Simulate and fit your own drift-duffison models (DDMs)!

Hierarchical DDMs in brms

We will again use the PDM data set, and now fit it properly with a simple drift-diffusion model! You can load it using the following code in `R`:

```
library(curl)
```

```
# See https://github.com/mdnunez/encodingN200 for more information about the data
pdmdat <- curl("https://tinyurl.com/PDMdataESCOP2022")
pdm <- read.csv(pdmdat)
```

```
colnames(pdm) <- c('N200_latencies', 'N200_amplitudes',
    'RT', 'accuracy', 'condition', 'EEG_session',
    'experiment', 'session', 'subject')

pdm <- pdm[pdm$experiment == 1, ]

pdm$N200_latencies <- pdm$N200_latencies/1000

pdm$RT <- pdm$RT/1000

head(pdm)
```

1. (5min) A random walk process is a model in which the next step of a process comes from a distribution which is then added to the result of the previous step.

Let us assume that a participant collects evidence in one experimental trial for the low or high spatial frequency target that follows a random walk. The random walk is *simulated* by the following R code:

```
set.seed(1)
nwalks = 50; nsteps = 100; step_length = .01
time = seq(0, 1, length.out=nsteps)
random_walks = matrix(0, nrow=nsteps, ncol=nwalks)
for(w in 1:nwalks)
{
    this_random_walk = 0.5
    for(s in 2:nsteps)
    {
        evidence_units = rnorm(1, 0.01, 0.1)
        this_random_walk[s] = this_random_walk[s-1] + evidence_units
    }
    random_walks[,w] = this_random_walk
}
```

```
matplot(time, random_walks,type='l')
```

1a. Change the standard deviation of the random walk to 0.001, what happens? How would you describe this graph?

1b. Change the standard deviation of the random walk to 1, what happens? What has changed compared to the original figure?

##Below is the code to fit the data using EZ Diffusion

```
ezdiff = function(rt, correct, s=1.0)
{
    if (length(rt) <= 0) stop('length(rt) <= 0')
    if (length(rt) != length(correct)) stop('RT and correct unequal lengths')
    if (max(correct, na.rm=TRUE) > 1) stop('Correct values larger than 1')
    if (min(correct, na.rm=TRUE) < 0) stop('Correct values smaller than 0')
    pc = mean(correct, na.rm=TRUE)
    if (pc <= 0) stop('Mean accuracy less than or equal to 0')
    # subtract or add 1/2 an error to prevent division by zero
    if (pc == 1.0) {pc=1 - 1/(2*length(correct))}
    if (pc == 0.5) {pc=0.5 + 1/(2*length(correct))}
    MRT = mean(rt[correct == 1], na.rm=TRUE)
    VRT = var(rt[correct == 1], na.rm=TRUE)
    if (VRT <= 0) stop('RT variance less than or equal to 0')
    r=(qlogis(pc)*(((pc^2)*qlogis(pc)) - pc*qlogis(pc) + pc - 0.5))/VRT
    drift=sign(pc-0.5)*s*(r)^0.25
    boundary=(s^2 * qlogis(pc))/drift
    y=(-1*drift*boundary)/(s^2)
    MDT=(boundary/(2*drift))*((1-exp(y))/(1+exp(y)))
    ndt=MRT-MDT
    return(list(boundary, drift, ndt))
}
est_params = ezdiff(pdm$RT, pdm$accuracy)

# The estimate of the boundary in evidence units
est_params[1]

# The estimate of the drift rate in evidence units per second
est_params[2]

# The estimate of the non-decision time in seconds
est_params[3]
```

##Below is the code to fit the data to a Bayesian DDM brms. Note there we are ignoring the effect of condition.

```
#Define the formula for brms
ddm_formula <- bf(RT | dec(accuracy) ~ 1,
                  bs ~ 1,
                  ndt ~ 1,
                  bias = 0.5)
```

```r
# Remove contaminant RTs
pdm_reduced <- pdm[pdm$RT > 0.2, ]

# Define the initial values for each Markov chain.
# Note that brms and Stan do not have good default start values for DDMs.
# Also note that we enforce the starting value of NDT to be less than the minimum RT
mcmc_initials <- function() {
  list(
    Intercept = runif(1, -4, 4),
    Intercept_bs = runif(1, 0.5, 2),
    Intercept_ndt = runif(1, 0, min(pdm$RT))
  )
}

bayes_ddm <- brm(ddm_formula,
                 family=wiener(link_bs ="identity",link_ndt="identity"),
                 init=mcmc_initials, data=pdm_reduced)
summary(bayes_ddm)
plot(bayes_ddm)
```

2. Fit this model and compare the output to `ezdiff()`. Note that the model fitting will take a long time. Do you draw the same conclusions? What do you learn from the Bayesian DDM that you do not learn from the EZ Diffusion?