MP - repetition - Serial position analysis (v5)

```
# do this in calling R script
# library(ggplot2)
# library(fmsb) # for TestModels
# library(lme4)
# library(kableExtra)
# library(MASS) # for dropterm
# library(tidyverse)
# library(dominanceanalysis)
# library(cowplot) # for multiple plots in one figure
# library(formatters) # to wrap strings for plot titles
# library(pals) # for continous color palettes
```

```
# load needed functions
# source(paste0(RootDir,"/src/function_library/sp_functions.R"))
# do this in the calling R script
```

```
# for testing - set by calling script normally
# ppt_parms <- read.csv(config$patient_param_file)
# CurPat <- ppt_parms$patient
# CurTask <- ppt_parms$task
# MinLength <- ppt_parms$min_length
# MaxLength <- ppt_parms$max_length
# RemoveFinalPosition<-ppt_parms$remove_final_position
```

```
# create patient directory if it doesn't already exist
if(!dir.exists(paste0(RootDir,"/output/",CurPat))){
  dir.create(paste0(RootDir,"/output/",CurPat),showWarnings = TRUE)
}
TablesDir <- paste0(RootDir,"/output/",CurPat,"/tables/")
if(!dir.exists(TablesDir)){
  dir.create(TablesDir,showWarnings = TRUE)
}
FigDir <- paste0(RootDir,"/output/",CurPat,"/fig/")
if(!dir.exists(FigDir)){
  dir.create(FigDir,showWarnings = TRUE)
}
ReportsDir <- paste0(RootDir,"/output/",CurPat,"/reports/")
if(!dir.exists(ReportsDir)){
  dir.create(ReportsDir,showWarnings = TRUE)
}
results_report_DF <- data.frame(Description=character(), ParameterValue=double())
```

```
ModelDatFilename<-paste0(RootDir,"/data/",CurPat,"/",CurPat,"_",CurTask,"_posdat.csv")
if(!file.exists(ModelDatFilename)){
  print("**ERROR** Input data file not found")
  print(sprintf("\tfile: ",ModelDatFilename))
  print(sprintf("\troot dir: ",RootDir))
```

```
}
PosDat<-read.csv(ModelDatFilename)
```

```
# may already be done in datafile
# if(RemoveFinalPosition){
#   PosDat <- PosDat[PosDat$pos < PosDat$stimlen,] # remove final position
# }
PosDat <- PosDat[PosDat$stimlen >= MinLength,] # include only lengths with sufficient N
PosDat <- PosDat[PosDat$stimlen <= MaxLength,]
# include only correct and nonword errors (not no response, word or w+nw errors)
PosDat <- PosDat[(PosDat$err_type == "correct") | (PosDat$err_type == "nonword"), ]
PosDat <- PosDat[(PosDat$pos) != (PosDat$stimlen), ]
# make sure final positions have been removed
PosDat$stim_number <- NumberStimuli(PosDat)
PosDat$raw_log_freq <- PosDat$log_freq
PosDat$log_freq <- PosDat$log_freq - mean(PosDat$log_freq) # centre frequency
PosDat$CumErr <- CalcCumErrFromPreserved(PosDat)
```

```
# plot distribution of complex onsets and codas
# across serial position in the stimuli -- do complexities concentrate
# on one part of the serial position curve?
# syll_component = 1 is coda
# syll_component = S is satellite

# get counts of syllable components in different serial positions
syll_comp_dist_N <- PosDat %>% mutate(pos_factor = as.factor(pos), syll_component_factor = as.factor(sy

syll_comp_dist_perc <- syll_comp_dist_N %>% ungroup() %>%
  mutate(across(O:S, ~ . / total))

kable(syll_comp_dist_N)
```

pos_factor	O	P	V	1	S	total
1	504	30	120	NA	NA	654
2	59	NA	406	91	98	654
3	290	NA	152	197	15	654
4	283	NA	220	60	36	599
5	211	NA	199	66	34	510
6	192	1	126	67	20	406
7	167	NA	96	22	16	301
8	79	NA	49	25	4	157
9	69	NA	2	NA	6	77

```
kable(syll_comp_dist_perc)
```

pos_factor	O	P	V	1	S	total
1	0.7706422	0.0458716	0.1834862	NA	NA	654
2	0.0902141	NA	0.6207951	0.1391437	0.1498471	654
3	0.4434251	NA	0.2324159	0.3012232	0.0229358	654
4	0.4724541	NA	0.3672788	0.1001669	0.0601002	599
5	0.4137255	NA	0.3901961	0.1294118	0.0666667	510
6	0.4729064	0.0024631	0.3103448	0.1650246	0.0492611	406

pos_factor	O	P	V	1	S	total
7	0.5548173	NA	0.3189369	0.0730897	0.0531561	301
8	0.5031847	NA	0.3121019	0.1592357	0.0254777	157
9	0.8961039	NA	0.0259740	NA	0.0779221	77

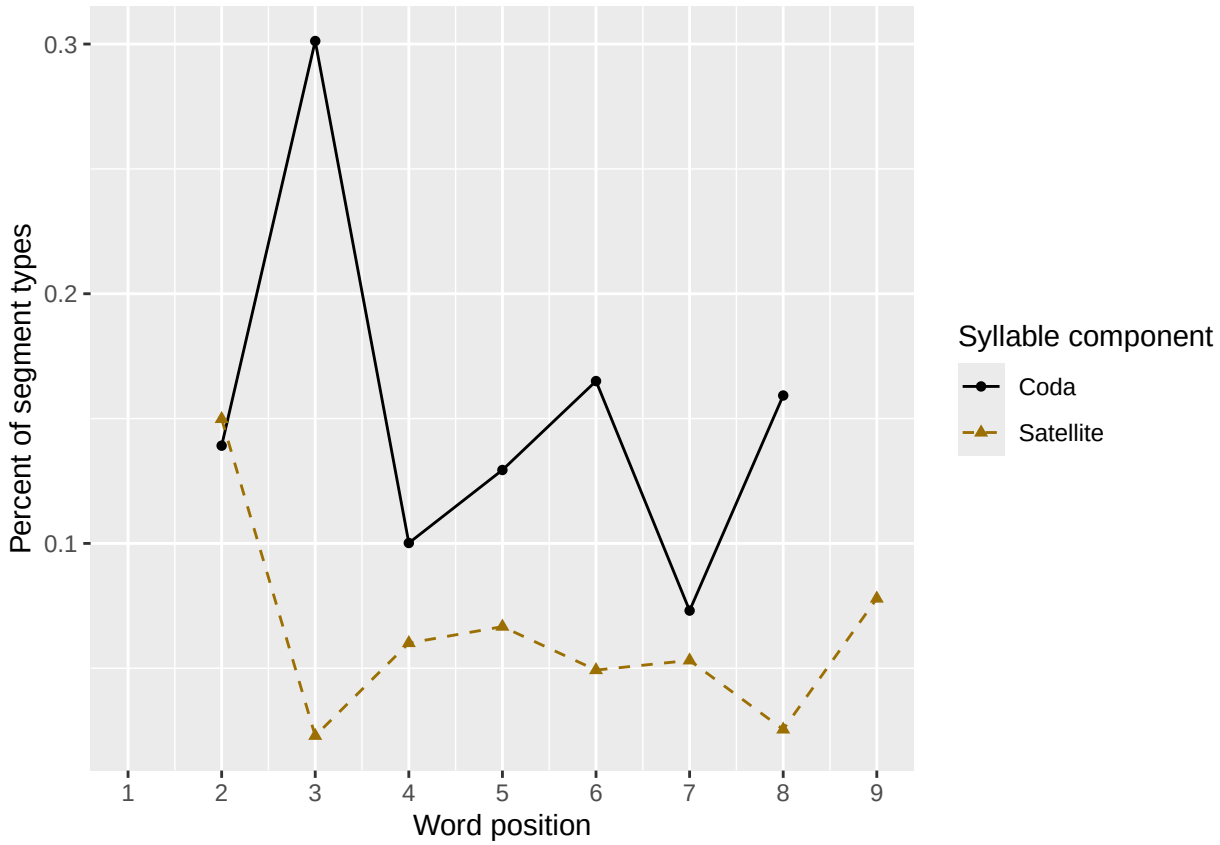
```
# get percentages only from summary table
syll_comp_perc <- syll_comp_dist_perc[,2:(ncol(syll_comp_dist_perc)-1)]
syll_comp_perc$pos <- as.integer(as.character(syll_comp_dist_perc$pos_factor))
syll_comp_perc <- syll_comp_perc %>% rename("Coda" ="1","Satellite"="S")

# pivot to long format for plotting
syll_comp_plot_data <- syll_comp_perc %>% pivot_longer(cols=seq(1:(ncol(syll_comp_perc)-1)),
            names_to="syll_component", values_to="percent") %>% filter(syll_component %in% c("Coda","Sa
# plot satellite and coda occurances across position

syll_comp_plot <- ggplot(syll_comp_plot_data,
                         aes(x=pos,y=percent,group=syll_component,
                             color=syll_component,
                             linetype = syll_component,
                             shape = syll_component)) +
  geom_line() + geom_point() + scale_color_manual(name="Syllable component", values = palette_values) +
syll_comp_plot <- syll_comp_plot + scale_y_continuous(name="Percent of segment types") + scale_linetype_
  scale_shape_discrete(name="Syllable component")
ggsave(paste0(FigDir,CurPat,"_",CurTask,"_pos_of_syll_complexities_in_stimuli.png"),plot=syll_comp_plot
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

```
syll_comp_plot
```

```
## Warning: Removed 3 rows containing missing values or values outside the scale range (`geom_line()`).
## Removed 3 rows containing missing values or values outside the scale range (`geom_point()`).
```




```
# len/pos table
pos_len_summary <- PosDat %>% group_by(stimlen,pos) %>% summarise(preserved = mean(preserved))
```

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

```
min_preserved_raw <- min(as.numeric(unlist(pos_len_summary$preserved)))
max_preserved_raw <- max(as.numeric(unlist(pos_len_summary$preserved)))
preserved_range <- (max_preserved_raw - min_preserved_raw)
min_preserved <- min_preserved_raw - (0.1 * preserved_range)
max_preserved <- max_preserved_raw + (0.1 * preserved_range)
pos_len_table <- pos_len_summary %>% pivot_wider(names_from = pos, values_from = preserved)
write.csv(pos_len_table,paste0(TablesDir,CurPat,"_",CurTask,"_preserved_percentages_by_len_position.csv
pos_len_table
```

```
## # A tibble: 7 x 10
## # Groups:   stimlen [7]
##   stimlen   `1`   `2`   `3`    `4`    `5`    `6`    `7`    `8`    `9`
##     <int> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1       4 0.945 0.964 0.927 NA     NA     NA     NA     NA     NA
## 2       5 0.966 0.989 0.978  0.955 NA     NA     NA     NA     NA
## 3       6 0.962 0.976 0.971  0.958  0.936 NA     NA     NA     NA
## 4       7 0.933 0.976 0.929  0.933  0.967  0.938 NA     NA     NA
## 5       8 0.889 0.965 0.944  0.962  0.958  0.955  0.962 NA     NA
## 6       9 0.925 0.956 0.956  0.95   0.969  0.925  0.944  0.925 NA
## 7      10 0.987 1     0.961  0.955  0.948  0.944  0.929  0.950  0.896
```

```
# len/pos table
pos_len_N <- PosDat %>% group_by(stimlen,pos) %>% summarise(preserved = mean(preserved),N=n()) %>% dply
```

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

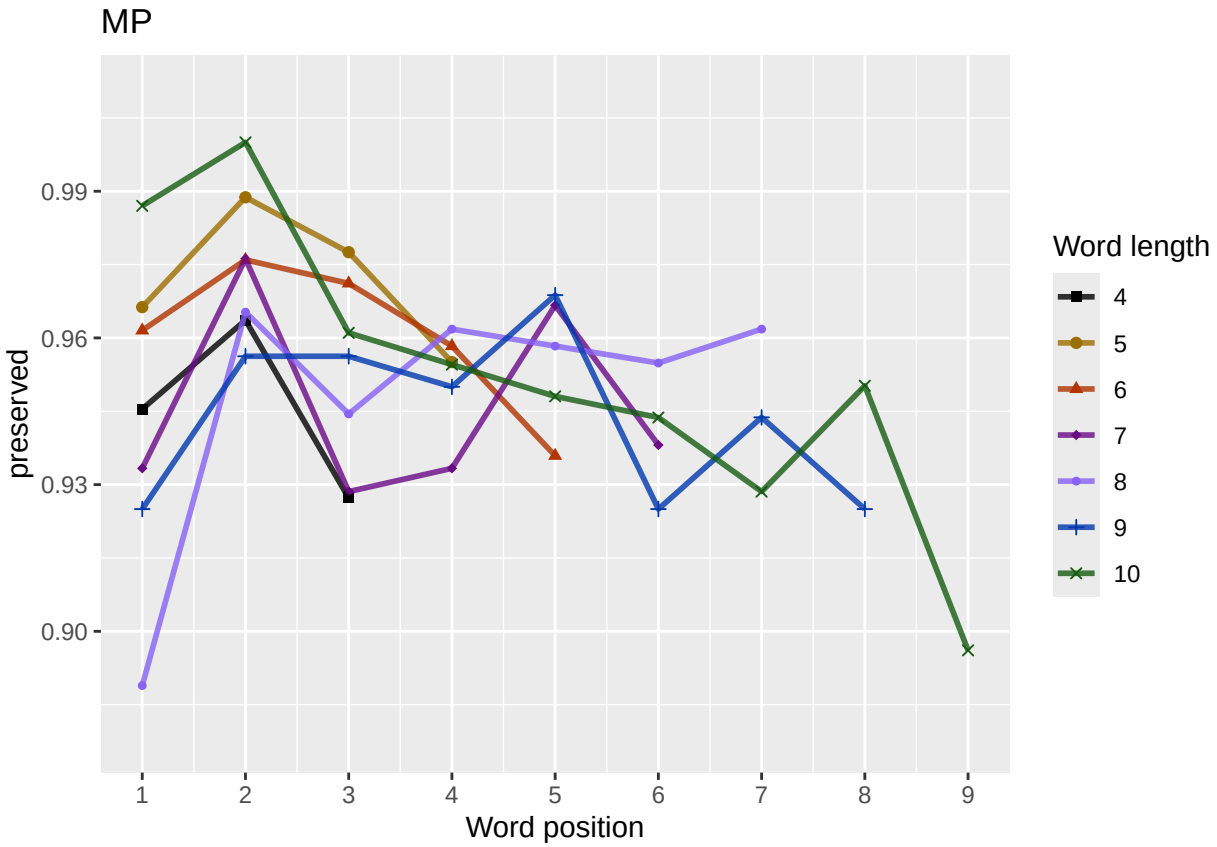
```
pos_len_N_table <- pos_len_N %>% pivot_wider(names_from = pos, values_from = N)
write.csv(pos_len_N_table,paste0(TablesDir,CurPat,"_",CurTask,"_N_by_len_position.csv"))
pos_len_N_table
```

```
## # A tibble: 7 x 10
## # Groups:   stimlen [7]
##   stimlen   `1`   `2`   `3`   `4`   `5`   `6`   `7`   `8`   `9`
##     <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
## 1       4    55    55    55    NA    NA    NA    NA    NA    NA
## 2       5    89    89    89    89    NA    NA    NA    NA    NA
## 3       6   104   104   104   104   104    NA    NA    NA    NA
## 4       7   105   105   105   105   105   105    NA    NA    NA
## 5       8   144   144   144   144   144   144   144    NA    NA
## 6       9    80    80    80    80    80    80    80    80    NA
## 7      10    77    77    77    77    77    77    77    77    77
```

```
obs_linetypes <- c("solid","solid","solid","solid",
                   "solid","solid","solid","solid","solid")
pred_linetypes <- "dashed"
pos_len_summary$stimlen<-factor(pos_len_summary$stimlen)
pos_len_summary$pos<-factor(pos_len_summary$pos)
# len_pos_plot <- ggplot(pos_len_summary,aes(x=pos,y=preserved,group=stimlen,shape=stimlen,color=stimle
len_pos_plot <- plot_len_pos_obs_predicted(PosDat,
                                           paste0(CurPat),
                                           NULL,
                                           c(min_preserved,max_preserved),
                                           palette_values,
                                           shape_values,
                                           obs_linetypes,
                                           pred_linetypes)
```

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

```
ggsave(paste0(FigDir,CurPat,"_",CurTask,"_percent_preserved_by_length_pos.png"),
       plot=len_pos_plot,device="png",unit="cm",width=15,height=11)
len_pos_plot
```



Length and position

```
# length and position

LPModelEquations<-c("preserved ~ 1",
            "preserved ~ stimlen",
            "preserved ~ pos",
            "preserved ~ stimlen + pos",
            "preserved ~ stimlen*pos",
            "preserved ~ I(pos^2)+pos",
            "preserved ~ stimlen + I(pos^2) + pos",
            "preserved ~ stimlen * (I(pos^2) + pos)"
)

LPRes<-TestModels(LPModelEquations,PosDat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## **************************
## model index:  6
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.74987     -0.02919      0.20856
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1632
## log likelihood:  -754.6963
## Nagelkerke R2:  0.005144461
## % pres/err predicted correctly:  -368.0201
## % of predictable range [ (model-null)/(1-null) ]:  0.00185947
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.17598     -0.05148
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -756.7923
## Nagelkerke R2:  0.001827377
## % pres/err predicted correctly:  -368.4056
## % of predictable range [ (model-null)/(1-null) ]:  0.0008167095
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##     3.38742     -0.05359
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -757.1979
## Nagelkerke R2:  0.001185227
## % pres/err predicted correctly:  -368.5278
## % of predictable range [ (model-null)/(1-null) ]:  0.0004862924
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
```

```
##
## Coefficients:
## (Intercept)
##       2.973
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4011 Residual
## Null Deviance:       1516
## Residual Deviance: 1516  AIC: 1633
## log likelihood:  -757.9461
## Nagelkerke R2:  0
## % pres/err predicted correctly:  -368.7075
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     2.86145     -0.01547     -0.02838      0.20638
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1634
## log likelihood:  -754.6469
## Nagelkerke R2:  0.005222613
## % pres/err predicted correctly:  -368.0019
## % of predictable range [ (model-null)/(1-null) ]:  0.001908671
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     3.38419     -0.03219     -0.04120
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1513  AIC: 1634
## log likelihood:  -756.5701
## Nagelkerke R2:  0.002179197
## % pres/err predicted correctly:  -368.3521
## % of predictable range [ (model-null)/(1-null) ]:  0.000961522
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
```

```
##     2.81614      0.03659      0.14048     -0.02122
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1512  AIC: 1635
## log likelihood:  -756.1378
## Nagelkerke R2:  0.002863618
## % pres/err predicted correctly:  -368.2452
## % of predictable range [ (model-null)/(1-null) ]:  0.001250441
## ***************************
## model index:  8
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          I(pos^2)               pos  stimlen:I(pos^2)
##         3.272740         -0.072025         -0.014054          0.008455         -0.002127
##      stimlen:pos
##         0.027528
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1638
## log likelihood:  -754.5441
## Nagelkerke R2:  0.005385125
## % pres/err predicted correctly:  -367.9927
## % of predictable range [ (model-null)/(1-null) ]:  0.001933551
## ***************************
```

```r
BestLPModel<-LPRes$ModelResult[[1]]
BestLPModelFormula<-LPRes$Model[[1]]

LPAICSummary<-data.frame(Model=LPRes$Model,
                         AIC=LPRes$AIC,
                         row.names = LPRes$Model)
LPAICSummary$DeltaAIC<-LPAICSummary$AIC-LPAICSummary$AIC[1]
LPAICSummary$AICexp<-exp(-0.5*LPAICSummary$DeltaAIC)
LPAICSummary$AICwt<-LPAICSummary$AICexp/sum(LPAICSummary$AICexp)
LPAICSummary$NagR2<-LPRes$NagR2

LPAICSummary <- merge(LPAICSummary,LPRes$CoefficientValues, by='row.names',sort=FALSE)
LPAICSummary <- subset(LPAICSummary, select = -c(Row.names))

write.csv(LPAICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_len_pos_model_summary.csv"),row.names = FA
kable(LPAICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ I(pos^2) + pos	1632.086	0.0000000	1.0000000	0.2466940	0.0051442	2.5749866	NA	0.2085560	NA	-0.0291870	NA
preserved ~ pos	1632.632	0.5454303	0.7613081	0.1878104	0.0018273	3.4175979	NA	-0.0514827	NA	NA	NA
preserved ~ stimlen	1633.081	0.9944994	0.6082001	0.1500398	0.0011853	3.2387423	-0.0535934	NA	NA	NA	NA

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ 1	1633.102	2.016298	0.3660160	0.0791484	0.0000000	2.0973323	NA	NA	NA	NA	NA
preserved ~ stimlen + I(pos^2) + pos	1633.805	0.719362	0.5142329	0.1104425	0.0052226	2.8614500	- 0.0154717	0.2063755	NA	- 0.0283800	NA
preserved ~ stimlen + pos	1633.901	0.817657	0.4902996	0.0994168	0.0021793	2.2384187	- 0.0321900	- 0.1041195	NA	NA	NA
preserved ~ stimlen * pos	1635.335	2.248924	0.1970176	0.0426083	0.0028626	2.8161390	0.0365932	1.1404754	- 0.0212226	NA	NA
preserved ~ stimlen * (I(pos^2) + pos)	1637.745	1.654628	0.3605917	0.0701459	0.0053853	3.1272740	- 0.0720253	0.0084555	0.0275280	- 0.0140536	- 0.0021271

```
print(BestLPModelFormula)
```

```
## [1] "preserved ~ I(pos^2) + pos"
```

```
print(BestLPModel)
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.74987     -0.02919      0.20856
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1632
```

```
PosDat$LPFitted<-fitted(BestLPModel)
fitted_len_pos_plot <- plot_len_pos_obs_predicted(PosDat, PosDat$patient[1],
                         NULL,palette_values=palette_values)
```

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

```
# len/pos table
fitted_pos_len_summary <- PosDat %>% group_by(stimlen,pos) %>% summarise(fitted = mean(LPFitted))
```

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

```
fitted_pos_len_table <- fitted_pos_len_summary %>% pivot_wider(names_from = pos, values_from = fitted)
fitted_pos_len_table
```

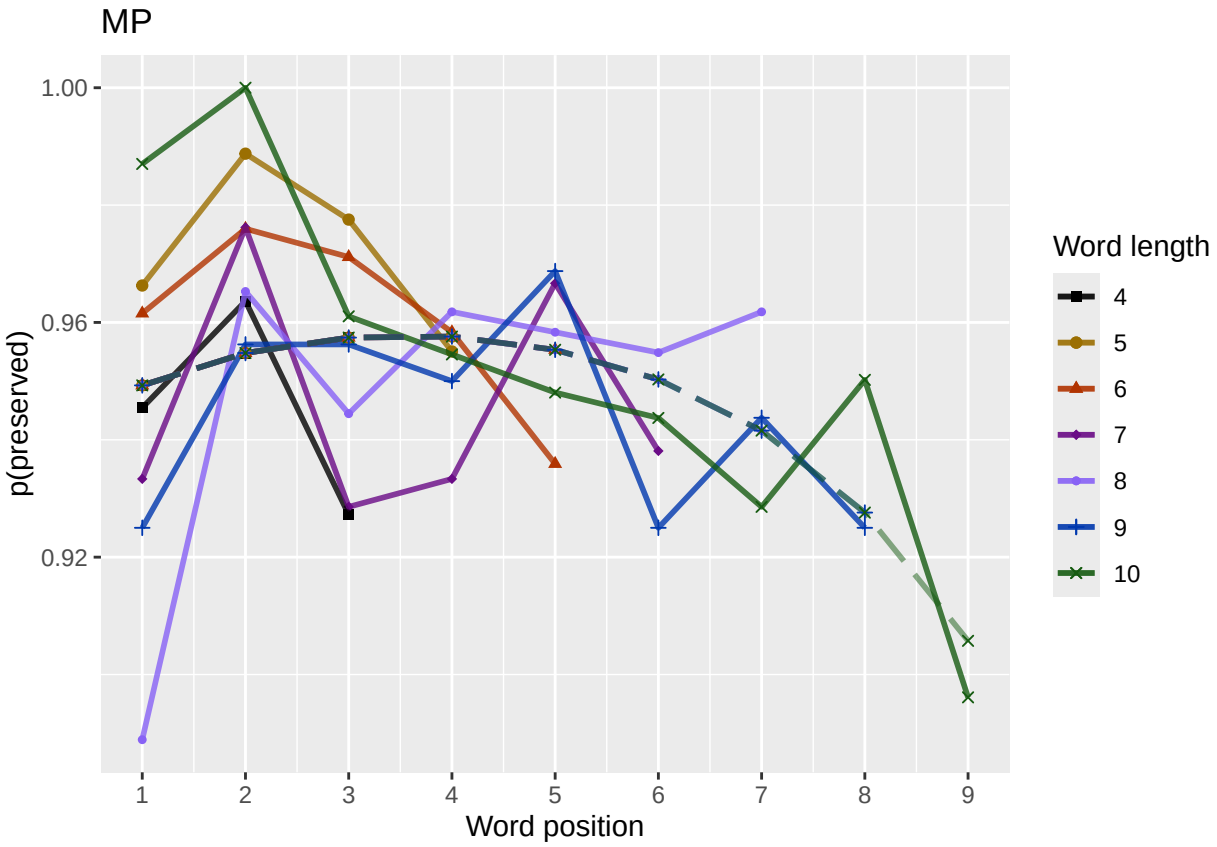
```
## # A tibble: 7 x 10
## # Groups:   stimlen [7]
##   stimlen   `1`   `2`   `3`    `4`    `5`    `6`    `7`    `8`    `9`
##     <int> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1       4 0.949 0.955 0.957 NA     NA     NA     NA     NA     NA
## 2       5 0.949 0.955 0.957  0.958 NA     NA     NA     NA     NA
## 3       6 0.949 0.955 0.957  0.958  0.955 NA     NA     NA     NA
## 4       7 0.949 0.955 0.957  0.958  0.955  0.950 NA     NA     NA
## 5       8 0.949 0.955 0.957  0.958  0.955  0.950  0.942 NA     NA
## 6       9 0.949 0.955 0.957  0.958  0.955  0.950  0.942  0.928 NA
```

```
## 7      10 0.949 0.955 0.957  0.958  0.955  0.950  0.942  0.928  0.906
```

```
fitted_pos_len_summary$stimlen<-factor(fitted_pos_len_summary$stimlen)
fitted_pos_len_summary$pos<-factor(fitted_pos_len_summary$pos)
# fitted_len_pos_plot <- ggplot(pos_len_summary,aes(x=pos,y=preserved,group=stimlen,shape=stimlen,color
# fitted_len_pos_plot <- fitted_len_pos_plot + geom_line(data=fitted_pos_len_summary, aes(x=pos,y=fitte
# geom_point(data=fitted_pos_len_summary,aes(x=pos,y=fitted,group=stimlen,shape=stimlen)) + ggtitle(pas

fitted_len_pos_plot <- plot_len_pos_obs_predicted(PosDat,
                                          paste0(PosDat$patient[1]),
                                          "LPFitted",
                                          NULL,
                                          palette_values,
                                          shape_values,
                                          obs_linetypes,
                                          pred_linetypes = c("longdash")
                                          )
```

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

```
ggsave(paste0(FigDir,CurPat,"_",CurTask,"_percent_preserved_by_length_pos_wfit.png"),plot=fitted_len_pos
fitted_len_pos_plot
```



length and position without fragments to see if this changes position^2 influence

```
# first number responses, then count resp with fragments - below we will eliminate fragments
# and re-run models
```

```
# number responses
resp_num<-0
prev_pos<-9999 # big number to initialize (so first position is smaller)
resp_num_array <- integer(length = nrow(PosDat))
for(i in seq(1,nrow(PosDat))){
  if(PosDat$pos[i] <= prev_pos){ # equal for resp length 1
    resp_num <- resp_num + 1
  }
  resp_num_array[i]<-resp_num
  prev_pos <- PosDat$pos[i]
}
PosDat$resp_num <- resp_num_array
```

```
# count responses with fragments
resp_with_frag <- PosDat %>% group_by(resp_num) %>% summarise(frag = as.logical(sum(fragment)))
num_frag <- resp_with_frag %>% summarise(frag_sum = sum(frag), N=n())
num_frag
```

```
## # A tibble: 1 x 2
##   frag_sum     N
##      <int> <int>
## 1        6   654
```

```
num_frag$percent_with_frag[1] <- (num_frag$frag_sum[1] / num_frag$N[1])*100
```

```
## Warning: Unknown or uninitialised column: `percent_with_frag`.
```

```
print(sprintf("The number of responses with fragments was %d / %d = %.2f percent",
              num_frag$frag_sum[1],num_frag$N[1],num_frag$percent_with_frag[1]))
```

```
## [1] "The number of responses with fragments was 6 / 654 = 0.92 percent"
```

```
write.csv(num_frag,paste0(TablesDir,CurPat,"_",CurTask,"_percent_fragments.csv"),row.names = FALSE)
```

```
# length and position models for data without fragments
NoFragData <- PosDat %>% filter(fragment != 1)

NoFrag_LPRes<-TestModels(LPModelEquations,NoFragData)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## **************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
```

```
##     3.62882     -0.07667
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3998 Residual
## Null Deviance:       1447
## Residual Deviance: 1444  AIC: 1560
## log likelihood:  -722.0887
## Nagelkerke R2:  0.002356592
## % pres/err predicted correctly:  -347.6561
## % of predictable range [ (model-null)/(1-null) ]:  0.0008898691
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.66657     -0.03576      0.27982
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3997 Residual
## Null Deviance:       1447
## Residual Deviance: 1440  AIC: 1560
## log likelihood:  -719.9592
## Nagelkerke R2:  0.005859847
## % pres/err predicted correctly:  -347.2711
## % of predictable range [ (model-null)/(1-null) ]:  0.00199328
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     3.02565     -0.04943     -0.03323      0.27293
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3996 Residual
## Null Deviance:       1447
## Residual Deviance: 1439  AIC: 1560
## log likelihood:  -719.4798
## Nagelkerke R2:  0.006648022
## % pres/err predicted correctly:  -347.1639
## % of predictable range [ (model-null)/(1-null) ]:  0.002300453
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
##     2.59273      0.05599      0.32334     -0.03937
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3996 Residual
```

```
## Null Deviance:       1447
## Residual Deviance: 1441  AIC: 1561
## log likelihood:  -720.6038
## Nagelkerke R2:  0.004799889
## % pres/err predicted correctly:  -347.3104
## % of predictable range [ (model-null)/(1-null) ]:  0.001880528
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       3.034
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3999 Residual
## Null Deviance:       1447
## Residual Deviance: 1447  AIC: 1561
## log likelihood:  -723.5199
## Nagelkerke R2:  7.314916e-16
## % pres/err predicted correctly:  -347.9667
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     3.62813     -0.06850     -0.01606
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3997 Residual
## Null Deviance:       1447
## Residual Deviance: 1444  AIC: 1561
## log likelihood:  -721.9971
## Nagelkerke R2:  0.00250738
## % pres/err predicted correctly:  -347.6159
## % of predictable range [ (model-null)/(1-null) ]:  0.001005308
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.17925     -0.03723
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3998 Residual
## Null Deviance:       1447
## Residual Deviance: 1446  AIC: 1562
## log likelihood:  -722.9538
```

```
## Nagelkerke R2:  0.0009323979
## % pres/err predicted correctly:  -347.8097
## % of predictable range [ (model-null)/(1-null) ]:  0.0004499388
## **************************
## model index:  8
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          I(pos^2)               pos  stimlen:I(pos^2)
##         3.041762         -0.044123         -0.008353          0.174642          -0.002289
##      stimlen:pos
##         0.006815
##
## Degrees of Freedom: 3999 Total (i.e. Null);  3994 Residual
## Null Deviance:       1447
## Residual Deviance: 1439  AIC: 1564
## log likelihood:  -719.4009
## Nagelkerke R2:  0.00677772
## % pres/err predicted correctly:  -347.1243
## % of predictable range [ (model-null)/(1-null) ]:  0.00241404
## **************************
```

```
NoFragBestLPModel<-NoFrag_LPRes$ModelResult[[1]]
NoFragBestLPModelFormula<-NoFrag_LPRes$Model[[1]]

NoFragLPAICSummary<-data.frame(Model=NoFrag_LPRes$Model,
                       AIC=NoFrag_LPRes$AIC,
                       row.names = NoFrag_LPRes$Model)
NoFragLPAICSummary$DeltaAIC<-NoFragLPAICSummary$AIC-NoFragLPAICSummary$AIC[1]
NoFragLPAICSummary$AICexp<-exp(-0.5*NoFragLPAICSummary$DeltaAIC)
NoFragLPAICSummary$AICwt<-NoFragLPAICSummary$AICexp/sum(NoFragLPAICSummary$AICexp)
NoFragLPAICSummary$NagR2<-NoFrag_LPRes$NagR2

NoFragLPAICSummary <- merge(NoFragLPAICSummary,NoFrag_LPRes$CoefficientValues, by='row.names',sort=FALSE
NoFragLPAICSummary <- subset(NoFragLPAICSummary, select = -c(Row.names))

write.csv(NoFragLPAICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_no_fragments_len_pos_model_summary.cs
kable(NoFragLPAICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ stimlen	1559.574	0.0000000	1.0000000	0.2368284	0.0023566	0.6628823	-0.0766700	NA	NA	NA	NA
preserved ~ I(pos^2) + pos	1559.907	0.3324301	0.8468641	0.2005572	0.0058529	0.8666569	NA	0.2798208	NA	-0.0357594	NA
preserved ~ stimlen + I(pos^2) + pos	1560.339	0.7645039	0.6823281	0.1615901	0.0066483	0.8025647	-0.0494273	0.2729344	NA	-0.0332293	NA
preserved ~ stimlen * pos	1561.123	1.5485250	0.4610487	0.1091859	0.0047929	0.5927260	0.0559898	0.3233389	-0.0393676	NA	NA
preserved ~ 1	1561.264	1.6897387	0.4296146	0.1017428	0.0000000	0.6033923	NA	NA	NA	NA	NA
preserved ~ stimlen + pos	1561.417	1.8429164	0.3979384	0.0942410	0.0025074	0.8628131	-0.0685018	0.2801603	NA	NA	NA

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ pos	1561.98	2.413698	0.299138	0.070842	0.000933	0.179252	NA	-0.0372333	NA	NA	NA
preserved ~ stimlen * (I(pos^2) + pos)	1564.07	0.495545	0.105634	0.025016	0.000677	0.037041762	-0.0441227	0.17464115	0.00068152	-0.0083529	-0.00022892

```
# plot no fragment data

NoFragData$LPFitted<-fitted(NoFragBestLPModel)
nofrag_obs_len_pos_plot <- plot_len_pos_obs_predicted(NoFragData, NoFragData$patient[1],
                          NULL,palette_values=palette_values)
```

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

```
# len/pos table
nofrag_fitted_pos_len_summary <- NoFragData %>% group_by(stimlen,pos) %>% summarise(fitted = mean(LPFit
```

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

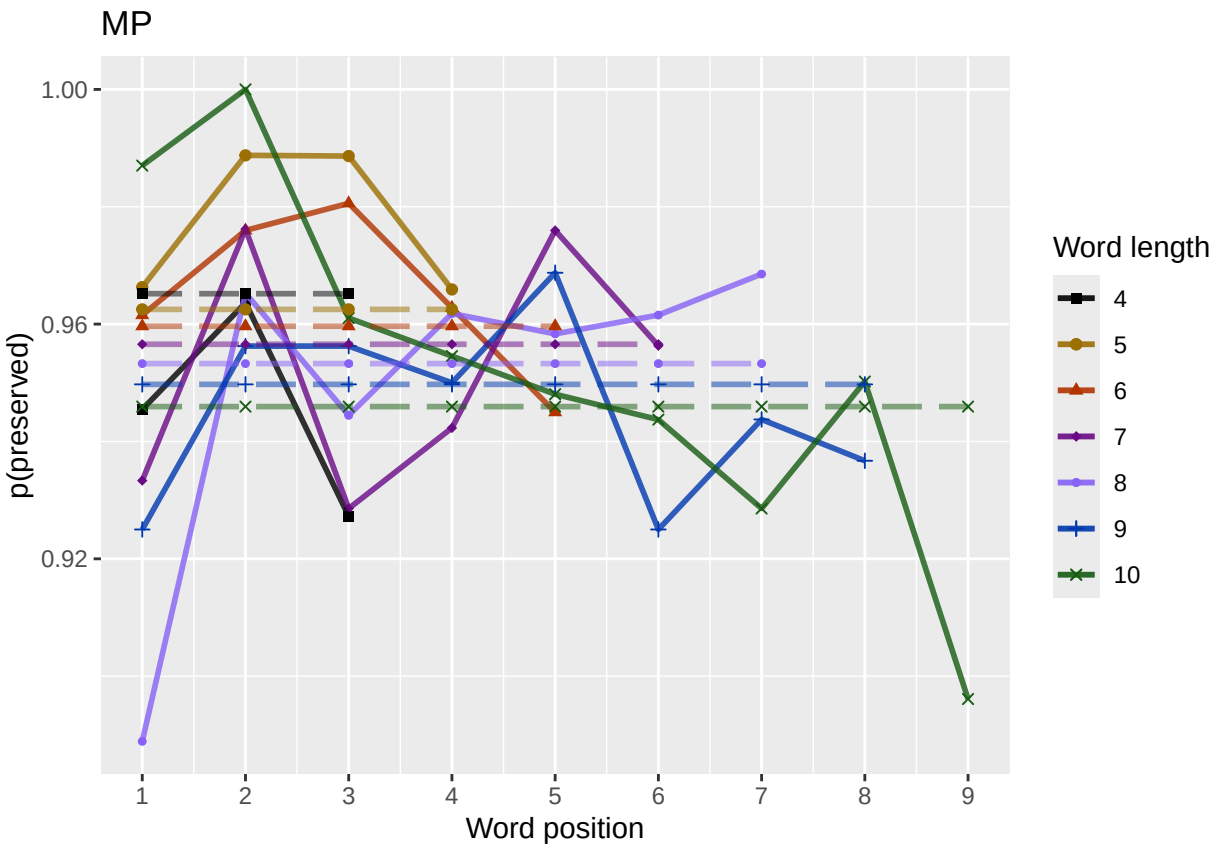
```
nofrag_fitted_pos_len_table <- nofrag_fitted_pos_len_summary %>% pivot_wider(names_from = pos, values_f
nofrag_fitted_pos_len_table
```

```
## # A tibble: 7 x 10
## # Groups:   stimlen [7]
##   stimlen   `1`   `2`   `3`    `4`    `5`    `6`    `7`    `8`    `9`
##     <int> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1       4 0.965 0.965 0.965 NA     NA     NA     NA     NA     NA
## 2       5 0.963 0.963 0.963  0.963 NA     NA     NA     NA     NA
## 3       6 0.960 0.960 0.960  0.960  0.960 NA     NA     NA     NA
## 4       7 0.957 0.957 0.957  0.957  0.957  0.957 NA     NA     NA
## 5       8 0.953 0.953 0.953  0.953  0.953  0.953  0.953 NA     NA
## 6       9 0.950 0.950 0.950  0.950  0.950  0.950  0.950  0.950 NA
## 7      10 0.946 0.946 0.946  0.946  0.946  0.946  0.946  0.946  0.946
```

```
fitted_pos_len_summary$stimlen<-factor(nofrag_fitted_pos_len_summary$stimlen)
fitted_pos_len_summary$pos<-factor(nofrag_fitted_pos_len_summary$pos)
# fitted_len_pos_plot <- ggplot(pos_len_summary,aes(x=pos,y=preserved,group=stimlen,shape=stimlen,color
# fitted_len_pos_plot <- fitted_len_pos_plot + geom_line(data=fitted_pos_len_summary, aes(x=pos,y=fitte
# geom_point(data=fitted_pos_len_summary,aes(x=pos,y=fitted,group=stimlen,shape=stimlen)) + ggtitle(pas

nofrag_fitted_len_pos_plot <- plot_len_pos_obs_predicted(NoFragData,
                                        paste0(NoFragData$patient[1]),
                                        "LPFitted",
                                        NULL,
                                        palette_values,
                                        shape_values,
                                        obs_linetypes,
                                        pred_linetypes = c("longdash")
                                        )
```

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

```
ggsave(paste0(FigDir,CurPat,"_",CurTask,"no_fragments_percent_preserved_by_length_pos_wfit.png"),plot=no
nofrag_fitted_len_pos_plot
```



back to full data

```
results_report_DF <- AddReportLine(results_report_DF,"min preserved",min_preserved)
results_report_DF <- AddReportLine(results_report_DF,"max preserved",max_preserved)
print(sprintf("Min/max preserved range: %.2f - %.2f",min_preserved,max_preserved))
```

```
## [1] "Min/max preserved range: 0.88 - 1.01"
```

```
# find the average difference between lengths and the average difference
# between positions taking the other factor into account

# choose fitted or raw -- we use fitted values because they are smoothed averages
# that take into account the influence of the other variable
table_to_use <- fitted_pos_len_table
# take away column with length information
table_to_use <- table_to_use[,2:ncol(table_to_use)]

# take averages for positions
# don't want downward estimates influenced by return upward of U
# therefore, for downward influence, use only the values before the min
# take the difference between each value (differences between position proportion correct) **NOTE** pro
# average the difference in probabilities

# in case min is close to the end or we are not using a min (for non-U-shaped)
# functions, we need to get differences _first_ and then average those (e.g.
```

```r
# if there is an effect of length and we just average position data,
# later positions) will have a lower average (because of the length effect)
# even if all position functions go upward

table_pos_diffs <- t(diff(t(as.matrix(table_to_use))))
ncol_pos_diff_matrix <- ncol(table_pos_diffs)
first_col_mean <- mean(as.numeric(unlist(table_pos_diffs[,1])))
potential_u_shape <- 0
if(first_col_mean < 0){ # downward, so potential U-shape
  # take only negative values from the table
  filtered_table_pos_diffs<-table_pos_diffs
  filtered_table_pos_diffs[table_pos_diffs >= 0] <- NA
  potential_u_shape <- 1
}else{
  # upward - use the positive values
  # take only negative values from the table
  filtered_table_pos_diffs<-table_pos_diffs
  filtered_table_pos_diffs[table_pos_diffs <= 0] <- NA
}
average_pos_diffs <- apply(filtered_table_pos_diffs,2,mean,na.rm=TRUE)
OA_mean_pos_diff <- mean(average_pos_diffs,na.rm=TRUE)

table_len_diffs <- diff(as.matrix(table_to_use))
average_len_diffs <- apply(table_len_diffs,1,mean,na.rm=TRUE)
OA_mean_len_diff <- mean(average_len_diffs,na.rm=TRUE)
CurrentLabel<-"mean change in probability for each additional length: "
print(CurrentLabel)
```

```
## [1] "mean change in probability for each additional length: "
```

```r
print(OA_mean_len_diff)
```

```
## [1] 0
```

```r
results_report_DF <- AddReportLine(results_report_DF, CurrentLabel,OA_mean_len_diff)

CurrentLabel<-"mean change in probability for each additional position (excluding  U-shape): "
print(CurrentLabel)
```

```
## [1] "mean change in probability for each additional position (excluding  U-shape): "
```

```r
print(OA_mean_pos_diff)
```

```
## [1] 0.002773221
```

```r
results_report_DF <- AddReportLine(results_report_DF, CurrentLabel,OA_mean_pos_diff)

if(potential_u_shape){ # downward, so potential U-shape
  # take only positive values from the table
  filtered_pos_upward_u<-table_pos_diffs
  filtered_pos_upward_u[table_pos_diffs <= 0] <- NA
  average_pos_u_diffs <- apply(filtered_pos_upward_u,
                               2,mean,na.rm=TRUE)
  OA_mean_pos_u_diff <- mean(average_pos_u_diffs,na.rm=TRUE)
  if(is.nan(OA_mean_pos_u_diff) | (OA_mean_pos_u_diff <= 0)){
    print("No U-shape in this participant")
    results_report_DF <- AddReportLine(results_report_DF, "U-shape", 0)
```

```
    potential_u_shape <- FALSE
  }else{
    results_report_DF <- AddReportLine(results_report_DF, "U-shape", 1)

    CurrentLabel<-"Average upward change after U minimum"
    print(CurrentLabel)
    print(OA_mean_pos_u_diff)
    results_report_DF <- AddReportLine(results_report_DF, CurrentLabel, OA_mean_pos_u_diff)

    CurrentLabel<-"Proportion of average downward change"
    prop_ave_down <- abs(OA_mean_pos_u_diff/OA_mean_pos_diff)
    results_report_DF <- AddReportLine(results_report_DF, CurrentLabel, prop_ave_down)
  }
}else{
  print("No U-shape in this participant")
  results_report_DF <- AddReportLine(results_report_DF, "U-shape", 0)
}
```

```
## [1] "No U-shape in this participant"
```

```
if(potential_u_shape){
  n_rows<-nrow(table_to_use)
  downward_dist <- numeric(n_rows)
  upward_dist <- numeric(n_rows)
  for(i in seq(1,n_rows)){
    current_row <- as.numeric(unlist(table_to_use[i,1:9]))
    current_row <- current_row[!is.na(current_row)]
    current_row_len <- length(current_row)
    row_min <- min(current_row,na.rm=TRUE)
    min_pos <- which(current_row == row_min)
    left_max <- max(current_row[1:min_pos],na.rm=TRUE)
    right_max <- max(current_row[min_pos:current_row_len])
    left_diff <- left_max - row_min
    right_diff <- right_max - row_min
    downward_dist[i]<-left_diff
    upward_dist[i]<-right_diff
  }
  print("differences from left max to min for each row: ")
  print(downward_dist)
  print("differences from min to right max for each row: ")
  print(upward_dist)

  # Use the max return upward (min of upward_dist) and then the
  # downward distance from the left for the same row

  print(" ")
  biggest_return_upward_row <- which(upward_dist == max(upward_dist))
  CurrentLabel <- "row with biggest return upward"
  print(CurrentLabel)
  print(biggest_return_upward_row)
  results_report_DF <- AddReportLine(results_report_DF, CurrentLabel, biggest_return_upward_row)

  print(" ")
  CurrentLabel<-"downward distance for row with the largest upward value"
```

```
  print(CurrentLabel)
  print(downward_dist[biggest_return_upward_row])
  results_report_DF <- AddReportLine(results_report_DF, CurrentLabel, downward_dist[biggest_return_upwa

  CurrentLabel<-"return upward value"
  print(upward_dist[biggest_return_upward_row])
   results_report_DF <- AddReportLine(results_report_DF,
                                      CurrentLabel,
                                      upward_dist[biggest_return_upward_row])

  print(" ")
  # percentage return upward
  percentage_return_upward <- upward_dist[biggest_return_upward_row]/downward_dist[biggest_return_upwar
  CurrentLabel <- "Return upward as a proportion of the downward distance:"
  print(CurrentLabel)
  print(percentage_return_upward)
  results_report_DF <- AddReportLine(results_report_DF, CurrentLabel,
                         percentage_return_upward)
}else{
  print("no U-shape in this participant")
}
```

```
## [1] "no U-shape in this participant"
```

```
FLPModelEquations<-c(
  # models with log frequency, but no three-way interactions (due to difficulty interpreting and small
            "preserved ~ stimlen*log_freq",
            "preserved ~ stimlen+log_freq",
            "preserved ~ pos*log_freq",
            "preserved ~ pos+log_freq",
            "preserved ~ stimlen*log_freq + pos*log_freq",
            "preserved ~ stimlen*log_freq + pos",
            "preserved ~ stimlen + pos*log_freq",
            "preserved ~ stimlen + pos + log_freq",
            "preserved ~ (I(pos^2)+pos)*log_freq",
            "preserved ~ stimlen*log_freq + (I(pos^2) + pos)*log_freq",
            "preserved ~ stimlen*log_freq + I(pos^2) + pos",
            "preserved ~ stimlen + (I(pos^2) + pos)*log_freq",
            "preserved ~ stimlen + I(pos^2) + pos + log_freq",

            # models without frequency
            "preserved ~ 1",
            "preserved ~ stimlen",
            "preserved ~ pos",
            "preserved ~ stimlen + pos",
            "preserved ~ stimlen*pos",
            "preserved ~ I(pos^2)+pos",
            "preserved ~ stimlen + I(pos^2) + pos",
            "preserved ~ stimlen * (I(pos^2) + pos)"
)

FLPRes<-TestModels(FLPModelEquations,PosDat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##  (Intercept)          pos      log_freq  pos:log_freq
##      3.19994     -0.05324       0.22166      -0.02629
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1503  AIC: 1626
## log likelihood:  -751.3289
## Nagelkerke R2:  0.01046618
## % pres/err predicted correctly:  -367.4996
## % of predictable range [ (model-null)/(1-null) ]:  0.003267208
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos     log_freq
##     3.15471     -0.04035      0.11633
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1505  AIC: 1626
## log likelihood:  -752.3931
## Nagelkerke R2:  0.008785358
```

```
## % pres/err predicted correctly:  -367.6925
## % of predictable range [ (model-null)/(1-null) ]:  0.002745511
## **************************
## model index:  9
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##       (Intercept)           I(pos^2)                pos           log_freq  I(pos^2):log_freq
##          2.815873          -0.026436           0.182160           0.308295           0.006087
##      pos:log_freq
##         -0.081092
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1498  AIC: 1627
## log likelihood:  -748.9086
## Nagelkerke R2:  0.01428569
## % pres/err predicted correctly:  -366.9066
## % of predictable range [ (model-null)/(1-null) ]:  0.004871325
## **************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq  stimlen:log_freq
##          3.13903          -0.02158           0.39675          -0.03573
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1504  AIC: 1627
## log likelihood:  -751.9112
## Nagelkerke R2:  0.009546552
## % pres/err predicted correctly:  -367.6273
## % of predictable range [ (model-null)/(1-null) ]:  0.002921877
## **************************
## model index:  11
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq          I(pos^2)               pos
##          2.57456           0.01834           0.41266          -0.03029           0.22228
## stimlen:log_freq
##         -0.03760
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1498  AIC: 1628
## log likelihood:  -749.1179
```

```
## Nagelkerke R2:  0.01395547
## % pres/err predicted correctly:  -367.0411
## % of predictable range [ (model-null)/(1-null) ]:  0.00450759
## **************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     log_freq
##     3.13633     -0.01816      0.11695
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1506  AIC: 1628
## log likelihood:  -753.0204
## Nagelkerke R2:  0.007794087
## % pres/err predicted correctly:  -367.845
## % of predictable range [ (model-null)/(1-null) ]:  0.002332995
## **************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq               pos  stimlen:log_freq
##        3.136e+00        -9.052e-05         3.969e-01        -4.134e-02        -3.574e-02
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1503  AIC: 1628
## log likelihood:  -751.2813
## Nagelkerke R2:  0.01054138
## % pres/err predicted correctly:  -367.484
## % of predictable range [ (model-null)/(1-null) ]:  0.00330951
## **************************
## model index:  13
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos     log_freq
##     2.59517      0.02088     -0.02909      0.21221      0.11821
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1501  AIC: 1628
## log likelihood:  -750.383
## Nagelkerke R2:  0.01195946
## % pres/err predicted correctly:  -367.279
## % of predictable range [ (model-null)/(1-null) ]:  0.00386394
```

```
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##  (Intercept)       stimlen           pos      log_freq  pos:log_freq
##      3.12317       0.01191      -0.05702       0.22617      -0.02682
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1503  AIC: 1628
## log likelihood:  -751.3012
## Nagelkerke R2:  0.01050994
## % pres/err predicted correctly:  -367.4948
## % of predictable range [ (model-null)/(1-null) ]:  0.003280385
## ***************************
## model index:  8
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos     log_freq
##    3.133004     0.003317    -0.041313     0.116994
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1505  AIC: 1628
## log likelihood:  -752.3909
## Nagelkerke R2:  0.00878881
## % pres/err predicted correctly:  -367.6932
## % of predictable range [ (model-null)/(1-null) ]:  0.002743663
## ***************************
## model index:  12
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##       (Intercept)            stimlen           I(pos^2)               pos           log_freq
##          2.590776           0.031178          -0.028138           0.187220           0.318382
## I(pos^2):log_freq       pos:log_freq
##          0.005982          -0.081538
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4005 Residual
## Null Deviance:       1516
## Residual Deviance: 1497  AIC: 1629
## log likelihood:  -748.7268
## Nagelkerke R2:  0.01457241
## % pres/err predicted correctly:  -366.8733
## % of predictable range [ (model-null)/(1-null) ]:  0.004961422
## ***************************
```

```
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq               pos  stimlen:log_freq
##         3.131909          0.005978          0.394999         -0.051543         -0.025580
##     log_freq:pos
##        -0.019094
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1502  AIC: 1629
## log likelihood:  -750.8252
## Nagelkerke R2:  0.01126149
## % pres/err predicted correctly:  -367.3893
## % of predictable range [ (model-null)/(1-null) ]:  0.003565603
## **************************
## model index:  10
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##       (Intercept)            stimlen           log_freq           I(pos^2)                pos
##          2.616416           0.023673           0.547930          -0.027343           0.186731
##  stimlen:log_freq  log_freq:I(pos^2)       log_freq:pos
##         -0.030889           0.007821          -0.088492
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4004 Residual
## Null Deviance:       1516
## Residual Deviance: 1496  AIC: 1629
## log likelihood:  -748.0612
## Nagelkerke R2:  0.01562181
## % pres/err predicted correctly:  -366.723
## % of predictable range [ (model-null)/(1-null) ]:  0.005367979
## **************************
## model index:  19
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.74987     -0.02919      0.20856
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1632
## log likelihood:  -754.6963
## Nagelkerke R2:  0.005144461
## % pres/err predicted correctly:  -368.0201
## % of predictable range [ (model-null)/(1-null) ]:  0.00185947
```

```
## **************************
## model index:  16
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.17598     -0.05148
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -756.7923
## Nagelkerke R2:  0.001827377
## % pres/err predicted correctly:  -368.4056
## % of predictable range [ (model-null)/(1-null) ]:  0.0008167095
## **************************
## model index:  15
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##     3.38742     -0.05359
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -757.1979
## Nagelkerke R2:  0.001185227
## % pres/err predicted correctly:  -368.5278
## % of predictable range [ (model-null)/(1-null) ]:  0.0004862924
## **************************
## model index:  14
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.973
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4011 Residual
## Null Deviance:       1516
## Residual Deviance: 1516  AIC: 1633
## log likelihood:  -757.9461
## Nagelkerke R2:  0
## % pres/err predicted correctly:  -368.7075
## % of predictable range [ (model-null)/(1-null) ]:  0
## **************************
## model index:  20
##
```

```
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     2.86145     -0.01547     -0.02838      0.20638
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1634
## log likelihood:  -754.6469
## Nagelkerke R2:  0.005222613
## % pres/err predicted correctly:  -368.0019
## % of predictable range [ (model-null)/(1-null) ]:  0.001908671
## ***************************
## model index:  17
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     3.38419     -0.03219     -0.04120
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1513  AIC: 1634
## log likelihood:  -756.5701
## Nagelkerke R2:  0.002179197
## % pres/err predicted correctly:  -368.3521
## % of predictable range [ (model-null)/(1-null) ]:  0.000961522
## ***************************
## model index:  18
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
##     2.81614      0.03659      0.14048     -0.02122
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1512  AIC: 1635
## log likelihood:  -756.1378
## Nagelkerke R2:  0.002863618
## % pres/err predicted correctly:  -368.2452
## % of predictable range [ (model-null)/(1-null) ]:  0.001250441
## ***************************
## model index:  21
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
##      (Intercept)           stimlen          I(pos^2)               pos  stimlen:I(pos^2)
##         3.272740         -0.072025         -0.014054          0.008455         -0.002127
##      stimlen:pos
##         0.027528
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1638
## log likelihood:  -754.5441
## Nagelkerke R2:  0.005385125
## % pres/err predicted correctly:  -367.9927
## % of predictable range [ (model-null)/(1-null) ]:  0.001933551
## ***************************
```

```r
BestFLPModel<-FLPRes$ModelResult[[1]]
BestFLPModelFormula<-FLPRes$Model[[1]]

FLPAICSummary<-data.frame(Model=FLPRes$Model,
                          AIC=FLPRes$AIC,row.names=FLPRes$Model)
FLPAICSummary$DeltaAIC<-FLPAICSummary$AIC-FLPAICSummary$AIC[1]
FLPAICSummary$AICexp<-exp(-0.5*FLPAICSummary$DeltaAIC)
FLPAICSummary$AICwt<-FLPAICSummary$AICexp/sum(FLPAICSummary$AICexp)
FLPAICSummary$NagR2<-FLPRes$NagR2

FLPAICSummary <- merge(FLPAICSummary,FLPRes$CoefficientValues,
                           by='row.names',sort=FALSE)
FLPAICSummary <- subset(FLPAICSummary, select = -c(Row.names))

write.csv(FLPAICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_freq_len_pos_model_summary.csv"),row.names
kable(FLPAICSummary)
```

Model	AIC DeltaAIC AICexp AICwt NagR2 (Intercept) [illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	stimlen:I(pos^2)
preserved ~ pos * log_freq	1626.[illegible] NA	0.2216650 NA	- 0.053[illegible]	-[illegible]62914	NA	NA	NA	NA	NA	NA	
preserved ~ pos + log_freq	1626.[illegible] NA	0.1163298 NA	-0.0403482	NA	NA	NA	NA	NA	NA	NA	
preserved ~ (I(pos^2) + pos) * log_freq	1626.[illegible] NA	0.3082047 NA	0.1821601	-0.0810923	NA	-0.0264355	0.0060872 NA	NA	NA		
preserved ~ stimlen * log_freq	1627.[illegible] -0.0215834	0.3967503 -0.0357280	NA	NA	NA	NA	NA	NA	NA	NA	
preserved ~ stimlen * log_freq + I(pos^2) + pos	1627.[illegible]	-0.0375989	0.2222773 NA	NA	-0.0302865	NA	NA	NA	NA		
preserved ~ stimlen + log_freq	1627.[illegible] -0.0181595	0.1169486 NA	NA	NA	NA	NA	NA	NA	NA	NA	

Model	AIC	Delta_AIC	AICwt	NagR2	(Intercept)	stimlen	log_freq	pos	log_freq:pos	I(pos^2)	log_freq:I(pos^2)	pos:I(pos^2)	log_freq:pos:I(pos^2)
preserved ~ stimlen * log_freq + pos	1628.0	0.0	0.3	0.0	0.0	0.0	0.3	-	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + I(pos^2) + pos + log_freq	1628.2	0.2	0.3	0.0	0.0	0.0	0.3	0.2	0.2	NA	-	NA	NA	NA	NA
preserved ~ stimlen + pos * log_freq	1628.3	0.3	0.3	0.0	0.0	0.0	0.3	-	-	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + pos + log_freq	1628.4	0.4	0.3	0.0	0.0	0.0	0.3	-	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + (I(pos^2) + pos) * log_freq	1628.5	0.5	0.3	0.0	0.0	0.0	0.3	0.1	NA	-	0.0	NA	NA	NA	NA
preserved ~ stimlen * log_freq + pos * log_freq	1629.1	1.1	0.2	0.0	0.0	0.0	0.3	-	NA	-	NA	NA	NA	NA	NA
preserved ~ stimlen * log_freq + (I(pos^2) + pos) * log_freq	1629.3	1.3	0.2	0.0	0.0	0.0	0.3	0.1	NA	-	-	NA	0.0	NA	NA
preserved ~ I(pos^2) + pos	1632.5	4.5	0.0	0.0	0.0	NA	NA	0.2	0.2	NA	-	NA	NA	NA	NA
preserved ~ pos	1632.6	4.6	0.0	0.0	0.0	NA	NA	-	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen	1633.0	5.0	0.0	0.0	0.0	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
preserved ~ 1	1633.1	5.1	0.0	0.0	0.0	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + I(pos^2) + pos	1633.7	5.7	0.0	0.0	0.0	NA	NA	0.2	0.2	NA	-	NA	NA	NA	NA
preserved ~ stimlen + pos	1633.7	5.7	0.0	0.0	0.0	NA	NA	-	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen * pos	1635.3	7.3	0.0	0.0	0.0	NA	NA	0.1	0.1	NA	NA	NA	NA	-	NA
preserved ~ stimlen * (I(pos^2) + pos)	1637.7	9.7	0.0	0.0	0.0	NA	NA	0.0	0.0	NA	-	NA	NA	0.0	0.0

```r
print(BestFLPModelFormula)
```

```
## [1] "preserved ~ pos * log_freq"
```

```r
print(BestFLPModel)
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##  (Intercept)           pos      log_freq  pos:log_freq
##      3.19994      -0.05324       0.22166      -0.02629
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1503  AIC: 1626
```

```r
# do a median split on frequency to plot hf/ls effects (analysis is continuous)
median_freq <- median(PosDat$log_freq)
PosDat$freq_bin[PosDat$log_freq >= median_freq]<-"hf"
PosDat$freq_bin[PosDat$log_freq <  median_freq]<-"lf"
```

```r
PosDat$FLPFitted<-fitted(BestFLPModel)

HFDat <- PosDat[PosDat$freq_bin == "hf",]
LFDat <- PosDat[PosDat$freq_bin == "lf",]

HF_Plot <- plot_len_pos_obs_predicted(HFDat,paste0(CurPat," - High frequency"),"FLPFitted",c(min_preser
```

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

```r
LF_Plot <- plot_len_pos_obs_predicted(LFDat,paste0(CurPat, " - Low frequency"),"FLPFitted",c(min_preser
```

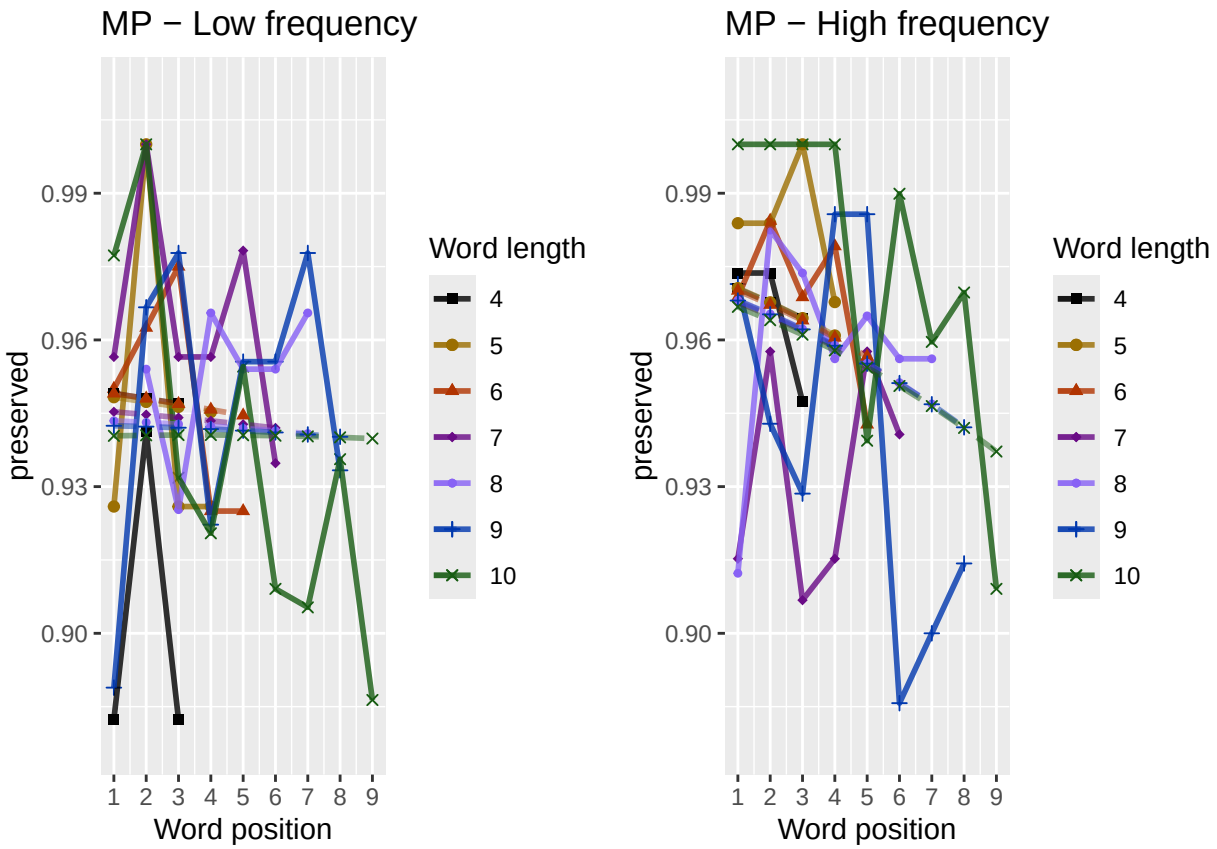
```
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## Scale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

```r
library(ggpubr)
Both_Plots <- ggarrange(LF_Plot,HF_Plot) # labels=c("LF","HF",ncol=2)
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_point()`).
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```

```r
ggsave(paste0(FigDir,CurPat,"_",CurTask,"_frequency_effect_length_pos_wfit.png"),device="png",unit="cm"
print(Both_Plots)
```




```
# only main effects
MEModelEquations<-c(
  "preserved ~ CumPres",
  "preserved ~ CumErr",
  "preserved ~ (I(pos^2)+pos)",
  "preserved ~ pos",
  "preserved ~ stimlen",
  "preserved ~ 1"
)
MERes<-TestModels(MEModelEquations,PosDat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ****************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
```

```
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      2.6277       0.1424
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1501  AIC: 1619
## log likelihood:  -750.5503
## Nagelkerke R2:  0.01169542
## % pres/err predicted correctly:  -367.358
## % of predictable range [ (model-null)/(1-null) ]:  0.003650207
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.74987     -0.02919      0.20856
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1632
## log likelihood:  -754.6963
## Nagelkerke R2:  0.005144461
## % pres/err predicted correctly:  -368.0201
## % of predictable range [ (model-null)/(1-null) ]:  0.00185947
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.17598     -0.05148
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
```

```
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -756.7923
## Nagelkerke R2:  0.001827377
## % pres/err predicted correctly:  -368.4056
## % of predictable range [ (model-null)/(1-null) ]:  0.0008167095
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##     3.38742     -0.05359
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -757.1979
## Nagelkerke R2:  0.001185227
## % pres/err predicted correctly:  -368.5278
## % of predictable range [ (model-null)/(1-null) ]:  0.0004862924
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.973
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4011 Residual
## Null Deviance:       1516
## Residual Deviance: 1516  AIC: 1633
## log likelihood:  -757.9461
## Nagelkerke R2:  0
## % pres/err predicted correctly:  -368.7075
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```
BestMEModel<-MERes$ModelResult[[ BestModelIndexL1 ]]
BestMEModelFormula<-MERes$Model[[BestModelIndexL1]]

MEAICSummary<-data.frame(Model=MERes$Model,
                         AIC=MERes$AIC,row.names=MERes$Model)
MEAICSummary$DeltaAIC<-MEAICSummary$AIC-MEAICSummary$AIC[1]
MEAICSummary$AICexp<-exp(-0.5*MEAICSummary$DeltaAIC)
MEAICSummary$AICwt<-MEAICSummary$AICexp/sum(MEAICSummary$AICexp)
MEAICSummary$NagR2<-MERes$NagR2

MEAICSummary <- merge(MEAICSummary,MERes$CoefficientValues,
                          by='row.names',sort=FALSE)
MEAICSummary <- subset(MEAICSummary, select = -c(Row.names))
```

```
write.csv(MEAICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_main_effects_model_summary.csv"),row.names
kable(MEAICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumPres	CumErr	I(posˆ2)	pos	stimlen
preserved ~ CumErr	1370.271	0.0000	1	1	0.1849986	3.453059	NA	-1.388724	NA	NA	NA
preserved ~ CumPres	1618.832	248.5612	0	0	0.0116952	2.627686	0.1424215	NA	NA	NA	NA
preserved ~ (I(posˆ2) + pos)	1632.086	261.8149	0	0	0.0051442	2.749866	NA	NA	-0.029187	0.2085560	NA
preserved ~ pos	1632.632	262.3603	0	0	0.0018273	3.175979	NA	NA	NA	-0.0514827	NA
preserved ~ stimlen	1633.081	262.8094	0	0	0.0011853	2.387423	NA	NA	NA	NA	-0.0535934
preserved ~ 1	1633.102	262.8312	0	0	0.0000000	2.973323	NA	NA	NA	NA	NA

```
if(DoSimulations){
      BestMEModelFormulaRnd <- BestMEModelFormula
    if(grepl("CumPres",BestMEModelFormulaRnd)){
      BestMEModelFormulaRnd <- gsub("CumPres","RndCumPres",BestMEModelFormulaRnd)
    }else if(grepl("CumErr",BestMEModelFormulaRnd)){
      BestMEModelFormulaRnd <- gsub("CumErr","RndCumErr",BestMEModelFormulaRnd)
    }

    RndModelAIC<-numeric(length=RandomSamples)
    for(rindex in seq(1,RandomSamples)){
    # Shuffle cumulative values
      PosDat$RndCumPres <- ShuffleWithinWord(PosDat,"CumPres")
      PosDat$RndCumErr <- ShuffleWithinWord(PosDat,"CumErr")
      BestModelRnd <- glm(as.formula(BestMEModelFormulaRnd),
                          family="binomial",data=PosDat)
      RndModelAIC[rindex] <- BestModelRnd$aic
    }
    ModelNames<-c(paste0("***",BestMEModelFormula),
                  rep(BestMEModelFormulaRnd,RandomSamples))
    AICValues <- c(BestMEModel$aic,RndModelAIC)
    BestMEModelRndDF <- data.frame(Name=ModelNames,AIC=AICValues)
    BestMEModelRndDF <- BestMEModelRndDF %>% arrange(AIC)
    BestMEModelRndDF <- rbind(BestMEModelRndDF,
                              data.frame(Name=c("Random average"),
                                         AIC=c(mean(RndModelAIC))))
    BestMEModelRndDF <- rbind(BestMEModelRndDF,
                              data.frame(Name=c("Random SD"),
                                         AIC=c(sd(RndModelAIC))))

    write.csv(BestMEModelRndDF,
              paste0(TablesDir,CurPat,"_",CurTask,
                     "_best_main_effects_model_with_random_cum_term.csv"),
              row.names = FALSE)
}
```

```r
syll_component_summary <- PosDat %>%
  group_by(syll_component) %>% summarise(MeanPres = mean(preserved),
                                         N = n())
write.csv(syll_component_summary,paste0(TablesDir,CurPat,"_",CurTask,"_syllable_component_summary.csv")
kable(syll_component_summary)
```

syll_component	MeanPres	N
1	0.9652778	528
O	0.9397699	1854
P	0.6774194	31
S	0.9679767	229
V	0.9650852	1370

```r
#  main effects models for data without satellite positions

keep_components = c("O","V","1")
OV1Data <- PosDat[PosDat$syll_component %in% keep_components,]
OV1Data <- OV1Data %>% select(stim_number,
                              stimlen,stim,pos,
                              preserved,syll_component)
OV1Data$CumPres <- CalcCumPres(OV1Data)
OV1Data$CumErr <- CalcCumErrFromPreserved(OV1Data)

SimpSyllMEAICSummary <- EvaluateSubsetData(OV1Data,MEModelEquations)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.468       -1.483
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3750 Residual
## Null Deviance:       1388
## Residual Deviance: 1162  AIC: 1258
## log likelihood:  -581.0585
## Nagelkerke R2:  0.1889702
## % pres/err predicted correctly:  -288.1109
## % of predictable range [ (model-null)/(1-null) ]:  0.1428079
## ***************************
## model index:  1
##
```

```
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      2.7454       0.1092
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3750 Residual
## Null Deviance:       1388
## Residual Deviance: 1381  AIC: 1500
## log likelihood:  -690.3057
## Nagelkerke R2:  0.00638848
## % pres/err predicted correctly:  -335.6172
## % of predictable range [ (model-null)/(1-null) ]:  0.001955284
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.33716     -0.08354
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3750 Residual
## Null Deviance:       1388
## Residual Deviance: 1382  AIC: 1503
## log likelihood:  -691.2004
## Nagelkerke R2:  0.004848854
## % pres/err predicted correctly:  -335.6376
## % of predictable range [ (model-null)/(1-null) ]:  0.001894827
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     3.07678     -0.01709      0.07070
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3749 Residual
## Null Deviance:       1388
## Residual Deviance: 1381  AIC: 1504
## log likelihood:  -690.5475
## Nagelkerke R2:  0.005972593
## % pres/err predicted correctly:  -335.5083
## % of predictable range [ (model-null)/(1-null) ]:  0.002278253
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
## (Intercept)
##       3.001
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3751 Residual
## Null Deviance:       1388
## Residual Deviance: 1388  AIC: 1507
## log likelihood:  -694.0154
## Nagelkerke R2:  7.180631e-16
## % pres/err predicted correctly:  -336.2767
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##     3.40440     -0.05225
##
## Degrees of Freedom: 3751 Total (i.e. Null);  3750 Residual
## Null Deviance:       1388
## Residual Deviance: 1387  AIC: 1507
## log likelihood:  -693.3633
## Nagelkerke R2:  0.001123871
## % pres/err predicted correctly:  -336.1248
## % of predictable range [ (model-null)/(1-null) ]:  0.0004500902
## ***************************
```

```
write.csv(SimpSyllMEAICSummary,
          paste0(TablesDir,CurPat,"_",CurTask,
                 "_OV1_main_effects_model_summary.csv"),row.names = FALSE)
kable(SimpSyllMEAICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumPres	CumErr	I(pos^2)	pos	stimlen
preserved ~ CumErr	1257.746	0.0000	1	1	0.1889703	3.468464	NA	-1.483216	NA	NA	NA
preserved ~ CumPres	1500.133	242.3865	0	0	0.0063883	2.745375	0.1091536	NA	NA	NA	NA
preserved ~ pos	1502.792	245.0464	0	0	0.0048489	3.337165	NA	NA	NA	-0.0835356	NA
preserved ~ (I(pos^2) + pos)	1504.418	246.6721	0	0	0.0059726	3.076783	NA	NA	-0.0170873	0.0707033	NA
preserved ~ 1	1506.615	248.8690	0	0	0.0000000	3.000640	NA	NA	NA	NA	NA
preserved ~ stimlen	1506.721	248.9747	0	0	0.0011239	3.404397	NA	NA	NA	NA	-0.0522482

```
#  main effects models for data without satellite or coda positions
# (tradeoff -- takes away more complex segments, in addition to satellites, but
#  also reduces data)

keep_components = c("O","V")
```

```r
OVData <- PosDat[PosDat$syll_component %in% keep_components,]
OVData <- OVData %>% select(stim_number,
                            stimlen,stim,pos,
                            preserved,syll_component)
OVData$CumPres <- CalcCumPres(OVData)
OVData$CumErr <- CalcCumErrFromPreserved(OVData)

SimpSyllMEAICSummary2<-EvaluateSubsetData(OVData,MEModelEquations)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.390       -1.618
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3222 Residual
## Null Deviance:       1233
## Residual Deviance: 1053  AIC: 1136
## log likelihood:  -526.4572
## Nagelkerke R2:  0.17114
## % pres/err predicted correctly:  -262.1449
## % of predictable range [ (model-null)/(1-null) ]:  0.12832
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##     3.28074     -0.08103
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3222 Residual
## Null Deviance:       1233
## Residual Deviance: 1228  AIC: 1331
## log likelihood:  -614.1117
## Nagelkerke R2:  0.004897266
## % pres/err predicted correctly:  -300.3112
## % of predictable range [ (model-null)/(1-null) ]:  0.001892355
## ***************************
## model index:  1
##
```

```
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##     2.76385      0.09634
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3222 Residual
## Null Deviance:       1233
## Residual Deviance: 1229  AIC: 1332
## log likelihood:  -614.6232
## Nagelkerke R2:  0.003900304
## % pres/err predicted correctly:  -300.5164
## % of predictable range [ (model-null)/(1-null) ]:  0.00121256
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     3.06280     -0.01483      0.05187
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3221 Residual
## Null Deviance:       1233
## Residual Deviance: 1227  AIC: 1332
## log likelihood:  -613.643
## Nagelkerke R2:  0.005810481
## % pres/err predicted correctly:  -300.2221
## % of predictable range [ (model-null)/(1-null) ]:  0.002187567
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.956
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3223 Residual
## Null Deviance:       1233
## Residual Deviance: 1233  AIC: 1334
## log likelihood:  -616.623
## Nagelkerke R2:  -6.985609e-16
## % pres/err predicted correctly:  -300.8824
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
## (Intercept)      stimlen
##     3.27780     -0.04185
##
## Degrees of Freedom: 3223 Total (i.e. Null);  3222 Residual
## Null Deviance:       1233
## Residual Deviance: 1232  AIC: 1335
## log likelihood:  -616.2428
## Nagelkerke R2:  0.0007418908
## % pres/err predicted correctly:  -300.7879
## % of predictable range [ (model-null)/(1-null) ]:  0.0003131826
## **************************
```

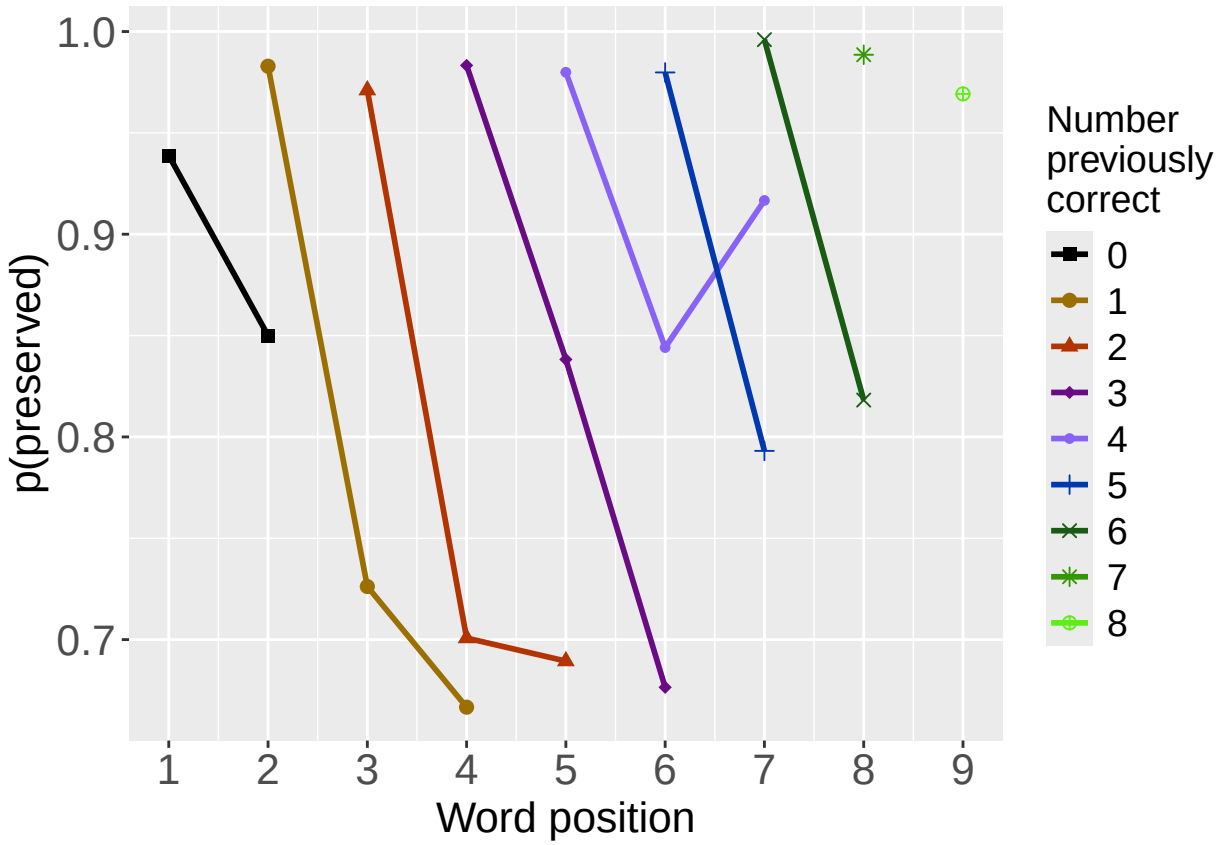
```
write.csv(SimpSyllMEAICSummary2,
          paste0(TablesDir,CurPat,"_",CurTask,
                 "_OV_main_effects_model_summary.csv"),row.names = FALSE)
kable(SimpSyllMEAICSummary2)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumPres	CumErr	I(posˆ2)	pos	stimlen
preserved ~ CumErr	1136.368	0.0000	1	1	0.171140	3.390245	NA	-1.617817	NA	NA	NA
preserved ~ pos	1330.685	194.3163	0	0	0.004897	3.280737	NA	NA	NA	-0.0810299	NA
preserved ~ CumPres	1331.704	195.3358	0	0	0.003900	2.763847	0.0963437	NA	NA	NA	NA
preserved ~ (I(posˆ2) + pos)	1332.371	196.0024	0	0	0.005810	3.062802	NA	NA	-0.0148274	0.0518667	NA
preserved ~ 1	1334.064	197.6955	0	0	0.000000	2.955596	NA	NA	NA	NA	NA
preserved ~ stimlen	1334.746	198.3774	0	0	0.000741	9.277799	NA	NA	NA	NA	-0.0418515

```
# plot prev err and prev cor plots
PrevCorPlot<-PlotObsPreviousCorrect(PosDat,palette_values,shape_values)
```

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

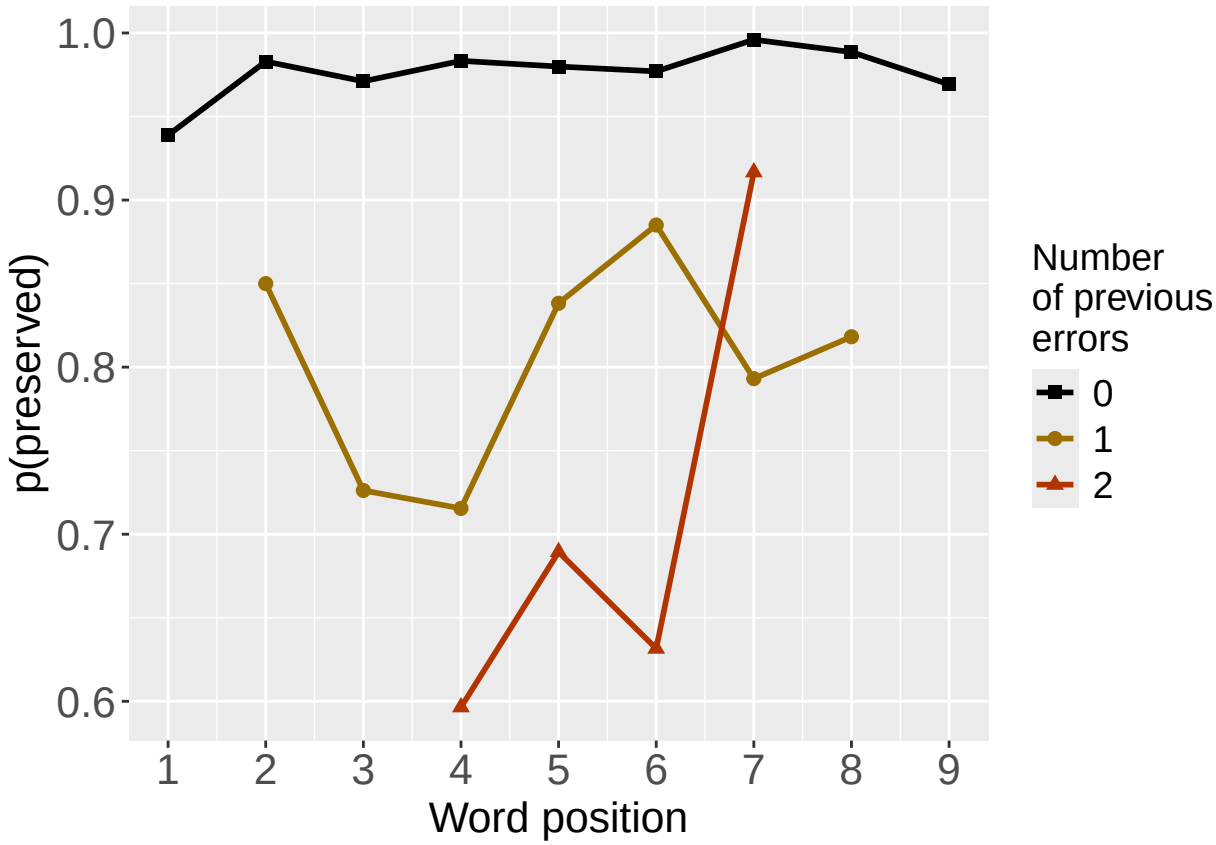
```
print(PrevCorPlot)
```




```
PrevErrPlot<-PlotObsPreviousError(PosDat,palette_values,shape_values)
```

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

```
print(PrevErrPlot)
```




```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"_PrevCorPlot_Obs")
ggsave(filename=paste0(PlotName,".tif"),plot=PrevCorPlot,device="tiff",compression = "lzw")
```

```
## Saving 6.5 x 4.5 in image
```

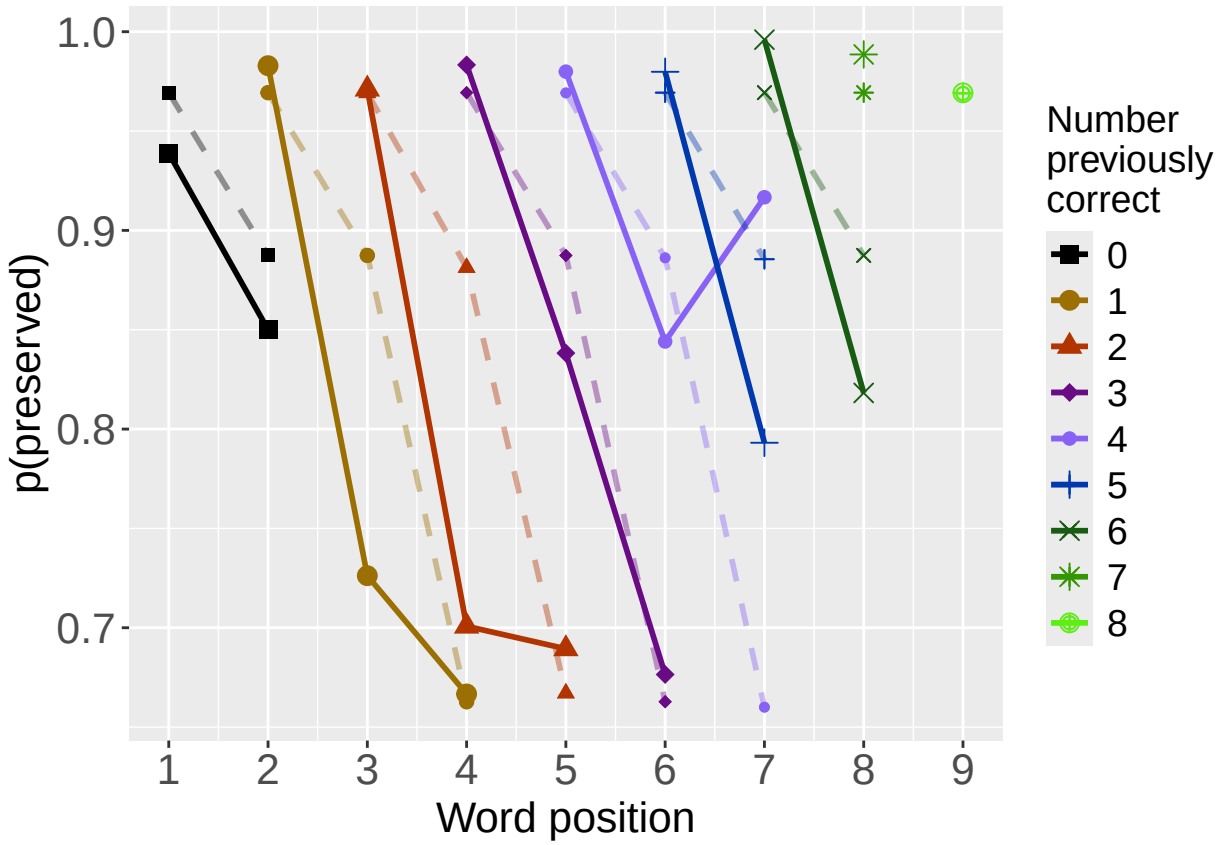
```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"_PrevErrPlot_Obs")
ggsave(filename=paste0(PlotName,".tif"),plot=PrevErrPlot,device="tiff",compression = "lzw")
```

```
## Saving 6.5 x 4.5 in image
```

```
# plot prev err and prev cor with predicted values
MEModel<-MERes$ModelResult[[ BestModelIndexL1 ]]
PosDat$MEPred<-fitted(MEModel)

PrevCorPlot<-PlotObsPredPreviousCorrect(PosDat,"preserved","MEPred",palette_values,shape_values)
```

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

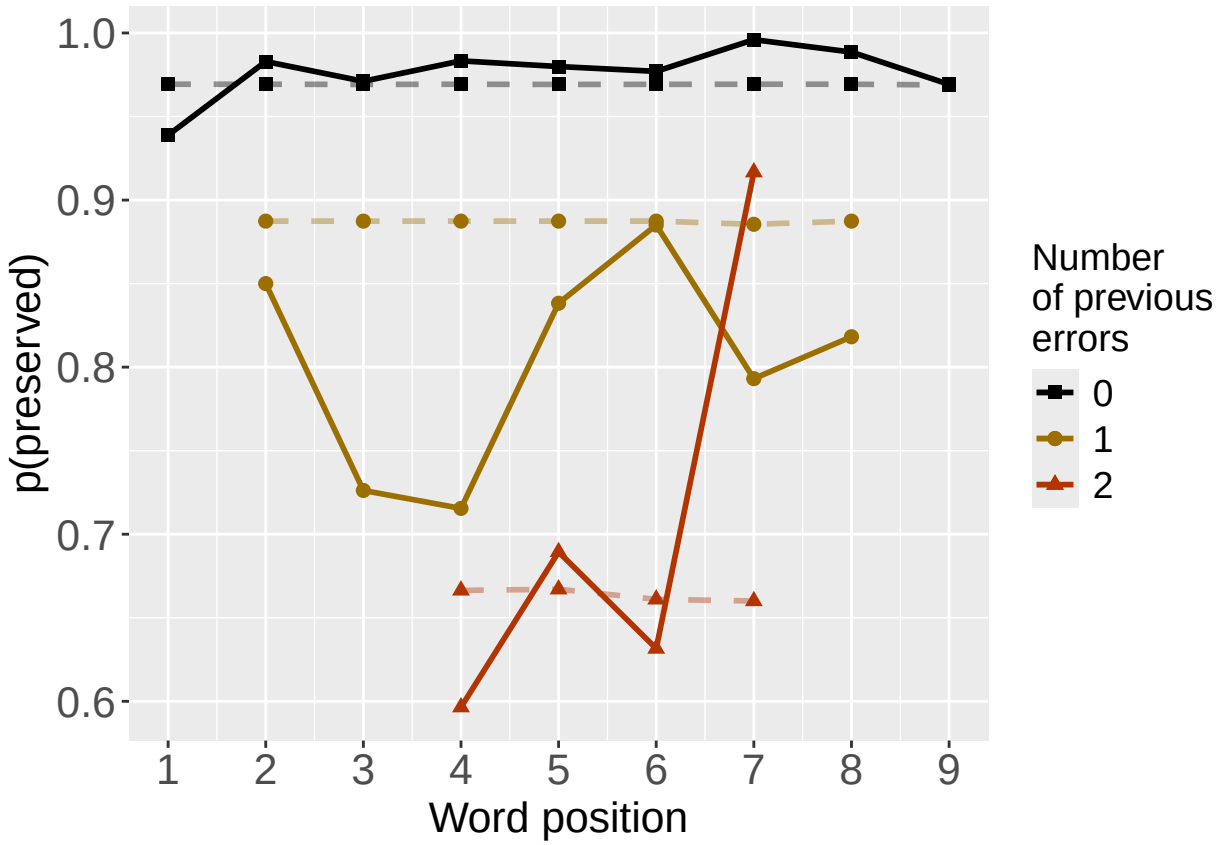
```
print(PrevCorPlot)
```




```
PrevErrPlot<-PlotObsPredPreviousError(PosDat,"preserved","MEPred",palette_values,shape_values)
```

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

```
print(PrevErrPlot)
```




```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"PrevCorPlot_ObsPredME")
ggsave(filename=paste0(PlotName,".tif"),plot=PrevCorPlot,device="tiff",compression = "lzw")
```

```
## Saving 6.5 x 4.5 in image
```

```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"PrevErrPlot_ObsPredME")
ggsave(filename=paste0(PlotName,".tif"),plot=PrevErrPlot,device="tiff",compression = "lzw")
```

```
## Saving 6.5 x 4.5 in image
```

```r
CumAICSummary <- NULL
```

```r
########
# level 2 -- Add position squared (quadratic with position)
########

# After establishing the primary variable, see about additions

BestMEFactor<-BestMEModelFormula
OtherFactor<-"I(pos^2) + pos"
OtherFactorName<-"quadratic_by_pos"

if(!grepl("pos",BestMEFactor,fixed = TRUE)){ # if best model does not include pos
  AICSummary<-TestLevel2Models(PosDat,BestMEFactor,OtherFactor,OtherFactorName)
  CumAICSummary <- AICSummary
  kable(AICSummary)
}
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ****************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos
##     2.47482     -1.56423     -0.03333      0.44460
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1257  AIC: 1356
## log likelihood:  -628.2755
## Nagelkerke R2:  0.1989338
## % pres/err predicted correctly:  -315.6968
## % of predictable range [ (model-null)/(1-null) ]:  0.1433857
```

```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     2.74987     -0.02919      0.20856
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1509  AIC: 1632
## log likelihood:  -754.6963
## Nagelkerke R2:  0.005144461
## % pres/err predicted correctly:  -368.0201
## % of predictable range [ (model-null)/(1-null) ]:  0.00185947
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(pos^2)	pos
preserved ~ CumErr + I(pos^2) + pos	1355.521	0.0000	1.0000000	0.9993739	0.1989338	2.474823	-1.564229	-0.0333336	0.4446014

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(posˆ2)	pos
preserved ~ CumErr	1370.271	14.7507	0.0006265	0.0006261	0.1849986	3.453059	-1.388724	NA	NA
preserved ~ I(posˆ2) + pos	1632.086	276.5656	0.0000000	0.0000000	0.0051445	2.749866	NA	-0.0291870	0.2085560

```
########
# level 2 -- Add length
########

BestMEFactor<-BestMEModelFormula
OtherFactor<-"stimlen"

if(!grepl(OtherFactor,BestMEFactor,fixed = TRUE)){
  AICSummary<-TestLevel2Models(PosDat,BestMEFactor,OtherFactor)
  CumAICSummary <- ConcatAndAddMissingColumns(CumAICSummary,AICSummary)
  kable(AICSummary)
}
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr      stimlen
##     3.19880     -1.39892      0.03358
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1372
## log likelihood:  -637.4052
## Nagelkerke R2:  0.1853445
## % pres/err predicted correctly:  -316.8122
## % of predictable range [ (model-null)/(1-null) ]:  0.1403686
## ***************************
## model index:  3
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##     3.38742     -0.05359
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -757.1979
## Nagelkerke R2:  0.001185227
## % pres/err predicted correctly:  -368.5278
## % of predictable range [ (model-null)/(1-null) ]:  0.0004862924
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	stimlen
preserved ~ CumErr	1370.271	0.000000	1.0000000	0.7033709	0.1849986	3.453059	-1.388724	NA
preserved ~ CumErr + stimlen	1371.998	1.726804	0.4217249	0.2966291	0.1853445	3.198804	-1.398918	0.0335758
preserved ~ stimlen	1633.081	262.809415	0.0000000	0.0000000	0.0011852	3.387423	NA	-0.0535934

```
########
# level 2 -- add cumulative preserved
########

BestMEFactor<-BestMEModelFormula
OtherFactor<-"CumPres"

if(!grepl(OtherFactor,BestMEFactor,fixed = TRUE)){
  AICSummary<-TestLevel2Models(PosDat,BestMEFactor,OtherFactor)
  CumAICSummary <- ConcatAndAddMissingColumns(CumAICSummary,AICSummary)
  kable(AICSummary)
}
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr      CumPres
##      3.0706      -1.3966       0.1606
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
```

```
## Null Deviance:       1516
## Residual Deviance: 1261  AIC: 1356
## log likelihood:  -630.3368
## Nagelkerke R2:  0.1958711
## % pres/err predicted correctly:  -316.0654
## % of predictable range [ (model-null)/(1-null) ]:  0.1423887
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      2.6277       0.1424
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1501  AIC: 1619
## log likelihood:  -750.5503
## Nagelkerke R2:  0.01169542
## % pres/err predicted correctly:  -367.358
## % of predictable range [ (model-null)/(1-null) ]:  0.003650207
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	CumPres
preserved ~ CumErr + CumPres	1355.974	0.0000	1.0000000	0.9992148	0.1958711	3.070585	-1.396593	0.1605824
preserved ~ CumErr	1370.271	14.2976	0.0007858	0.0007852	0.1849986	3.453059	-1.388724	NA
preserved ~ CumPres	1618.832	262.8588	0.0000000	0.0000000	0.0116954	2.627686	NA	0.1424215

```
########
# level 2 -- Add linear position (NOT quadratic)
########
```

```
BestMEFactor<-BestMEModelFormula
OtherFactor<-"pos"

if(!grepl(OtherFactor,BestMEFactor,fixed = TRUE)){
  AICSummary<-TestLevel2Models(PosDat,BestMEFactor,OtherFactor)
  CumAICSummary <- ConcatAndAddMissingColumns(CumAICSummary,AICSummary)
  kable(AICSummary)
}
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ****************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr          pos
##      2.9100      -1.5572       0.1606
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1261  AIC: 1356
## log likelihood:  -630.3368
## Nagelkerke R2:  0.1958711
## % pres/err predicted correctly:  -316.0654
## % of predictable range [ (model-null)/(1-null) ]:  0.1423887
## ****************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## ****************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
## (Intercept)          pos
##     3.17598     -0.05148
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1514  AIC: 1633
## log likelihood:  -756.7923
## Nagelkerke R2:  0.001827377
## % pres/err predicted correctly:  -368.4056
## % of predictable range [ (model-null)/(1-null) ]:  0.0008167095
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	pos
preserved ~ CumErr + pos	1355.974	0.0000	1.0000000	0.9992148	0.1958711	2.910002	-1.557175	0.1605824
preserved ~ CumErr	1370.271	14.2976	0.0007858	0.0007852	0.1849986	3.453059	-1.388724	NA
preserved ~ pos	1632.632	276.6580	0.0000000	0.0000000	0.0018274	3.175979	NA	-0.0514827

```
CumAICSummary <- CumAICSummary %>% arrange(AIC)
BestModelFormulaL2 <- CumAICSummary$Model[BestModelIndexL2]
write.csv(CumAICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_main_effects_plus_one_model_summary.csv")
kable(CumAICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(pos^2)	pos	stimlen	CumPres
preserved ~ CumErr + I(pos^2) + pos	1355.520	0.000000	1.000000	0.9993739	0.1989333	2.8474823	-1.564229	-0.0333336	0.4446014	NA	NA
preserved ~ CumErr + pos	1355.974	0.000000	1.000000	0.9992148	0.1958712	2.910002	-1.557175	NA	0.1605824	NA	NA
preserved ~ CumErr + CumPres	1355.974	0.000000	1.000000	0.9992148	0.1958713	3.1070585	-1.396593	NA	NA	NA	0.1605824
preserved ~ CumErr	1370.271	14.750704	0.000626	0.0006261	0.1849986	3.453059	-1.388724	NA	NA	NA	NA
preserved ~ CumErr	1370.270	0.000000	1.000000	0.9703370	0.1849986	3.453059	-1.388724	NA	NA	NA	NA
preserved ~ CumErr	1370.271	14.297606	0.000785	0.0007852	0.1849986	3.453059	-1.388724	NA	NA	NA	NA
preserved ~ CumErr	1370.271	14.297606	0.000785	0.0007852	0.1849986	3.453059	-1.388724	NA	NA	NA	NA
preserved ~ CumErr + stimlen	1371.998	18.726804	0.421724	0.2966291	0.1853445	3.5198804	-1.398918	NA	NA	0.0335758	NA
preserved ~ CumPres	1618.832	262.858788	0.000000	0.0000000	0.0011695	4.4627686	NA	NA	NA	NA	0.1424215
preserved ~ I(pos^2) + pos	1632.080	276.565620	0.000000	0.0000000	0.0005144	3.5749866	NA	-0.0291870	0.2085560	NA	NA
preserved ~ pos	1632.632	276.657955	0.000000	0.0000000	0.0018273	3.4175979	NA	NA	-0.0514827	NA	NA
preserved ~ stimlen	1633.082	262.809405	0.000000	0.0000000	0.0011852	3.2387423	NA	NA	NA	-0.0535934	NA

```r
# explore influence of frequency and length

if(grepl("stimlen",BestModelFormulaL2)){ # if length is already in the formula
  Level3ModelEquations<-c(
      BestModelFormulaL2, # best model from level 2
    "preserved ~ 1", # null model
      paste0(BestModelFormulaL2," + log_freq")
  )
}else if(grepl("log_freq",BestModelFormulaL2)){ # if log_freq is already in the formula
    Level3ModelEquations<-c(
      BestModelFormulaL2, # best model from level 2
    "preserved ~ 1", # null model
      paste0(BestModelFormulaL2," + stimlen")
  )
}else{
  Level3ModelEquations<-c(
      BestModelFormulaL2, # best model from level 2
    "preserved ~ 1", # null model
      paste0(BestModelFormulaL2," + log_freq"),
      paste0(BestModelFormulaL2," + stimlen"),
      paste0(BestModelFormulaL2," + stimlen + log_freq")
  )
}

Level3Res<-TestModels(Level3ModelEquations,PosDat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos     log_freq
##     2.47960     -1.55906     -0.03171      0.44270      0.11070
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1250  AIC: 1352
## log likelihood:  -624.9677
## Nagelkerke R2:  0.2038423
## % pres/err predicted correctly:  -316.3779
## % of predictable range [ (model-null)/(1-null) ]:  0.1415433
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
```

```
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos      stimlen     log_freq
##    2.421714    -1.558954    -0.032105     0.443967     0.007942     0.112245
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4006 Residual
## Null Deviance:       1516
## Residual Deviance: 1250  AIC: 1354
## log likelihood:  -624.9572
## Nagelkerke R2:  0.2038578
## % pres/err predicted correctly:  -316.3896
## % of predictable range [ (model-null)/(1-null) ]:  0.1415118
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos
##     2.47482     -1.56423     -0.03333      0.44460
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1257  AIC: 1356
## log likelihood:  -628.2755
## Nagelkerke R2:  0.1989338
## % pres/err predicted correctly:  -315.6968
## % of predictable range [ (model-null)/(1-null) ]:  0.1433857
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos      stimlen
##     2.65942     -1.56432     -0.03198      0.44060     -0.02539
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1256  AIC: 1357
## log likelihood:  -628.1603
## Nagelkerke R2:  0.1991049
## % pres/err predicted correctly:  -315.6882
## % of predictable range [ (model-null)/(1-null) ]:  0.1434088
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
```

```
## (Intercept)
##       2.973
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4011 Residual
## Null Deviance:       1516
## Residual Deviance: 1516  AIC: 1633
## log likelihood:  -757.9461
## Nagelkerke R2:  0
## % pres/err predicted correctly:  -368.7075
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```
BestModelL3<-Level3Res$ModelResult[[ BestModelIndexL3 ]]
BestModelFormulaL3<-Level3Res$Model[[BestModelIndexL3]]

AICSummary<-data.frame(Model=Level3Res$Model,
                       AIC=Level3Res$AIC,
                       row.names = Level3Res$Model)
AICSummary$DeltaAIC<-AICSummary$AIC-AICSummary$AIC[1]
AICSummary$AICexp<-exp(-0.5*AICSummary$DeltaAIC)
AICSummary$AICwt<-AICSummary$AICexp/sum(AICSummary$AICexp)
AICSummary$NagR2<-Level3Res$NagR2

AICSummary <- merge(AICSummary,Level3Res$CoefficientValues,
                        by='row.names',sort=FALSE)
AICSummary <- subset(AICSummary, select = -c(Row.names))

write.csv(AICSummary,paste0(TablesDir,CurPat,"_",CurTask,"_main_effects_plus_one_len_freq_summary.csv")

kable(AICSummary)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(pos^2)	pos	log_freq	stimlen
preserved ~ CumErr + I(pos^2) + pos + log_freq	1351.735	0.000000	1.000000	0.6346205	0.2038423	2.479595	-1.559061	-0.0317081	0.4427034	0.4106950	NA
preserved ~ CumErr + I(pos^2) + pos + stimlen + log_freq	1353.818	2.082889	0.3529405	0.2239862	0.2038578	2.421714	-1.558950	-0.0321054	0.4439673	0.4112245	0.0079420
preserved ~ CumErr + I(pos^2) + pos	1355.523	3.787517	0.1506802	0.0956265	0.1989328	2.474823	-1.564229	-0.0333336	0.4446012	NA	NA
preserved ~ CumErr + I(pos^2) + pos + stimlen	1356.995	5.259894	0.0721167	0.0457668	0.1991024	2.659424	-1.564318	-0.0319842	0.4405986	NA	-0.0253938
preserved ~ 1	1633.102	281.367	0.000000	0.000000	0.000000	2.973323	NA	NA	NA	NA	NA

```
# BestModelIndex is set by the parameter file and is used to choose
# a model that is essentially equivalent to the lowest AIC model, but
# simpler (and with a slightly higher AIC value, so not in first position)
BestModelL3<-Level3Res$ModelResult[[ BestModelIndexL3 ]]
BestModelFormulaL3<-Level3Res$Model[BestModelIndexL3]

BestModel<-BestModelL3
BestModelDeletions<-arrange(dropterm(BestModel),desc(AIC))
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
# order by terms that increase AIC the most when they are the one dropped
BestModelDeletions
```

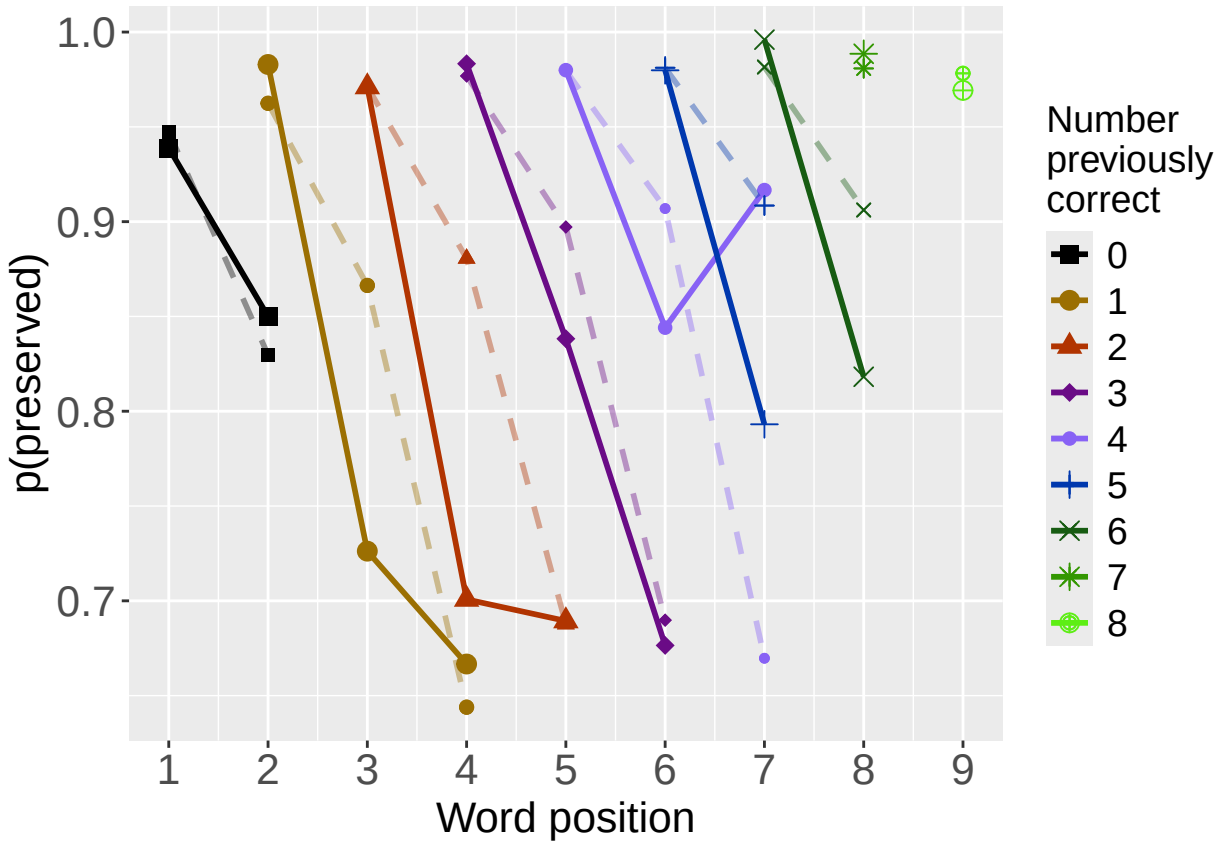
```
## Single term deletions
##
## Model:
## preserved ~ CumErr + I(pos^2) + pos + log_freq
##          Df Deviance    AIC
## CumErr    1   1500.9 1600.7
## pos       1   1259.0 1358.8
## log_freq  1   1256.5 1356.3
## I(pos^2)  1   1253.7 1353.5
## <none>        1249.9 1351.7
```

```
################################
# Single deletions from best model
################################

write.csv(BestModelDeletions,paste0(TablesDir,CurPat,"_",CurTask,"_best_model_single_term_deletions.csv
```

```
# plot prev err and prev cor with predicted values
PosDat$OAPred<-fitted(BestModel)

PrevCorPlot<-PlotObsPredPreviousCorrect(PosDat,"preserved","OAPred",palette_values,shape_values)
```

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

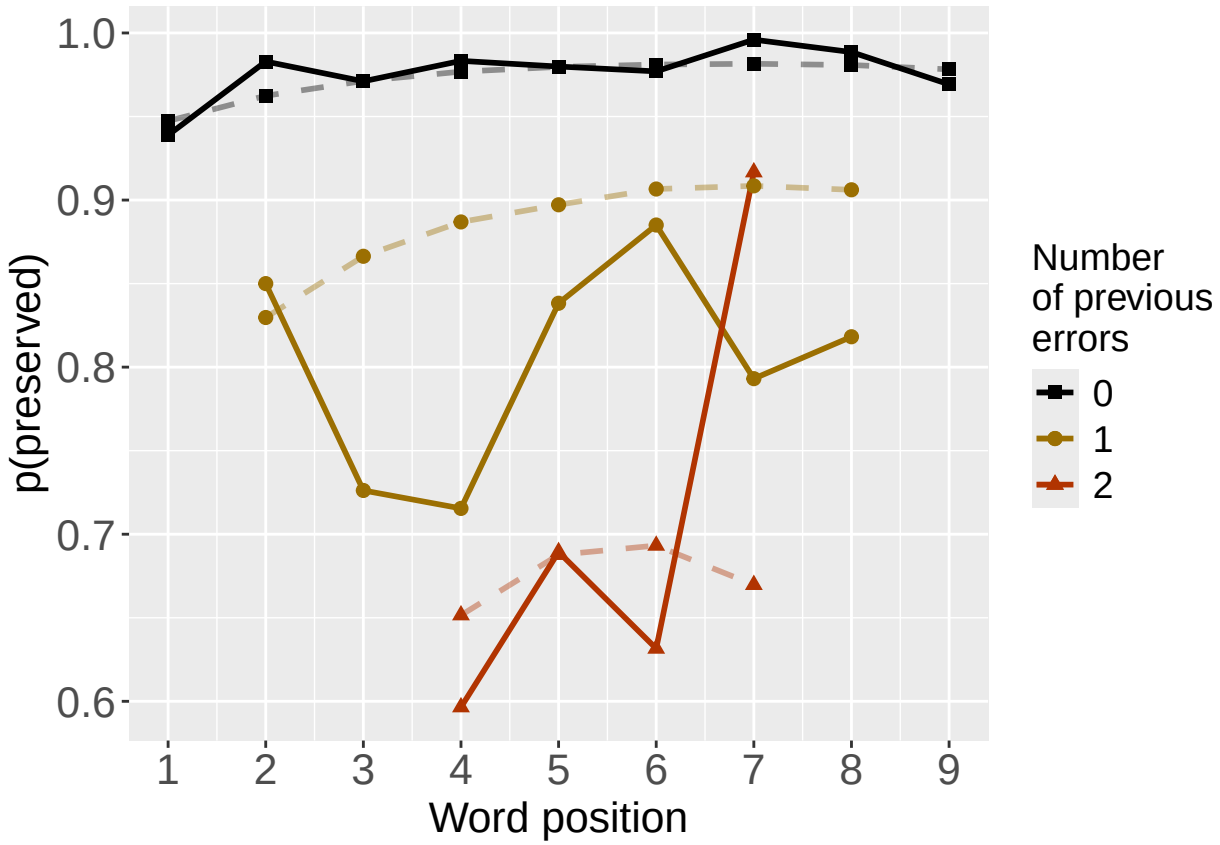
```
print(PrevCorPlot)
```




```
PrevErrPlot<-PlotObsPredPreviousError(PosDat,"preserved","OAPred",palette_values,shape_values)
```

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

```
print(PrevErrPlot)
```




```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"_ObsPred_BestFullModel")
ggsave(filename=paste(PlotName,"_prev_correct.tif",sep=""),plot=PrevCorPlot,device="tiff",compression =
```

```
## Saving 6.5 x 4.5 in image
```

```
ggsave(filename=paste(PlotName,"_prev_error.tif",sep=""),plot=PrevErrPlot,device="tiff",compression = "
```

```
## Saving 6.5 x 4.5 in image
```

```
if(DoSimulations){
  BestModelFormulaL3Rnd <- BestModelFormulaL3
  if(grepl("CumPres",BestModelFormulaL3Rnd)){
    BestModelFormulaL3Rnd <- gsub("CumPres","RndCumPres",BestModelFormulaL3Rnd)
  }
  if(grepl("CumErr",BestModelFormulaL3Rnd)){
    BestModelFormulaL3Rnd <- gsub("CumErr","RndCumErr",BestModelFormulaL3Rnd)
  }

  RndModelAIC<-numeric(length=RandomSamples)
  for(rindex in seq(1,RandomSamples)){
  # Shuffle cumulative values
    PosDat$RndCumPres <- ShuffleWithinWord(PosDat,"CumPres")
    PosDat$RndCumErr <- ShuffleWithinWord(PosDat,"CumErr")
    BestModelRnd <- glm(as.formula(BestModelFormulaL3Rnd),
                        family="binomial",data=PosDat)
    RndModelAIC[rindex] <- BestModelRnd$aic
  }
  ModelNames<-c(paste0("***",BestModelFormulaL3),
```

```
                rep(BestModelFormulaL3Rnd,RandomSamples))
  AICValues <- c(BestModelL3$aic,RndModelAIC)
  BestModelRndDF <- data.frame(Name=ModelNames,AIC=AICValues)
  BestModelRndDF <- BestModelRndDF %>% arrange(AIC)
  BestModelRndDF <- rbind(BestModelRndDF,
                          data.frame(Name=c("Random average"),
                                     AIC=c(mean(RndModelAIC))))
  BestModelRndDF <- rbind(BestModelRndDF,
                          data.frame(Name=c("Random SD"),
                                     AIC=c(sd(RndModelAIC))))
  write.csv(BestModelRndDF,paste0(TablesDir,CurPat,"_",CurTask,
                                  "_best_model_with_random_cum_term.csv"),
            row.names = FALSE)
}
```

```
# plot influence factors
num_models <- nrow(BestModelDeletions) - 1
# minus 1 because last
# model is always <none> and we don't want to include that

if(num_models > 4){
  last_model_num <- 4
}else{
  last_model_num <- num_models
}

FinalModelSet<-ModelSetFromDeletions(BestModelFormulaL3,
                                     BestModelDeletions,
                                     last_model_num)

PlotName<-paste0(FigDir,CurPat,"_",CurTask,"_FactorPlots")

FactorPlot<-PlotModelSetWithFreq(PosDat,FinalModelSet,
                                 palette_values,FinalModelSet,PptID=CurPat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

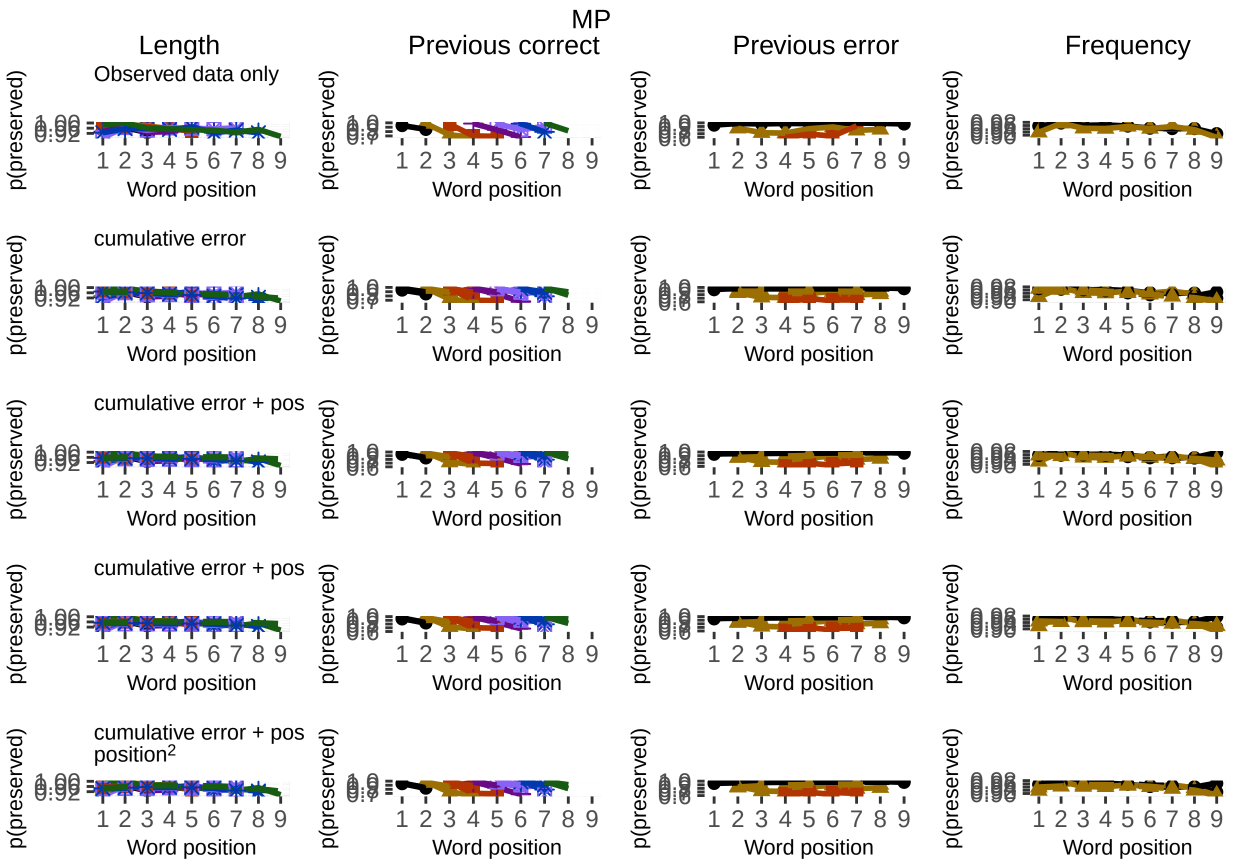
```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.453       -1.389
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4010 Residual
## Null Deviance:       1516
## Residual Deviance: 1275  AIC: 1370
## log likelihood:  -637.637
```

```
## Nagelkerke R2:  0.1849986
## % pres/err predicted correctly:  -316.8303
## % of predictable range [ (model-null)/(1-null) ]:  0.1403198
## **************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr          pos
##      2.9100      -1.5572       0.1606
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4009 Residual
## Null Deviance:       1516
## Residual Deviance: 1261  AIC: 1356
## log likelihood:  -630.3368
## Nagelkerke R2:  0.1958711
## % pres/err predicted correctly:  -316.0654
## % of predictable range [ (model-null)/(1-null) ]:  0.1423887
## **************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr          pos     log_freq
##      2.8939      -1.5527       0.1728       0.1138
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4008 Residual
## Null Deviance:       1516
## Residual Deviance: 1254  AIC: 1352
## log likelihood:  -626.8295
## Nagelkerke R2:  0.2010806
## % pres/err predicted correctly:  -316.7959
## % of predictable range [ (model-null)/(1-null) ]:  0.1404128
## **************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr          pos     log_freq     I(pos^2)
##     2.47960     -1.55906      0.44270      0.11070     -0.03171
##
## Degrees of Freedom: 4011 Total (i.e. Null);  4007 Residual
## Null Deviance:       1516
## Residual Deviance: 1250  AIC: 1352
## log likelihood:  -624.9677
## Nagelkerke R2:  0.2038423
## % pres/err predicted correctly:  -316.3779
## % of predictable range [ (model-null)/(1-null) ]:  0.1415433
```

```
## ***************************
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumPres'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumErr'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'freq_bin'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumPres'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumErr'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'freq_bin'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumPres'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumErr'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'freq_bin'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumPres'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumErr'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'freq_bin'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'stimlen'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumPres'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'CumErr'. You can override using the `.groups` argument.
## `summarise()` has grouped output by 'freq_bin'. You can override using the `.groups` argument.

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 9 rows containing missing values or values outside the scale range
## (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 9 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 4 rows containing missing values or values outside the scale range
## (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 9 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 9 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 9 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 4 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 4 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need that many have
##   them.
```

```
## Warning: Removed 9 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 9 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 9 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 4 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 4 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 9 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 9 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 9 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 4 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 4 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 7 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 9 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 9 rows containing missing values or values outside the scale range (`geom_point()`).

## Warning: The shape palette can deal with a maximum of 6 discrete values because more than 6 becomes
## difficult to discriminate
## i you have requested 9 values. Consider specifying shapes manually if you need that many have
##   them.

## Warning: Removed 4 rows containing missing values or values outside the scale range (`geom_point()`)
## Removed 4 rows containing missing values or values outside the scale range (`geom_point()`).
```

```
ggsave(paste0(PlotName,".tif"),plot=FactorPlot,width = 360,height=400,units="mm",device="tiff",compressi
# use \blandscape and \elandscape to make markdown plots landscape if needed
FactorPlot
```




```
DA.Result<-dominanceAnalysis(BestModel)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
DAContributionAverage<-ConvertDominanceResultToTable(DA.Result)
write.csv(DAContributionAverage,paste0(TablesDir,CurPat,"_",CurTask,"_dominance_analysis_table.csv"),ro
kable(DAContributionAverage)
```

	CumErr	I(pos^2)	pos	log_freq
McFadden	0.1665828	0.0026968	0.0037997	0.0043265
SquaredCorrelation	0.0653700	0.0010487	0.0014567	0.0017104
Nagelkerke	0.0653700	0.0010487	0.0014567	0.0017104
Estrella	0.0716041	0.0011797	0.0016993	0.0018439

```
deviance_differences_df<-GetDevianceSet(FinalModelSet,PosDat)
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
write.csv(deviance_differences_df,paste0(TablesDir,CurPat,"_",CurTask,"_deviance_differences_table.csv"),row.names = FALSE)
deviance_differences_df
```

```
##                                                                 model deviance
## CumErr + pos + log_freq + I(pos^2) CumErr + pos + log_freq + I(pos^2) 1249.935
## CumErr + pos + log_freq                       CumErr + pos + log_freq 1253.659
## CumErr + pos                                        CumErr + pos 1260.674
## CumErr                                                    CumErr 1275.274
## null                                                        null 1515.892
##                                    deviance_explained percent_explained
## CumErr + pos + log_freq + I(pos^2)           265.9569          17.54458
## CumErr + pos + log_freq                      262.2333          17.29894
## CumErr + pos                                 255.2187          16.83620
## CumErr                                       240.6182          15.87304
## null                                           0.0000           0.00000
##                                    percent_of_explained_deviance increment_in_explained
## CumErr + pos + log_freq + I(pos^2)                     100.00000               1.400086
## CumErr + pos + log_freq                                 98.59991               2.637495
## CumErr + pos                                            95.96242               5.489789
## CumErr                                                  90.47263              90.472630
## null                                                          NA               0.000000
```

```
kable(deviance_differences_df[,2:3], format="latex", booktabs=TRUE) %>%
  kable_styling(latex_options="scale_down")
```

```
kable(deviance_differences_df[,c(4:6)], format="latex", booktabs=TRUE) %>%
  kable_styling(latex_options="scale_down")
```

```
NagContributions <- t(DAContributionAverage[3,])
NagTotal<-sum(NagContributions)
NagPercents <- NagContributions / NagTotal
NagResultTable <- data.frame(NagContributions = NagContributions, NagPercents = NagPercents)
```

	deviance	deviance_explained
CumErr + pos + log_freq + I(pos^2)	1249.935	265.9569
CumErr + pos + log_freq	1253.659	262.2333
CumErr + pos	1260.674	255.2187
CumErr	1275.274	240.6182
null	1515.892	0.0000

	percent_explained	percent_of_explained_deviance	increment_in_explained
CumErr + pos + log_freq + I(pos^2)	17.54458	100.00000	1.400086
CumErr + pos + log_freq	17.29894	98.59991	2.637495
CumErr + pos	16.83620	95.96242	5.489789
CumErr	15.87304	90.47263	90.472630
null	0.00000	NA	0.000000

```
names(NagResultTable)<-c("NagContributions","NagPercents")
write.csv(NagResultTable,paste0(TablesDir,CurPat,"_",CurTask,"_nagelkerke_contributions_table.csv"),row.names = TRUE)
NagPercents
```

```
##          Nagelkerke
## CumErr   0.93941653
## I(pos^2) 0.01507082
## pos      0.02093342
## log_freq 0.02457923
```

```
sse_results_list<-compare_SS_accounted_for(FinalModelSet,"preserved ~ 1",PosDat,N_cutoff=10)
```

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

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

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

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

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

## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length

## Warning in (null_cumcor_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length

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

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

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

## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length
## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length

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

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

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

## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length
## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length

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

model	p_accounted_for	model_deviance
preserved ~ CumErr	0.8637594	1275.274
preserved ~ CumErr+pos	0.8646299	1260.674
preserved ~ CumErr+pos+log_freq+I(pos^2)	0.8732845	1249.935
preserved ~ CumErr+pos+log_freq	0.8794862	1253.659

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

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

```
## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length
## Warning in (null_cumerr_resid^2) * (N_values$len_pos_N): longer object length is not a
## multiple of shorter object length
```

```
sse_table<-sse_results_table(sse_results_list)
write.csv(sse_table,paste0(TablesDir,CurPat,"_",CurTask,"_sse_results_table.csv"),row.names = TRUE)
sse_table
```

```
##                                   model p_accounted_for model_deviance  diff_CumErr
## 1                     preserved ~ CumErr       0.8637594       1275.274 0.0000000000
## 2                  preserved ~ CumErr+pos       0.8646299       1260.674 0.0008704981
## 3 preserved ~ CumErr+pos+log_freq+I(pos^2)       0.8732845       1249.935 0.0095250887
## 4          preserved ~ CumErr+pos+log_freq       0.8794862       1253.659 0.0157268016
##   diff_CumErr+pos diff_CumErr+pos+log_freq+I(pos^2) diff_CumErr+pos+log_freq
## 1   -0.0008704981                      -0.009525089             -0.015726802
## 2    0.0000000000                      -0.008654591             -0.014856303
## 3    0.0086545906                       0.000000000             -0.006201713
## 4    0.0148563035                       0.006201713              0.000000000
```

```
kable(sse_table[,1:3], format="latex", booktabs=TRUE) %>%
  kable_styling(latex_options="scale_down")
```

```
write.csv(results_report_DF,paste0(TablesDir,CurPat,"_",CurTask,"_results_report_df.csv"),row.names = FALSE)
```

```
kable(sse_table[,c(1,4:6)], format="latex", booktabs=TRUE) %>%
  kable_styling(latex_options="scale_down")
```

model	diff_CumErr	diff_CumErr+pos	diff_CumErr+pos+log_freq+I(pos^2)
preserved ~ CumErr	0.0000000	-0.0008705	-0.0095251
preserved ~ CumErr+pos	0.0008705	0.0000000	-0.0086546
preserved ~ CumErr+pos+log_freq+I(pos^2)	0.0095251	0.0086546	0.0000000
preserved ~ CumErr+pos+log_freq	0.0157268	0.0148563	0.0062017