Perception example of EZBHDDM

Adriana F. Chávez De la Peña and Joachim Vandekerckhove

2024-01-16

Introduction

Data from Ratcliff and Rouder (1998), experiment 1, participant N.H.:

> '"…subjects were asked to decide **whether the overall brightness of pixel arrays displayed on a computer monitor was "high" or "low"** (Fig. 3a). The brightness of a display was controlled by the proportion of the pixels that were white. For each trial, the proportion of white pixels was chosen from one of two distributions, a high distribution or a low distribution, each with fixed mean and standard deviation (Fig. 3b). Feedback was given after each trial to tell the subject whether his or her decision had correctly indicated the distribution…"

There are 66 cells in the design with two main factors:

- **Accuracy vs Speed instructions**: Conditions 1-33 had an accuracy instruction, 34-66 a speed instruction.
- **More black vs More white pixels**: Conditions 1-16 and 34-49 had more black pixels; conditions 18-33 and 51-66 had more white, and conditions 17 and 50 were ambiguous and will not be used here because they can't provide accuracy measures.

```
# Load necessary libraries/packages
library(R2jags)
```

Loading and cleaning the data

Load the data

```
# Load the data from one participant
data_raw <- read.csv("./nh.tsv", sep = "")
colnames(data_raw) <- c("index","cond","response","RT")
head(data_raw)
```

```
##   index cond response  RT
## 1     1    1        1 457
## 2     1    1        1 569
## 3     1    1        1 390
## 4     1    1        1 534
## 5     1    1        1 501
## 6     1    1        1 488
```

Data set dimensions
```r
# Data set dimensions
dim(data_raw)
```

```
## [1] 7889    4
```

Clean the data

```r
# Create a copy of the raw data file
data <- data_raw

# Get 'accuracy' binary coding based on the condition and response in raw data file
accuracy <- as.integer(data_raw$cond > 0  & data_raw$cond < 17 & data_raw$response==1|
                       data_raw$cond > 17 & data_raw$cond < 34 & data_raw$response==2|
                       data_raw$cond > 33 & data_raw$cond < 50 & data_raw$response==1|
                       data_raw$cond > 50 & data_raw$cond < 67 & data_raw$response==2)

# Update the 'response' column of the data copied so it reflects accuracy
data$response <- accuracy

# Remove rows where RT > 3000ms
data <- data[which(data$RT<=3000),]
```

Get summary statistics

Write custom function `ez_summaries`

```r
# Define a function to compute the summary statistics used by EZ-DDM
ez_summaries <- function(subset){
  # Identify condition ID
  cond <- unique(subset$cond)
  # Return relevant summary statistics
  return(data.frame("nTrials" = nrow(subset),
                    "score" = sum(subset$response),
                    "meanRT"  = mean(subset$RT[which(subset$response==1)]/1000),
                    "varRT"   = var(subset$RT[which(subset$response==1)]/1000),
                  # Index variable: Accuracy (-0.5) vs Speed (0.5) condition
                    "Xi"  = as.integer(cond>33)-0.5,
                  # Arbitrary scale of stimulus configuration | 0 is 50/50 black and white
                    "Xs"  = ((cond-1) %% 33 - 16)/5))
}
```

Compute summary statistics from data

```r
# Initialize an empty output data frame (df)
tmp <- matrix(0,nrow = max(data$cond),ncol = 6)
df <- as.data.frame(tmp)
colnames(df) <- c("nTrials", "sum_accuracy", "mean_rt_correct",
```

```r
                   "variance_rt_correct", "Xi", "Xs")

# Populate the df output using the ez_summaries function
for(i in 1:max(data$cond)){
  df[i,] <- ez_summaries(data[which(data$cond==i),])
}

# Remove the two ambiguous conditions (17 and 50, with 50/50 black and white)
df <- df[-which(df$Xs==0),]
head(df,3)
```

```
##   nTrials sum_accuracy mean_rt_correct variance_rt_correct   Xi   Xs
## 1      30           30       0.4917333          0.03115648 -0.5 -3.2
## 2      24           24       0.4888750          0.03118168 -0.5 -3.0
## 3      36           36       0.4994722          0.02872146 -0.5 -2.8
```

The model

The model incorporates an effect (β) of instruction (i.e., x_i, `Xi`) on the *bound parameter (α)*.

$$\alpha \sim \text{Normal}(\mu_\alpha + \beta x_i, \sigma_\alpha)$$

We also include a nonlinear regression on the drift rate δ using instruction (i.e., x_i, `Xi`) and stimulus configuration (i.e., x_s, `Xs`) as predictors. For the latter, we used the absolute value (i.e., $|x_s|$, `abs(Xs)`) to represent the task difficulty getting easier as the black/white balance departs from 50%.

$$\begin{aligned} Y &= \Phi(\beta_1 + \beta_2|x_s| + \beta_3 x_i|x_s|) \\ \delta_{\text{pred}} &= \mu_\delta + \beta_0 Y + \beta_4 x_i \\ \delta &\sim \text{Normal}(\delta_{\text{pred}}, \sigma_\delta) \end{aligned}$$

In the present example, we focus on the regression parameters capturing the effects of instruction (β_3, `Beta3`, and β_4, `Beta4`).

Write the model in JAGS

EZ JAGS code for the model discussed above:

```r
model <- write("
model {
        ##### Priors for hierarchical DDM parameters
        betaweight ~ dnorm(0.00, 1.00)
        beta0 ~ dnorm(0.00, 1.00)
        beta1 ~ dnorm(0.00, 1.00)
        beta2 ~ dnorm(0.00, 1.00)
        beta3 ~ dnorm(0.00, 1.00)
        beta4 ~ dnorm(0.00, 1.00)
        bound_mean ~ dnorm(1.50, (0.20^-2))T( 0.10, 3.00)
```

```
        drift_mean ~ dnorm(0.50, (0.50^-2))
        nondt_mean ~ dnorm(0.30, (0.06^-2))T( 0, )
        bound_sdev ~ dunif(0.01, 1.00)
        drift_sdev ~ dunif(0.01, 3.00)
        nondt_sdev ~ dunif(0.01, 0.50)

        # Hierarchical distributions of individual DDM parameters.
        for (p in 1:length(meanRT)) {
            # Here, drift rate is the criterion.
            drift_pred[p] = beta0*phi(beta1 + beta2*abs(Xs[p])
                            + beta3*Xi[p]*abs(Xs[p])) + beta4 * Xi[p] + drift_mean
            drift[p] ~ dnorm(drift_pred[p], (drift_sdev^-2))
            bound_pred[p] = bound_mean + betaweight * Xi[p]
            bound[p] ~ dnorm(bound_pred[p],(bound_sdev^-2))T( 0.10, 3.00)
            nondt[p] ~ dnorm(nondt_mean, (nondt_sdev^-2))T( 0.05, )

            # Forward equations from EZ DDM
            ey[p]  = exp(-bound[p] * drift[p])
            Pc[p]  = 1 / (1 + ey[p])
            PRT[p] = 2 * pow(drift[p], 3) / bound[p] *
                     pow(ey[p] + 1, 2) / (2 * -bound[p] *
                     drift[p] * ey[p] - ey[p] * ey[p] + 1)
            MDT[p] = (bound[p] / (2 * drift[p])) * (1 - ey[p]) / (1 + ey[p])
            MRT[p] = MDT[p] + nondt[p]

            # Noiseless predictions from forward EZ DDM
            ey_pred[p]  = exp(-bound_pred[p] * drift_pred[p])
            Pc_pred[p]  = 1 / (1 + ey_pred[p])
            PRT_pred[p] = 2 * pow(drift_pred[p], 3) / bound_pred[p] *
                     pow(ey_pred[p] + 1, 2) / (2 * -bound_pred[p] *
                     drift_pred[p] * ey_pred[p] - ey_pred[p] * ey_pred[p] + 1)
            MDT_pred[p] = (bound_pred[p] / (2 * drift_pred[p])) *
                     (1 - ey_pred[p]) / (1 + ey_pred[p])
            MRT_pred[p] = MDT_pred[p] + nondt_mean

            # Sampling distributions for summary statistics
            correct[p] ~ dbin(Pc[p], nTrials[p])
            varRT[p]   ~ dnorm(1/PRT[p], 0.5*(correct[p]-1)
                                      * PRT[p] * PRT[p])
            meanRT[p]  ~ dnorm(MRT[p], PRT[p] * correct[p])
    }
}", "./model_perception.bug")
```

JAGS

Specify JAGS setup

```
# General setup
n.chains  <- 4
n.iter    <- 5000
n.burnin  <- 250
n.thin    <- 1

# Pass data to JAGS
data_toJAGS <- list("nTrials"  =  df$nTrials,
                    "meanRT"   =  df$mean_rt_correct,
                    "varRT"    =  df$variance_rt_correct,
                    "correct"  =  df$sum_accuracy,
                    "Xi"   =  df$Xi,
                    "Xs"   =  df$Xs)

# Prepare initial values
myinits <- rep(list(list()), n.chains)
for(i in 1:n.chains){
    myinits[[i]] <- list(drift = rnorm(length(data_toJAGS$nTrials),0,0.1))
}

# Specify parameters to keep track of
parameters <- c('beta3', 'beta4', 'drift', 'drift_pred',
                "Pc_pred", "MRT_pred", "PRT_pred")
```

Run JAGS

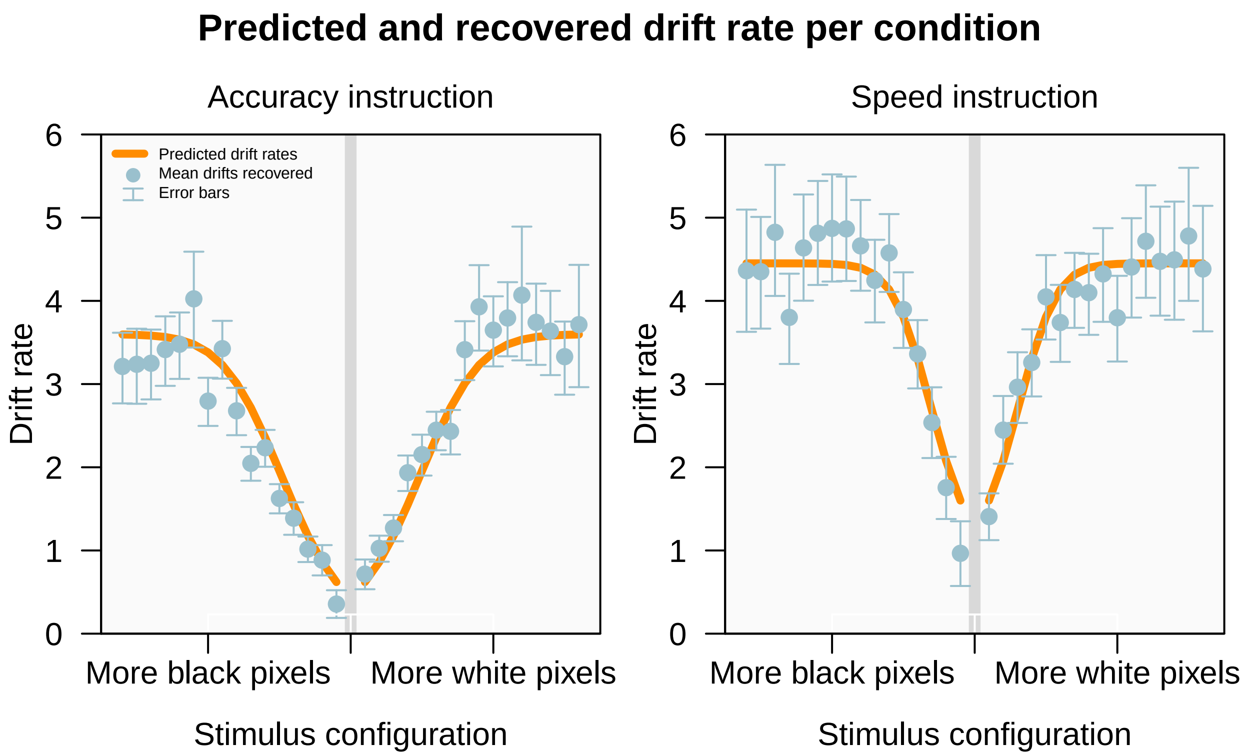
```
samples <- jags(data=data_toJAGS,
                parameters.to.save=parameters,
                model="./model_perception.bug",
                n.chains=n.chains,
                n.iter=n.iter,
                n.burnin=n.burnin,
                n.thin=n.thin,
                DIC=T,
                inits=myinits)
```

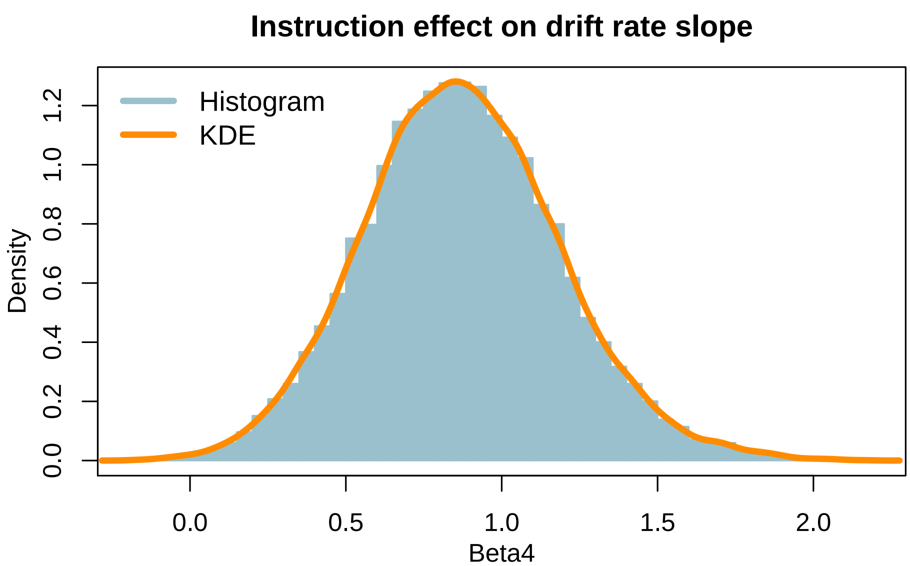
```
## module glm loaded

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 192
##    Unobserved stochastic nodes: 204
##    Total graph size: 3149
##
## Initializing model
```

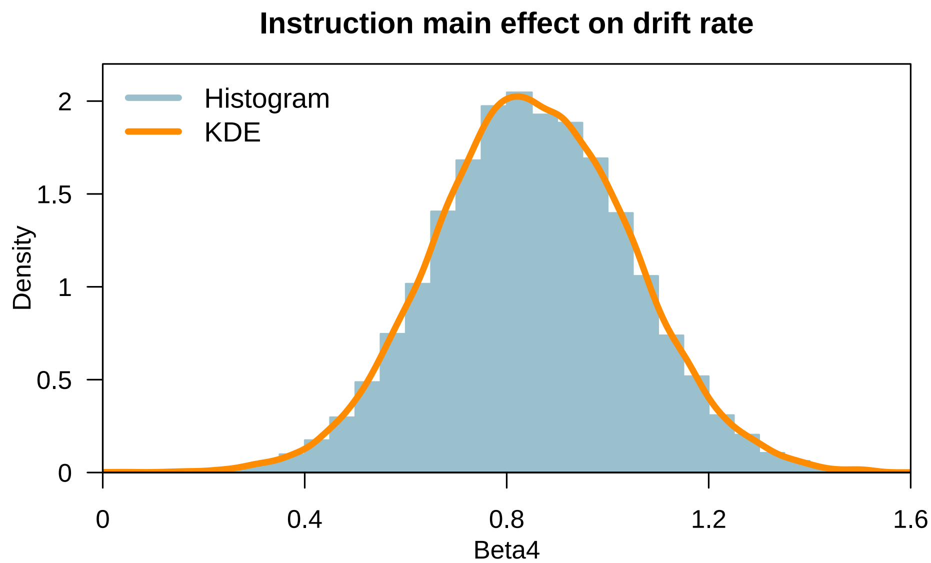
Extract samples

```
##### Drift rate parameters
# Recovered drift rates
drift <- samples$BUGSoutput$sims.list$drift
# Effects of instruction
beta3 <- as.vector(samples$BUGSoutput$sims.list$beta3) # Main
beta4 <- samples$BUGSoutput$sims.list$beta4 # Interaction
# Fitted values / Predicted drift rates
drift_pred <- samples$BUGSoutput$sims.list$drift_pred

##### Summary statistics sampled
accRate_hat   <- samples$BUGSoutput$sims.list$Pc_pred
rtCorMean_hat <- samples$BUGSoutput$sims.list$MRT_pred
rtCorVar_hat  <- 1/samples$BUGSoutput$sims.list$PRT_pred
```

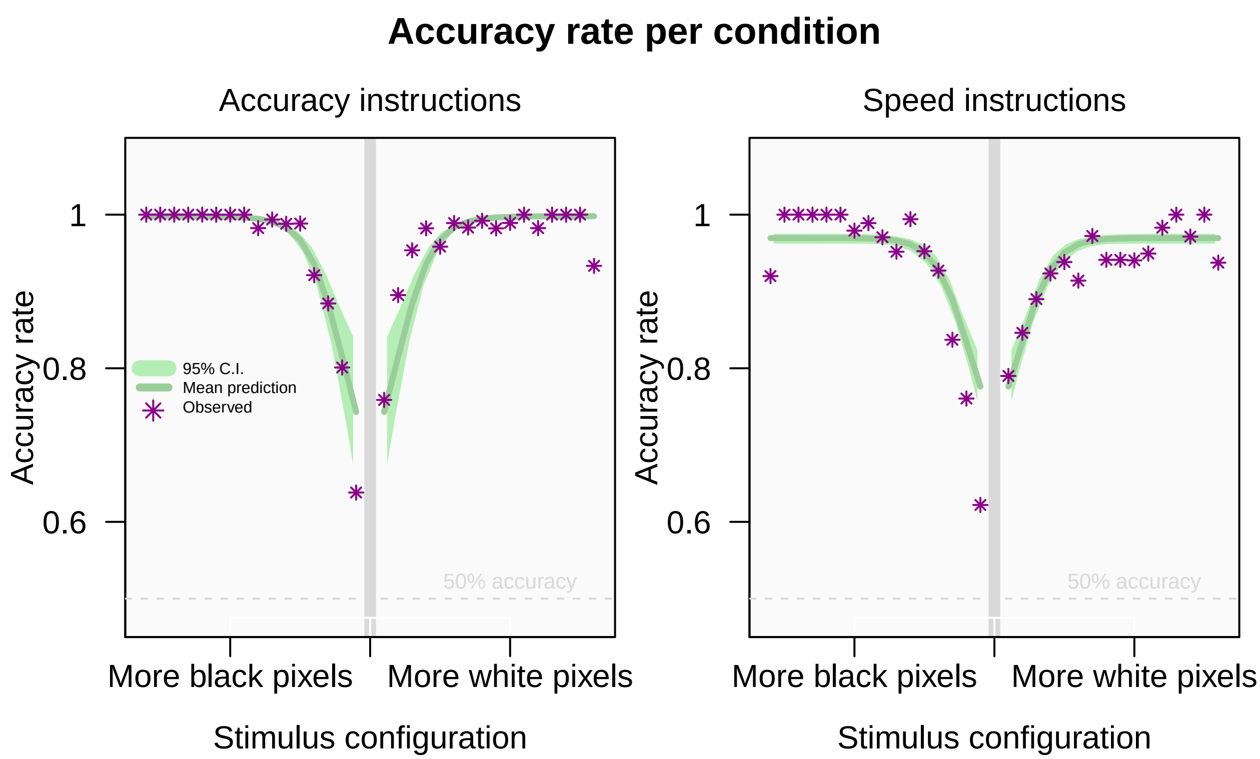
Results


Predicted and recovered drift rate per condition
Accuracy instruction
Speed instruction
Predicted drift rates
Mean drifts recovered
Error bars
Drift rate
0
1
2
3
4
5
6
More black pixels
More white pixels
Stimulus configuration

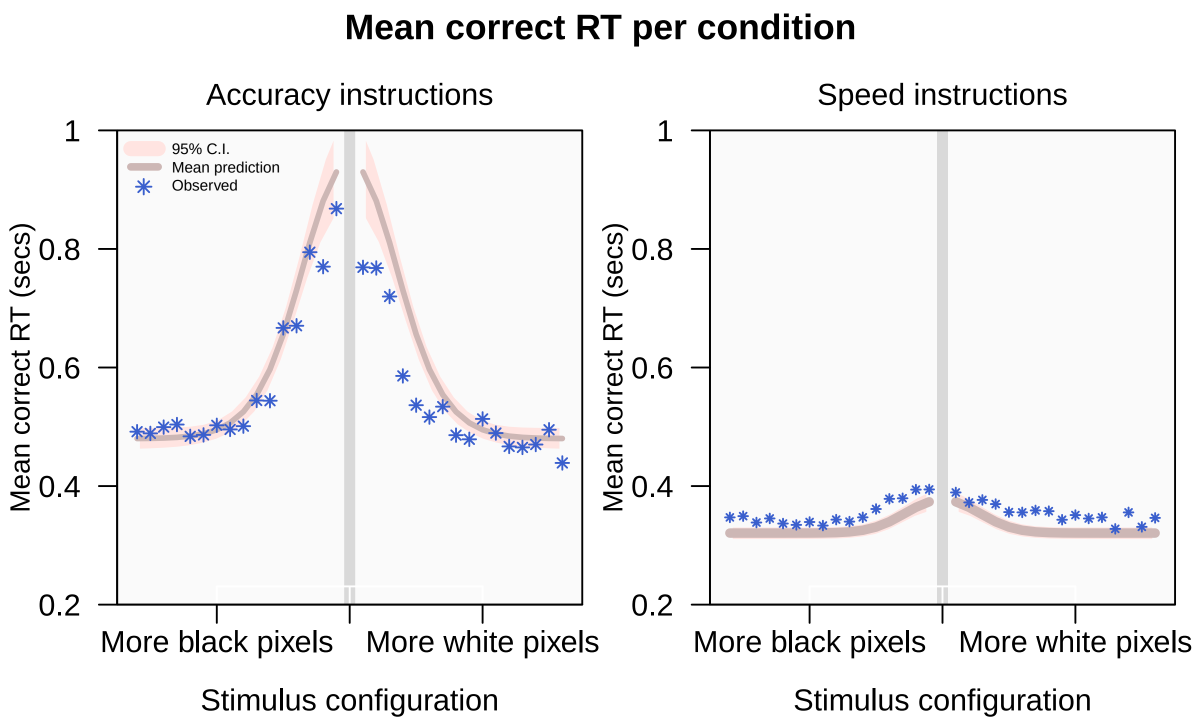

Instruction effect on drift rate slope
Histogram
KDE
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0.0
0.5
1.0
1.5
2.0
Beta4

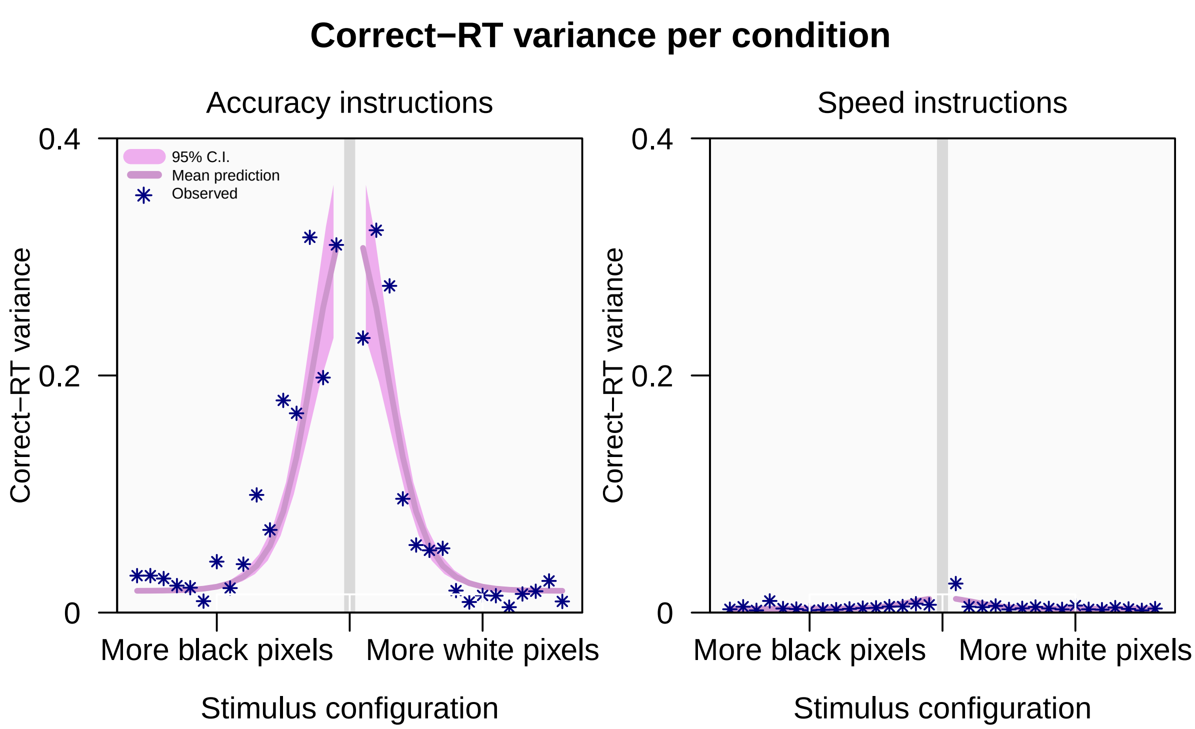


Instruction main effect on drift rate
Histogram
KDE
Density
Beta4
0
0.5
1
1.5
2
0.4
0.8
1.2
1.6



Accuracy rate per condition
Accuracy instructions
Speed instructions
Accuracy rate
1
0.8
0.6
95% C.I.
Mean prediction
Observed
50% accuracy
More black pixels
More white pixels
Stimulus configuration


Mean correct RT per condition
Accuracy instructions
Speed instructions
95% C.I.
Mean prediction
Observed
Mean correct RT (secs)
1
0.8
0.6
0.4
0.2
More black pixels
More white pixels
Stimulus configuration


Correct−RT variance per condition
Accuracy instructions
Speed instructions
95% C.I.
Mean prediction
Observed
Correct−RT variance
0.4
0.2
0
More black pixels
More white pixels
Stimulus configuration