EZ Bayesian Hierarchical Drift Diffusion Model

Based on Joachim's python code

Adriana F. Chavez

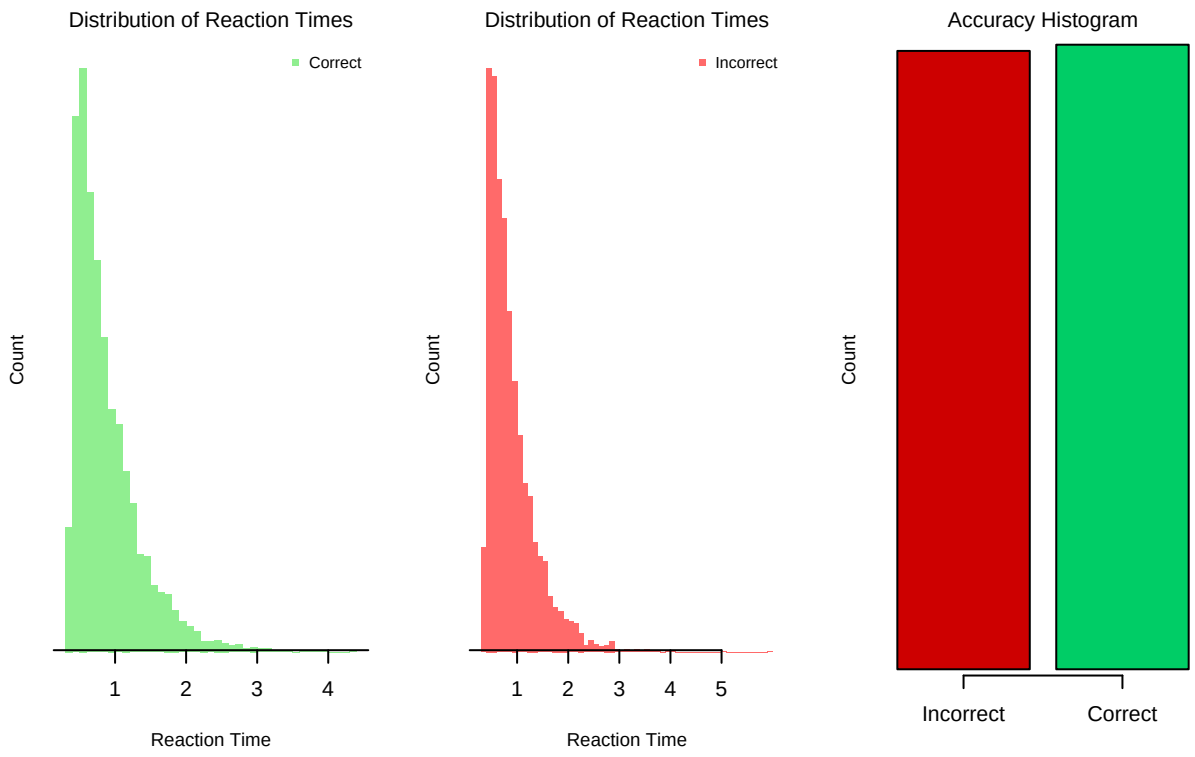
08 September, 2023

Set up functions

```
# Simulate DDM the dumb way

# Part 1: Simulate single trial outcome
simulate_ddm <- function(a, v, dt, max_steps){
     x <- 0
     random_dev <- rnorm(max_steps)
     # Scale step changes by dt
     noise <- random_dev * sqrt(dt)
     drift <- v * dt

     for(i in 2:max_steps){
        this_step = drift + noise[i]
        x = x + this_step
        if(abs(x)>=(a/2)){  break  }
     }
     output <- list("RT" = (i+1)*dt, "C"  = x)
     return(output)
}

# Part 2: Simulate over 'n' trials
wdmrnd <- function(a,v,t,n){
    dt = 0.001
    max_steps = 10 / dt
    rt = rep(NA,n)
    accuracy = rep(NA,n)

    for(i in 1:n){
        X <- simulate_ddm(a, v, dt, max_steps)
        rt[i] <- X$RT
        if(X$C>0){  accuracy[i] <- 1
          }else{    accuracy[i] <- 0  }
    }

    rt = rt + t
    output <- data.frame("RT" = rt, "Accuracy" = accuracy)
    return(output)
}
```

Example data

```
a = 1.50
v = 0.00
t = 0.30
n = 10000

data <- wdmrnd(a, v, t, n)
rt <- data$RT
accuracy <- data$Accuracy
```



Define Simulation Study environment and variables

```
sample_parameters <- function(settings){
  prior <- settings$prior
  nP <- settings$nP
  bound_mean = rnorm(1,prior$bound_mean_mean,prior$bound_mean_sdev)
  drift_mean = rnorm(1,prior$drift_mean_mean,prior$drift_mean_sdev)
  nondt_mean = rnorm(1,prior$nondt_mean_mean,prior$nondt_mean_sdev)
  bound_sdev = runif(1,prior$bound_sdev_lower,prior$bound_sdev_upper)
  drift_sdev = runif(1,prior$drift_sdev_lower,prior$drift_sdev_upper)
```

```
  nondt_sdev = runif(1,prior$nondt_sdev_lower,prior$nondt_sdev_upper)
  bound = rnorm(nP,bound_mean, bound_sdev)
  drift = rnorm(nP,drift_mean, drift_sdev)
  nondt = rnorm(nP,nondt_mean, nondt_sdev)

  parameter_set <- list("bound_mean" = bound_mean, "drift_mean" = drift_mean,
                        "nondt_mean" = nondt_mean, "bound_sdev" = bound_sdev,
                        "drift_sdev" = drift_sdev, "nondt_sdev" = nondt_sdev,
                        "bound" = bound,   "drift" = drift,   "nondt" = nondt)

  return(parameter_set)
}

sample_data <- function(settings, parameter_set){
      for(i in 1:settings$nP){
          data <- wdmrnd(a = parameter_set$bound[i],
                         v = parameter_set$drift[i],
                         t = parameter_set$nondt[i],
                         n = settings$nT[i])
      }
}
```

```
prior <- default_priors()
```

```
## ========== EZBHDDM Priors: ===============
## Bound Mean Mean:    1.5
## Bound Mean Std Dev: 0.2
## Drift Mean Mean:    0
## Drift Mean Std Dev: 0.5
## Non-decision Time Mean Mean: 0.3
## Non-decision Time Mean Std:  0.06
## Bound Std Dev Shape: 0.1
## Bound Std Dev Scale: 0.2
## Drift Std Dev Shape: 0.2
## Drift Std Dev Scale: 0.4
## Non-decision Time Shape: 0.01
## Non-decision Time Scale: 0.05
```

Run simulations

Simple example

```
prior = Hddm_Prior()
np.random.seed(seed = 188) # This doesn't work
design = Hddm_Design(participants=20, trials=50, prior=prior)
design.sample_parameters()
design.sample_data()
design.estimate_parameters()
```

```
nSim <- 200
settings <- list("nPart"   = 50,
                 "nTrials" = 150,
                 "prior"   = prior)
prior <- default_priors()
```

```
tru = [Hddm_Parameter_Set()] * K
est = [Hddm_Parameter_Set()] * K
err = [Hddm_Parameter_Set()] * K

for(k in 1:nSim){
    set.seed(k)
    cat("Iteration", k+1, "of", nSim)
    design = Hddm_Design(participants=20, trials=50, prior=prior)
    design.sample_parameters()
    design.sample_data()
    #print(design.parameter_set)
    #design.data.summary()
    design.estimate_parameters()
    tru[k] = design.parameter_set
    est[k] = design.estimate
    if design.estimate is not None:
        err[k] = (design.estimate - design.parameter_set)
    else:
        err[k] = None
    if (k+1) % 100 == 0:
        print(f'. {k+1} of {K}\n', end='')
    else:
        print('.', end='')
}
```