TC - 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	506	34	120	NA	NA	660
2	64	NA	400	90	106	660
3	279	NA	166	200	15	660
4	291	NA	211	67	33	602
5	210	NA	200	65	36	511
6	191	1	117	67	20	396
7	156	NA	90	24	18	288
8	80	NA	48	20	4	152
9	63	NA	1	NA	4	68

```
kable(syll_comp_dist_perc)
```

pos_factor	O	P	V	1	S	total
1	0.7666667	0.0515152	0.1818182	NA	NA	660
2	0.0969697	NA	0.6060606	0.1363636	0.1606061	660
3	0.4227273	NA	0.2515152	0.3030303	0.0227273	660
4	0.4833887	NA	0.3504983	0.1112957	0.0548173	602
5	0.4109589	NA	0.3913894	0.1272016	0.0704501	511
6	0.4823232	0.0025253	0.2954545	0.1691919	0.0505051	396

pos_factor	O	P	V	1	S	total
7	0.5416667	NA	0.3125000	0.0833333	0.0625000	288
8	0.5263158	NA	0.3157895	0.1315789	0.0263158	152
9	0.9264706	NA	0.0147059	NA	0.0588235	68

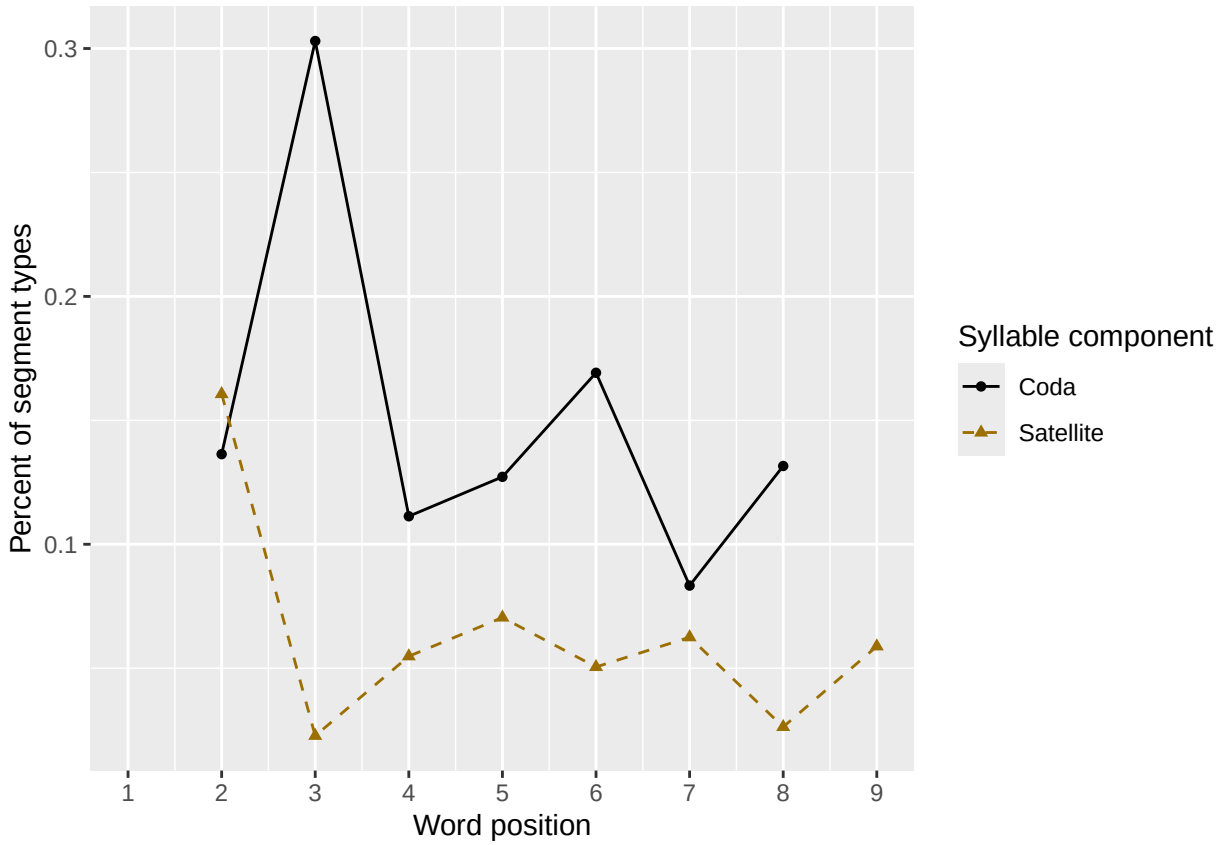
```
# get percentages only from summary table
syll_comp_perc <- syll_comp_dist_perc[,2:(ncol(syll_comp_dist_perc)-1)]
syll_comp_perc$pos <- as.integer(as.character(syll_comp_dist_perc$pos_factor))
syll_comp_perc <- syll_comp_perc %>% rename("Coda" ="1","Satellite"="S")

# 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 1     1     1     NA     NA     NA     NA     NA     NA
## 2       5 0.978 0.965 0.967  0.938 NA     NA     NA     NA     NA
## 3       6 0.991 0.974 0.977  0.929  0.899 NA     NA     NA     NA
## 4       7 0.991 0.963 0.981  0.981  0.954  0.926 NA     NA     NA
## 5       8 0.990 0.966 0.979  0.934  0.934  0.893  0.833 NA     NA
## 6       9 0.976 0.958 0.976  0.966  0.935  0.917  0.883  0.839 NA
## 7      10 1     0.978 0.963  0.926  0.897  0.838  0.868  0.868  0.772
```

```
# 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.
```

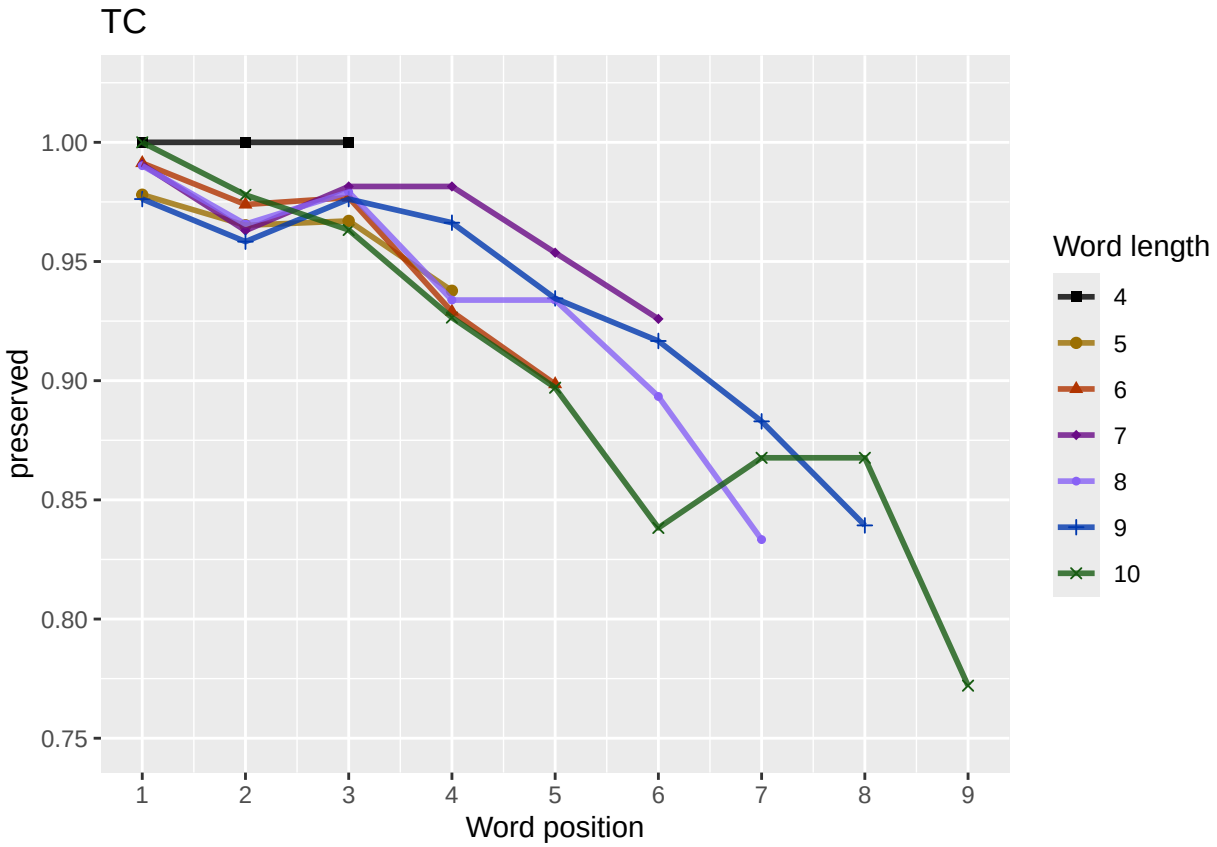
```
pos_len_N_table <- pos_len_N %>% pivot_wider(names_from = pos, values_from = N)
write.csv(pos_len_N_table,paste0(TablesDir,CurPat,"_",CurTask,"_N_by_len_position.csv"))
pos_len_N_table
```

```
## # 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    58    58    58    NA    NA    NA    NA    NA    NA
## 2       5    91    91    91    91    NA    NA    NA    NA    NA
## 3       6   115   115   115   115   115    NA    NA    NA    NA
## 4       7   108   108   108   108   108   108    NA    NA    NA
## 5       8   136   136   136   136   136   136   136    NA    NA
## 6       9    84    84    84    84    84    84    84    84    NA
## 7      10    68    68    68    68    68    68    68    68    68
```

```
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
##     4.85878      0.01809     -0.55558
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1733
## log likelihood:  -786.6536
## Nagelkerke R2:  0.1012647
## % pres/err predicted correctly:  -417.3579
## % of predictable range [ (model-null)/(1-null) ]:  0.04131391
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##      4.4709      -0.3713
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1733
## log likelihood:  -787.4638
## Nagelkerke R2:  0.1001447
## % pres/err predicted correctly:  -416.9457
## % of predictable range [ (model-null)/(1-null) ]:  0.04225867
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     5.15492     -0.04193      0.02012     -0.55956
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1734
## log likelihood:  -786.3549
## Nagelkerke R2:  0.1016775
## % pres/err predicted correctly:  -417.3028
## % of predictable range [ (model-null)/(1-null) ]:  0.04144019
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
```

```
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
##     5.67638     -0.15107     -0.61433      0.02924
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1734
## log likelihood:  -786.5142
## Nagelkerke R2:  0.1014573
## % pres/err predicted correctly:  -417.0748
## % of predictable range [ (model-null)/(1-null) ]:  0.04196278
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     4.63930     -0.02793     -0.36020
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1734
## log likelihood:  -787.3263
## Nagelkerke R2:  0.1003348
## % pres/err predicted correctly:  -416.8799
## % of predictable range [ (model-null)/(1-null) ]:  0.04240935
## ***************************
## 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)
##         5.126793         -0.053810         -0.012213         -0.403528          0.002925
##      stimlen:pos
##        -0.010688
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1572  AIC: 1738
## log likelihood:  -786.2057
## Nagelkerke R2:  0.1018836
## % pres/err predicted correctly:  -417.2521
## % of predictable range [ (model-null)/(1-null) ]:  0.04155646
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
## (Intercept)      stimlen
##      4.8262      -0.2591
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1680  AIC: 1838
## log likelihood:  -839.9217
## Nagelkerke R2:  0.02665592
## % pres/err predicted correctly:  -431.1588
## % of predictable range [ (model-null)/(1-null) ]:  0.009688705
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.782
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3996 Residual
## Null Deviance:       1717
## Residual Deviance: 1717  AIC: 1873
## log likelihood:  -858.614
## Nagelkerke R2:  6.357761e-16
## % pres/err predicted correctly:  -435.3868
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```
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(posˆ2)	stimlen:I(posˆ2)
preserved ~ I(posˆ2) + pos	1732.582	0.0000000	1.0000000	0.6032507	0.0101264	7.7858780	NA	-0.5555836	NA	0.0180911	NA
preserved ~ pos	1732.798	0.2161890	0.8975427	0.5412180	0.0100144	7.7470944	NA	-0.3713085	NA	NA	NA

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ stimlen + I(pos^2) + pos	1733.82	0.2473160	1.535980	0.2162536	0.4101677	5.154915	-	-	NA	0.0201213	NA
							0.0419321	1.5595628			
preserved ~ stimlen * pos	1734.43	5.8528137	7.395974	0.0120079	0.4101457	3.676383	-	-	0.0292417	NA	NA
							0.1510678	0.6143268			
preserved ~ stimlen + pos	1734.46	5.8824905	7.390140	0.0118310	0.4100334	8.639296	-	-	NA	NA	NA
							0.0279309	1.3602032			
preserved ~ stimlen * (I(pos^2) + pos)	1737.68	5.1030500	1.077962	0.7023642	0.4101885	6.126793	-	-	-	-	0.0029245
							0.0538098	1.8403528	1.0106880	0.0122132	
preserved ~ stimlen	1837.73	105.1514	3.08e-24	0.0000000	0.0000000	0.0266549	0.826209	-	NA	NA	NA
							0.2591030				
preserved ~ 1	1873.23	140.6520	6.66e-32	0.0000000	0.0000000	0.0000000	0.782199	NA	NA	NA	NA	NA

```
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
##     4.85878      0.01809     -0.55558
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1733
```

```
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
```

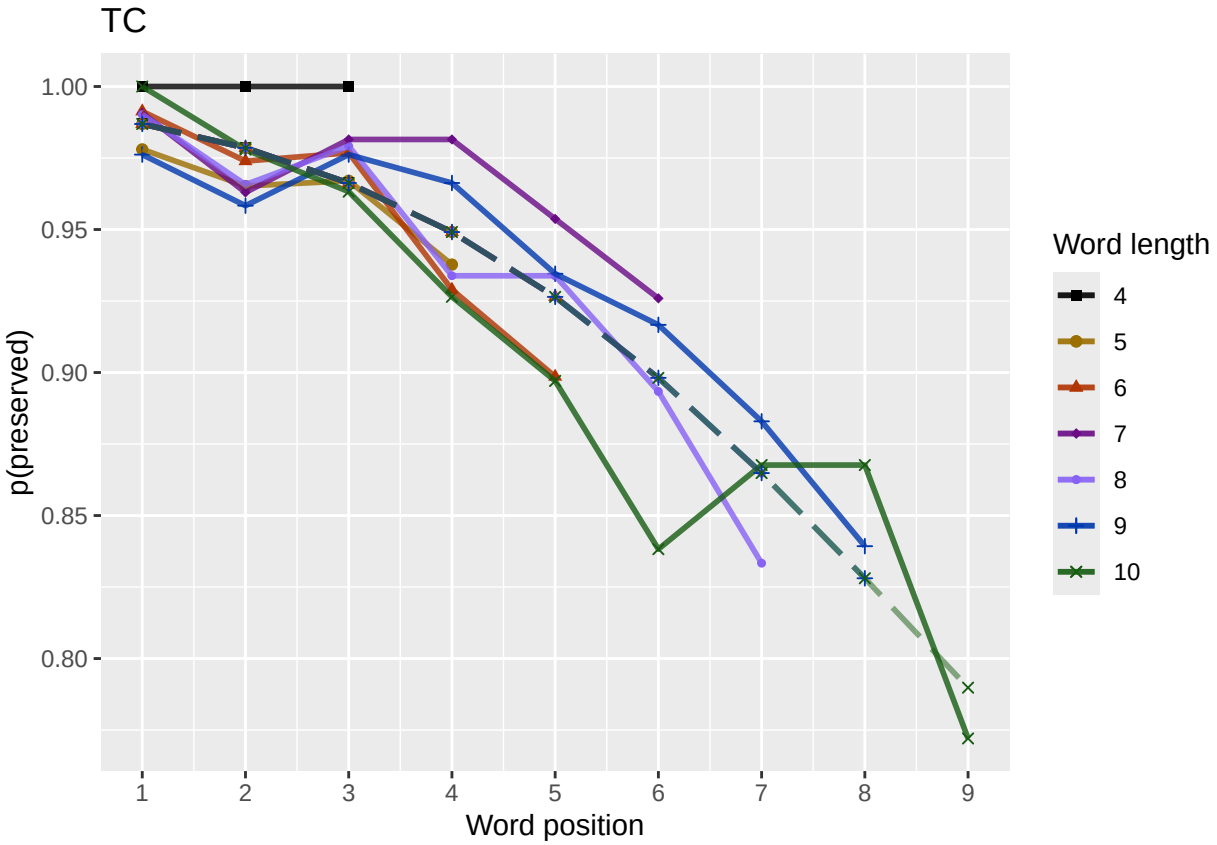
```
## # A tibble: 7 x 10
## # Groups:   stimlen [7]
##   stimlen   `1`   `2`   `3`    `4`    `5`    `6`    `7`    `8`    `9`
##     <int> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
## 1       4 0.987 0.979 0.966 NA     NA     NA     NA     NA     NA
## 2       5 0.987 0.979 0.966  0.949 NA     NA     NA     NA     NA
## 3       6 0.987 0.979 0.966  0.949  0.926 NA     NA     NA     NA
## 4       7 0.987 0.979 0.966  0.949  0.926  0.898 NA     NA     NA
```

```
## 5       8 0.987 0.979 0.966  0.949  0.926  0.898  0.865 NA     NA
## 6       9 0.987 0.979 0.966  0.949  0.926  0.898  0.865  0.828 NA
## 7      10 0.987 0.979 0.966  0.949  0.926  0.898  0.865  0.828  0.790
```

```
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       26   660
```

```
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 26 / 660 = 3.94 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:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
```

```
##
## Coefficients:
## (Intercept)          pos
##      4.4849      -0.3015
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3915 Residual
## Null Deviance:       1286
## Residual Deviance: 1220  AIC: 1350
## log likelihood:  -609.9537
## Nagelkerke R2:  0.05977576
## % pres/err predicted correctly:  -296.627
## % of predictable range [ (model-null)/(1-null) ]:  0.01952972
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     4.72281      0.01211     -0.42100
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3914 Residual
## Null Deviance:       1286
## Residual Deviance: 1219  AIC: 1352
## log likelihood:  -609.685
## Nagelkerke R2:  0.06025783
## % pres/err predicted correctly:  -296.7142
## % of predictable range [ (model-null)/(1-null) ]:  0.01924237
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     4.28635      0.03324     -0.31478
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3914 Residual
## Null Deviance:       1286
## Residual Deviance: 1220  AIC: 1352
## log likelihood:  -609.8047
## Nagelkerke R2:  0.0600431
## % pres/err predicted correctly:  -296.6094
## % of predictable range [ (model-null)/(1-null) ]:  0.01958784
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
```

```
##     5.34900     -0.09520     -0.58808      0.03191
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3913 Residual
## Null Deviance:       1286
## Residual Deviance: 1218  AIC: 1352
## log likelihood:  -609.0696
## Nagelkerke R2:  0.06136145
## % pres/err predicted correctly:  -296.5922
## % of predictable range [ (model-null)/(1-null) ]:  0.01964444
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     4.54354      0.02571      0.01080     -0.41835
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3913 Residual
## Null Deviance:       1286
## Residual Deviance: 1219  AIC: 1353
## log likelihood:  -609.5989
## Nagelkerke R2:  0.06041218
## % pres/err predicted correctly:  -296.6909
## % of predictable range [ (model-null)/(1-null) ]:  0.0193192
## ***************************
## 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)
##         4.710263         -0.030959         -0.054539         -0.177527          0.005549
##      stimlen:pos
##        -0.009510
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3911 Residual
## Null Deviance:       1286
## Residual Deviance: 1218  AIC: 1357
## log likelihood:  -608.9386
## Nagelkerke R2:  0.06159632
## % pres/err predicted correctly:  -296.5069
## % of predictable range [ (model-null)/(1-null) ]:  0.01992541
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##       4.429       -0.163
```

```
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3915 Residual
## Null Deviance:       1286
## Residual Deviance: 1275  AIC: 1406
## log likelihood:  -637.5802
## Nagelkerke R2:  0.009864071
## % pres/err predicted correctly:  -301.6891
## % of predictable range [ (model-null)/(1-null) ]:  0.002853579
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       3.164
##
## Degrees of Freedom: 3916 Total (i.e. Null);  3916 Residual
## Null Deviance:       1286
## Residual Deviance: 1286  AIC: 1415
## log likelihood:  -642.9943
## Nagelkerke R2:  7.934133e-16
## % pres/err predicted correctly:  -302.5554
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```
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 ~ pos	1350.372	0.000000	1.000000	0.3845948	0.0597754	0.8484893	NA	-0.3015464	NA	NA	NA
preserved ~ I(pos^2) + pos	1351.509	1.136847	0.566417	0.2178410	0.0602574	0.8722810	NA	-0.4209992	NA	0.0121095	NA
preserved ~ stimlen + pos	1352.124	1.752231	0.416397	0.1601402	0.0600434	1.2863490	0.0332400	-0.3147840	NA	NA	NA
preserved ~ stimlen * pos	1352.442	2.074790	0.354376	0.1362904	0.0613655	1.5348999	-0.0952038	-0.5880848	0.0319068	NA	NA

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	stimlen	pos	stimlen:pos	I(pos^2)	stimlen:I(pos^2)
preserved ~ stimlen + I(pos^2) + pos	1353.413	1.038728	0.2188501	0.1084169	0.0604124	25435350.0257141	- 0.4183457	NA	0.0108023	NA	
preserved ~ stimlen * (I(pos^2) + pos)	1356.616	5.242739	0.0440967	0.0161695	0.0615963	7710263	- 0.0309585	- 5.1775274	- 0.0095095	0.0055493 0.0545392	
preserved ~ stimlen	1405.725	55.347497	0	0	0	0.0098641	4429076	- 0.1630006	NA	NA	NA
preserved ~ 1	1414.645	64.272566	0	0	0	0	0.0163831	NA	NA	NA	NA

```
# 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.
```

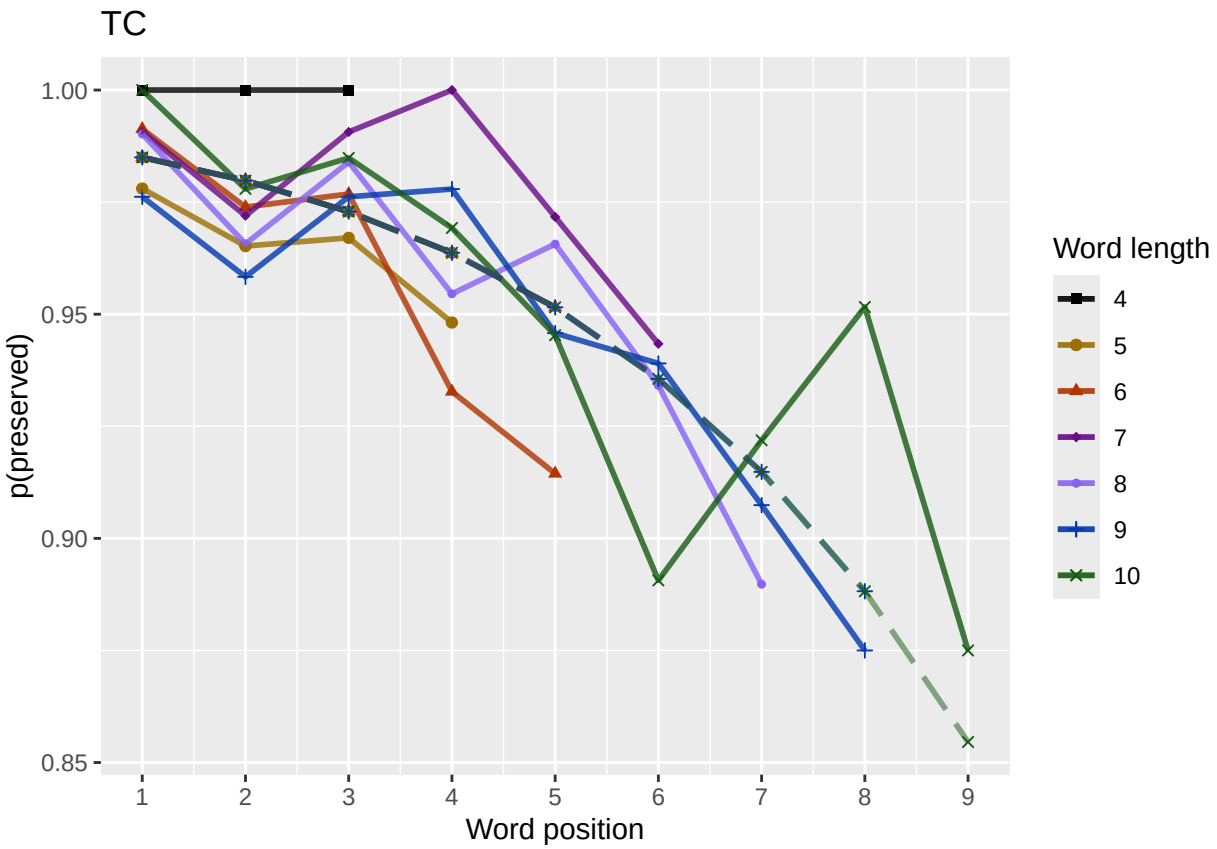
```
nofrag_fitted_pos_len_table <- nofrag_fitted_pos_len_summary %>% pivot_wider(names_from = pos, values_fi
nofrag_fitted_pos_len_table
```

```
## # 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 0.985 0.980 0.973 NA     NA     NA     NA     NA     NA
## 2       5 0.985 0.980 0.973  0.964 NA     NA     NA     NA     NA
## 3       6 0.985 0.980 0.973  0.964  0.952 NA     NA     NA     NA
## 4       7 0.985 0.980 0.973  0.964  0.952  0.936 NA     NA     NA
## 5       8 0.985 0.980 0.973  0.964  0.952  0.936  0.915 NA     NA
## 6       9 0.985 0.980 0.973  0.964  0.952  0.936  0.915  0.888 NA
## 7      10 0.985 0.980 0.973  0.964  0.952  0.936  0.915  0.888  0.855
```

```
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: 0.75 - 1.02"
```

```
# 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.
# 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: "
```

```
print(OA_mean_len_diff)
```

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

```
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): "
```

```
print(OA_mean_pos_diff)
```

```
## [1] -0.02463252
```

```
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:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq               pos  stimlen:log_freq
##         4.433375         -0.005488          0.577052         -0.344365         -0.063983
##     log_freq:pos
##         0.025448
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1545  AIC: 1706
## log likelihood:  -772.6099
## Nagelkerke R2:  0.1206056
## % pres/err predicted correctly:  -412.0275
## % of predictable range [ (model-null)/(1-null) ]:  0.05352891
## ***************************
## model index:  11
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq          I(pos^2)               pos
##         4.869669         -0.004362          0.545273          0.017647         -0.537049
## stimlen:log_freq
##        -0.043727
##
```

```
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1545  AIC: 1707
## log likelihood:  -772.6162
## Nagelkerke R2:  0.1205971
## % pres/err predicted correctly:  -412.9556
## % of predictable range [ (model-null)/(1-null) ]:  0.05140204
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq               pos  stimlen:log_freq
##         4.416998          0.008257          0.551733         -0.362642         -0.044430
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1547  AIC: 1707
## log likelihood:  -773.3549
## Nagelkerke R2:  0.1195831
## % pres/err predicted correctly:  -412.6231
## % of predictable range [ (model-null)/(1-null) ]:  0.05216406
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos     log_freq
##      4.4550      -0.3537       0.1846
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1550  AIC: 1708
## log likelihood:  -775.1141
## Nagelkerke R2:  0.1171668
## % pres/err predicted correctly:  -412.4515
## % of predictable range [ (model-null)/(1-null) ]:  0.05255717
## ***************************
## model index:  10
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##       (Intercept)            stimlen           log_freq           I(pos^2)                pos
##          4.817473          -0.016155           0.362077           0.014289          -0.487619
##  stimlen:log_freq  log_freq:I(pos^2)       log_freq:pos
##         -0.059085          -0.007951           0.106954
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3989 Residual
```

```
## Null Deviance:       1717
## Residual Deviance: 1543  AIC: 1708
## log likelihood:  -771.273
## Nagelkerke R2:  0.1224398
## % pres/err predicted correctly:  -412.3004
## % of predictable range [ (model-null)/(1-null) ]:  0.05290359
## ***************************
## model index:  13
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos     log_freq
##     4.77488      0.01284      0.01874     -0.54863      0.18710
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1548  AIC: 1709
## log likelihood:  -774.1671
## Nagelkerke R2:  0.1184679
## % pres/err predicted correctly:  -412.7655
## % of predictable range [ (model-null)/(1-null) ]:  0.0518376
## ***************************
## model index:  8
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos     log_freq
##     4.29367      0.02635     -0.36339      0.18832
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1550  AIC: 1709
## log likelihood:  -774.9988
## Nagelkerke R2:  0.1173253
## % pres/err predicted correctly:  -412.3961
## % of predictable range [ (model-null)/(1-null) ]:  0.05268412
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##  (Intercept)           pos      log_freq  pos:log_freq
##     4.450134     -0.352624      0.176462      0.001561
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1550  AIC: 1709
## log likelihood:  -775.11
```

```
## Nagelkerke R2:  0.1171725
## % pres/err predicted correctly:  -412.4062
## % of predictable range [ (model-null)/(1-null) ]:  0.05266108
## ***************************
## 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
##           4.75537            0.01348           -0.49252           -0.05190            -0.01027
##      pos:log_freq
##           0.10768
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1547  AIC: 1710
## log likelihood:  -773.2821
## Nagelkerke R2:  0.119683
## % pres/err predicted correctly:  -412.5975
## % of predictable range [ (model-null)/(1-null) ]:  0.05222268
## ***************************
## model index:  7
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##  (Intercept)       stimlen           pos      log_freq  pos:log_freq
##    4.2934001     0.0261433    -0.3629884     0.1857697     0.0004827
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1550  AIC: 1711
## log likelihood:  -774.9984
## Nagelkerke R2:  0.1173259
## % pres/err predicted correctly:  -412.3827
## % of predictable range [ (model-null)/(1-null) ]:  0.0527149
## ***************************
## model index:  12
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##       (Intercept)            stimlen           I(pos^2)                pos           log_freq
##           4.66638            0.01254            0.01283           -0.49089           -0.04765
## I(pos^2):log_freq       pos:log_freq
##          -0.01029            0.10734
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3990 Residual
## Null Deviance:       1717
## Residual Deviance: 1547  AIC: 1712
```

```
## log likelihood:  -773.2576
## Nagelkerke R2:  0.1197167
## % pres/err predicted correctly:  -412.5762
## % of predictable range [ (model-null)/(1-null) ]:  0.0522715
## **************************
## model index:  19
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     4.85878      0.01809     -0.55558
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1733
## log likelihood:  -786.6536
## Nagelkerke R2:  0.1012647
## % pres/err predicted correctly:  -417.3579
## % of predictable range [ (model-null)/(1-null) ]:  0.04131391
## **************************
## model index:  16
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##      4.4709      -0.3713
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1733
## log likelihood:  -787.4638
## Nagelkerke R2:  0.1001447
## % pres/err predicted correctly:  -416.9457
## % of predictable range [ (model-null)/(1-null) ]:  0.04225867
## **************************
## model index:  20
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     I(pos^2)          pos
##     5.15492     -0.04193      0.02012     -0.55956
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1734
## log likelihood:  -786.3549
## Nagelkerke R2:  0.1016775
## % pres/err predicted correctly:  -417.3028
```

```
## % of predictable range [ (model-null)/(1-null) ]:  0.04144019
## ***************************
## model index:  18
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos  stimlen:pos
##     5.67638     -0.15107     -0.61433      0.02924
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1734
## log likelihood:  -786.5142
## Nagelkerke R2:  0.1014573
## % pres/err predicted correctly:  -417.0748
## % of predictable range [ (model-null)/(1-null) ]:  0.04196278
## ***************************
## model index:  17
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen          pos
##     4.63930     -0.02793     -0.36020
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1734
## log likelihood:  -787.3263
## Nagelkerke R2:  0.1003348
## % pres/err predicted correctly:  -416.8799
## % of predictable range [ (model-null)/(1-null) ]:  0.04240935
## ***************************
## 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)
##         5.126793         -0.053810         -0.012213         -0.403528          0.002925
##      stimlen:pos
##        -0.010688
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1572  AIC: 1738
## log likelihood:  -786.2057
## Nagelkerke R2:  0.1018836
## % pres/err predicted correctly:  -417.2521
## % of predictable range [ (model-null)/(1-null) ]:  0.04155646
```

```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq  stimlen:log_freq
##          4.61841          -0.22649           0.55235          -0.04518
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1653  AIC: 1811
## log likelihood:  -826.2919
## Nagelkerke R2:  0.0459359
## % pres/err predicted correctly:  -428.4064
## % of predictable range [ (model-null)/(1-null) ]:  0.0159958
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen     log_freq
##      4.4895      -0.2082       0.1815
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1656  AIC: 1814
## log likelihood:  -828.039
## Nagelkerke R2:  0.04347187
## % pres/err predicted correctly:  -428.2269
## % of predictable range [ (model-null)/(1-null) ]:  0.01640717
## ***************************
## model index:  15
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##      4.8262      -0.2591
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1680  AIC: 1838
## log likelihood:  -839.9217
## Nagelkerke R2:  0.02665592
## % pres/err predicted correctly:  -431.1588
## % of predictable range [ (model-null)/(1-null) ]:  0.009688705
## ***************************
## model index:  14
##
```

```
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.782
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3996 Residual
## Null Deviance:       1717
## Residual Deviance: 1717  AIC: 1873
## log likelihood:  -858.614
## Nagelkerke R2:  6.357761e-16
## % pres/err predicted correctly:  -435.3868
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```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 DeltaAICexpAICwtNagR2(Intercept)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
preserved ~ stimlen * log_freq + pos * log_freq	[illegible]	-0.0054885	0.5770521	-0.0639833	[illegible]	NA	0.0254477	NA	NA	NA	NA	NA
preserved ~ stimlen * log_freq + I(pos^2) + pos	[illegible]	-0.0043624	0.5452729	-0.0437271	[illegible]	NA	NA	0.0176468	NA	NA	NA	NA
preserved ~ stimlen * log_freq + pos	[illegible]	[illegible]	[illegible]	-0.0444293	[illegible]	NA	NA	NA	NA	NA	NA	NA
preserved ~ pos + log_freq	[illegible]	NA	0.1846209	NA	-0.3536655	NA	NA	NA	NA	NA	NA	NA

Model	AIC	DeltaAIC	expAIC	AICw	NagR2	(Intercept)	stimlen	log_freq	stimlen:log_freq	pos	log_freq:pos	I(pos^2)	I(pos^2):log_freq	[illegible]	stimlen:pos	stimlen:I(pos^2)
preserved ~ stimlen * log_freq + (I(pos^2) + pos) * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.0161551	0.3620772	-0.0590846	-0.0876187	NA	[illegible]	[illegible]	[illegible]	-0.007951	NA	NA
preserved ~ stimlen + I(pos^2) + pos + log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.5486336	NA	NA	0.0187406	NA	NA	NA	NA
preserved ~ stimlen + pos + log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.3633867	NA	NA	NA	NA	NA	NA	NA
preserved ~ pos * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	NA	0.1764622	NA	-0.3526240	0.0015614	NA	NA	NA	NA	NA	NA
preserved ~ (I(pos^2) + pos) * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	NA	-0.0518958	NA	-0.4925236	0.1076778	NA	0.0134790	-0.0102660	NA	NA	NA
preserved ~ stimlen + pos * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.3629884	0.0004827	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + (I(pos^2) + pos) * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.0476455	NA	-0.4908911	0.1073394	NA	0.0128258	-0.0102891	NA	NA	NA
preserved ~ I(pos^2) + pos	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	NA	NA	NA	-0.5555836	NA	NA	0.0180911	NA	NA	NA	NA
preserved ~ pos	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	NA	NA	NA	-0.3713085	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen + I(pos^2) + pos	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.0419321	NA	NA	-0.5595628	NA	NA	0.0201213	NA	NA	NA	NA
preserved ~ stimlen * pos	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.1510678	NA	NA	-0.6143268	NA	NA	NA	NA	NA	0.0292417	NA
preserved ~ stimlen + pos	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.0279304	NA	NA	-0.3602032	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen * (I(pos^2) + pos)	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.0538098	NA	NA	-0.4035281	NA	NA	-0.0122132	NA	NA	-0.0106883	0.0029245
preserved ~ stimlen * log_freq	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.2264888	0.5523502	-0.0451768	NA	NA	NA	NA	NA	NA	NA	NA

Model	AIC DeltaAICexpAICwNagR2(Intercept)	stimlen	log_freqstimlen:log_freqposlog_freq:posI(pos^2)log_freq:I(pos^2)stimlen:I(pos^2)								
preserved ~ stimlen + log_freq	1813.517205.09690000000434.7189499	-0.2081626	0.1814888 NA	NA	NA	NA	NA	NA	NA	NA	NA
preserved ~ stimlen	1837.731 3.27022689000000264.5526209	-0.2591030	NA	NA	NA	NA	NA	NA	NA	NA	NA
preserved ~ 1	1873.236.477029040000000002.7821NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

```r
print(BestFLPModelFormula)
```

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

```r
print(BestFLPModel)
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
##      (Intercept)           stimlen          log_freq               pos  stimlen:log_freq
##         4.433375         -0.005488          0.577052         -0.344365         -0.063983
##     log_freq:pos
##         0.025448
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1545  AIC: 1706
```

```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.
```

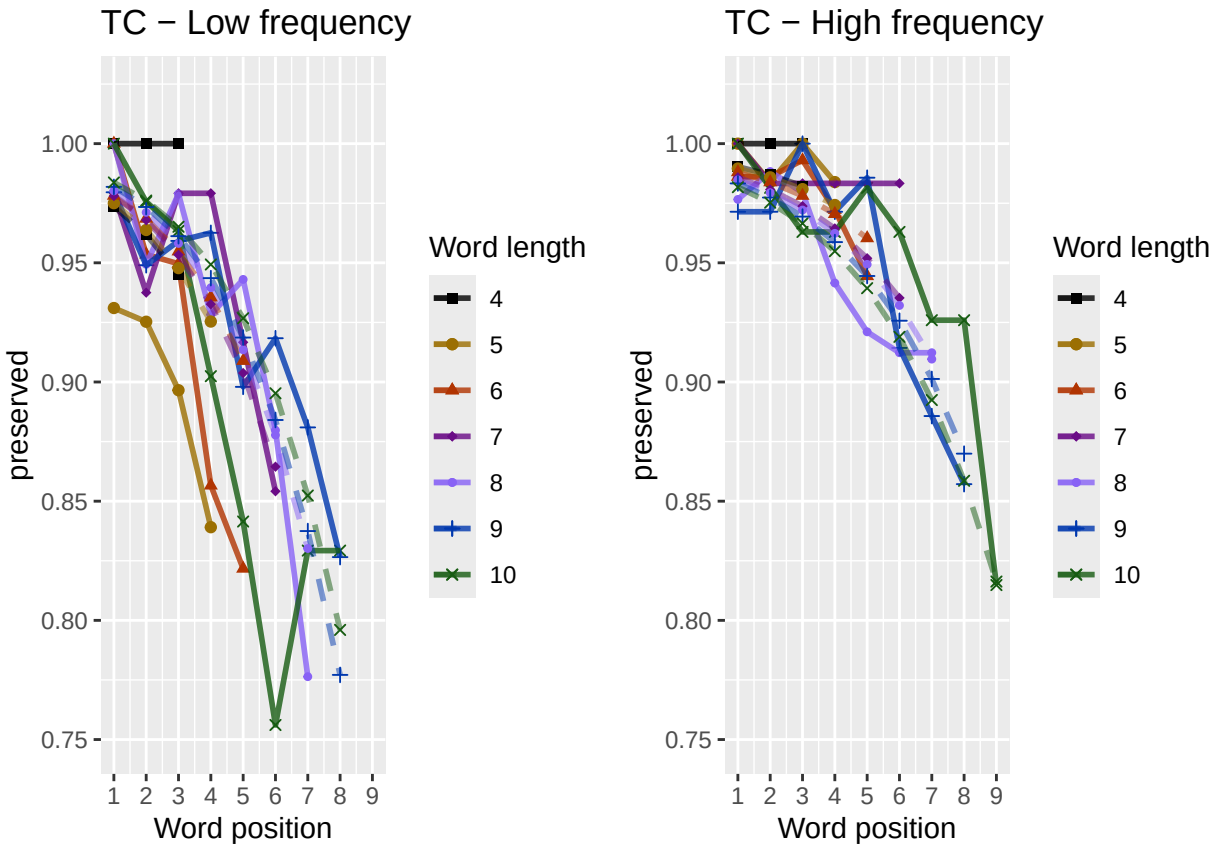
```
library(ggpubr)
Both_Plots <- ggarrange(LF_Plot,HF_Plot) # labels=c("LF","HF",ncol=2)
```

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

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

```
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
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     4.85878      0.01809     -0.55558
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1733
## log likelihood:  -786.6536
## Nagelkerke R2:  0.1012647
## % pres/err predicted correctly:  -417.3579
## % of predictable range [ (model-null)/(1-null) ]:  0.04131391
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##      4.4709      -0.3713
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1733
## log likelihood:  -787.4638
## Nagelkerke R2:  0.1001447
```

```
## % pres/err predicted correctly:  -416.9457
## % of predictable range [ (model-null)/(1-null) ]:  0.04225867
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##      4.8262      -0.2591
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1680  AIC: 1838
## log likelihood:  -839.9217
## Nagelkerke R2:  0.02665592
## % pres/err predicted correctly:  -431.1588
## % of predictable range [ (model-null)/(1-null) ]:  0.009688705
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      3.2039      -0.1426
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1697  AIC: 1858
## log likelihood:  -848.622
## Nagelkerke R2:  0.01428005
## % pres/err predicted correctly:  -433.4101
## % of predictable range [ (model-null)/(1-null) ]:  0.004529765
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.782
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3996 Residual
## Null Deviance:       1717
## Residual Deviance: 1717  AIC: 1873
## log likelihood:  -858.614
## Nagelkerke R2:  6.357761e-16
## % pres/err predicted correctly:  -435.3868
## % 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(pos^2)	pos	stimlen
preserved ~ CumErr	1313.550	0.0000	1	1	0.335186	3.431129	NA	-2.33789	NA	NA	NA
preserved ~ (I(pos^2) + pos)	1732.582	419.0318	0	0	0.101264	7.858780	NA	NA	0.0180911	-0.5555836	NA
preserved ~ pos	1732.798	419.2479	0	0	0.100144	7.470944	NA	NA	NA	-0.3713085	NA
preserved ~ stimlen	1837.733	524.1832	0	0	0.026655	9.826209	NA	NA	NA	NA	-0.259103
preserved ~ CumPres	1857.585	544.0351	0	0	0.014280	8.203861	-0.1425876	NA	NA	NA	NA
preserved ~ 1	1873.234	559.6838	0	0	0.000000	2.782199	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)
}
```

```
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	0.9424640	533
O	0.9372283	1840
P	0.9714286	35
S	0.9173729	236
V	0.9509731	1353

```
#  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
##       3.440       -2.509
##
## Degrees of Freedom: 3725 Total (i.e. Null);  3724 Residual
## Null Deviance:       1572
## Residual Deviance: 1115  AIC: 1206
## log likelihood:  -557.6976
## Nagelkerke R2:  0.3350179
## % pres/err predicted correctly:  -273.4016
## % of predictable range [ (model-null)/(1-null) ]:  0.311281
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     4.84793      0.01867     -0.55050
##
## Degrees of Freedom: 3725 Total (i.e. Null);  3723 Residual
## Null Deviance:       1572
## Residual Deviance: 1448  AIC: 1605
## log likelihood:  -723.8615
## Nagelkerke R2:  0.09522122
## % pres/err predicted correctly:  -382.4404
## % of predictable range [ (model-null)/(1-null) ]:  0.03760523
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)          pos
##      4.4480      -0.3604
##
## Degrees of Freedom: 3725 Total (i.e. Null);  3724 Residual
## Null Deviance:       1572
## Residual Deviance: 1449  AIC: 1605
## log likelihood:  -724.6529
## Nagelkerke R2:  0.09402722
## % pres/err predicted correctly:  -382.0614
## % of predictable range [ (model-null)/(1-null) ]:  0.03855664
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##      4.8746      -0.2619
##
```

```
## Degrees of Freedom: 3725 Total (i.e. Null);  3724 Residual
## Null Deviance:       1572
## Residual Deviance: 1537  AIC: 1692
## log likelihood:  -768.4998
## Nagelkerke R2:  0.02707347
## % pres/err predicted correctly:  -393.5375
## % of predictable range [ (model-null)/(1-null) ]:  0.009752911
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      3.1427      -0.1238
##
## Degrees of Freedom: 3725 Total (i.e. Null);  3724 Residual
## Null Deviance:       1572
## Residual Deviance: 1559  AIC: 1716
## log likelihood:  -779.6165
## Nagelkerke R2:  0.009846562
## % pres/err predicted correctly:  -396.2691
## % of predictable range [ (model-null)/(1-null) ]:  0.002896681
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.805
##
## Degrees of Freedom: 3725 Total (i.e. Null);  3725 Residual
## Null Deviance:       1572
## Residual Deviance: 1572  AIC: 1724
## log likelihood:  -785.9409
## Nagelkerke R2:  0
## % pres/err predicted correctly:  -397.4232
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

```
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	1206.470	0.0000	1	1	0.3350179	3.440013	NA	-2.509035	NA	NA	NA
preserved ~ (I(pos^2) + pos)	1604.863	398.3937	0	0	0.0952212	2.847926	NA	NA	0.0186688	-0.5505012	NA

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumPres	CumErr	I(pos^2)	pos	stimlen
preserved ~ pos	1605.134	398.6646	0	0	0.0940272	2.447977	NA	NA	NA	-0.3604055	NA
preserved ~ stimlen	1691.868	485.3981	0	0	0.0270734	4.874625	NA	NA	NA	NA	-0.2619488
preserved ~ CumPres	1716.387	509.9171	0	0	0.0098466	3.142734	-0.1237699	NA	NA	NA	NA
preserved ~ 1	1724.486	518.0161	0	0	0.0000000	2.805426	NA	NA	NA	NA	NA

```
#  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")
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
##       3.360       -2.671
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3191 Residual
## Null Deviance:       1350
## Residual Deviance: 1010  AIC: 1085
## log likelihood:  -505.0096
## Nagelkerke R2:  0.2926743
## % pres/err predicted correctly:  -247.9019
## % of predictable range [ (model-null)/(1-null) ]:  0.2703295
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
```

```
##
## Coefficients:
## (Intercept)          pos
##      4.4467      -0.3582
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3191 Residual
## Null Deviance:       1350
## Residual Deviance: 1241  AIC: 1357
## log likelihood:  -620.2922
## Nagelkerke R2:  0.09735848
## % pres/err predicted correctly:  -326.3058
## % of predictable range [ (model-null)/(1-null) ]:  0.04048381
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##      4.7589       0.0148      -0.5087
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3190 Residual
## Null Deviance:       1350
## Residual Deviance: 1240  AIC: 1357
## log likelihood:  -619.8232
## Nagelkerke R2:  0.09818188
## % pres/err predicted correctly:  -326.6091
## % of predictable range [ (model-null)/(1-null) ]:  0.03959471
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##       4.774       -0.250
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3191 Residual
## Null Deviance:       1350
## Residual Deviance: 1322  AIC: 1440
## log likelihood:  -660.9287
## Nagelkerke R2:  0.02507899
## % pres/err predicted correctly:  -337.008
## % of predictable range [ (model-null)/(1-null) ]:  0.009109688
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
```

```
##      3.2682      -0.1952
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3191 Residual
## Null Deviance:       1350
## Residual Deviance: 1329  AIC: 1449
## log likelihood:  -664.2697
## Nagelkerke R2:  0.01905413
## % pres/err predicted correctly:  -337.9293
## % of predictable range [ (model-null)/(1-null) ]:  0.00640888
## ***************************
## model index:  6
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)
##       2.807
##
## Degrees of Freedom: 3192 Total (i.e. Null);  3192 Residual
## Null Deviance:       1350
## Residual Deviance: 1350  AIC: 1466
## log likelihood:  -674.7902
## Nagelkerke R2:  9.6624e-16
## % pres/err predicted correctly:  -340.1155
## % of predictable range [ (model-null)/(1-null) ]:  0
## ***************************
```

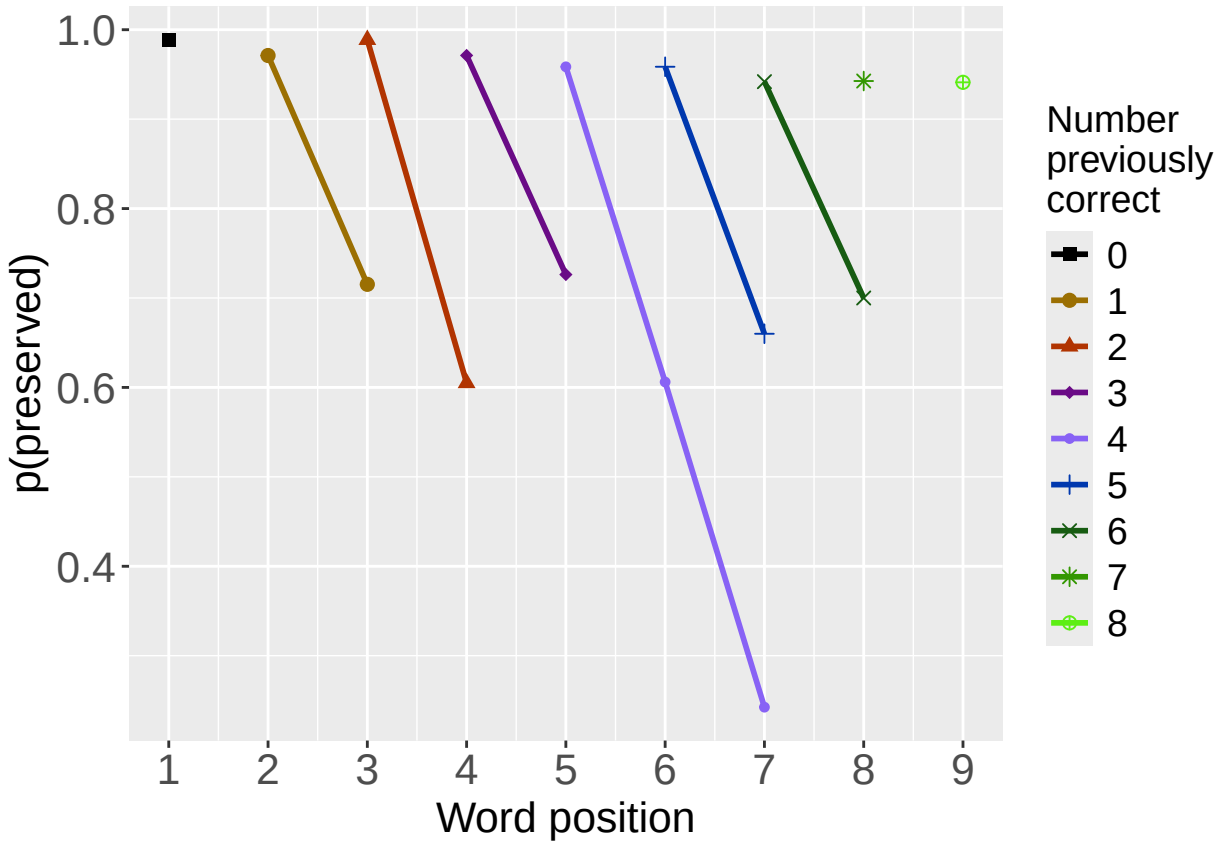
```
write.csv(SimpSyllMEAICSummary2,
          paste0(TablesDir,CurPat,"_",CurTask,
                 "_OV_main_effects_model_summary.csv"),row.names = FALSE)
kable(SimpSyllMEAICSummary2)
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumPres	CumErr	I(pos^2)	pos	stimlen
preserved ~ CumErr	1084.608	0.0000	1	1	0.2926743	3.359649	NA	-2.670571	NA	NA	NA
preserved ~ pos	1356.895	272.2869	0	0	0.0973585	4.446663	NA	NA	NA	-0.3582125	NA
preserved ~ (I(pos^2) + pos)	1357.485	272.8763	0	0	0.0981819	4.758867	NA	NA	0.0148048	-0.5086949	NA
preserved ~ stimlen	1439.616	355.0078	0	0	0.0250790	4.774463	NA	NA	NA	NA	-0.2499895
preserved ~ CumPres	1448.607	363.9986	0	0	0.0190543	3.268156	-0.1951761	NA	NA	NA	NA
preserved ~ 1	1465.545	380.9370	0	0	0.0000000	2.806992	NA	NA	NA	NA	NA

```
# 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.
```

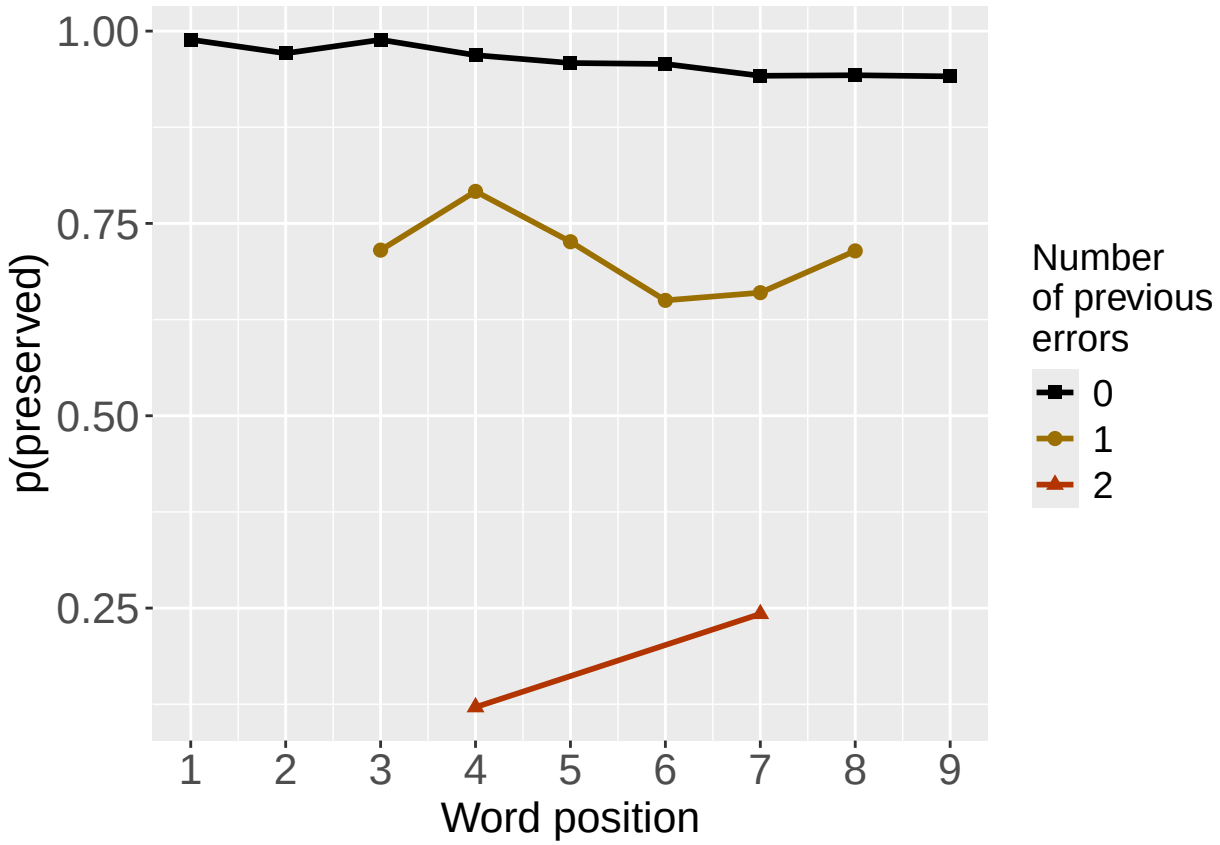
```
print(PrevCorPlot)
```




```
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
```

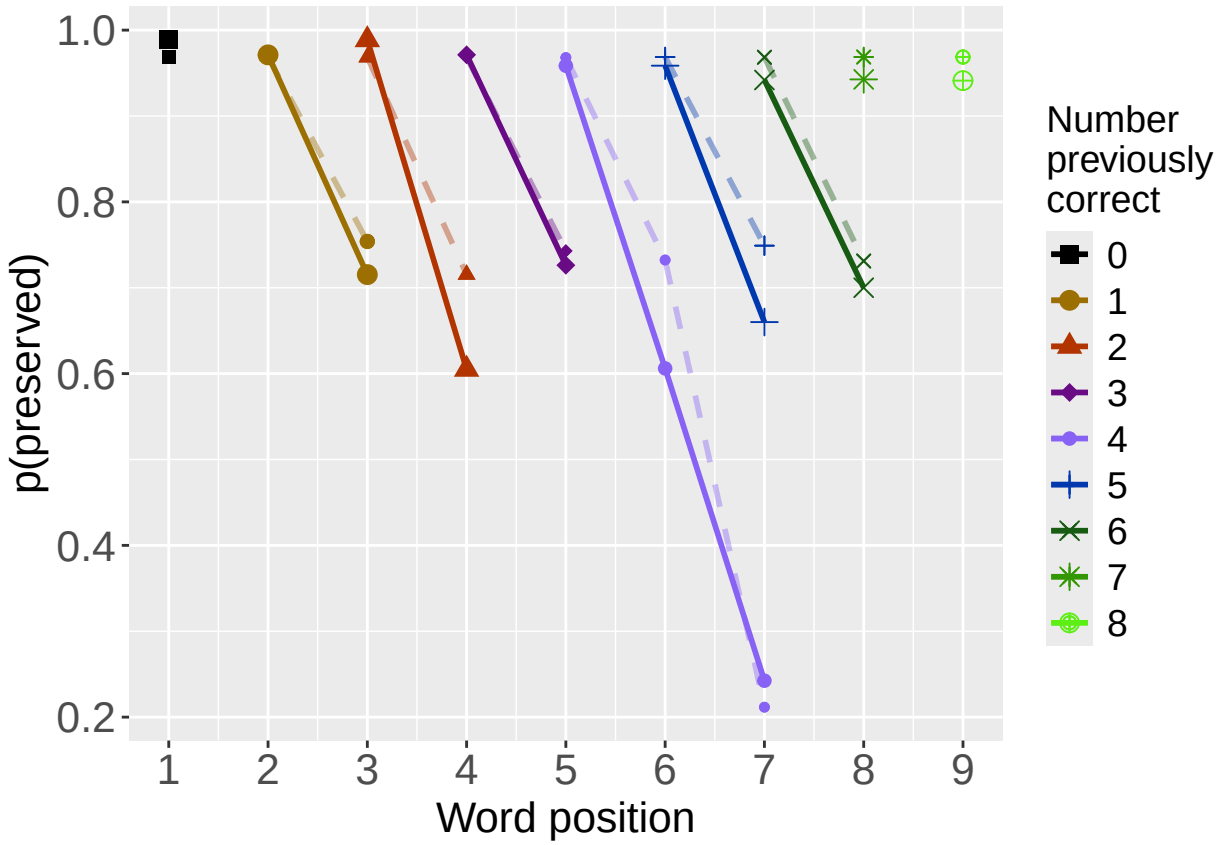
```
PlotName<-paste0(FigDir,CurPat,"_",CurTask,"_PrevErrPlot_Obs")
ggsave(filename=paste0(PlotName,".tif"),plot=PrevErrPlot,device="tiff",compression = "lzw")
```

```
## 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.
```

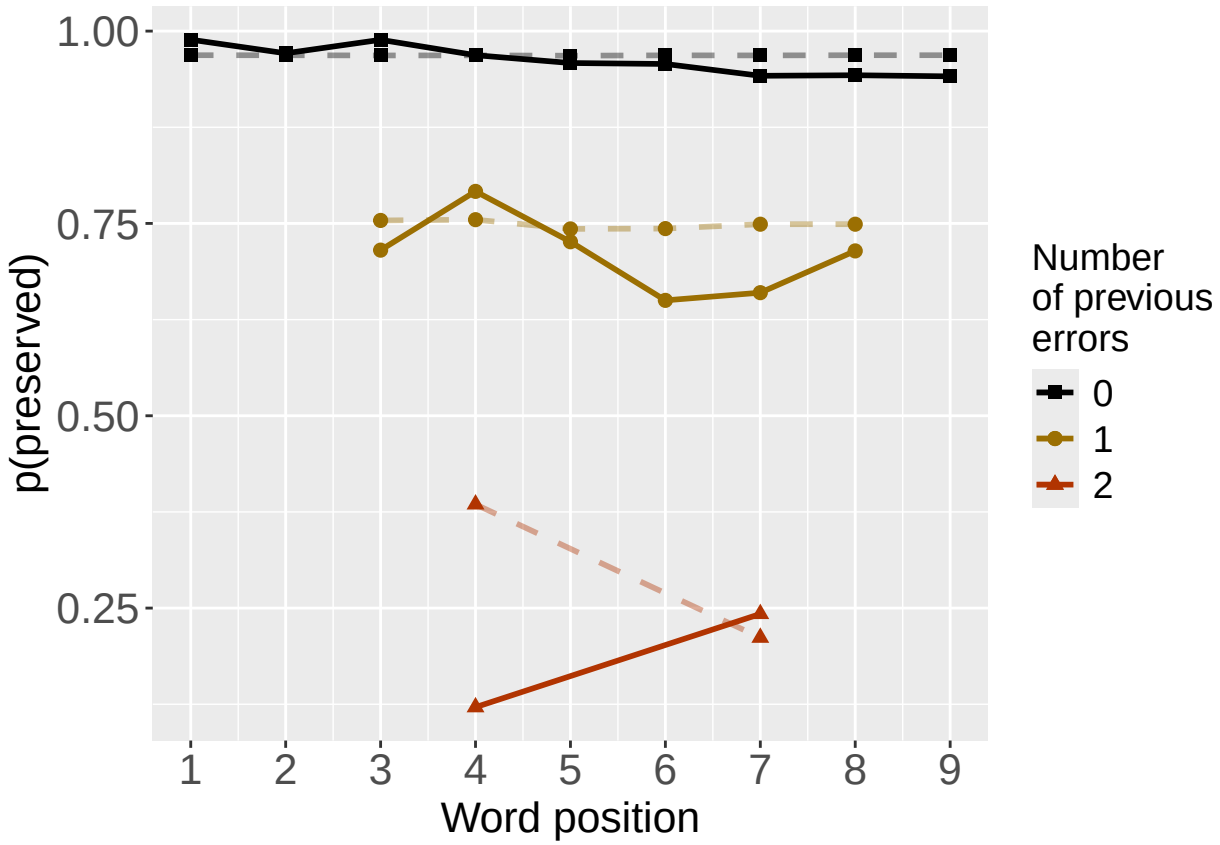
```
print(PrevCorPlot)
```




```
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
```

```
CumAICSummary <- NULL
```

```
########
# 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
##      4.6911      -2.1080       0.0262      -0.4411
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1194  AIC: 1296
## log likelihood:  -596.9931
## Nagelkerke R2:  0.3513305
## % pres/err predicted correctly:  -297.7671
## % of predictable range [ (model-null)/(1-null) ]:  0.3153617
```

```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)     I(pos^2)          pos
##     4.85878      0.01809     -0.55558
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1573  AIC: 1733
## log likelihood:  -786.6536
## Nagelkerke R2:  0.1012647
## % pres/err predicted correctly:  -417.3579
## % of predictable range [ (model-null)/(1-null) ]:  0.04131391
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(pos^2)	pos
preserved ~ CumErr + I(pos^2) + pos	1296.283	0.00000	1.0000000	0.999822	0.3513305	4.691056	-2.108033	0.0262045	-0.4411165

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	I(posˆ2)	pos
preserved ~ CumErr	1313.550	17.26669	0.0001781	0.000178	0.3351862	3.431129	-2.337890	NA	NA
preserved ~ I(posˆ2) + pos	1732.582	436.29845	0.0000000	0.000000	0.1012647	4.858780	NA	0.0180911	-0.5555836

```
########
# 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:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr      stimlen
##       4.292       -2.290        -0.112
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1215  AIC: 1312
## log likelihood:  -607.3588
## Nagelkerke R2:  0.3382678
## % pres/err predicted correctly:  -299.1262
## % of predictable range [ (model-null)/(1-null) ]:  0.3122474
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  3
```

```
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      stimlen
##      4.8262      -0.2591
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1680  AIC: 1838
## log likelihood:  -839.9217
## Nagelkerke R2:  0.02665592
## % pres/err predicted correctly:  -431.1588
## % of predictable range [ (model-null)/(1-null) ]:  0.009688705
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	stimlen
preserved ~ CumErr + stimlen	1311.605	0.000000	1.0000000	0.7255972	0.3382678	4.291905	-2.290258	-0.1119992
preserved ~ CumErr	1313.550	1.944796	0.3781751	0.2744028	0.3351862	3.431129	-2.337890	NA
preserved ~ stimlen	1837.733	526.127986	0.0000000	0.0000000	0.0266559	4.826209	NA	-0.2591030

```
########
# 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
##      4.0141      -2.3133      -0.1884
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
```

```
## Null Deviance:       1717
## Residual Deviance: 1196  AIC: 1297
## log likelihood:  -598.1762
## Nagelkerke R2:  0.3498431
## % pres/err predicted correctly:  -297.6706
## % of predictable range [ (model-null)/(1-null) ]:  0.315583
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)      CumPres
##      3.2039      -0.1426
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1697  AIC: 1858
## log likelihood:  -848.622
## Nagelkerke R2:  0.01428005
## % pres/err predicted correctly:  -433.4101
## % of predictable range [ (model-null)/(1-null) ]:  0.004529765
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	CumPres
preserved ~ CumErr + CumPres	1297.182	0.00000	1.0000000	0.999721	0.3498431	4.014062	-2.313281	-0.1883570
preserved ~ CumErr	1313.550	16.36778	0.0002791	0.000279	0.3351862	3.431129	-2.337890	NA
preserved ~ CumPres	1857.585	560.40287	0.0000000	0.000000	0.0142800	3.203861	NA	-0.1425876

```
########
# 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
##      4.2024      -2.1249      -0.1884
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1196  AIC: 1297
## log likelihood:  -598.1762
## Nagelkerke R2:  0.3498431
## % pres/err predicted correctly:  -297.6706
## % of predictable range [ (model-null)/(1-null) ]:  0.315583
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
```

```
## Coefficients:
## (Intercept)          pos
##      4.4709      -0.3713
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1575  AIC: 1733
## log likelihood:  -787.4638
## Nagelkerke R2:  0.1001447
## % pres/err predicted correctly:  -416.9457
## % of predictable range [ (model-null)/(1-null) ]:  0.04225867
## ***************************
```

Model	AIC	DeltaAIC	AICexp	AICwt	NagR2	(Intercept)	CumErr	pos
preserved ~ CumErr + pos	1297.182	0.00000	1.0000000	0.999721	0.3498431	4.202419	-2.124924	-0.1883570
preserved ~ CumErr	1313.550	16.36778	0.0002791	0.000279	0.3351862	3.431129	-2.337890	NA
preserved ~ pos	1732.798	435.61573	0.0000000	0.000000	0.1001447	4.470944	NA	-0.3713085

```r
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	1296.283	0.000000	1.0000000	0.999822	0.3513304	5.691056	-2.108033	0.0262045	-0.4411165	NA	NA
preserved ~ CumErr + CumPres	1297.182	0.000000	1.0000000	0.999721	0.3498431	4.014062	-2.313281	NA	NA	NA	-0.1883570
preserved ~ CumErr + pos	1297.182	0.000000	1.0000000	0.999721	0.3498431	4.202419	-2.124924	NA	-0.1883570	NA	NA
preserved ~ CumErr + stimlen	1311.606	0.000000	1.0000000	0.725597	0.2338267	4.291905	-2.290258	NA	NA	-0.1119992	NA
preserved ~ CumErr	1313.550	17.266690	0.0001781	0.000178	0.3351862	3.431129	-2.337890	NA	NA	NA	NA
preserved ~ CumErr	1313.550	0.944796	0.3781751	0.274402	0.3351862	3.431129	-2.337890	NA	NA	NA	NA
preserved ~ CumErr	1313.550	16.367780	0.0002791	0.000279	0.3351862	3.431129	-2.337890	NA	NA	NA	NA
preserved ~ CumErr	1313.550	16.367780	0.0002791	0.000279	0.3351862	3.431129	-2.337890	NA	NA	NA	NA
preserved ~ I(pos^2) + pos	1732.584	436.298451	0.0000000	0.000000	0.1012644	4.858780	NA	0.0180911	-0.5555836	NA	NA
preserved ~ pos	1732.798	435.615733	0.0000000	0.000000	0.1001447	4.470944	NA	NA	-0.3713085	NA	NA
preserved ~ stimlen	1837.735	526.127986	0.0000000	0.000000	0.0266554	3.826209	NA	NA	NA	-0.2591030	NA
preserved ~ CumPres	1857.585	560.402874	0.0000000	0.000000	0.0142803	3.203861	NA	NA	NA	NA	-0.1425876

```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
##     4.71480     -2.06355      0.02985     -0.45255      0.17592
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1178  AIC: 1281
## log likelihood:  -588.9676
## Nagelkerke R2:  0.3613978
## % pres/err predicted correctly:  -296.8795
## % of predictable range [ (model-null)/(1-null) ]:  0.3173957
## ***************************
## model index:  5
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
```

```
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos      stimlen     log_freq
##     4.50532     -2.06414      0.02848     -0.44926      0.02944      0.18033
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3991 Residual
## Null Deviance:       1717
## Residual Deviance: 1178  AIC: 1283
## log likelihood:  -588.8532
## Nagelkerke R2:  0.3615409
## % pres/err predicted correctly:  -296.778
## % of predictable range [ (model-null)/(1-null) ]:  0.3176283
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos
##      4.6911      -2.1080       0.0262      -0.4411
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1194  AIC: 1296
## log likelihood:  -596.9931
## Nagelkerke R2:  0.3513305
## % pres/err predicted correctly:  -297.7671
## % of predictable range [ (model-null)/(1-null) ]:  0.3153617
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     I(pos^2)          pos      stimlen
##     4.83591     -2.10691      0.02722     -0.44335     -0.02050
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1194  AIC: 1298
## log likelihood:  -596.9346
## Nagelkerke R2:  0.351404
## % pres/err predicted correctly:  -297.7992
## % of predictable range [ (model-null)/(1-null) ]:  0.3152882
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
```

```
## (Intercept)
##       2.782
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3996 Residual
## Null Deviance:       1717
## Residual Deviance: 1717  AIC: 1873
## log likelihood:  -858.614
## Nagelkerke R2:  6.357761e-16
## % pres/err predicted correctly:  -435.3868
## % 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	1281.331	0.000000	1.000000	0.7118104	0.3613974	8.714800	- 2.063551	0.0298464	- 0.4525520	0.1759178	NA
preserved ~ CumErr + I(pos^2) + pos + stimlen + log_freq	1283.144	1.812353	0.4040662	0.2876203	0.3615409	5.505321	- 2.064139	0.0284752	- 0.4492567	0.1803380	0.0294365
preserved ~ CumErr + I(pos^2) + pos	1296.284	14.952184	0.0005663	0.0004032	0.3513305	3.591056	- 2.108033	0.0262045	- 0.4411165	NA	NA
preserved ~ CumErr + I(pos^2) + pos + stimlen	1298.116	16.784572	0.0002269	0.0001615	0.3514040	3.835914	- 2.106910	0.0272243	- 0.4433513	NA	- 0.0204973
preserved ~ 1	1873.235	591.902700	0.0000000	0.0000000	0.0000000	2.782199	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
```

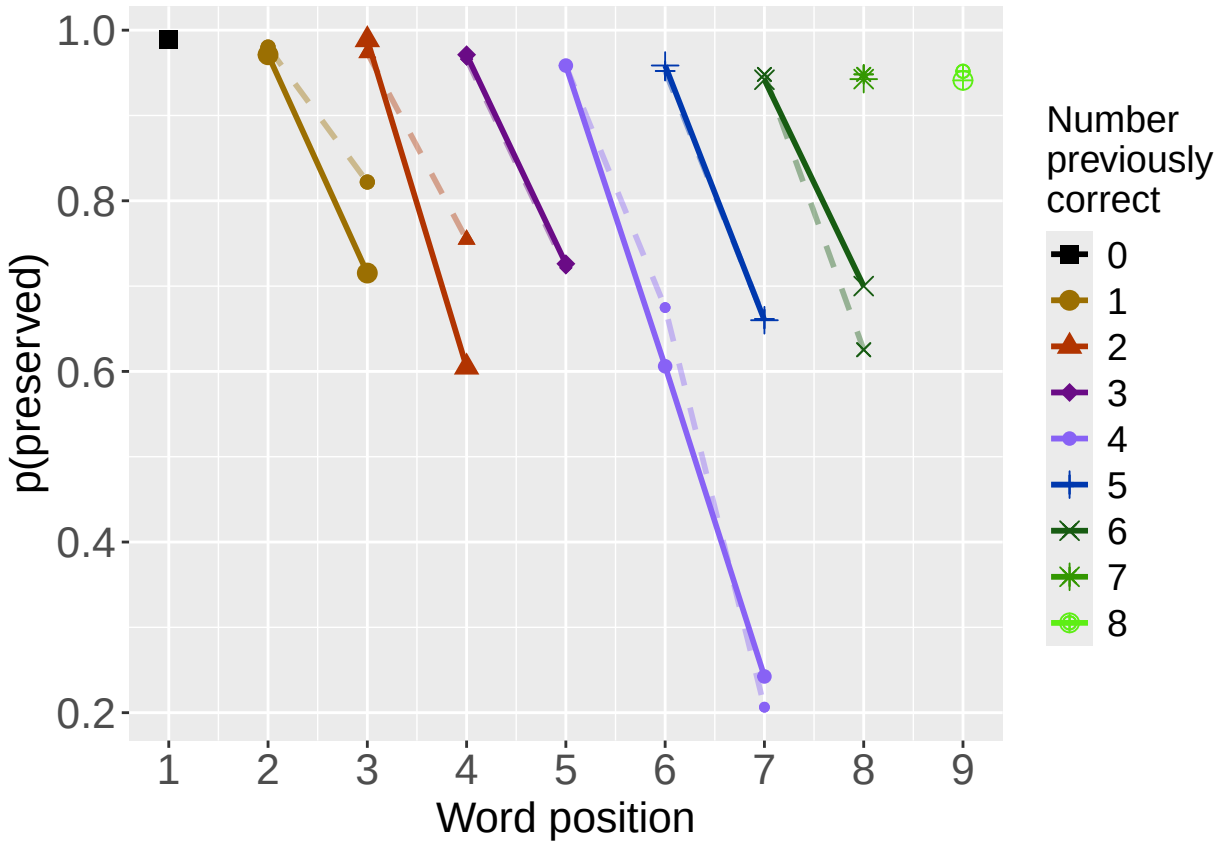
```
## Single term deletions
##
## Model:
## preserved ~ CumErr + I(pos^2) + pos + log_freq
##          Df Deviance    AIC
## CumErr    1   1548.4 1649.8
## log_freq  1   1194.0 1295.4
## pos       1   1185.2 1286.6
## I(pos^2)  1   1181.0 1282.3
## <none>        1177.9 1281.3
```

```
################################
# 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.
```

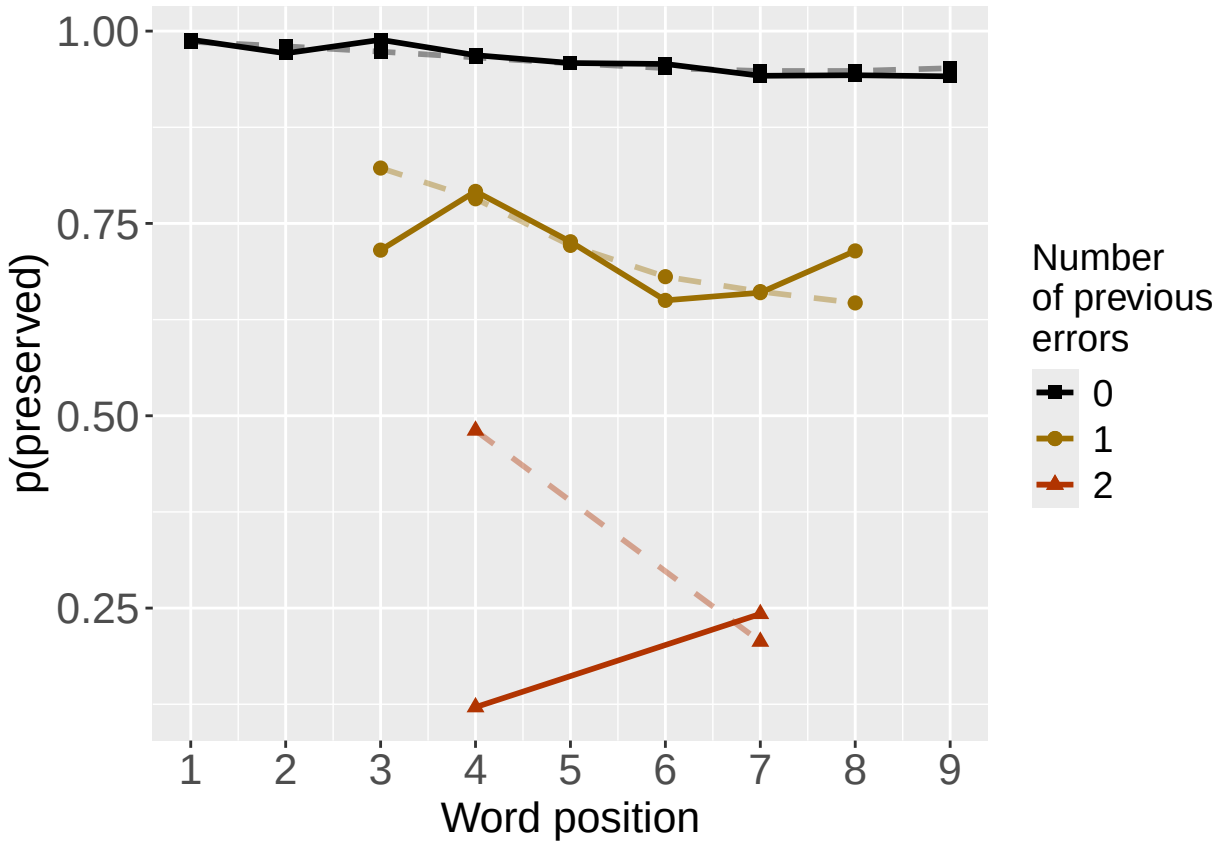
```
print(PrevCorPlot)
```




```
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!
```

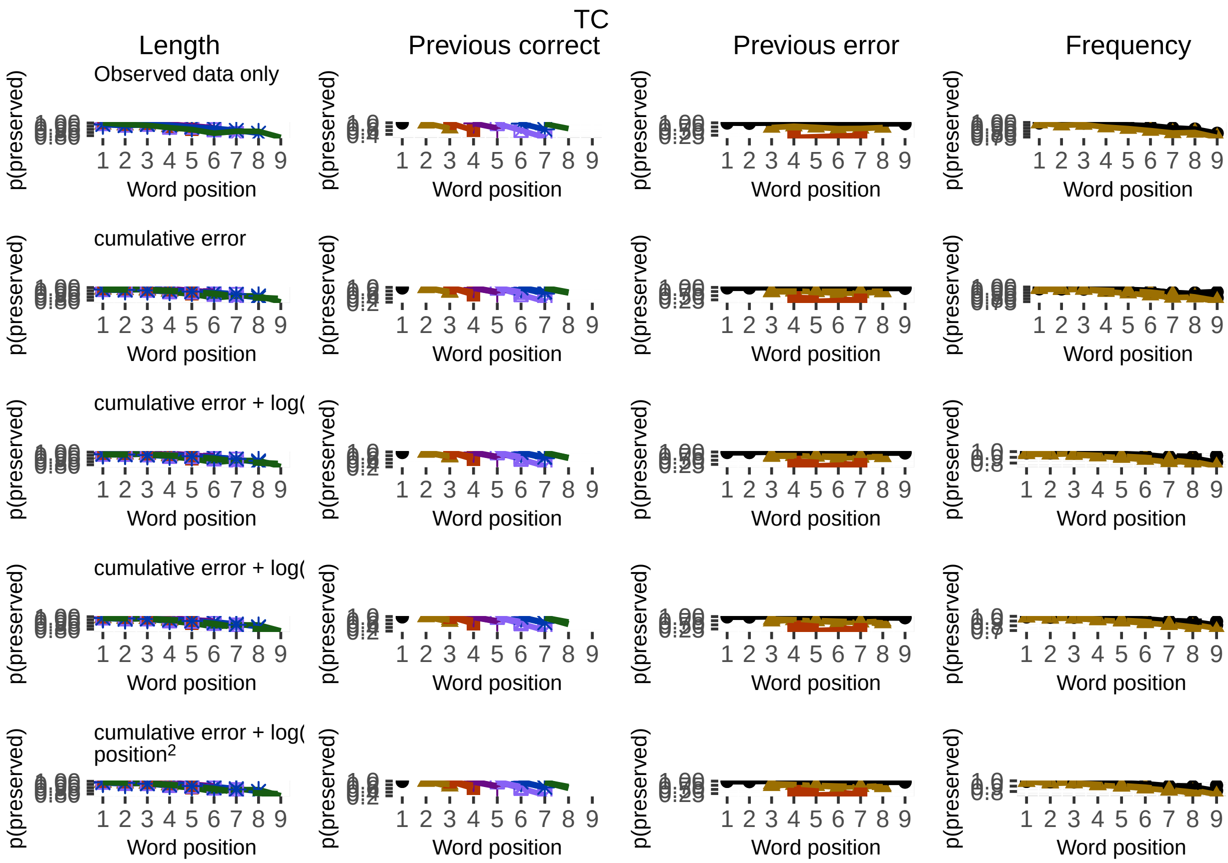
```
## ***************************
## model index:  1
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr
##       3.431       -2.338
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3995 Residual
## Null Deviance:       1717
## Residual Deviance: 1220  AIC: 1314
## log likelihood:  -609.7964
```

```
## Nagelkerke R2:  0.3351862
## % pres/err predicted correctly:  -299.3174
## % of predictable range [ (model-null)/(1-null) ]:  0.3118092
## ***************************
## model index:  2
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     log_freq
##      3.4965      -2.2666       0.1973
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3994 Residual
## Null Deviance:       1717
## Residual Deviance: 1199  AIC: 1294
## log likelihood:  -599.3434
## Nagelkerke R2:  0.3483746
## % pres/err predicted correctly:  -298.3978
## % of predictable range [ (model-null)/(1-null) ]:  0.3139164
## ***************************
## model index:  3
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     log_freq          pos
##      4.1650      -2.0821       0.1722      -0.1667
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3993 Residual
## Null Deviance:       1717
## Residual Deviance: 1181  AIC: 1283
## log likelihood:  -590.4732
## Nagelkerke R2:  0.3595122
## % pres/err predicted correctly:  -296.9093
## % of predictable range [ (model-null)/(1-null) ]:  0.3173274
## ***************************
## model index:  4
##
## Call:  glm(formula = as.formula(ModelEquations[Index]), family = "binomial",
##     data = PosDat)
##
## Coefficients:
## (Intercept)       CumErr     log_freq          pos     I(pos^2)
##     4.71480     -2.06355      0.17592     -0.45255      0.02985
##
## Degrees of Freedom: 3996 Total (i.e. Null);  3992 Residual
## Null Deviance:       1717
## Residual Deviance: 1178  AIC: 1281
## log likelihood:  -588.9676
## Nagelkerke R2:  0.3613978
## % pres/err predicted correctly:  -296.8795
## % of predictable range [ (model-null)/(1-null) ]:  0.3173957
```

```
## ***************************
## `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	0.2536415	0.0239748	0.0289486	0.0140271
SquaredCorrelation	0.1091220	0.0109308	0.0131246	0.0061905
Nagelkerke	0.1091220	0.0109308	0.0131246	0.0061905
Estrella	0.1326240	0.0116194	0.0141414	0.0071460

```
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 + log_freq + pos + I(pos^2) CumErr + log_freq + pos + I(pos^2) 1177.935
## CumErr + log_freq + pos                         CumErr + log_freq + pos 1180.946
## CumErr + log_freq                                     CumErr + log_freq 1198.687
## CumErr                                                           CumErr 1219.593
## null                                                               null 1717.228
##                                    deviance_explained percent_explained
## CumErr + log_freq + pos + I(pos^2)           539.2929          31.40485
## CumErr + log_freq + pos                      536.2816          31.22949
## CumErr + log_freq                            518.5411          30.19641
## CumErr                                       497.6352          28.97898
## null                                           0.0000           0.00000
##                                    percent_of_explained_deviance increment_in_explained
## CumErr + log_freq + pos + I(pos^2)                     100.00000              0.5583797
## CumErr + log_freq + pos                                 99.44162              3.2895794
## CumErr + log_freq                                       96.15204              3.8765489
## CumErr                                                  92.27549             92.2754920
## null                                                          NA              0.0000000
```

```
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 + log_freq + pos + I(pos^2)	1177.935	539.2929
CumErr + log_freq + pos	1180.946	536.2816
CumErr + log_freq	1198.687	518.5411
CumErr	1219.593	497.6352
null	1717.228	0.0000

	percent_explained	percent_of_explained_deviance	increment_in_explained
CumErr + log_freq + pos + I(pos^2)	31.40485	100.00000	0.5583797
CumErr + log_freq + pos	31.22949	99.44162	3.2895794
CumErr + log_freq	30.19641	96.15204	3.8765489
CumErr	28.97898	92.27549	92.2754920
null	0.00000	NA	0.0000000

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

```
##          Nagelkerke
## CumErr   0.78297778
## I(pos^2) 0.07843131
## pos      0.09417249
## log_freq 0.04441843
```

```
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	0.9656278	1219.593
preserved ~ CumErr+log_freq+pos	0.9695263	1180.946
preserved ~ CumErr+log_freq	0.9733065	1198.687
preserved ~ CumErr+log_freq+pos+I(pos^2)	0.9736846	1177.935

```
## 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       0.9656278       1219.593 0.000000000
## 2          preserved ~ CumErr+log_freq+pos       0.9695263       1180.946 0.003898458
## 3              preserved ~ CumErr+log_freq       0.9733065       1198.687 0.007678678
## 4 preserved ~ CumErr+log_freq+pos+I(pos^2)       0.9736846       1177.935 0.008056756
##   diff_CumErr+log_freq+pos diff_CumErr+log_freq diff_CumErr+log_freq+pos+I(pos^2)
## 1             -0.003898458        -0.0076786779                      -0.0080567560
## 2              0.000000000        -0.0037802195                      -0.0041582976
## 3              0.003780220         0.0000000000                      -0.0003780781
## 4              0.004158298         0.0003780781                       0.0000000000
```

```
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+log_freq+pos	diff_CumErr+log_freq
preserved ~ CumErr	0.0000000	-0.0038985	-0.0076787
preserved ~ CumErr+log_freq+pos	0.0038985	0.0000000	-0.0037802
preserved ~ CumErr+log_freq	0.0076787	0.0037802	0.0000000
preserved ~ CumErr+log_freq+pos+I(pos^2)	0.0080568	0.0041583	0.0003781