RP_obs_gap_bias$rep_bias = RP_obs_gap_bias$m_repDur - RP_obs_gap_bias$curDur


ezANOVA(data = RP_obs_gap_bias%>%filter(Exp =='Exp4b'), dv= rep_bias, wid=NSub, within= .(curDur, gap, W
```

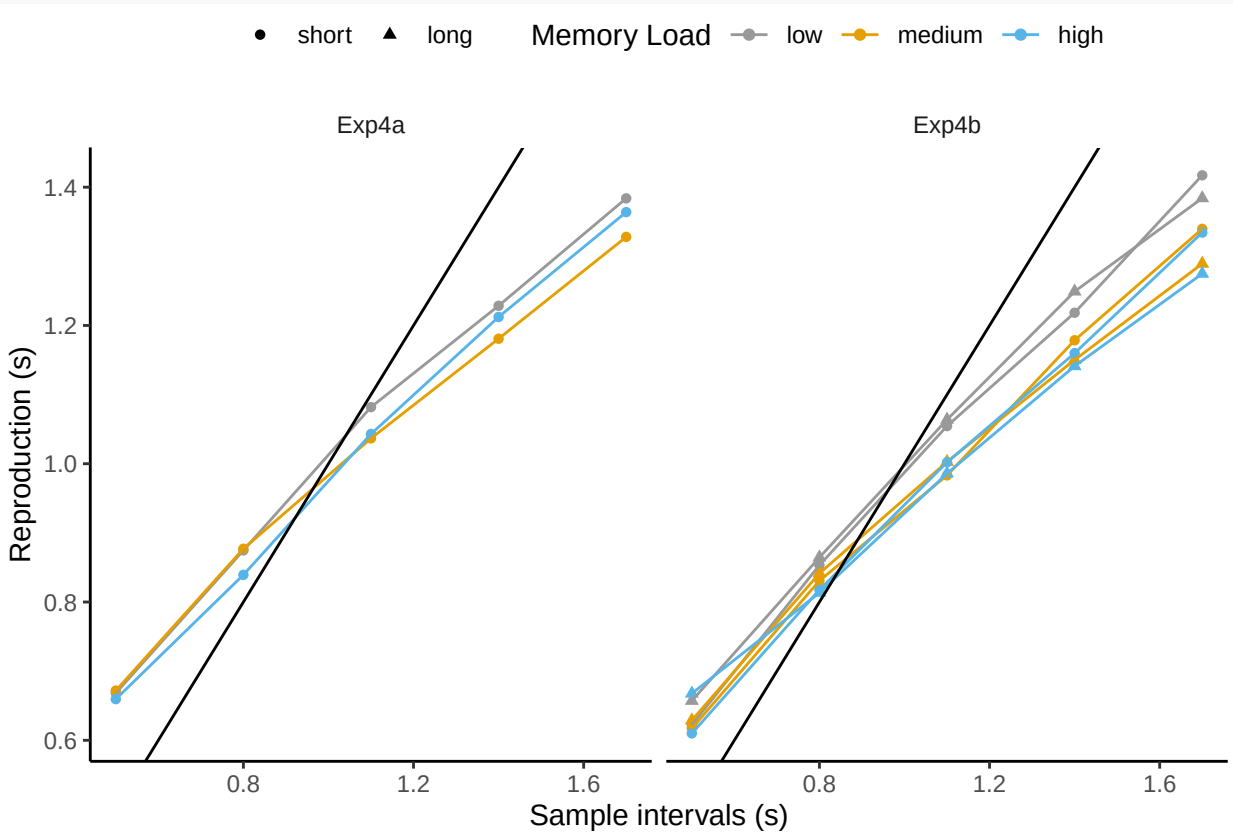
```
## Warning: Converting "NSub" to factor for ANOVA.

## Warning: "curDur" will be treated as numeric.

## Warning: You have removed one or more levels from variable "gap". Refactoring
## for ANOVA.

## Warning: There is at least one numeric within variable, therefore aov() will be
## used for computation and no assumption checks will be obtained.

## $ANOVA
```

```
##              Effect DFn DFd           F            p p<.05          ges
## 1            curDur   1  15 96.33440029 6.384821e-08     * 0.8176127533
## 2               gap   1  15  0.03549505 8.530873e-01       0.0001018033
## 3            WMSize   2  30  9.48732989 6.415750e-04     * 0.0895670700
## 4        curDur:gap   1  15 16.35270907 1.060464e-03     * 0.0220610259
## 5     curDur:WMSize   2  30  9.34453850 7.003993e-04     * 0.0280287517
## 6        gap:WMSize   2  30  2.51037940 9.816022e-02       0.0027254302
## 7 curDur:gap:WMSize   2  30  1.63535038 2.117823e-01       0.0021874657
```

```
RP_obs_gap  <- ggplot( data = RP_obs_gap_bias%>%
  dplyr::group_by(Exp, curDur, WMSize, gap) %>%
  dplyr::summarize(m_m_repDur = mean(m_repDur),
                   m_se_repDur = mean(se_repDur)), aes(x = curDur, y = m_m_repDur,  color=as.factor(WMS
  geom_point()+
  geom_line()+
  geom_abline(slope=1, intercept=0)+
  facet_grid(cols = vars(Exp)) +
  labs(x="Sample intervals (s)", y="Reproduction (s)", shape=" ", color = "Memory Load")+
  theme_new+colorSet3+
   scale_x_continuous(breaks=seq(0, 1.6, 0.4))+ theme(legend.position="top")
```

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

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/RP_obs.png"), RP_obs_gap, width = 6, height = 4)

RP_obs_gap
```



4 load parameter estimation

4.1 check model parameters

```
### Average Parameters
mm_Baypar <- dplyr::group_by(Bayparlist, Exp) %>%
  dplyr::summarize( m_sig_s2 = mean(sig_s2),
                    m_sig_pr2_log = mean(sig_pr2_log),
                    m_ks= mean(ks),
                    m_kr = mean(kr),
                    m_ls = mean(ls),
                    m_ts = mean(ts),
                    m_mu_pr = mean(mu_pr),
                    m_sig_pr2 = mean(sig_pr2),
                    m_mu_pr_log= mean(mu_pr_log),
                    m_sig_mn2 = mean(sig_mn2),
                    n= n(),
                    se_sig_s2 = sd(sig_s2)/sqrt(n-1),
                    se_sig_mn2 = sd(sig_mn2)/sqrt(n-1),
                    se_sig_pr2_log = sd(sig_pr2_log)/sqrt(n-1),
                    se_ks = sd(ks)/sqrt(n-1),
                    se_kr = sd(kr)/sqrt(n-1),
                    se_ls =sd(ls)/sqrt(n-1),
                    se_mu_pr_log = sd(mu_pr_log)/sqrt(n-1))
mm_Baypar
```

```
## # A tibble: 5 x 19
##   Exp   m_sig_s2 m_sig_pr2_log  m_ks   m_kr   m_ls    m_ts m_mu_pr m_sig_pr2
##   <fct>    <dbl>         <dbl> <dbl>  <dbl>  <dbl>   <dbl>   <dbl>     <dbl>
## 1 Exp1    0.0341        0.139  0      0      0      0.00545   0.966    0.107
## 2 Exp2    0.0801        0.119  0.120 0      0.114  0.0125    1.04     0.0322
## 3 Exp3    0.0388        0.0940 0      0.0318 0      0.00513   0.910    0.0155
## 4 Exp4a   0.0302        0.153  0.421 0.224  0.0141 0.00483   0.982    0.0661
## 5 Exp4b   0.0221        0.104  0.408 0.205  0.0171 0.00484   0.985    0.0143
## # ... with 10 more variables: m_mu_pr_log <dbl>, m_sig_mn2 <dbl>, n <int>,
## #   se_sig_s2 <dbl>, se_sig_mn2 <dbl>, se_sig_pr2_log <dbl>, se_ks <dbl>,
## #   se_kr <dbl>, se_ls <dbl>, se_mu_pr_log <dbl>
```

5 Prediction results

5.1 load model Prediction results

```
#load prediction
AllDat_predY <- read.csv(paste0(modelPath, "/rlt/AllDat_predY_",modelversion,".csv"))
AllDat_predY$WMSize <- as.factor(AllDat_predY$WMSize)
levels(AllDat_predY$WMSize) = c("low", "medium",  "high")
AllDat_predY$pred_Bias = AllDat_predY$mu_r - AllDat_predY$curDur
AllDat_predY$predErr = AllDat_predY$mu_r - AllDat_predY$repDur
AllDat_predY$relatErr = AllDat_predY$predErr / AllDat_predY$repDur
AllDat_predY[which(AllDat_predY$Exp == "Exp4"),"Exp"] = "Exp4a"
AllDat_predY[which(AllDat_predY$Exp == "Exp5"),"Exp"] = "Exp4b"
AllDat_predY$Exp = as.factor(AllDat_predY$Exp)
AllDat_predY$gap <- factor(AllDat_predY$gap,  labels = c('short','long', 'not sure'))
```

```r
#calculate the mean reproduction biases for the five given intervals for all subjects
mpredY_sub <- dplyr::group_by(AllDat_predY, curDur, Exp, NSub, WMSize) %>%
  dplyr::summarize(n = n(),
                   m_repDur = mean(repDur),
                   sd_repDur = sd(repDur),
                   m_mu_r = mean(mu_r),
                   m_sig_r = mean(sig_r),
                   m_wp = mean(wp),
                   se_wp = sd(wp)/sqrt(n-1),
                   log_lik =mean(log_lik),
                   cv =sd_repDur/ m_repDur,
                   pred_cv = mean(sig_r/mu_r),
                   predRP_err = mean(m_mu_r-m_repDur),
                   predVar_err = mean(m_sig_r-sd_repDur),
                   predRP_rerr = mean(abs(m_mu_r-m_repDur)/m_repDur),
                   predVar_rerr = mean(abs(m_sig_r-sd_repDur)/sd_repDur),
                   predcv_err = pred_cv-cv,
                   predcv_rerr = mean(abs(pred_cv-cv)/cv))
```

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

```r
mpredY_sub_split <-split(mpredY_sub %>%select(c('Exp', 'NSub', 'WMSize', 'm_repDur', 'curDur')), mpredY
mpredY_sub_cv_split <-split(mpredY_sub %>%select(c('Exp', 'NSub', 'WMSize', 'cv', 'curDur')), mpredY_su

mpredY_sub_split1 <-split(mpredY_sub %>%select(c('Exp', 'NSub', 'WMSize', 'curDur' ,'m_repDur')), mpred
mpredY_sub_cv_split1 <-split(mpredY_sub %>%select(c('Exp', 'NSub', 'WMSize','curDur', 'cv')), mpredY_su
mpredY_sub_jasp_RP = NULL
mpredY_sub_jasp_cv = NULL
mpredY_sub_jasp_RP1 = NULL
mpredY_sub_jasp_cv1 = NULL
for(i in 1: length(mpredY_sub_split)){
  temp = mpredY_sub_split[[i]]
  curDur = unique(temp$curDur)
  temp$curDur = NULL
  colnames(temp) = c('Exp', 'NSub', 'WMSize', paste0('m_repDur_', curDur))

  temp_cv = mpredY_sub_cv_split[[i]]
  curDur = unique(temp_cv$curDur)
  temp_cv$curDur = NULL
  colnames(temp_cv) = c('Exp', 'NSub', 'WMSize', paste0('cv_', curDur))
  if(i == 1){
    mpredY_sub_jasp_RP = temp
    mpredY_sub_jasp_cv = temp_cv
  }
  else{
    mpredY_sub_jasp_RP = left_join(mpredY_sub_jasp_RP, temp, by=c("Exp",  "NSub", "WMSize"))
    mpredY_sub_jasp_cv = left_join(mpredY_sub_jasp_cv, temp_cv, by=c("Exp",  "NSub", "WMSize"))
  }

}
for (i in 1: length(mpredY_sub_split1)){
  temp1 = mpredY_sub_split1[[i]]
  WMSize = unique(temp1$WMSize)
```

```
  temp1$WMSize = NULL
  colnames(temp1) = c('Exp', 'NSub', 'curDur', paste0('m_repDur_', WMSize))

  temp_cv1 = mpredY_sub_cv_split1[[i]]
  WMSize = unique(temp_cv1$WMSize)
  temp_cv1$WMSize = NULL
  colnames(temp_cv1) = c('Exp', 'NSub', 'curDur', paste0('cv_', WMSize))
  if(i == 1){
    mpredY_sub_jasp_RP1 = temp1
    mpredY_sub_jasp_cv1 = temp_cv1
  }
  else{
    mpredY_sub_jasp_RP1 = left_join(mpredY_sub_jasp_RP1, temp1, by=c("Exp",  "NSub", "curDur"))
    mpredY_sub_jasp_cv1 = left_join(mpredY_sub_jasp_cv1, temp_cv1, by=c("Exp",  "NSub", "curDur"))
  }
}

write_csv(mpredY_sub_jasp_RP, paste0(modelPath, '/rlt/RP_Bias_jasp.csv'))
write_csv(mpredY_sub_jasp_cv, paste0(modelPath, '/rlt/cv_jasp.csv'))
write_csv(mpredY_sub_jasp_RP1, paste0(modelPath, '/rlt/RP_Bias_jasp1.csv'))
write_csv(mpredY_sub_jasp_cv1, paste0(modelPath, '/rlt/cv_jasp1.csv'))
```

```
write_csv(dplyr::group_by(mpredY_sub, curDur, NSub) %>%
  dplyr::summarize(m_cv = mean(cv))%>%spread(curDur, m_cv), paste0(modelPath, '/rlt/m_cv.csv'))
```

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

```
mpredY_sub$RP_bias = mpredY_sub$m_repDur -mpredY_sub$curDur
mpredY_sub_new <- dplyr::group_by(mpredY_sub, curDur, Exp, NSub) %>%
  dplyr::summarize(m_RP_bias = mean(RP_bias))%>% spread(curDur, m_RP_bias)
```

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

```
write_csv(mpredY_sub_new%>%filter(Exp == 'Exp1'), paste0(modelPath, '/rlt/RP_Bias_exp1.csv'))
write_csv(mpredY_sub_new%>%filter(Exp == 'Exp2'), paste0(modelPath, '/rlt/RP_Bias_exp2.csv'))
write_csv(mpredY_sub_new%>%filter(Exp == 'Exp3'), paste0(modelPath, '/rlt/RP_Bias_exp3.csv'))
write_csv(mpredY_sub_new%>%filter(Exp == 'Exp4a'), paste0(modelPath, '/rlt/RP_Bias_exp4a.csv'))
write_csv(mpredY_sub_new%>%filter(Exp == 'Exp4b'), paste0(modelPath, '/rlt/RP_Bias_exp4b.csv'))



mpredY_sub_WMsize <- dplyr::group_by(mpredY_sub, WMSize, Exp, NSub) %>%
  dplyr::summarize(m_RP_bias = mean(RP_bias))%>% spread(WMSize, m_RP_bias)
```

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

```
write_csv(mpredY_sub_WMsize%>%filter(Exp == 'Exp3'), paste0(modelPath, '/rlt/RP_Bias_WMsize_exp3.csv'))
write_csv(mpredY_sub_WMsize%>%filter(Exp == 'Exp4a'), paste0(modelPath, '/rlt/RP_Bias_WMsize_exp4a.csv'))
write_csv(mpredY_sub_WMsize%>%filter(Exp == 'Exp4b'), paste0(modelPath, '/rlt/RP_Bias_WMsize_exp4b.csv'))
```

```
#### predicted data
m_predY <- mpredY_sub%>%
  dplyr::group_by(Exp, curDur, WMSize) %>%
  dplyr::summarize(m_m_repDur = mean(m_repDur),
```

```
                  m_sd_repDur = mean(sd_repDur),
                  m_m_sig_r =mean(m_sig_r),
                  m_m_mu_r = mean(m_mu_r),
                  m_m_wp = mean(m_wp),
                  n = n(),
                  m_se_wp = sd(se_wp)/sqrt(n-1),
                  log_lik =mean(log_lik),
                  mpredRP_err = mean(predRP_err),
                  mpredVar_err = mean(predVar_err),
                  mpredRP_rerr = mean(predRP_rerr),
                  mpredVar_rerr = mean(predVar_rerr),
                  cv= mean(cv),
                  pred_cv = mean(pred_cv),
                  mpredcv_err = mean(predcv_err),
                  mpredcv_rerr = mean(predcv_rerr))
m_predY_acc =  mpredY_sub%>%
  dplyr::group_by(Exp) %>%
  dplyr::summarize(mpred_rerr = mean(predRP_rerr)*100,
                  mpredVar_rerr = mean(predVar_rerr)*100,
                  mpredcv_rerr = mean(predcv_rerr)*100)
m_predY_acc
```

```
## # A tibble: 5 x 4
##   Exp   mpred_rerr mpredVar_rerr mpredcv_rerr
##   <fct>      <dbl>         <dbl>        <dbl>
## 1 Exp1        3.33          17.8         17.7
## 2 Exp2        4.17          13.6         13.9
## 3 Exp3        3.40          16.1         15.9
## 4 Exp4a       3.47          18.5         18.5
## 5 Exp4b       2.81          16.5         15.7
```

6 WAIC and LOO-CV

```
#extract waic and loo-cv from parameter list
m_WAIC <- dplyr::group_by(Bayparlist, Exp) %>%
  dplyr::summarize(n =n(),
                  m_looic = mean(looic),
                  m_waic = mean(waic),
                  se_waic = sd(waic)/sqrt(n-1),
                  se_looic = sd(looic)/sqrt(n-1),
                  m_p_loo = mean(p_loo),
                  m_elpd_loo = mean(elpd_loo),
                  m_se_looic = mean(se_looic),
                  m_se_p_loo = mean(se_p_loo),
                  m_p_waic = mean(p_waic),
                  m_se_waic = mean(se_waic))

m_WAIC
```

```
## # A tibble: 5 x 12
##   Exp       n m_looic m_waic se_waic se_looic m_p_loo m_elpd_loo m_se_looic
##   <fct> <int>   <dbl>  <dbl>   <dbl>    <dbl>   <dbl>      <dbl>      <dbl>
## 1 Exp1     16    374.   125.    38.2     38.2    305.      -187.       38.2
```

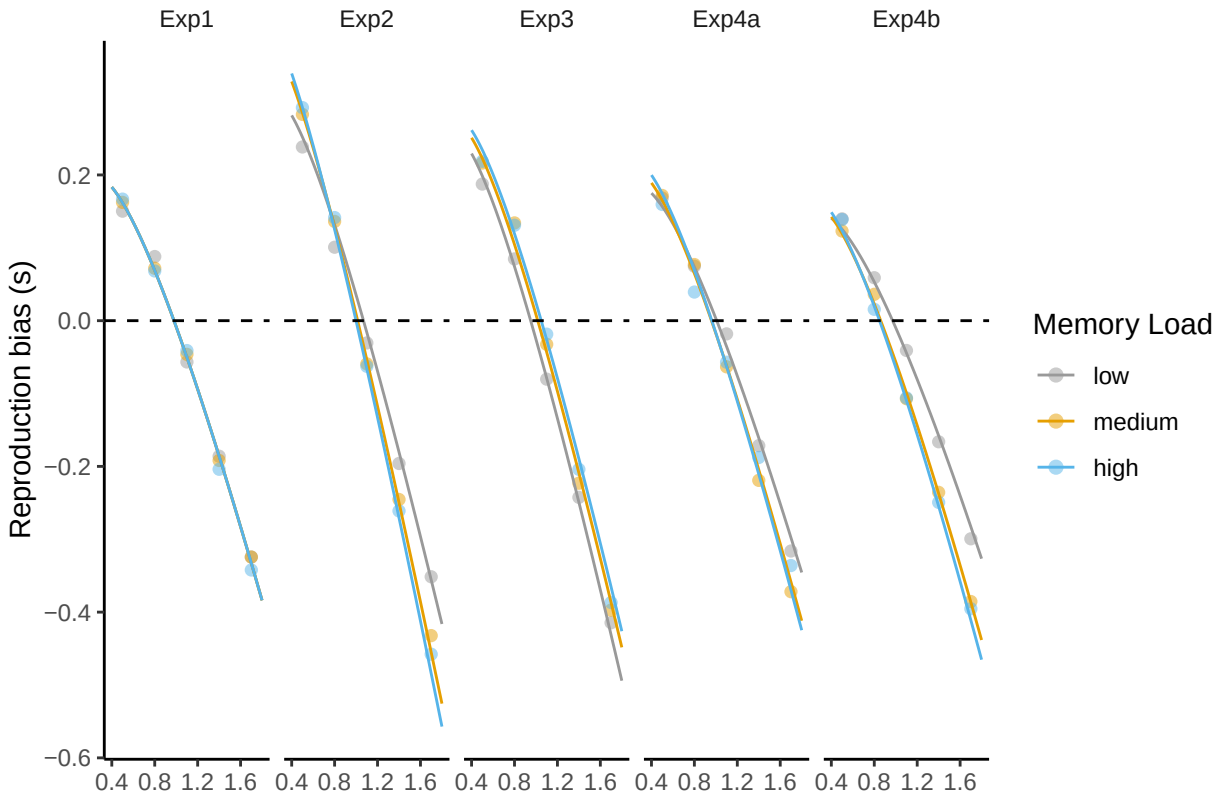
```
## 2 Exp2     16    557.   311.    40.5     40.1    303.      -278.       40.1
## 3 Exp3     16    421.   174.    21.7     21.4    304.      -210.       21.4
## 4 Exp4a    16    418.   170.    36.2     36.0    305.      -209.       36.0
## 5 Exp4b    16    813.   317.   103.     103.     609.      -406.      103.
## # ... with 3 more variables: m_se_p_loo <dbl>, m_p_waic <dbl>, m_se_waic <dbl>
```

```
#load test results
AllDat_newY <- read.csv(paste0(modelPath, "/rlt/AllDat_newY_",modelversion,".csv"))
AllDat_newY$WMSize <- as.factor(AllDat_newY$WMSize)
levels(AllDat_newY$WMSize) = c("low", "medium",  "high")
AllDat_newY[which(AllDat_newY$Exp == "Exp4"), "Exp"] = "Exp4a"
AllDat_newY[which(AllDat_newY$Exp == "Exp5"), "Exp"] = "Exp4b"
AllDat_newY$Exp = as.factor(AllDat_newY$Exp)
```

7 Plot Results

7.1 RP biase

```
  RP_bias  <- ggplot(data = m_predY, aes(x = curDur, y = m_m_repDur - curDur,color=WMSize, shape = as.fa
  geom_point(size=2, alpha = 0.5)+
  geom_line(data= m_newY, aes(x=curDur, y=m_mu_r-curDur, color=WMSize)) +
  geom_hline(yintercept = 0, linetype='dashed')+
  facet_grid(cols = vars(Exp)) +
  labs(x=" ", y="Reproduction bias (s)", shape=" ", color = "Memory Load")+theme_new+
  colorSet3+guides(shape="none")

ggsave(paste0(getwd(), "/", modelPath, "/figures/RP_bias.png"), RP_bias, width = 6, height = 6)

RP_bias
```




```
fig = ggplot(data = AllExpData %>% filter(Exp == 'Exp3') %>%
  dplyr::group_by(curDur, WMSize, NSub) %>%
  dplyr::summarize(n = n(),
                   m_repDur = mean(repDur),
                   se_repDur = sd(repDur)/sqrt(n-1)), aes(x=curDur, y = m_repDur-curDur, group = WMSize
  geom_point()+
  geom_line()+
  #geom_errorbar(width=.2,  aes(ymin = m_repDur - se_repDur, ymax = m_repDur + se_repDur)) +
  geom_hline(yintercept = 0, linetype='dashed')+
  facet_wrap(~NSub) +
  labs(x="Sample intervals (s)", y="Reproduction bias in Exp 4b(s)", shape=" ", color = "Memory Load")+
  theme_new+colorSet3+guides(shape="none")+
   scale_x_continuous(breaks=seq(0, 1.6, 0.4))+ theme(legend.position="top")
```

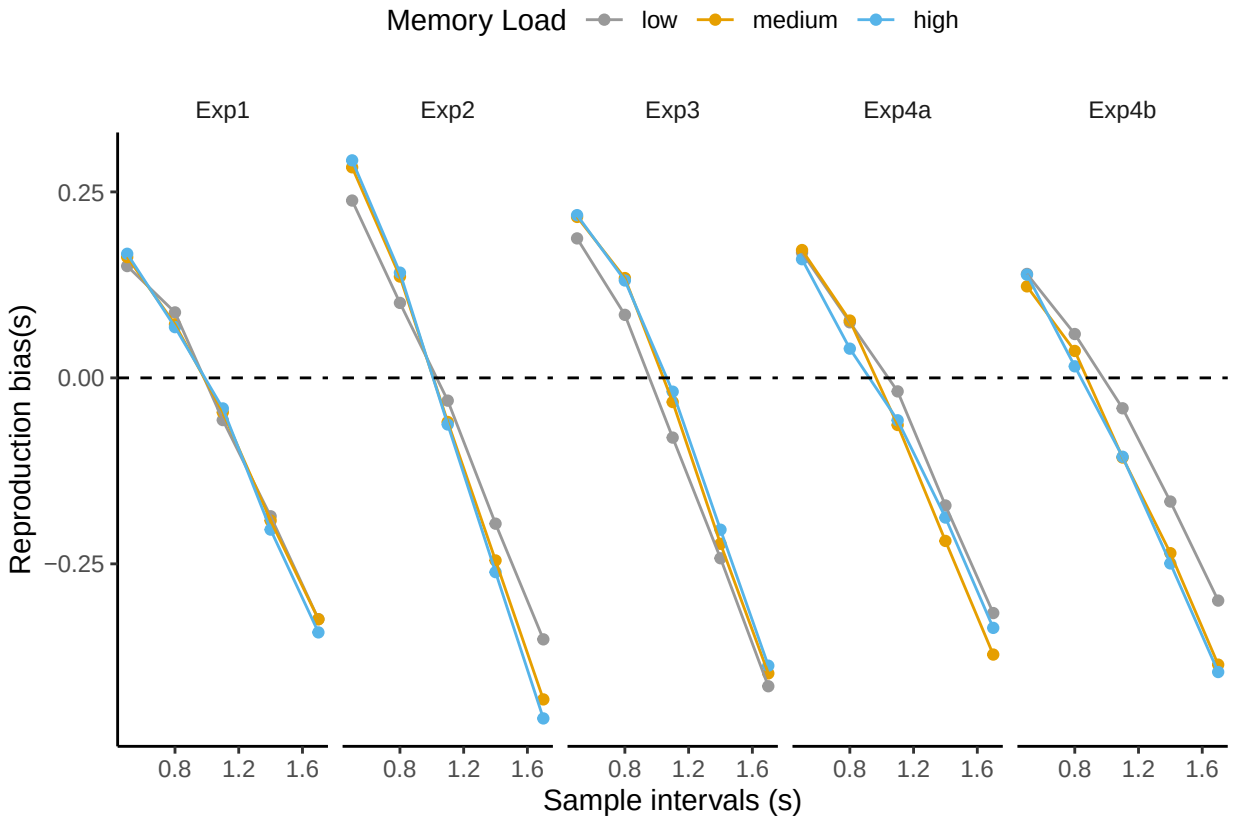
```
## `summarise()` has grouped output by 'curDur', 'WMSize'. You can override using
## the `.groups` argument.
```

```
RP_bias_obs  <- ggplot(data = AllExpData %>%
  dplyr::group_by(Exp, curDur, WMSize, NSub) %>%
  dplyr::summarize(n = n(),
                   m_repDur = mean(repDur),
                   se_repDur = sd(repDur)/sqrt(n-1)) %>%
  dplyr::group_by(Exp, curDur, WMSize) %>%
  dplyr::summarize(m_m_repDur = mean(m_repDur),
                   m_se_repDur = mean(se_repDur)), aes(x = curDur, y = m_m_repDur-curDur,  color=as.fac
  geom_point()+
  geom_line()+
  #geom_errorbar(width=.2,  aes(ymin = m_m_repDur-curDur - m_se_repDur, ymax = m_m_repDur -curDur + m_s
```

```
  geom_hline(yintercept = 0, linetype='dashed')+
  facet_grid(cols = vars(Exp)) +
  labs(x="Sample intervals (s)", y="Reproduction bias(s)", shape=" ", color = "Memory Load")+
  theme_new+colorSet3+guides(shape="none")+
   scale_x_continuous(breaks=seq(0, 1.6, 0.4))+ theme(legend.position="top")
```

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

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/RP_bias_obs.png"), RP_bias_obs, width = 6, height = 4)

RP_bias_obs
```



7.2 RP Slope

```
# calculate the slope of the cv curve
RPslope_model <- function(df) {
  lm(m_repDur ~ log(curDur), data = df)
}

RPslopes <- mpredY_sub %>%
  dplyr::group_by(NSub, Exp, WMSize) %>% nest() %>%
  mutate(model = map(data, RPslope_model)) %>%
  mutate(slope = map(model, broom::tidy)) %>%  # get estimates
  unnest(slope, .drop = TRUE) %>% # remove raw data
  select(-std.error,-statistic, -p.value) %>%  # remove unnessary columns
  spread(term, estimate) %>%   # spread stimates
```

```
  dplyr::rename(Intercept = `(Intercept)`, slope = `log(curDur)`)
```

```
## Warning: The `.drop` argument of `unnest()` is deprecated as of tidyr 1.0.0.
## All list-columns are now preserved.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

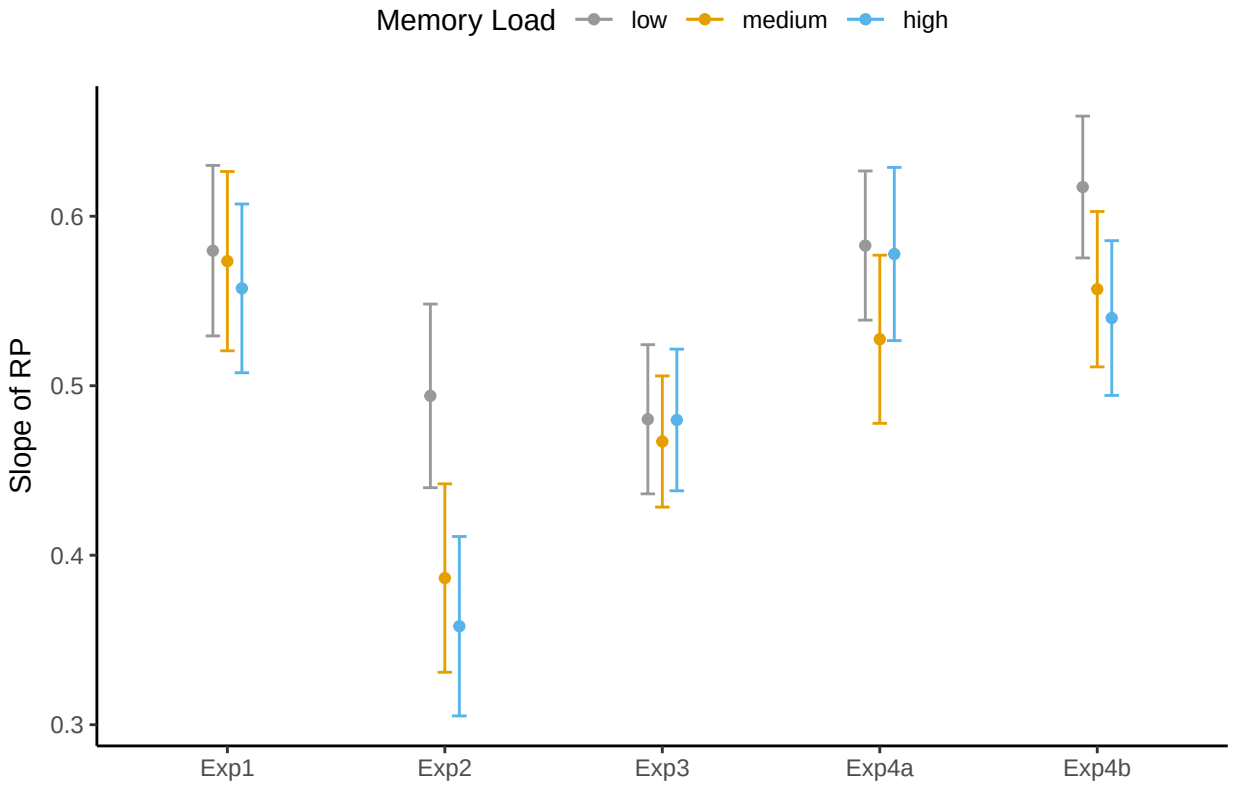
```
RPslopes$data <- NULL
RPslopes$model <- NULL
```

```
mRPslopes <- RPslopes%>% dplyr::group_by(WMSize, Exp) %>%
  dplyr::summarize(m_Intercept = mean(Intercept),
                   m_slope = mean(slope),
                   n = n(),
                   se_slope = sd(slope)/sqrt(n-1),
                   se_Intercept = sd(Intercept)/sqrt(n-1))
```

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

```
plt_CVslope <- ggplot(mRPslopes, aes(Exp, m_slope, ymin = m_slope - se_slope, ymax = m_slope + se_slope
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  position = position_dodge(width = 0.2)) +
  colorSet5+
  labs(x = "", y = TeX("Slope of RP"), color = 'Memory Load') +
  theme_new +theme(legend.position="top")

plt_CVslope
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



7.2.1 Anova analysis on slopes of RP

```
RPslopes$WMSize = as.factor(RPslopes$WMSize)
ezANOVA(data = RPslopes, dv= slope, wid=NSub, within=.(WMSize), between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## $ANOVA
##       Effect DFn DFd         F            p p<.05        ges
## 2        Exp   4  75  2.465821 5.210071e-02       0.11127986
## 3     WMSize   2 150 24.943347 4.446854e-10     * 0.01567449
## 4 Exp:WMSize   8 150  6.439500 3.455596e-07     * 0.01617814
##
## $`Mauchly's Test for Sphericity`
##       Effect        W         p p<.05
## 3     WMSize 0.943444 0.1160104
## 4 Exp:WMSize 0.943444 0.1160104
##
## $`Sphericity Corrections`
##       Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 3     WMSize 0.9464714 1.164847e-09         * 0.9702459 7.594066e-10         *
## 4 Exp:WMSize 0.9464714 6.643447e-07         * 0.9702459 4.968822e-07         *
```

```
ezANOVA(data = RPslopes %>% filter(Exp =='Exp1'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F         p p<.05         ges
## 2 WMSize   2  30 1.628257 0.2131416       0.002398051
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.8320858 0.2761702
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05       HFe    p[HF] p[HF]<.05
## 2 WMSize 0.8562273 0.2172718           0.9557549 0.214484
```

```
ezANOVA(data = RPslopes %>% filter(Exp =='Exp2'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F            p p<.05        ges
## 2 WMSize   2  30 21.20007 1.822755e-06     * 0.07646424
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.8461525 0.3105563
##
## $`Sphericity Corrections`
##   Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2 WMSize 0.8666656 7.302883e-06         * 0.9698478 2.493639e-06         *
```

```
ezANOVA(data = RPslopes %>% filter(Exp =='Exp3'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 2 WMSize   2  30 0.4137346 0.6648914       0.00152841
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9800567 0.8684769
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05      HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9804466 0.6610089           1.126392 0.6648914
```

```
ezANOVA(data = RPslopes %>% filter(Exp =='Exp4a'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 7.246942 0.002705914     * 0.01861911
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9730381 0.8258648
##
## $`Sphericity Corrections`
##   Effect      GGe       p[GG] p[GG]<.05      HFe       p[HF] p[HF]<.05
```

```
## 2 WMSize 0.973746 0.002974256         * 1.117022 0.002705914         *
```

```
ezANOVA(data = RPslopes %>% filter(Exp =='Exp4b'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 8.551317 0.001151174     * 0.03802106
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.8930404 0.4529994
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05      HFe       p[HF] p[HF]<.05
## 2 WMSize 0.9033753 0.001756187         * 1.019764 0.001151174         *
```

7.2.2 Anova analysis on Intercept of RP

```
ezANOVA(data = RPslopes, dv= Intercept, wid=NSub, within=.(WMSize), between = .(Exp))
```

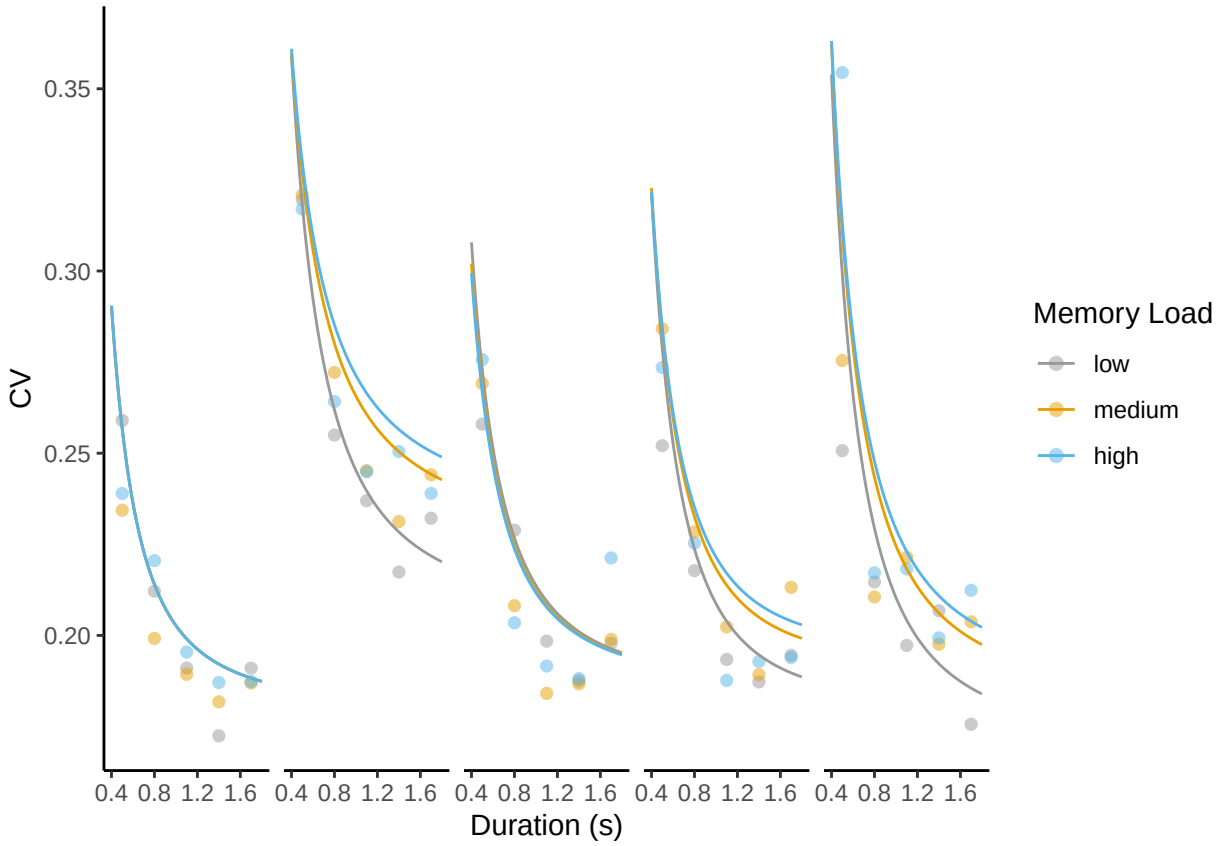
```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## $ANOVA
##       Effect DFn DFd         F            p p<.05         ges
## 2        Exp   4  75 0.6157298 6.526476e-01       0.028876673
## 3     WMSize   2 150 4.6236001 1.125794e-02     * 0.005792574
## 4 Exp:WMSize   8 150 6.6943967 1.762289e-07     * 0.032641724
##
## $`Mauchly's Test for Sphericity`
##       Effect         W            p p<.05
## 3     WMSize 0.8020579 0.0002855036     *
## 4 Exp:WMSize 0.8020579 0.0002855036     *
##
## $`Sphericity Corrections`
##       Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 3     WMSize 0.8347649 1.634417e-02         * 0.8515168 1.573692e-02         *
## 4 Exp:WMSize 0.8347649 1.479632e-06         * 0.8515168 1.191853e-06         *
```

7.3 CV

```
curDurItem <- unique(m_predY$curDur)
RP_CV <- ggplot(data= m_predY, aes(x=curDur, y= cv, color=WMSize, shape = as.factor('Observation'))) +
  geom_point(size=2, alpha = 0.5)+
  geom_line(data = m_newY, aes(x=curDur, y= m_sig_r/m_mu_r, color=WMSize)) +
  facet_grid(~Exp) +
  labs(x="Duration (s)", y="CV", shape=" ", color = "Memory Load")+ theme_new+
  colorSet3+guides(shape="none")+theme(strip.text.x = element_blank())
RP_CV
```




```
  ggsave(paste0(getwd(), "/", modelPath, "/figures/RP_CV.png"), RP_CV, width = 6, height = 6)
```

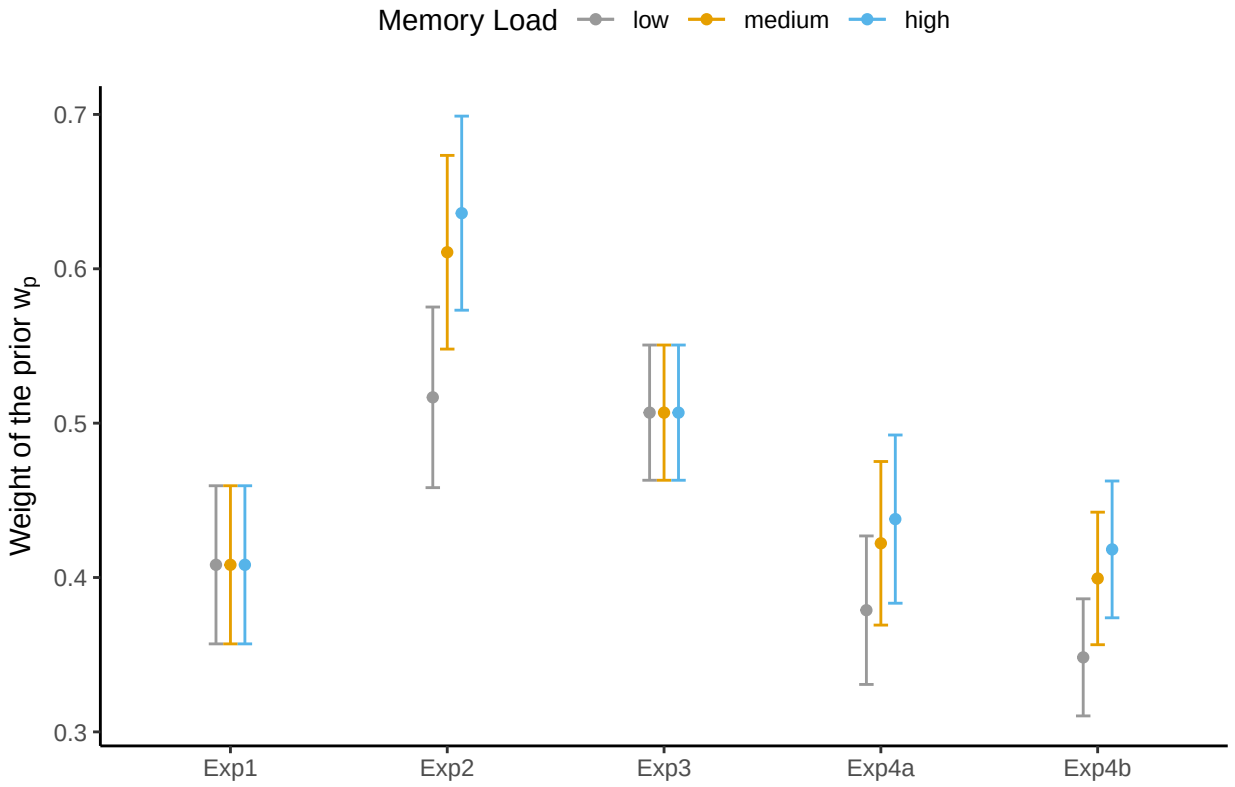
8 weight of prior

```
plt_wp <- ggplot(data = AllDat_predY %>%dplyr::group_by(NSub, Exp, WMSize) %>% dplyr::summarise(m_wp = m
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  position = position_dodge(width = 0.2)) +
  #coord_cartesian(ylim = c(0.5, 1)) +
  colorSet5+
  labs(x = "", y = TeX("Weight of the prior $w_p$"), color = 'Memory Load') +
  theme_new + theme(legend.position="top")
```

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

```
plt_wp
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```




```
ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_wp.png"), plt_wp, width = 3, height = 3)
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

9 Indifference Point and slope

9.1 Inifference Point (curve)

```
AllDat_newY$predErr = AllDat_newY$mu_r - AllDat_newY$curDur

temp_newY <- AllDat_newY %>% filter(curDur > 0.8, curDur < 1.1) %>% select(Exp, WMSize, NSub, predErr, c

InP_curve<- temp_newY%>% dplyr::group_by(Exp, WMSize, NSub)%>%
  dplyr::summarise(minErr = min(abs(predErr)), idx = which.min(abs(predErr)))
```

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

```
InP_curve$InP_curve = temp_newY[InP_curve$idx,]$curDur
InP_curve$y = temp_newY[InP_curve$idx,]$predErr + temp_newY[InP_curve$idx,]$curDur
InP_curve
```

```
## # A tibble: 240 x 7
## # Groups:   Exp, WMSize [15]
##    Exp   WMSize  NSub    minErr   idx InP_curve     y
##    <fct> <fct>  <int>     <dbl> <int>     <dbl> <dbl>
##  1 Exp1  low        1 0.00303      12      0.92 0.917
##  2 Exp1  low        2 0.0243       29      1.09 0.957
```

```
##  3 Exp1  low        3 0.00226      22      1.02 0.941
##  4 Exp1  low        4 0.00295      17      0.97 0.929
##  5 Exp1  low        5 0.000893     16      0.96 0.927
##  6 Exp1  low        6 0.00278       6      0.86 0.902
##  7 Exp1  low        7 0.0613        1      0.81 0.888
##  8 Exp1  low        8 0.0000331    26      1.06 0.950
##  9 Exp1  low        9 0.0707       29      1.09 0.957
## 10 Exp1  low       10 0.00140      19      0.99 0.934
## # ... with 230 more rows
```

```
#plot indifference points (the intersections of the Prediction curve with the diagonal)
plt_InP_curve<- ggplot(data = InP_curve%>%dplyr::group_by(Exp, WMSize)%>% dplyr::summarise(m_InP = mean
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  aes(ymin = m_InP - se_InP, ymax = m_InP + se_InP), position = position_dodge
  labs(colour = "Memory Load")+colorSet3+
  xlab(' ')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")
```

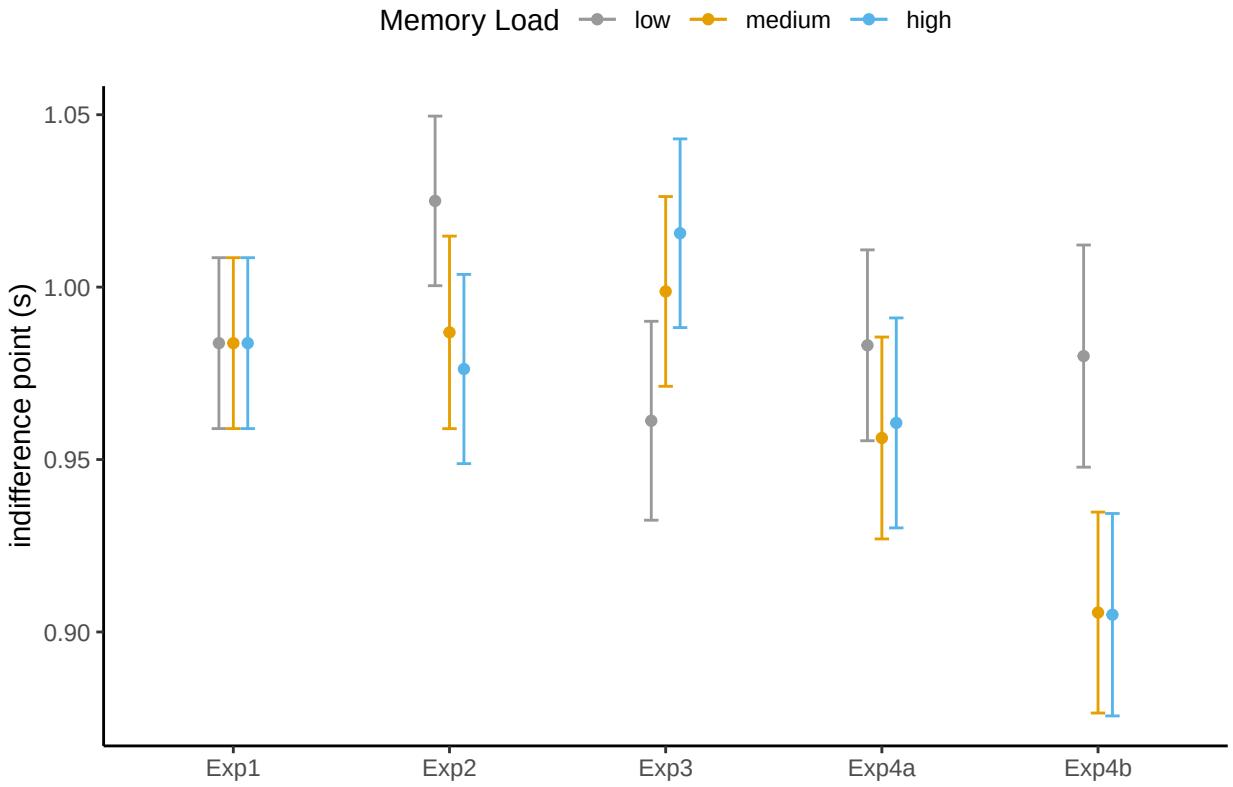
```
## `summarise()` has grouped output by 'Exp'. You can override using the `.groups`
## argument.
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_InP_curve.png"), plt_InP_curve, width = 3, height =
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
plt_InP_curve
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```



9.2 Indifference point and slope (linear regression)

9.2.1 Observed data

```
#Observed Indifference Point for Exp.4b
obs_model <- function(df) {
  lm(repDur ~ curDur, data = df)
}
#Observed Indifference Point
obs_Inp_list <- AllDat_predY %>%
  dplyr::group_by(NSub, Exp, WMSize, gap) %>% nest()  %>%
  mutate(model = map(data, obs_model)) %>%  # linear regression
  mutate(slope = map(model, broom::tidy)) %>%  # get estimates
  unnest(slope, .drop = TRUE) %>% # remove raw data
  select(-std.error,-statistic, -p.value) %>%  # remove unnessary columns
  spread(term, estimate) %>%   # spread stimates
  dplyr::rename(Intercept = `(Intercept)`, slope = curDur)  # rename columns
obs_Inp_list$model = NULL
obs_Inp_list$data = NULL
obs_Inp_list$inP = obs_Inp_list$Intercept /(1-obs_Inp_list$slope)
```

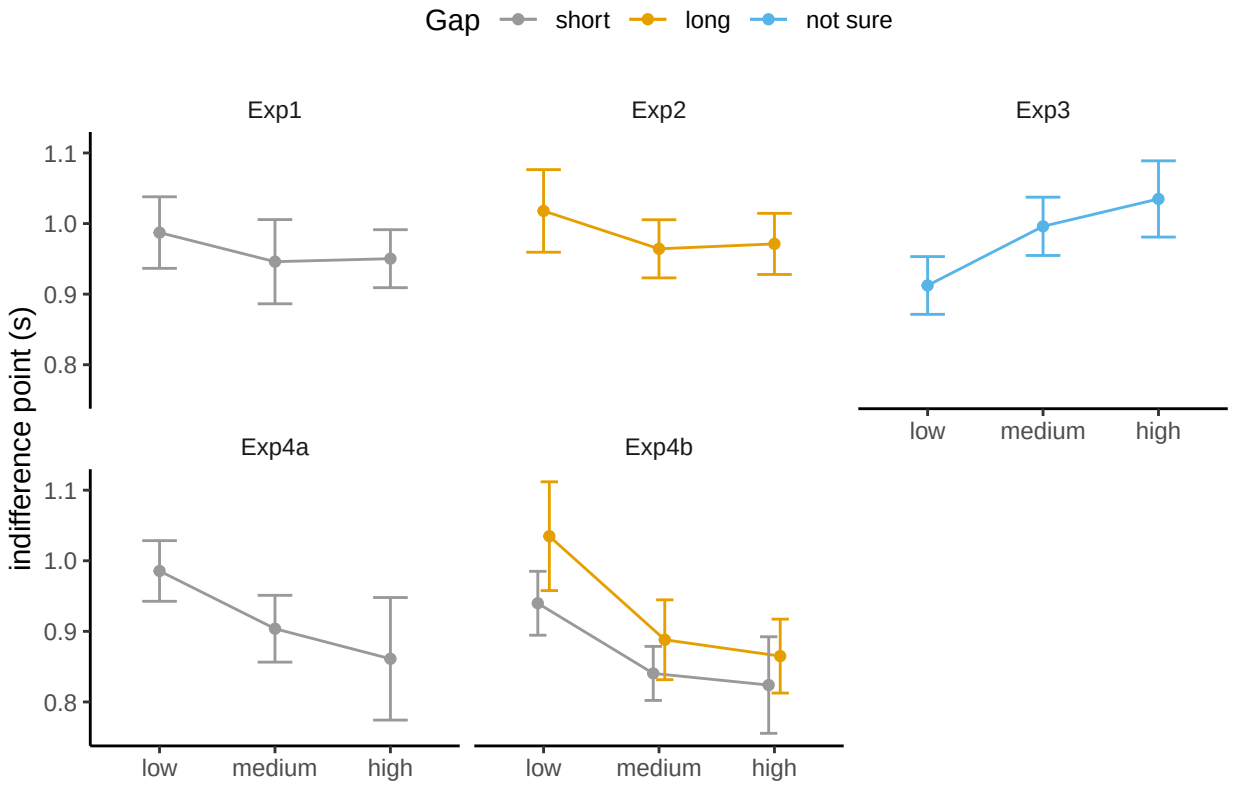
```
obs_Inp_list_no_gap <- AllDat_predY %>%
  dplyr::group_by(NSub, Exp, WMSize) %>% nest()  %>%
  mutate(model = map(data, obs_model)) %>%  # linear regression
  mutate(slope = map(model, broom::tidy)) %>%  # get estimates
  unnest(slope, .drop = TRUE) %>% # remove raw data
  select(-std.error,-statistic, -p.value) %>%
  spread(term, estimate) %>%   # spread stimates
```

```
  dplyr::rename(Intercept = `(Intercept)`, slope = curDur)  # rename columns
obs_Inp_list_no_gap$model = NULL
obs_Inp_list_no_gap$data = NULL
obs_Inp_list_no_gap$inP = obs_Inp_list_no_gap$Intercept /(1-obs_Inp_list_no_gap$slope)
```

```
m_obs_Inp_list = obs_Inp_list %>% group_by(Exp, WMSize, gap)%>%
  dplyr::summarise(n=n(),
                   m_Intercept = mean(Intercept),
                   se_Intercept= sd(Intercept)/sqrt(n-1),
                   m_inP = mean(inP),
                   se_inP = sd(inP)/sqrt(n-1),
                   m_slope = mean(slope),
                   se_slope = sd(slope)/sqrt(n-1))
```

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

```
plt_InP_linear_gap<- ggplot(data = m_obs_Inp_list, aes(x=WMSize, y=m_inP, group = gap, color = gap))+
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.3,  aes(ymin = m_inP - se_inP, ymax = m_inP + se_inP), position = position_dodge
  labs(colour = "Gap")+colorSet3+
  facet_wrap(~Exp)+
  xlab(' ')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")

ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_InP_linear_gap.png"), plt_InP_linear_gap, width = 5

plt_InP_linear_gap
```




```
ezANOVA(data = obs_Inp_list%>%filter(Exp =='Exp4b'), dv= inP, wid=NSub, within= .(gap, WMSize) )
```

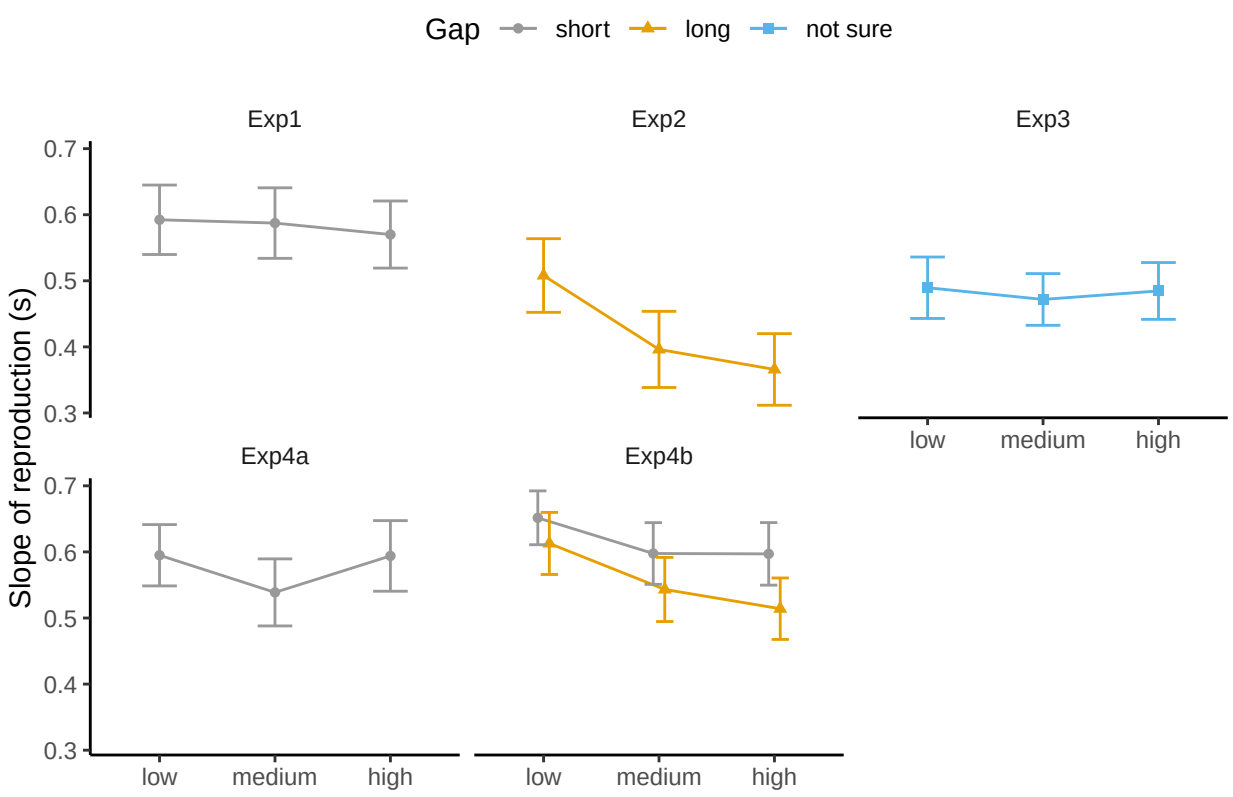
```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: You have removed one or more levels from variable "gap". Refactoring
## for ANOVA.
```

```
## $ANOVA
##       Effect DFn DFd        F           p p<.05         ges
## 2        gap   1  15 4.065468 0.062041074       0.019543220
## 3     WMSize   2  30 8.762649 0.001006806     * 0.078253848
## 4 gap:WMSize   2  30 0.584382 0.563670241       0.003061788
##
## $`Mauchly's Test for Sphericity`
##       Effect         W           p p<.05
## 3     WMSize 0.4515805 0.003829536     *
## 4 gap:WMSize 0.5210587 0.010428128     *
##
## $`Sphericity Corrections`
##       Effect       GGe       p[GG] p[GG]<.05       HFe       p[HF] p[HF]<.05
## 3     WMSize 0.6458198 0.004991546         * 0.6808339 0.004255437         *
## 4 gap:WMSize 0.6761593 0.502620038           0.7194299 0.512209537
```

```
plt_RP_slope_linear_gap<- ggplot(data = m_obs_Inp_list, aes(x= WMSize, y=m_slope, group = gap,color = ga
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.3,  aes(ymin = m_slope - se_slope, ymax = m_slope + se_slope), position = positi
  facet_wrap(~Exp)+
  labs(colour = "Gap", shape = "Gap")+colorSet3+
```

```
  xlab(' ')+ylab("Slope of reproduction (s)")+
  theme(legend.position = "top")

ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_RP_slope_linear_gap.png"), plt_RP_slope_linear_gap

plt_RP_slope_linear_gap
```




```
# plot the observed indifference points and slopes of RP
plt_obs_InP_slope_err<- ggplot(data = obs_Inp_list %>% group_by(Exp, WMSize)%>%
  dplyr::summarise(n=n(),
                   m_inP = mean(inP),
                   se_inP = sd(inP)/sqrt(n-1),
                   m_slope = mean(slope),
                   se_slope = sd(slope)/sqrt(n-1)), aes(x= m_slope, y=m_inP, color = WMSize))+
  geom_line(stat = "identity")+
  geom_point(stat = "identity")+
  geom_errorbar(width = 0.02, aes(ymin = m_inP - se_inP, ymax = m_inP + se_inP)) +
  geom_errorbarh(height =0.02, aes(xmin = m_slope - se_slope, xmax = m_slope + se_slope)) +
  theme_new+
  labs(colour = "Memory Load")+colorSet3+
  facet_grid(~Exp)+
  xlab('slope of reproduction')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")
```

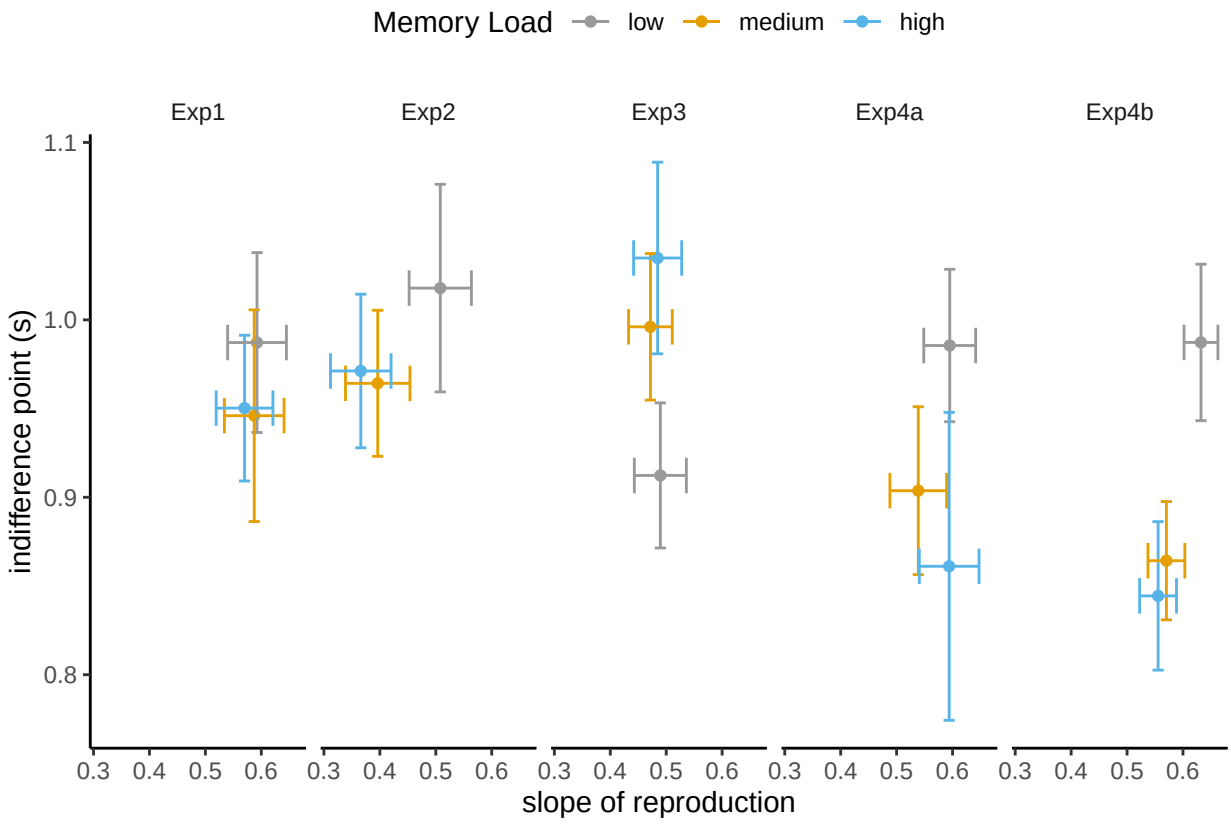
```
## `summarise()` has grouped output by 'Exp'. You can override using the `.groups`
## argument.
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_obs_InP_slope_err.png"), plt_obs_InP_slope_err, wi
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
plt_obs_InP_slope_err
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```




```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp1'), dv= inP, wid=NSub, within = .(WMSize))
```

9.2.1.1 anova on observed InP

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 2 WMSize   2  30 0.8958593 0.4188984       0.00926657
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.8099834 0.2287352
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.8403244 0.4045455           0.9343692 0.4133529
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp2'), dv= inP, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 2 WMSize   2  30 0.8571025 0.4345216       0.01703543
##
## $`Mauchly's Test for Sphericity`
##   Effect         W            p p<.05
## 2 WMSize 0.3308603 0.0004340228     *
##
## $`Sphericity Corrections`
##   Effect      GGe     p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.599111 0.3868109           0.6220776 0.3904242
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp3'), dv= inP, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 7.414842 0.002417317     * 0.08142895
##
## $`Mauchly's Test for Sphericity`
##   Effect         W          p p<.05
## 2 WMSize 0.5265272 0.01121878     *
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05       HFe       p[HF] p[HF]<.05
## 2 WMSize 0.6786688 0.008009349         * 0.7226376 0.006790677         *
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp4a'), dv= inP, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 2.821558 0.07536744       0.04668729
##
## $`Mauchly's Test for Sphericity`
##   Effect         W            p p<.05
## 2 WMSize 0.3560906 0.0007259782     *
```

```
##
## $`Sphericity Corrections`
##   Effect      GGe     p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.608306 0.1044842           0.6335813 0.1023756
```

```
ezANOVA(data = obs_Inp_list %>% filter(Exp =='Exp4b'), dv= inP, wid=NSub, within = .(WMSize, gap))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: You have removed one or more levels from variable "gap". Refactoring
## for ANOVA.
```

```
## $ANOVA
##       Effect DFn DFd        F           p p<.05         ges
## 2     WMSize   2  30 8.762649 0.001006806     * 0.078253848
## 3        gap   1  15 4.065468 0.062041074       0.019543220
## 4 WMSize:gap   2  30 0.584382 0.563670241       0.003061788
##
## $`Mauchly's Test for Sphericity`
##       Effect         W           p p<.05
## 2     WMSize 0.4515805 0.003829536     *
## 4 WMSize:gap 0.5210587 0.010428128     *
##
## $`Sphericity Corrections`
##       Effect       GGe       p[GG] p[GG]<.05       HFe       p[HF] p[HF]<.05
## 2     WMSize 0.6458198 0.004991546         * 0.6808339 0.004255437         *
## 4 WMSize:gap 0.6761593 0.502620038           0.7194299 0.512209537
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp1'), dv= slope, wid=NSub, within = .(WMSize))
```

9.2.1.2 anova on observed slope

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F         p p<.05         ges
## 2 WMSize   2  30 1.384048 0.2661036       0.002383075
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.8971654 0.4678511
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05      HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9067543 0.2663499           1.024387 0.2661036
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp2'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F            p p<.05        ges
## 2 WMSize   2  30 20.41381 2.533844e-06     * 0.07849941
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9058129 0.5003449
```

```
##
## $`Sphericity Corrections`
##   Effect       GGe        p[GG] p[GG]<.05      HFe        p[HF] p[HF]<.05
## 2 WMSize 0.9139205 6.037556e-06         * 1.034206 2.533844e-06         *
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp3'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05         ges
## 2 WMSize   2  30 0.5498427 0.5827455       0.002157507
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9757984 0.8424043
##
## $`Sphericity Corrections`
##   Effect       GGe    p[GG] p[GG]<.05      HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9763702 0.578772           1.120689 0.5827455
```

```
ezANOVA(data = obs_Inp_list_no_gap %>% filter(Exp =='Exp4a'), dv= slope, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 7.224838 0.002746564     * 0.01905119
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9501229 0.6989697
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05      HFe       p[HF] p[HF]<.05
## 2 WMSize 0.9524924 0.003257332         * 1.087427 0.002746564         *
```

```
ezANOVA(data = obs_Inp_list %>% filter(Exp =='Exp4b'), dv= slope, wid=NSub, within = .(WMSize, gap))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: You have removed one or more levels from variable "gap". Refactoring
## for ANOVA.
```

```
## $ANOVA
##       Effect DFn DFd         F            p p<.05         ges
## 2     WMSize   2  30  9.344539 0.0007003993     * 0.035428935
## 3        gap   1  15 16.352709 0.0010604639     * 0.027930881
## 4 WMSize:gap   2  30  1.635350 0.2117822688       0.002784549
##
## $`Mauchly's Test for Sphericity`
##       Effect         W          p p<.05
## 2     WMSize 0.8669475 0.36808622
## 4 WMSize:gap 0.6136631 0.03277255     *
##
## $`Sphericity Corrections`
##       Effect       GGe       p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2     WMSize 0.8825716 0.001234014         * 0.9914082 0.0007299619         *
```

```
## 4 WMSize:gap 0.7213254 0.219115993           0.7775271 0.2179558985
```

9.2.2 Predicated data

```
#Predicated Indifference Point for Exp.4b
pred_model <- function(df) {
  lm(mu_r ~ curDur, data = df)
}

pred_Inp_list <- AllDat_predY %>%
  dplyr::group_by(NSub, Exp, WMSize, gap) %>% nest()  %>%
  mutate(model = map(data, pred_model)) %>%  # linear regression
  mutate(slope = map(model, broom::tidy)) %>%  # get estimates
  unnest(slope, .drop = TRUE) %>% # remove raw data
  select(-std.error,-statistic, -p.value) %>%  # remove unnessary columns
  spread(term, estimate) %>%   # spread stimates
  dplyr::rename(Intercept = `(Intercept)`, pred_slope = curDur)  # rename columns
pred_Inp_list$model = NULL
pred_Inp_list$data = NULL
pred_Inp_list$pred_inP = pred_Inp_list$Intercept /(1-pred_Inp_list$pred_slope)
```

```
pred_Inp_slope_no_gap <- AllDat_predY %>%
  dplyr::group_by(NSub, Exp, WMSize) %>% nest()  %>%
  mutate(model = map(data, pred_model)) %>%  # linear regression
  mutate(slope = map(model, broom::tidy)) %>%  # get estimates
  unnest(slope, .drop = TRUE) %>% # remove raw data
  select(-std.error,-statistic, -p.value) %>%  # remove unnessary columns
  spread(term, estimate) %>%   # spread stimates
  dplyr::rename(Intercept = `(Intercept)`, pred_slope = curDur)  # rename columns
pred_Inp_slope_no_gap$model = NULL
pred_Inp_slope_no_gap$data = NULL
pred_Inp_slope_no_gap$pred_inP = pred_Inp_slope_no_gap$Intercept /(1-pred_Inp_slope_no_gap$pred_slope)
```

```
m_pred_Inp_slope_no_gap = pred_Inp_slope_no_gap %>% group_by(Exp, WMSize)%>%
  dplyr::summarise(n=n(),
                   m_Intercept = mean(Intercept),
                   se_Intercept= sd(Intercept)/sqrt(n-1),
                   m_pred_inP = mean(pred_inP),
                   se_pred_inP = sd(pred_inP)/sqrt(n-1),
                   m_pred_slope = mean(pred_slope),
                   se_pred_slope = sd(pred_slope)/sqrt(n-1))
```

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

```
# plot the observed indifference points and slopes of RP
plt_pred_InP_slope_err<- ggplot(data = m_pred_Inp_slope_no_gap, aes(x= m_pred_slope, y=m_pred_inP, colo
  geom_line(stat = "identity")+
  geom_errorbar(width = 0.02, aes(ymin = m_pred_inP - se_pred_inP, ymax = m_pred_inP + se_pred_inP)) +
  geom_errorbarh(height =0.02, aes(xmin = m_pred_slope - se_pred_slope, xmax = m_pred_slope + se_pred_s
  geom_point(data = obs_Inp_list%>% group_by(Exp, WMSize)%>%
  dplyr::summarise(n=n(),
                   m_inP = mean(inP),
                   se_inP = sd(inP)/sqrt(n-1),
                   m_slope = mean(slope),
```

```
                  se_slope = sd(slope)/sqrt(n-1)), aes(x= m_slope, y =m_inP, color = WMSize))+
  theme_new+
  labs(colour = "Memory Load")+colorSet3+
  facet_grid(~Exp)+
  xlab('slope of reproduction')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")
```

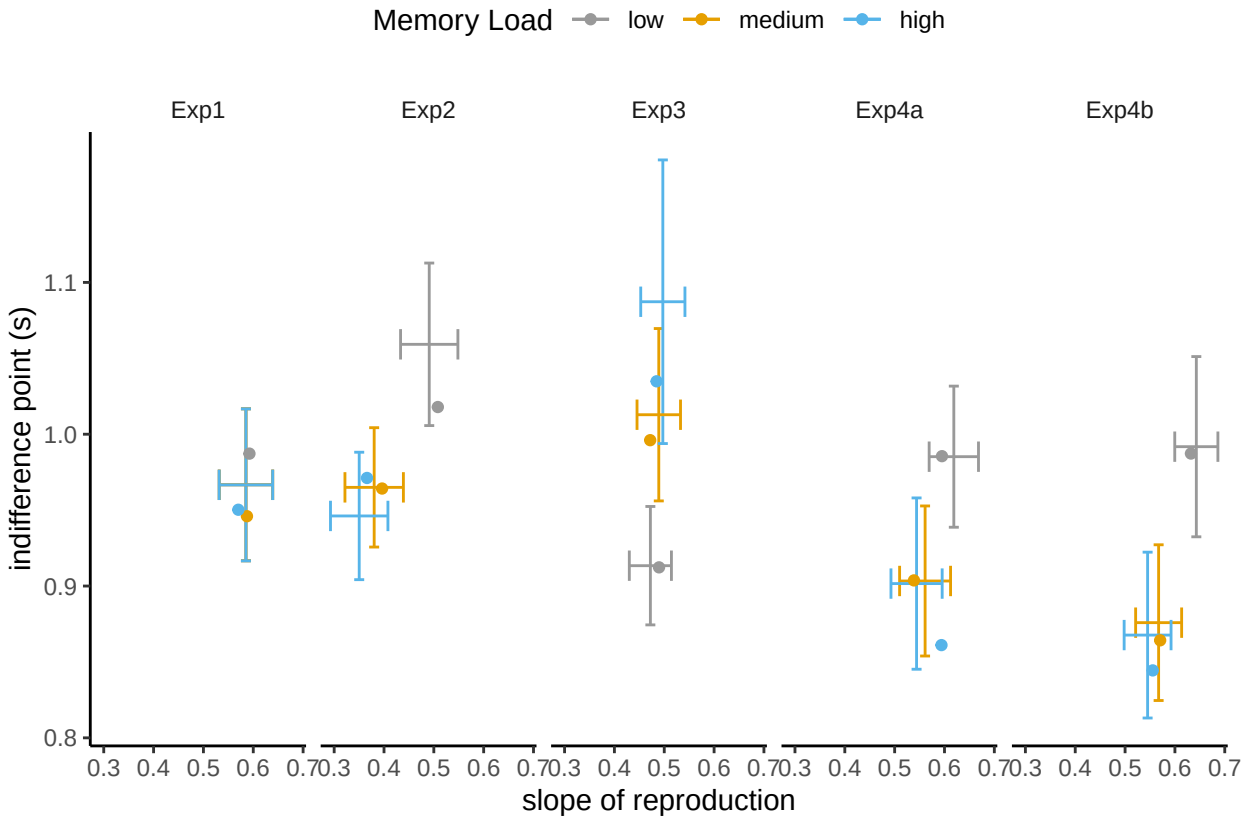
```
## `summarise()` has grouped output by 'Exp'. You can override using the `.groups`
## argument.
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_pred_InP_slope_err.png"), plt_pred_InP_slope_err, w
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
plt_pred_InP_slope_err
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```




```
InP_obs<-  ggplot(data = obs_Inp_list_no_gap %>%dplyr::group_by(WMSize, Exp) %>%dplyr::summarise(m_inP =
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  aes(ymin = m_inP - se_inP, ymax = m_inP + se_inP), position = position_dodge
  labs(colour = "Memory Load")+colorSet3+
  xlab(' ')+ylab("observed indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")
```

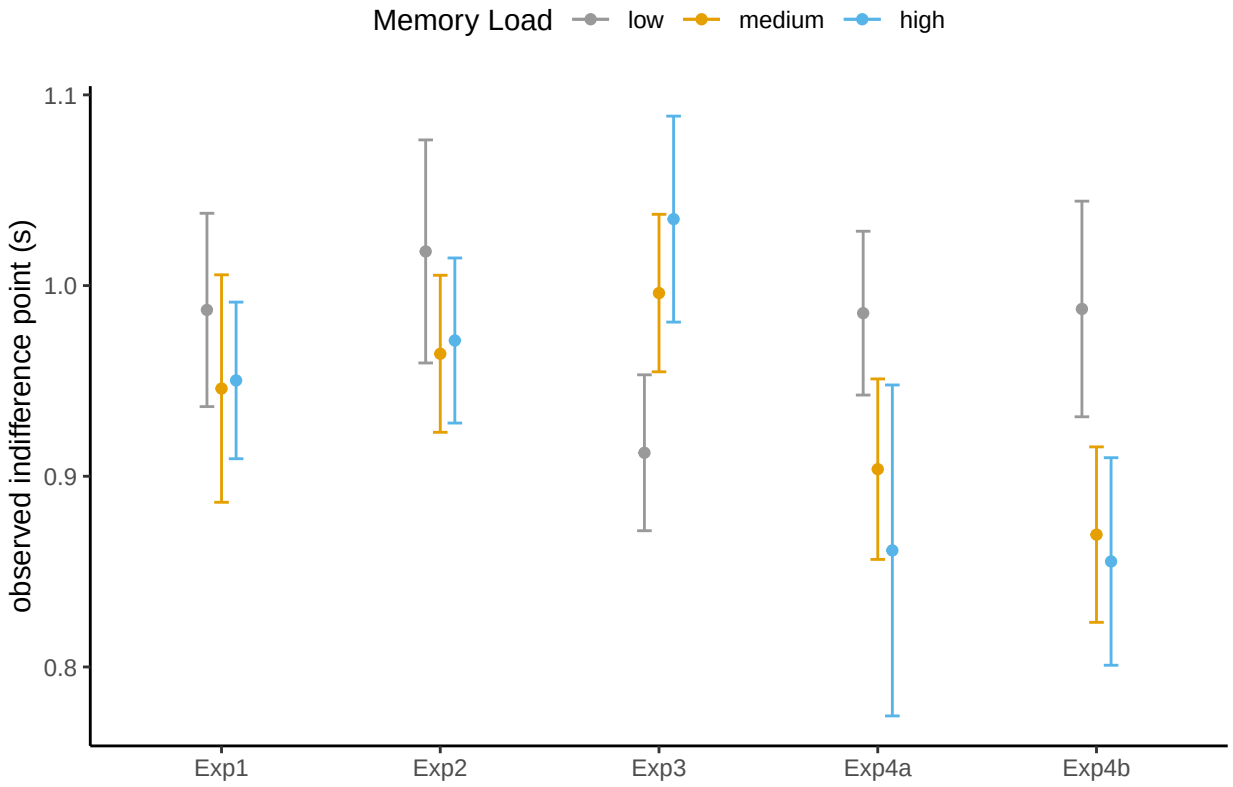
```
## `summarise()` has grouped output by 'WMSize'. You can override using the
## `.groups` argument.
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/InP_obs.png"), InP_obs, width = 3, height = 3)
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
InP_obs
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

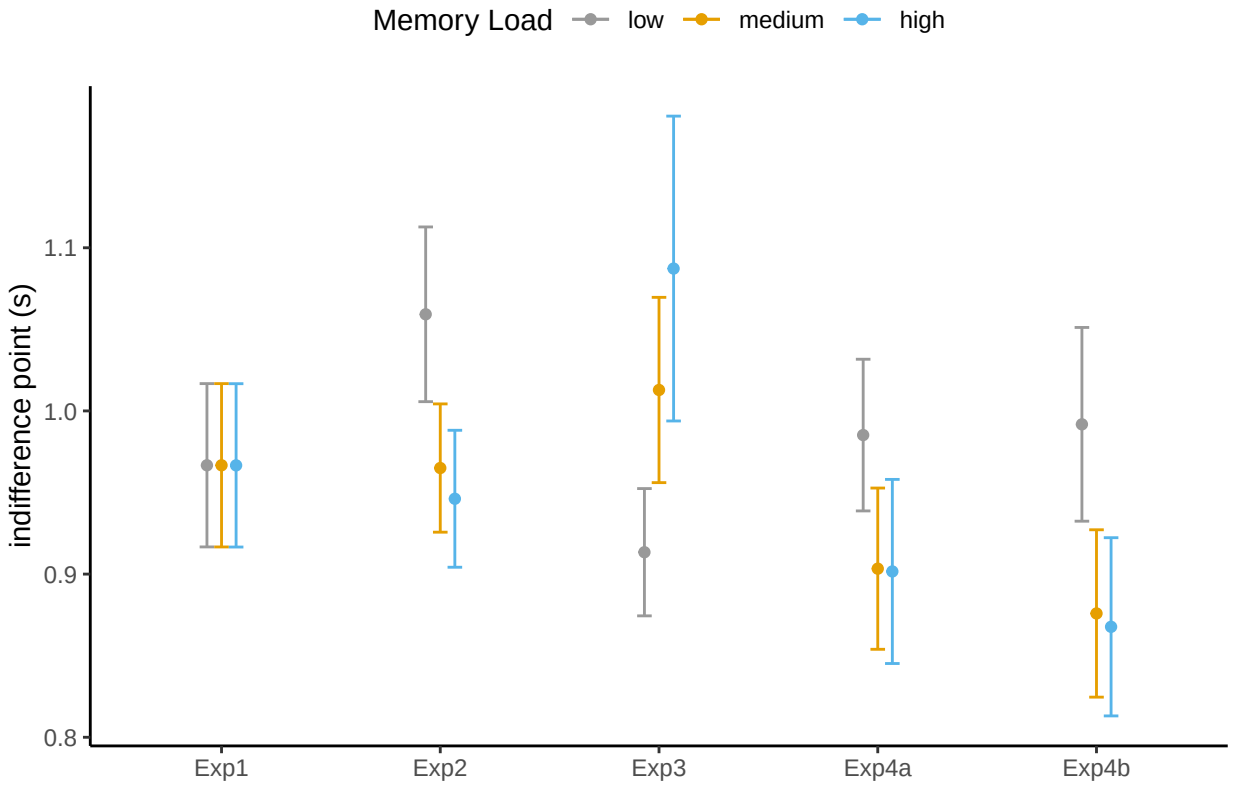



```
InP_pred<-  ggplot(data = m_pred_Inp_slope_no_gap, aes(x= Exp, y=m_pred_inP, color = WMSize))+
  geom_line(stat = "identity",position = position_dodge(width = 0.2))+
  geom_point(stat = "identity",position = position_dodge(width = 0.2))+
  geom_errorbar(width=.2,  aes(ymin = m_pred_inP - se_pred_inP, ymax = m_pred_inP + se_pred_inP), positi
  labs(colour = "Memory Load")+colorSet3+
  xlab(' ')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")


ggsave(paste0(getwd(), "/", modelPath, "/figures/InP_pred.png"), InP_pred, width = 3, height = 3)
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
InP_pred
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```




```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp1'), dv= pred_inP, wid=NSub, within = .(WMSiz
```

9.2.2.1 anova on predicated InP

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05          ges
## 2 WMSize   2  30 0.5370534 0.5899823       1.383912e-08
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9352103 0.6256984
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05     HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9391526 0.5793662           1.06895 0.5899823
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp2'), dv= pred_inP, wid=NSub, within = .(WMSiz
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 5.024835 0.01311701     * 0.07794038
##
## $`Mauchly's Test for Sphericity`
##   Effect           W            p p<.05
## 2 WMSize 0.001101406 1.966225e-21     *
```

```
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.5002755 0.04051021         * 0.5003346 0.04050482         *
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp3'), dv= pred_inP, wid=NSub, within = .(WMSiz
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 4.861495 0.01483166     * 0.07435496
##
## $`Mauchly's Test for Sphericity`
##   Effect           W            p p<.05
## 2 WMSize 0.005556219 1.634766e-16     *
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.5013929 0.04335867         * 0.5016917 0.04333085         *
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp4a'), dv= pred_inP, wid=NSub, within = .(WMSi
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 4.368778 0.02161901     * 0.04002161
##
## $`Mauchly's Test for Sphericity`
##   Effect          W            p p<.05
## 2 WMSize 0.01959382 1.108752e-12     *
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.5049469 0.05356217           0.5060112 0.05345865
```

```
ezANOVA(data = pred_Inp_list %>% filter(Exp =='Exp4b'), dv= pred_inP, wid=NSub, within = .(WMSize, gap)
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: You have removed one or more levels from variable "gap". Refactoring
## for ANOVA.
```

```
## $ANOVA
##       Effect DFn DFd         F            p p<.05         ges
## 2     WMSize   2  30  8.176676 1.464204e-03     * 0.069841635
## 3        gap   1  15 21.824653 3.010862e-04     * 0.004110952
## 4 WMSize:gap   2  30 19.346529 4.009831e-06     * 0.001823944
##
## $`Mauchly's Test for Sphericity`
##       Effect          W            p p<.05
## 2     WMSize 0.02048631 1.514420e-12     *
## 4 WMSize:gap 0.04046991 1.777976e-10     *
##
## $`Sphericity Corrections`
##       Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2     WMSize 0.5051746 0.0116757954         * 0.5062881 0.0116207400         *
```

```
## 4 WMSize:gap 0.5103264 0.0004684234         * 0.5125578 0.0004582596         *
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp1'), dv= pred_slope, wid=NSub, within = .(WMS
```

9.2.2.2 anova on predicated slope

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F         p p<.05          ges
## 2 WMSize   2  30 1.157496 0.3279124       3.874751e-08
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9444019 0.6700358
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05      HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9473302 0.3262934           1.080268 0.3279124
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp2'), dv= pred_slope, wid=NSub, within = .(WMS
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 78.75142 1.15266e-12     * 0.07176615
##
## $`Mauchly's Test for Sphericity`
##   Effect          W            p p<.05
## 2 WMSize 0.06006657 2.821175e-09     *
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2 WMSize 0.5154816 1.60398e-07         * 0.5188408 1.476789e-07         *
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp3'), dv= pred_slope, wid=NSub, within = .(WMS
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F            p p<.05         ges
## 2 WMSize   2  30 39.89814 3.531867e-09     * 0.004080136
##
## $`Mauchly's Test for Sphericity`
##   Effect           W            p p<.05
## 2 WMSize 0.002410559 4.729598e-19     *
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2 WMSize 0.5006034 1.36904e-05         * 0.5007327 1.366088e-05         *
```

```
ezANOVA(data = pred_Inp_slope_no_gap %>% filter(Exp =='Exp4a'), dv= pred_slope, wid=NSub, within = .(WM
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F            p p<.05        ges
```

```
## 2 WMSize   2  30 76.72359 1.600123e-12     * 0.02752745
##
## $`Mauchly's Test for Sphericity`
##   Effect          W           p p<.05
## 2 WMSize 0.02624338 8.573113e-12     *
##
## $`Sphericity Corrections`
##   Effect       GGe        p[GG] p[GG]<.05       HFe        p[HF] p[HF]<.05
## 2 WMSize 0.5066481 2.357933e-07         * 0.5080803 2.277348e-07         *
```

```
#ezANOVA(data = pred_Inp_list %>% filter(Exp =='Exp4b'), dv= pred_inP, wid=NSub, within = .(WMSize, gap
```

9.2.3 calculate the predication error

```
Inp_list_no_gap = left_join(obs_Inp_list_no_gap, pred_Inp_slope_no_gap, by = c("NSub", "Exp", "WMSize"))
Inp_list_no_gap$InP_err = Inp_list_no_gap$pred_inP -Inp_list_no_gap$inP
Inp_list_no_gap$InP_rerr = 100*Inp_list_no_gap$InP_err/ Inp_list_no_gap$inP


Inp_list_no_gap$slope_err = Inp_list_no_gap$pred_slope - Inp_list_no_gap$slope
Inp_list_no_gap$slope_rerr = 100* Inp_list_no_gap$slope_err/Inp_list_no_gap$slope

m_Inp_list_no_gap = Inp_list_no_gap %>% dplyr::group_by(Exp) %>% dplyr::summarise(m_InP_rerr = mean(InP

m_Inp_list_no_gap$InP_auc = 100- m_Inp_list_no_gap$m_InP_rerr_abs
m_Inp_list_no_gap$slope_auc = 100-  m_Inp_list_no_gap$m_slope_rerr_abs
```

9.3 Indifference Point (bootstraps)

```
# Custom function to find predicted indifference point
getPredInP_boot <- function(df, idx){
  vars <- c('NSub', 'Exp', 'WMSize')
  gp_vars = syms(vars)
  slopes <- df[idx, ] %>%
    dplyr::group_by(!!!gp_vars) %>% nest()  %>%  # nested data
    mutate(model = map(data, pred_model)) %>%  # linear regression
    mutate(slope = map(model, broom::tidy)) %>%  # get estimates out
    unnest(slope, .drop = TRUE) %>% # remove raw data
    select(-std.error,-statistic, -p.value) %>%  # remove unnessary clumns
    spread(term, estimate) %>%   # spread stimates
    dplyr::rename(minRP = `(Intercept)`, slope = curDur)  # rename columns
  slopes$inP = slopes$minRP /(1-slopes$slope)
  return(c(slopes$inP, slopes$slope))
}


# Custom function to find observed indifference point
getRPInP_boot <- function(df, idx){
  vars <- c('NSub', 'Exp', 'WMSize')
  gp_vars = syms(vars)
  slopes <- df[idx, ] %>%
    dplyr::group_by(!!!gp_vars) %>% nest()  %>%  # nested data
    mutate(model = map(data, obs_model)) %>%  # linear regression
```

```
    mutate(slope = map(model, broom::tidy)) %>%  # get estimates out
    unnest(slope, .drop = TRUE) %>% # remove raw data
    select(-std.error,-statistic, -p.value) %>%  # remove unnessary clumns
    spread(term, estimate) %>%   # spread stimates
    dplyr::rename(minRP = `(Intercept)`, slope = curDur)  # rename columns
  slopes$inP = slopes$minRP /(1-slopes$slope)
  return(c(slopes$inP, slopes$slope))
}
```

```
#calculate the bootstrapped 95% confidence intervals
generateCI = FALSE # tag for generation CI
if(generateCI){
  cilist <- NULL
  for(expname in unique(AllDat_predY$Exp)){
    for(nsub in unique(AllDat_predY$NSub)){
      for(WMsize in unique(AllDat_predY$WMSize)){
        dat = AllDat_predY %>% filter(Exp == expname, NSub == nsub, WMSize ==WMsize)
        set.seed(100)
        num = 1000
        bs_predInP <- boot(dat, getPredInP_boot, R = num)
        bs_RPInP <- boot(dat, getRPInP_boot, R = num)
        ci = data.frame(
          Exp = expname,
          NSub = nsub,
          WMSize = WMsize,
          sd_predInP_boot = sd(bs_predInP$t[,1]),
          m_predInP_boot = median(bs_predInP$t[,1]),
          sd_RPInP_boot = sd(bs_RPInP$t[,1]),
          mRP_InP_boot = median(bs_RPInP$t[,1]),
          sd_pred_slope_boot = sd(bs_predInP$t[,2]),
          m_pred_slope_boot = median(bs_predInP$t[,2]),
          sd_RP_slope_boot = sd(bs_RPInP$t[,2]),
          mRP_slope_boot = median(bs_RPInP$t[,2])
        )
        cilist = data.frame(rbind(cilist, ci))
      }
    }
  }
  write.csv(cilist, file = paste0("ci_list_median_1000.csv"))
}
```

```
# load the generated indifference point values and mark the outlier
cilist = read.csv(paste0("ci_list_median_1000.csv"))
cilist$Exp = as.factor(cilist$Exp)
cilist$WMSize= as.factor(cilist$WMSize)
cilist$inPOutlier = FALSE
cilist[which(cilist$mRP_InP_boot > 1.7 |cilist$mRP_InP_boot < 0.5 | cilist$m_predInP_boot < 0.5|cilist$m

#check if the outlier is the same as the outliers in variable slope_pr
cilist%>% filter(inPOutlier == TRUE)
```

```
##     X   Exp NSub WMSize sd_predInP_boot m_predInP_boot sd_RPInP_boot
## 1  34  Exp1   12 medium     0.006375669      0.5656030    3.86379115
## 2 111  Exp3    5   high     0.047276315      2.3286653    3.73026121
```

```
## 3 174 Exp4a   10   high     0.006521927      0.5408569   18.74752865
## 4 192 Exp4a   16   high     0.005482814      0.4780972    0.08010245
##   mRP_InP_boot sd_pred_slope_boot m_pred_slope_boot sd_RP_slope_boot
## 1    0.2385747        0.001152515         0.8735302       0.03800103
## 2    1.5867678        0.001112801         0.9708236       0.03448870
## 3    0.3606472        0.001119425         0.8743957       0.05437083
## 4    0.4603227        0.002504120         0.6482563       0.04006518
##   mRP_slope_boot inPOutlier
## 1      0.8958322       TRUE
## 2      0.9448729       TRUE
## 3      0.9595400       TRUE
## 4      0.6677330       TRUE
```

```
library(Rmisc)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
## Loading required package: plyr
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:ggpubr':
##
##     mutate
```

```
## The following objects are masked from 'package:rstatix':
##
##     desc, mutate
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

```
mCI <- cilist%>% filter(inPOutlier == FALSE) %>% dplyr::group_by(Exp, WMSize) %>%
  dplyr::summarise(n = n(),
                   m_RPInP_boot = mean(mRP_InP_boot),
                   m_predInP_boot = mean(m_predInP_boot),
                   m_sd_predInP_boot = mean(sd_predInP_boot),
                   m_sd_RPInP_boot= mean(sd_RPInP_boot),
                   m_RPSlope_boot = mean(mRP_slope_boot),
```

```
                  m_predSlope_boot = mean(m_pred_slope_boot),
                  m_sd_predSlope_boot = mean(sd_pred_slope_boot),
                  m_sd_RPSlope_boot= mean(sd_RP_slope_boot))
```

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

```
ezANOVA(data = cilist %>% filter(Exp =='Exp1'), dv=mRP_InP_boot, wid=NSub, within = .(WMSize))
```

9.3.0.1 anova on InP

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05         ges
## 2 WMSize   2  30 0.4850599 0.6204053       0.004762842
##
## $`Mauchly's Test for Sphericity`
##   Effect         W          p p<.05
## 2 WMSize 0.5333264 0.01227302     *
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.6818149 0.5514381           0.7266625 0.5628124
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp1'), dv=m_predInP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 2 WMSize   2  30 0.5300565 0.5939821       4.4502e-08
##
## $`Mauchly's Test for Sphericity`
##   Effect         W         p p<.05
## 2 WMSize 0.9194148 0.5553675
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05      HFe     p[HF] p[HF]<.05
## 2 WMSize 0.9254245 0.5807494           1.050014 0.5939821
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp2'), dv=mRP_InP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 2 WMSize   2  30 0.7605213 0.4762228       0.01376635
##
## $`Mauchly's Test for Sphericity`
##   Effect        W           p p<.05
## 2 WMSize 0.382407 0.001195862     *
##
## $`Sphericity Corrections`
##   Effect       GGe     p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.6182025 0.4213848           0.6459969 0.4264353
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp2'), dv=m_predInP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05       ges
## 2 WMSize   2  30 5.027847 0.01308745     * 0.0778555
##
## $`Mauchly's Test for Sphericity`
##   Effect           W            p p<.05
## 2 WMSize 0.001021933 1.164013e-21     *
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.5002556 0.04045981         * 0.5003104 0.04045481         *
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp3'), dv= mRP_InP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F           p p<.05        ges
## 2 WMSize   2  30 8.285907 0.001364493     * 0.08288682
##
## $`Mauchly's Test for Sphericity`
##   Effect         W          p p<.05
## 2 WMSize 0.5473213 0.01471284     *
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05       HFe       p[HF] p[HF]<.05
## 2 WMSize 0.6883835 0.005114773         * 0.7350777 0.004190498         *
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp3'), dv=m_predInP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 4.880417 0.01462132     * 0.07442553
##
## $`Mauchly's Test for Sphericity`
##   Effect          W            p p<.05
## 2 WMSize 0.00559242 1.710798e-16     *
##
## $`Sphericity Corrections`
##   Effect      GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.501402 0.04300346         * 0.5017028 0.04297553         *
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp4a'), dv= mRP_InP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 4.082462 0.02703042     * 0.04458098
##
## $`Mauchly's Test for Sphericity`
##   Effect         W           p p<.05
```

```
## 2 WMSize 0.4969286 0.007482688     *
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe     p[HF] p[HF]<.05
## 2 WMSize 0.6653044 0.04694059         * 0.7055815 0.0439202         *
```

```
ezANOVA(data = cilist %>% filter(Exp =='Exp4a'), dv=m_predInP_boot, wid=NSub, within = .(WMSize))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05        ges
## 2 WMSize   2  30 4.366367 0.02165941     * 0.03999416
##
## $`Mauchly's Test for Sphericity`
##   Effect          W            p p<.05
## 2 WMSize 0.02003665 1.296511e-12     *
##
## $`Sphericity Corrections`
##   Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
## 2 WMSize 0.5050599 0.05360968           0.5061486 0.05350379
```

```
#ezANOVA(data = cilist_exp4b %>% filter(Exp =='Exp4b'), dv= pred_inP, wid=NSub, within = .(WMSize, gap),
```

9.4 plot figures in manuscript

```
#plot the predicated indifference points and slope of predicated RP
plt_pred_InP_slope_err<- ggplot(data = obs_Inp_list_no_gap%>% dplyr::group_by(Exp, WMSize)%>%
  dplyr::summarise(n=n(),
                   m_inP = mean(inP),
                   se_inP = sd(inP)/sqrt(n-1),
                   m_slope = mean(slope),
                   se_slope = sd(slope)/sqrt(n-1)), aes(x= m_slope, y=m_inP, color = WMSize))+
  geom_line(stat = "identity")+
  geom_point(stat = "identity")+
  geom_errorbar(width = 0.02, aes(ymin = m_inP - se_inP, ymax = m_inP + se_inP)) +
  geom_errorbarh(height =0.02, aes(xmin = m_slope - se_slope, xmax = m_slope + se_slope)) +
  theme_new+
  labs(colour = "Memory Load")+colorSet3+
  facet_grid(~Exp)+
  xlab('slope of reproduction')+ylab("indifference point (s)")+guides(shape="none")+
  theme(legend.position = "top")
```

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

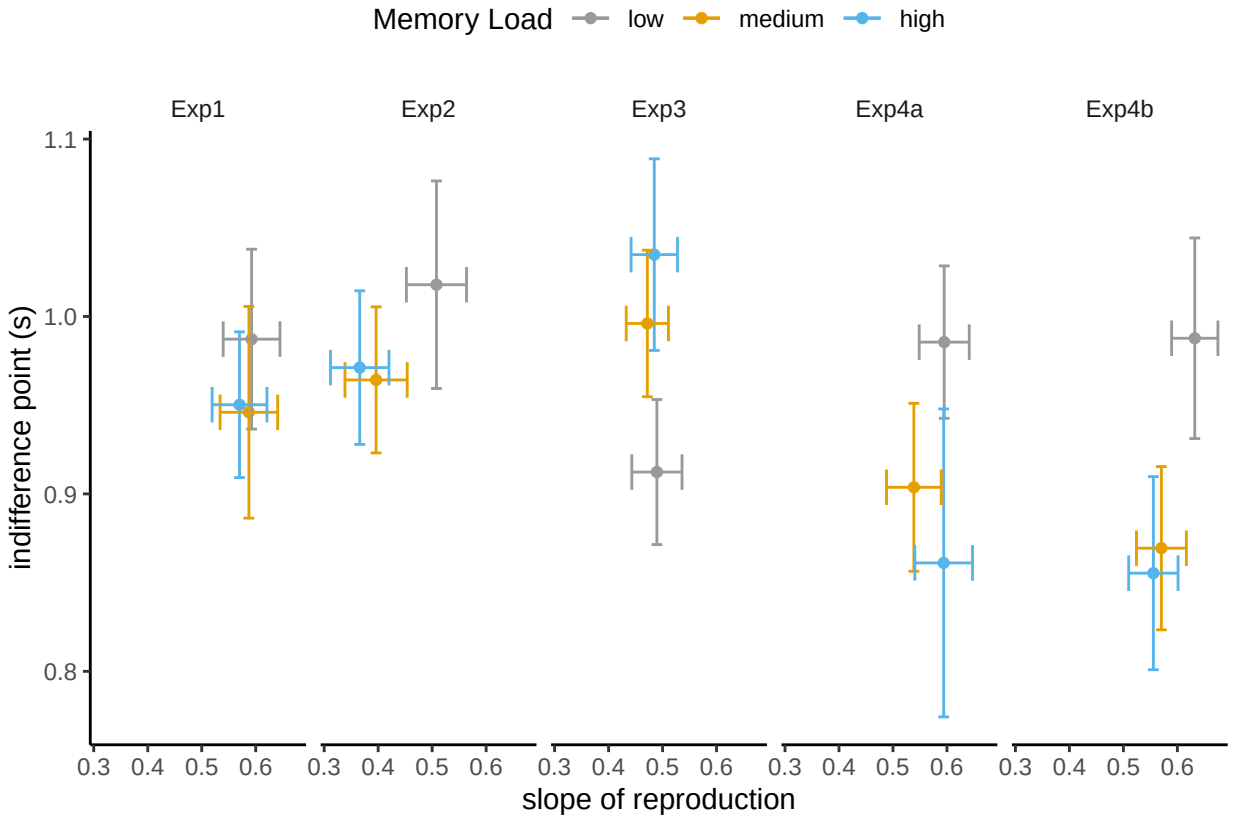
```
ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_pred_InP_slope_err.png"), plt_pred_InP_slope_err,
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
plt_pred_InP_slope_err
```

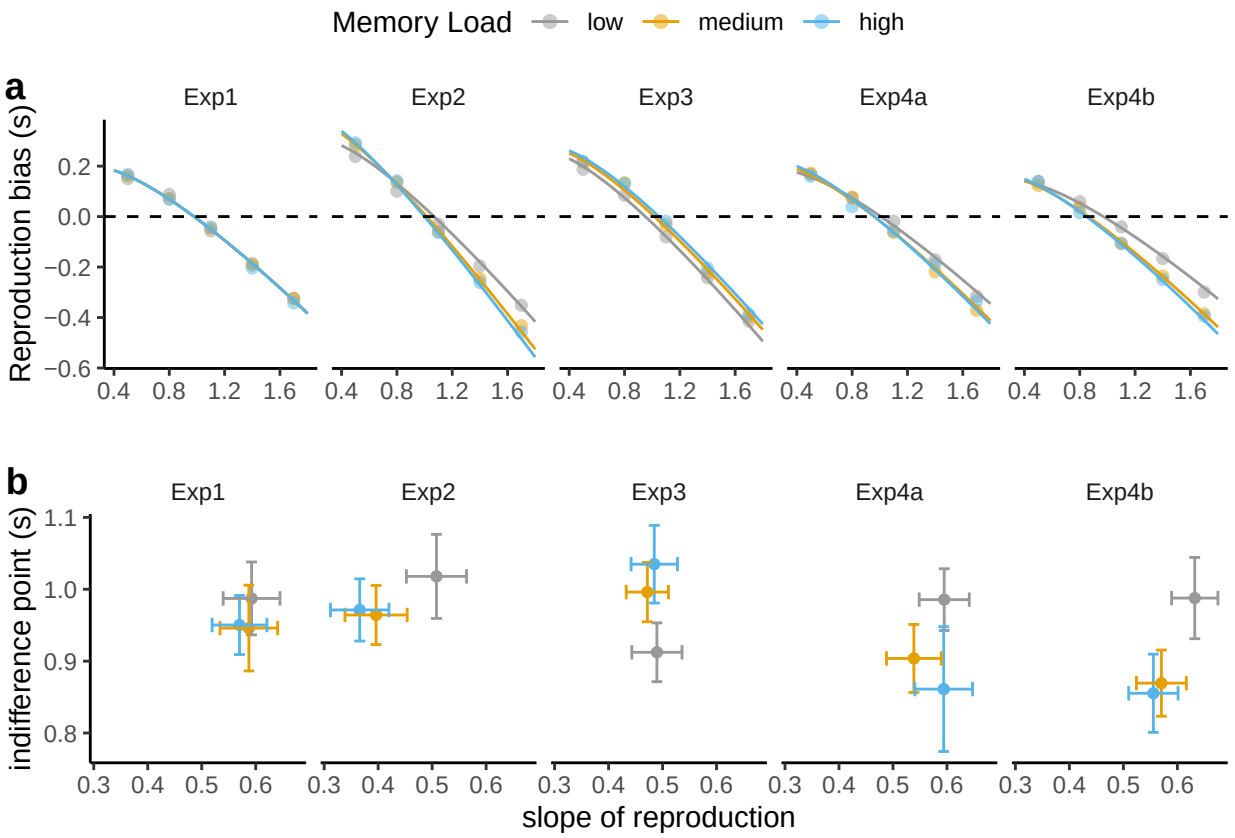
```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```




```
## Figures in the MS
fig3<-ggarrange(RP_bias, plt_pred_InP_slope_err, common.legend = TRUE, ncol=1, nrow=2,  labels = c("a",
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/fig3.png"), fig3, width = 6, height = 5)
fig3
```



10 anova analysis

10.1 Anova on mean reproduction biases

```
mpredY_sub$RP_Bias = mpredY_sub$m_repDur-mpredY_sub$curDur
RP_bias_Anova <- ezANOVA(data = mpredY_sub, dv= RP_Bias, wid=NSub, within= .(curDur, WMSize), between =
```

```
## Warning: Converting "NSub" to factor for ANOVA.

## Warning: "curDur" will be treated as numeric.

## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.

## Warning: There is at least one numeric within variable, therefore aov() will be
## used for computation and no assumption checks will be obtained.
```

```
RP_bias_Anova
```

```
## $ANOVA
##              Effect DFn DFd         SSn       SSd           F            p
## 1               Exp   4  75  0.20632235 6.7357598   0.5743293 6.820927e-01
## 2            curDur   1  75 47.70152785 7.0760974 505.5914872 4.589992e-35
## 4            WMSize   2 150  0.04568339 0.6961599   4.9216486 8.506722e-03
```

```
## 3        Exp:curDur   4  75  0.94184534 7.0760974   2.4956695 4.985449e-02
## 5        Exp:WMSize   8 150  0.25075426 0.6961599   6.7536818 1.507627e-07
## 6     curDur:WMSize   2 150  0.11975287 0.3666986  24.4927739 6.240642e-10
## 7 Exp:curDur:WMSize   8 150  0.12549340 0.3666986   6.4167178 3.670623e-07
##   p<.05         ges
## 1       0.013680912
## 2     * 0.762294526
## 4     * 0.003061808
## 3     * 0.059548049
## 5     * 0.016578279
## 6     * 0.007986470
## 7     * 0.008366110
```

```
# main effect of Duration  F(1.177, 3.532) = 377.965, p < .001,  p² = .863.
(RP_bias_Anova$ANOVA)$DFn[3] *(RP_bias_Anova$`Sphericity Corrections`)$GGe[1]
```

```
## numeric(0)
```

```
(RP_bias_Anova$ANOVA)$DFd[3] *(RP_bias_Anova$`Sphericity Corrections`)$GGe[1]
```

```
## numeric(0)
```

```
#Duration × Experiment,  F(12, 240) = 2.506, p = .004,  p² = .111
(RP_bias_Anova$ANOVA)$DFn[5] *(RP_bias_Anova$`Sphericity Corrections`)$GGe[2]
```

```
## numeric(0)
```

```
(RP_bias_Anova$ANOVA)$DFd[5] *(RP_bias_Anova$`Sphericity Corrections`)$GGe[2]
```

```
## numeric(0)
```

```
mpredY_sub <- ungroup(mpredY_sub)
res.aov <-  rstatix::anova_test(data = mpredY_sub, dv = RP_Bias, wid = NSub, within = c(curDur, WMSize)
```

```
## Warning: The 'wid' column contains duplicate ids across between-subjects
## variables. Automatic unique id will be created
```

```
get_anova_table(res.aov, correction = "GG")
```

```
## ANOVA Table (type II tests)
##
##              Effect   DFn    DFd       F        p p<.05   ges
## 1               Exp  4.00  75.00   0.574 6.82e-01       0.012
## 2            curDur  1.17  87.82 467.114 2.76e-39     * 0.745
## 3            WMSize  1.66 124.80   4.922 1.30e-02     * 0.003
## 4        Exp:curDur  4.68  87.82   2.395 4.70e-02     * 0.056
## 5        Exp:WMSize  6.66 124.80   6.754 1.35e-06     * 0.015
## 6     curDur:WMSize  6.65 498.39   7.228 5.84e-08     * 0.008
## 7 Exp:curDur:WMSize 26.58 498.39   2.257 3.96e-04     * 0.010
```

10.2 variance of prior (anova)

```
ezANOVA(data = Bayparlist, dv= sig_pr2_log, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## Coefficient covariances computed by hccm()
```

```
## $ANOVA
##   Effect DFn DFd         F       p p<.05        ges
## 1    Exp   4  75 0.5384571 0.70791       0.02791603
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd        SSn      SSd         F         p p<.05
## 1   4  75 0.04608107 1.121874 0.7701578 0.5480179
```

10.3 variance of motor noise (anova)

```
ezANOVA(data = Bayparlist, dv= sig_mn2, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## Coefficient covariances computed by hccm()
```

```
## $ANOVA
##   Effect DFn DFd        F         p p<.05        ges
## 1    Exp   4  75 0.602902 0.6617231       0.03115306
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd         SSn        SSd        F         p p<.05
## 1   4  75 0.001201703 0.03964056 0.568406 0.6863391
```

10.4 ls (anova)

```
ezANOVA(data = Bayparlist, dv= ls, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## Coefficient covariances computed by hccm()
```

```
## $ANOVA
##   Effect DFn DFd        F            p p<.05       ges
## 1    Exp   4  75 6.948456 8.182695e-05     * 0.2703842
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd        SSn       SSd        F            p p<.05
## 1   4  75 0.09541916 0.3464434 5.164218 0.0009928616     *
```

10.5 ts (anova)

```
ezANOVA(data = Bayparlist, dv= ts, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
```

```
## generating unique wid labels.

## Coefficient covariances computed by hccm()

## $ANOVA
##   Effect DFn DFd        F         p p<.05       ges
## 1    Exp   4  75 1.839334 0.1301531       0.0893343
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd         SSn         SSd        F         p p<.05
## 1   4  75 0.000555571 0.006384121 1.631698 0.1751162
```

10.6 Ks anova

```
ezANOVA(data = Bayparlist, dv= ks, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.

## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.

## Coefficient covariances computed by hccm()

## $ANOVA
##   Effect DFn DFd        F            p p<.05       ges
## 1    Exp   4  75 85.93971 3.114241e-27     * 0.8208993
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd       SSn       SSd        F            p p<.05
## 1   4  75 0.1584934 0.3319383 8.952723 5.765718e-06     *
```

10.7 mean of prior (anova)

```
ezANOVA(data = Bayparlist, dv= mu_pr, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.

## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.

## Coefficient covariances computed by hccm()

## $ANOVA
##   Effect DFn DFd         F         p p<.05        ges
## 1    Exp   4  75 0.9353417 0.4482419       0.04751463
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd        SSn      SSd         F         p p<.05
## 1   4  75 0.02732829 1.139767 0.4495704 0.7723812
```

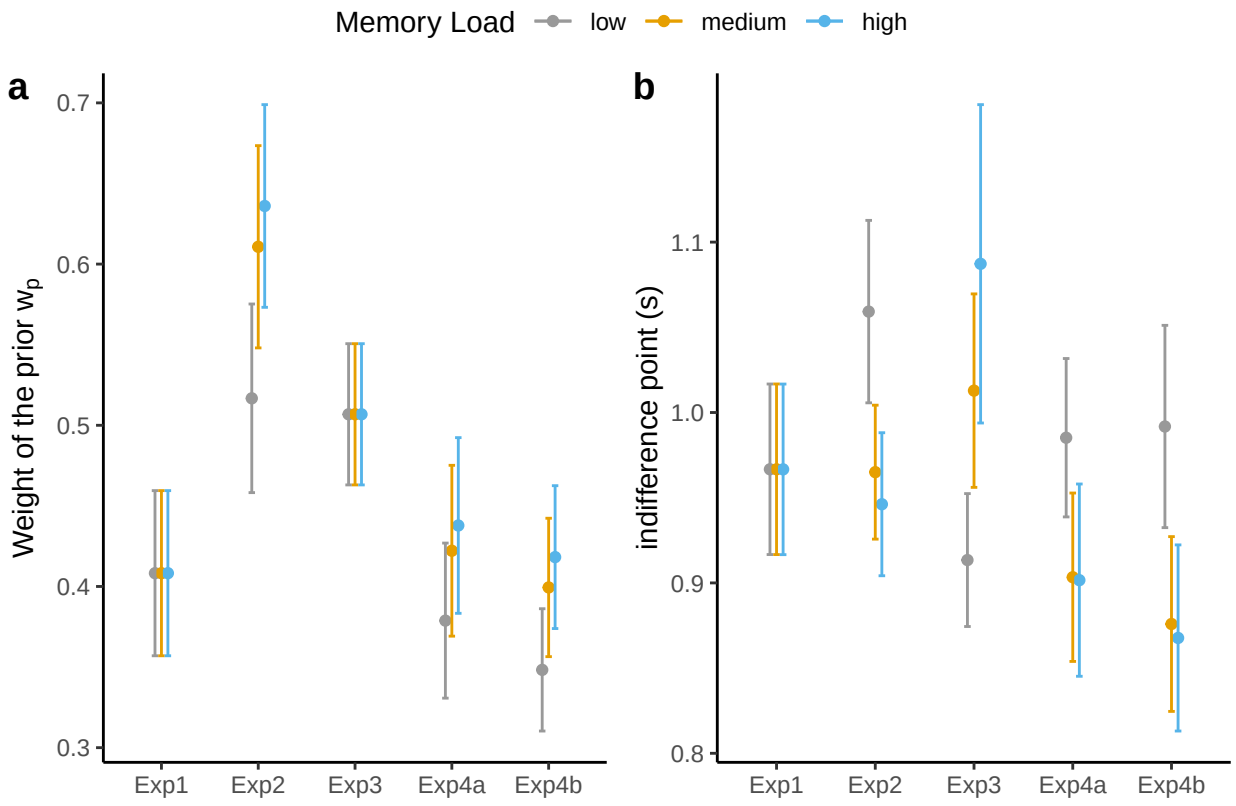
10.8 combine InP and wp

```
fig4<-ggarrange(plt_wp, InP_pred, common.legend = TRUE, ncol=2, nrow=1,  labels = c("a", "b"))
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

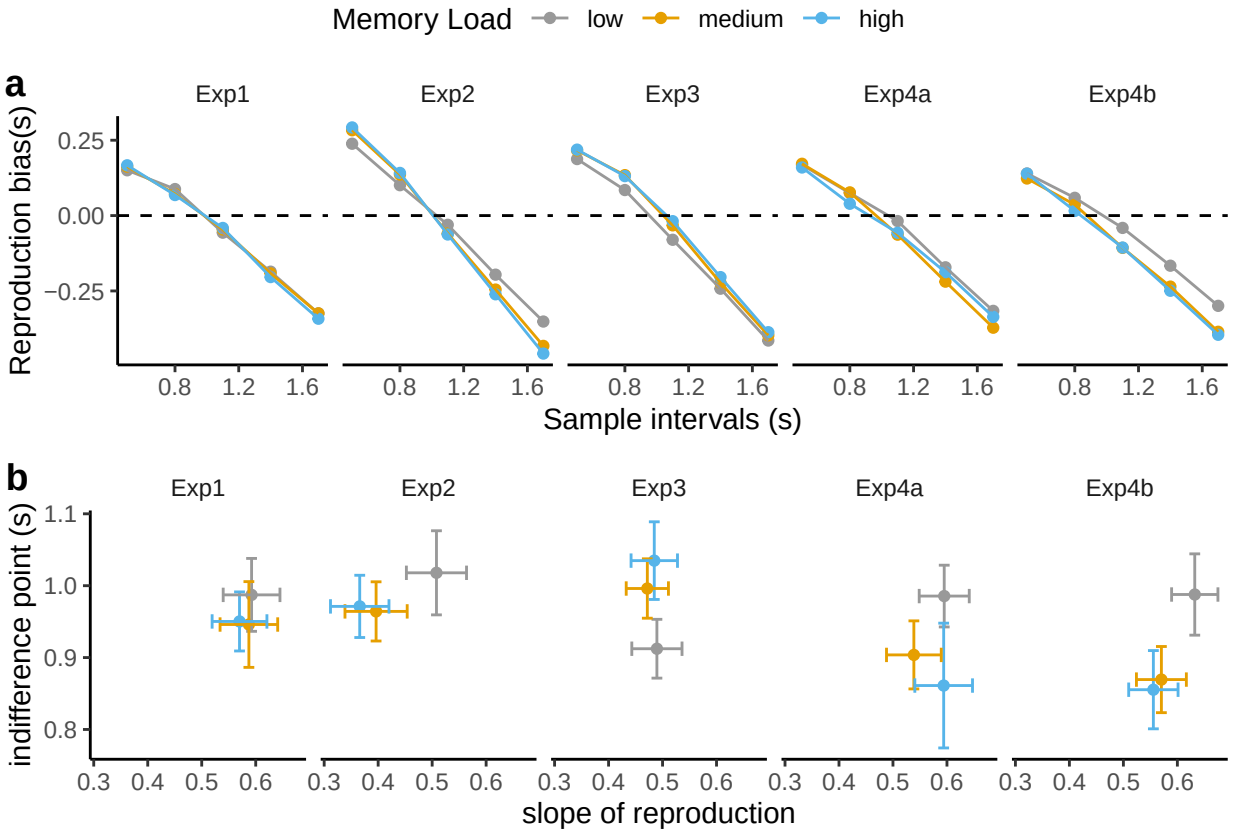
```
ggsave(paste0(getwd(), "/", modelPath, "/figures/fig4.png"), fig4, width = 6, height = 3)
fig4
```




```
fig5<-ggarrange(RP_bias_obs, plt_pred_InP_slope_err, common.legend = TRUE, ncol=1, nrow=2, labels = c("
```

```
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
## geom_path: Each group consists of only one observation. Do you need to adjust
## the group aesthetic?
```

```
ggsave(paste0(getwd(), "/", modelPath, "/figures/fig5.png"), fig5, width = 6, height = 5)
fig5
```



10.9 standard variance of Ds σ_s

```
ezANOVA(data = Bayparlist, dv= sig_s2, wid=NSub, between = .(Exp))
```

```
## Warning: Converting "NSub" to factor for ANOVA.
```

```
## Warning: The column supplied as the wid variable contains non-unique values
## across levels of the supplied between-Ss variables. Automatically fixing this by
## generating unique wid labels.
```

```
## Coefficient covariances computed by hccm()
```

```
## $ANOVA
##   Effect DFn DFd        F          p p<.05       ges
## 1    Exp   4  75 3.409153 0.01287802     * 0.1538485
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd        SSn       SSd        F          p p<.05
## 1   4  75 0.01726871 0.1448521 2.235302 0.07314935
```

```
##independent T test
t.test((Bayparlist%>%filter(Exp %in%c('Exp1')))$sig_s2)
```

```
##
##  One Sample t-test
##
## data:  (Bayparlist %>% filter(Exp %in% c("Exp1")))$sig_s2
## t = 7.2054, df = 15, p-value = 3.046e-06
## alternative hypothesis: true mean is not equal to 0
```

```
## 95 percent confidence interval:
##  0.02398402 0.04413435
## sample estimates:
##  mean of x
## 0.03405919
```

```
t.test((Bayparlist%>%filter(Exp %in%c('Exp2')))$sig_s2)
```

```
##
##  One Sample t-test
##
## data:  (Bayparlist %>% filter(Exp %in% c("Exp2")))$sig_s2
## t = 3.0831, df = 15, p-value = 0.007574
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.0247191 0.1354437
## sample estimates:
##  mean of x
## 0.08008139
```

```
t.test((Bayparlist%>%filter(Exp %in%c('Exp3')))$sig_s2)
```

```
##
##  One Sample t-test
##
## data:  (Bayparlist %>% filter(Exp %in% c("Exp3")))$sig_s2
## t = 11.682, df = 15, p-value = 6.232e-09
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.03170408 0.04585477
## sample estimates:
##  mean of x
## 0.03877943
```

```
t.test((Bayparlist%>%filter(Exp %in%c('Exp4a')))$sig_s2)
```

```
##
##  One Sample t-test
##
## data:  (Bayparlist %>% filter(Exp %in% c("Exp4a")))$sig_s2
## t = 6.5979, df = 15, p-value = 8.467e-06
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.02042601 0.03992116
## sample estimates:
##  mean of x
## 0.03017359
```

```
t.test((Bayparlist%>%filter(Exp %in%c('Exp4b')))$sig_s2)
```

```
##
##  One Sample t-test
##
## data:  (Bayparlist %>% filter(Exp %in% c("Exp4b")))$sig_s2
## t = 4.5291, df = 15, p-value = 0.0003995
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
```

```
##  0.01172269 0.03256513
## sample estimates:
##  mean of x
## 0.02214391
```

11 Model prediction error

```
m_predErr_sub<- mpredY_sub%>%
  dplyr::group_by(Exp, WMSize, NSub) %>% dplyr::summarise(
    mpredRP_err=mean(predRP_err),
    mpredVar_err=mean(predVar_err),
    mpredcv_err = mean(predcv_err),
    mpredRP_rerr = mean(predRP_rerr),
    mpredVar_rerr = mean(predVar_rerr),
    mpredcv_rerr = mean(predcv_rerr))
```

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

```
m_predErr<- m_predY%>%
  dplyr::group_by(Exp, WMSize) %>% dplyr::summarise(
    mmpredcv_err = mean(mpredcv_err),
    mmpredRP_err=mean(mpredRP_err),
    mmpredVar_err=mean(mpredVar_err),
    mmpredRP_rerr = mean(mpredRP_rerr),
    mmpredVar_rerr = mean(mpredVar_rerr),
    mmpredcv_rerr = mean(mpredcv_rerr))
```

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

```
m_predErr
```

```
## # A tibble: 15 x 8
## # Groups:   Exp [5]
##    Exp   WMSize mmpredcv_err mmpredRP_err mmpredVar_err mmpredRP_rerr
##    <fct> <fct>         <dbl>        <dbl>         <dbl>         <dbl>
##  1 Exp1  low        0.00594     -0.00193       0.00675         0.0283
##  2 Exp1  medium     0.0128      -0.00205       0.0117          0.0384
##  3 Exp1  high       0.00520      0.00274       0.00487         0.0333
##  4 Exp2  low        0.000861     0.0141        0.00576         0.0393
##  5 Exp2  medium     0.00862     -0.00541       0.00727         0.0364
##  6 Exp2  high       0.0135      -0.0111        0.0112          0.0494
##  7 Exp3  low        0.00725     -0.00433       0.00584         0.0361
##  8 Exp3  medium     0.0102      -0.00196       0.0113          0.0322
##  9 Exp3  high       0.00276      0.00645       0.00401         0.0339
## 10 Exp4a low        0.00833     -0.000218      0.00670         0.0303
## 11 Exp4a medium     0.00276      0.00211       0.00406         0.0354
## 12 Exp4a high       0.0144      -0.00363       0.0147          0.0383
## 13 Exp4b low        0.0153      -0.00151       0.0132          0.0210
## 14 Exp4b medium     0.0156       0.00147       0.0141          0.0321
## 15 Exp4b high       0.00117     -0.00130       0.000249        0.0310
## # ... with 2 more variables: mmpredVar_rerr <dbl>, mmpredcv_rerr <dbl>
```

12 model comparison (logarithmic vs. linear)

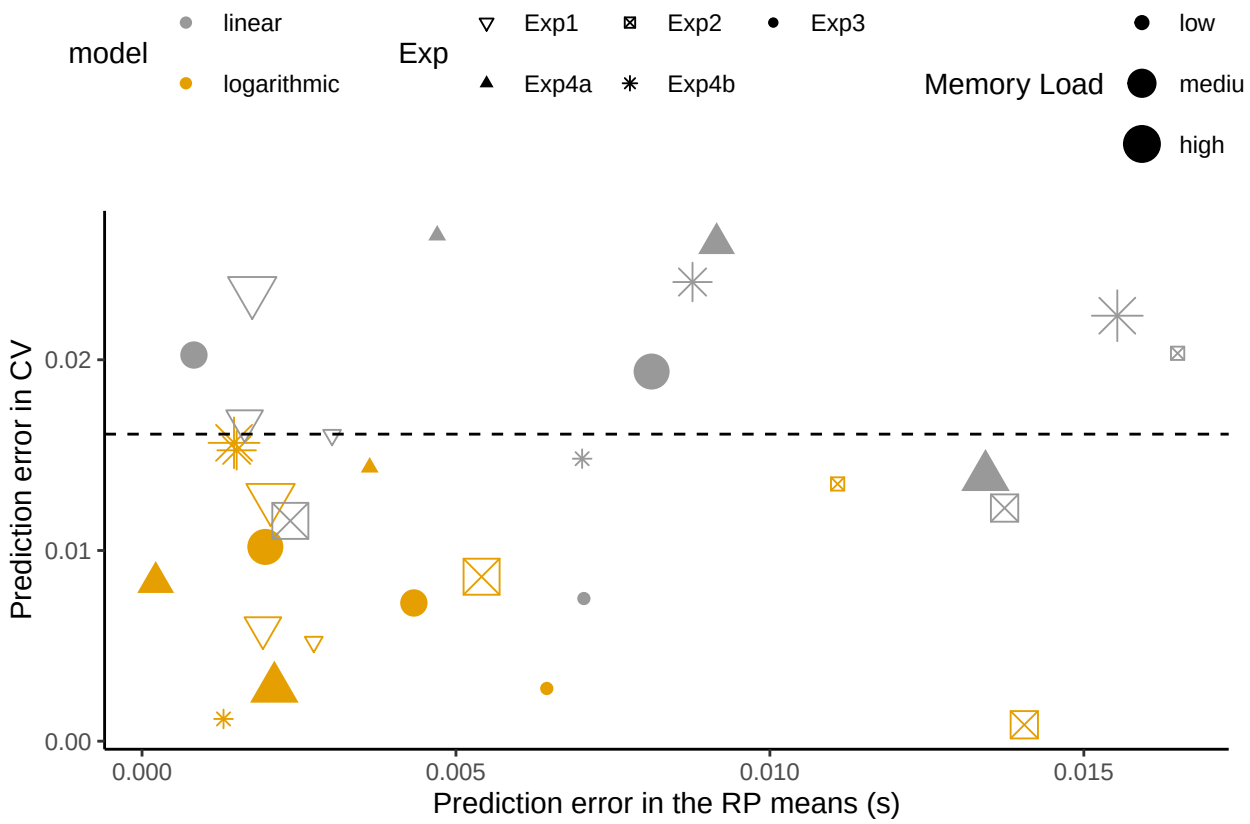
```
m_predErr_sub$model = 'logarithmic'
m_predErr$model = 'logarithmic'
linear_model = 'gap_linear_rstan'
m_predErr_linear = read.csv(paste0(getwd(), "/", rstanmodelPath, '/models/', linear_model, "/rlt/m_predl
m_predErr_linear$X = NULL
m_predErr_sub_linear = read.csv(paste0(getwd(), "/", rstanmodelPath, '/models/', linear_model, "/rlt/m_p
m_predErr_sub_linear$X = NULL

m_predErr_sub_all = rbind(m_predErr_sub, m_predErr_sub_linear)
m_predErr_all = rbind(m_predErr, m_predErr_linear)
```

```
m_predErr_all$WMSize = as.factor(m_predErr_all$WMSize)
levels(m_predErr_all$WMSize) =  c("low", "medium",  "high")
temp = m_predErr_all %>% filter(model == 'logarithmic') %>%summarise(abs_mmpredcv_err = abs(mmpredcv_er

plt_Err_CV_all = ggplot(m_predErr_all, aes(abs(mmpredRP_err), abs(mmpredcv_err), group = interaction(mo
  geom_point() +
  geom_hline(yintercept = round(max(temp$abs_mmpredcv_err), 4)+0.0005, linetype='dashed')+
  xlab('Prediction error in the RP means (s)')+ ylab('Prediction error in CV')+colorSet3+
  scale_shape_manual(values = c(6, 7, 16, 17,8)) +
  theme_new+
  theme(legend.position = 'top')+
 labs(size = 'Memory Load')+
  guides(colour = guide_legend(order = 1, nrow=2,byrow=TRUE),
         shape = guide_legend(order =2, nrow=2,byrow=TRUE),
            size = guide_legend(order = 3, nrow=3,byrow=TRUE))

ggsave(paste0(getwd(), "/", modelPath, "/figures/plt_Err_CV_all.png"), plt_Err_CV_all, width = 7, heigh
```

```
## Warning: Using size for a discrete variable is not advised.
```

```
plt_Err_CV_all
```

```
## Warning: Using size for a discrete variable is not advised.
```




```
m_predY_acc =  m_predErr_sub_all%>%
  dplyr::group_by(Exp, model) %>%
  dplyr::summarize(mmpredRP_rerr = mean(mpredRP_rerr)*100,
                   mmpredVar_rerr = mean(mpredVar_rerr)*100,
                   mmpredcv_rerr = mean(mpredcv_rerr)*100,
                   mmpredRP_acc = (1-mean(mpredRP_rerr))*100,
                   mmpredVar_acc = (1-mean(mpredVar_rerr))*100,
                   mmpredCV_acc = (1-mean(mpredcv_rerr))*100)
```

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

```
m_predY_acc
```

```
## # A tibble: 10 x 8
## # Groups:   Exp [5]
##    Exp   model       mmpredRP_rerr mmpredVar_rerr mmpredcv_rerr mmpredRP_acc
##    <chr> <chr>               <dbl>          <dbl>         <dbl>        <dbl>
##  1 Exp1  linear               3.59           25.3          25.3         96.4
##  2 Exp1  logarithmic          3.33           17.8          17.7         96.7
##  3 Exp2  linear               4.66           18.6          19.0         95.3
##  4 Exp2  logarithmic          4.17           13.6          13.9         95.8
##  5 Exp3  linear               4.06           22.2          22.5         95.9
##  6 Exp3  logarithmic          3.40           16.1          15.9         96.6
##  7 Exp4a linear               3.91           28.2          28.7         96.1
##  8 Exp4a logarithmic          3.47           18.5          18.5         96.5
##  9 Exp4b linear               4.10           24.1          23.0         95.9
## 10 Exp4b logarithmic          2.81           16.5          15.7         97.2
## # ... with 2 more variables: mmpredVar_acc <dbl>, mmpredCV_acc <dbl>
```

12.1 Export data for spss

```
obs_Inp_slope_Exp1_jasp <- obs_Inp_list_no_gap %>% filter(Exp == 'Exp1') %>% select(c("WMSize", "NSub",
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(obs_Inp_slope_Exp1_jasp, paste0(modelPath, '/rlt/obs_Inp_slope_Exp1_jasp.csv'))
```

```
obs_Inp_slope_Exp2_jasp <- obs_Inp_list_no_gap %>% filter(Exp == 'Exp2') %>% select(c("WMSize", "NSub",
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(obs_Inp_slope_Exp2_jasp, paste0(modelPath, '/rlt/obs_Inp_slope_Exp2_jasp.csv'))
```

```
obs_Inp_slope_Exp3_jasp <- obs_Inp_list_no_gap %>% filter(Exp == 'Exp3') %>% select(c("WMSize", "NSub",
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(obs_Inp_slope_Exp3_jasp, paste0(modelPath, '/rlt/obs_Inp_slope_Exp3_jasp.csv'))
```

```
obs_Inp_slope_Exp4a_jasp <- obs_Inp_list_no_gap %>% filter(Exp =='Exp4a') %>% select(c("WMSize", "NSub"
  pivot_wider(names_from = c("WMSize"), values_from = c(inP, slope), names_sep="_")
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(obs_Inp_slope_Exp4a_jasp, paste0(modelPath, '/rlt/obs_Inp_slope_Exp4a_jasp.csv'))
```

```
obs_Inp_list_Exp4b_jasp <- obs_Inp_list %>% filter(Exp == 'Exp4b') %>% select(c("WMSize", "NSub", "gap"
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(obs_Inp_list_Exp4b_jasp, paste0(modelPath, '/rlt/obs_Inp_slope_Exp4b_jasp.csv'))
```

```
pred_Inp_slope_Exp1_jasp <- pred_Inp_slope_no_gap %>% filter(Exp == 'Exp1') %>% select(c("WMSize", "NSu
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(pred_Inp_slope_Exp1_jasp, paste0(modelPath, '/rlt/pred_Inp_slope_Exp1_jasp.csv'))
```

```
pred_Inp_slope_Exp2_jasp <- pred_Inp_slope_no_gap %>% filter(Exp == 'Exp2') %>% select(c("WMSize", "NSu
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(pred_Inp_slope_Exp2_jasp, paste0(modelPath, '/rlt/pred_Inp_slope_Exp2_jasp.csv'))
```

```
pred_Inp_slope_Exp3_jasp <- pred_Inp_slope_no_gap %>% filter(Exp == 'Exp3') %>% select(c("WMSize", "NSu
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(pred_Inp_slope_Exp3_jasp, paste0(modelPath, '/rlt/pred_Inp_slope_Exp3_jasp.csv'))
```

```
pred_Inp_slope_Exp4a_jasp <- pred_Inp_slope_no_gap %>% filter(Exp =='Exp4a') %>% select(c("WMSize", "NS
  pivot_wider(names_from = c("WMSize"), values_from = c(pred_inP, pred_slope), names_sep="_")
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(pred_Inp_slope_Exp4a_jasp, paste0(modelPath, '/rlt/pred_Inp_slope_Exp4a_jasp.csv'))
```

```
pred_Inp_slope_Exp4b_jasp <- pred_Inp_list %>% filter(Exp == 'Exp4b') %>% select(c("WMSize", "NSub", "g
```

```
## Adding missing grouping variables: `Exp`
```

```
write.csv(pred_Inp_slope_Exp4b_jasp, paste0(modelPath, '/rlt/pred_Inp_slope_Exp4b_jasp.csv'))

boot_Inp_slope_Exp1_jasp <- cilist %>% filter(Exp == 'Exp1') %>% select(c("WMSize", "NSub","m_predInP_b

write.csv(boot_Inp_slope_Exp1_jasp, paste0(modelPath, '/rlt/boot_Inp_slope_Exp1_jasp.csv'))

boot_Inp_slope_Exp2_jasp <- cilist %>% filter(Exp == 'Exp2') %>% select(c("WMSize", "NSub","m_predInP_b

write.csv(boot_Inp_slope_Exp2_jasp, paste0(modelPath, '/rlt/boot_Inp_slope_Exp1_jasp.csv'))

boot_Inp_slope_Exp3_jasp <- cilist %>% filter(Exp == 'Exp3') %>% select(c("WMSize", "NSub","m_predInP_b

write.csv(boot_Inp_slope_Exp1_jasp, paste0(modelPath, '/rlt/boot_Inp_slope_Exp3_jasp.csv'))

boot_Inp_slope_Exp4a_jasp <- cilist %>% filter(Exp =='Exp4a') %>% select(c("WMSize", "NSub","m_predInP_
  pivot_wider(names_from = c("WMSize"), values_from = c(m_predInP_boot, mRP_InP_boot, m_pred_slope_boot

write.csv(boot_Inp_slope_Exp4a_jasp, paste0(modelPath, '/rlt/boot_Inp_slope_Exp4a_jasp.csv'))


# boot_Inp_slope_Exp4b_jasp <- cilist %>% filter(Exp == 'Exp4b') %>% select(c("WMSize", "NSub", "gap","
#
# write.csv(boot_Inp_slope_Exp4b_jasp, paste0(modelPath, '/rlt/boot_Inp_slope_Exp4b_jasp.csv'))
```