Bayesian Cognitive Modeling: Drift-Diffusion Models

Michael D. Nunez

29-August-2022

Bayesian Modeling in brms

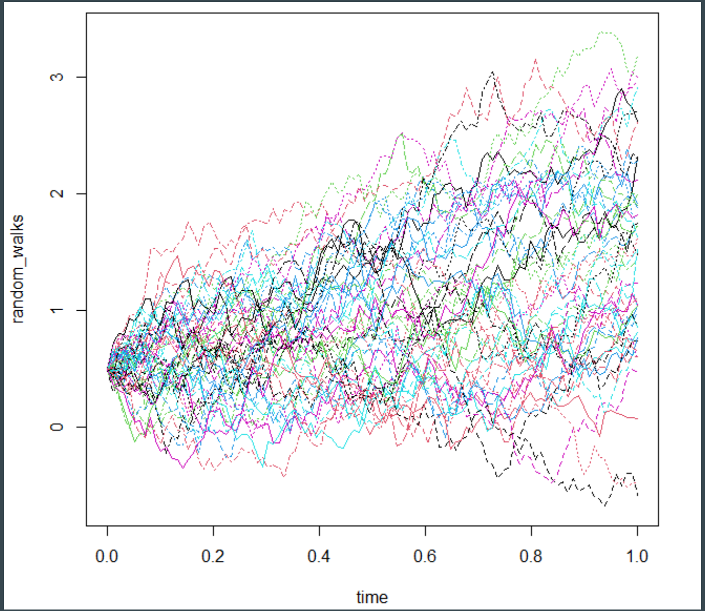
Random Walks

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.

```
set.seed(1)
nwalks = 50; nsteps = 100; step_length = .01; standard_deviation = 0.1
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, standard_deviation)
        this_random_walk[s] = this_random_walk[s-1] + evidence_units
    }
    random_walks[,w] = this_random_walk
}

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



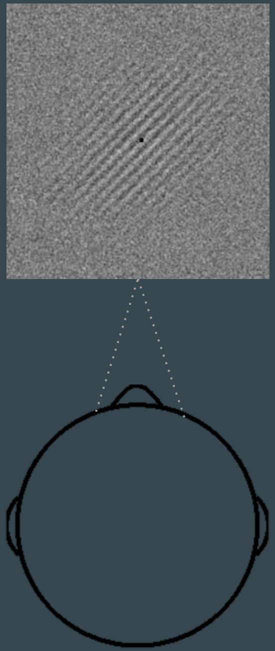
Do it yourself: Random Walks

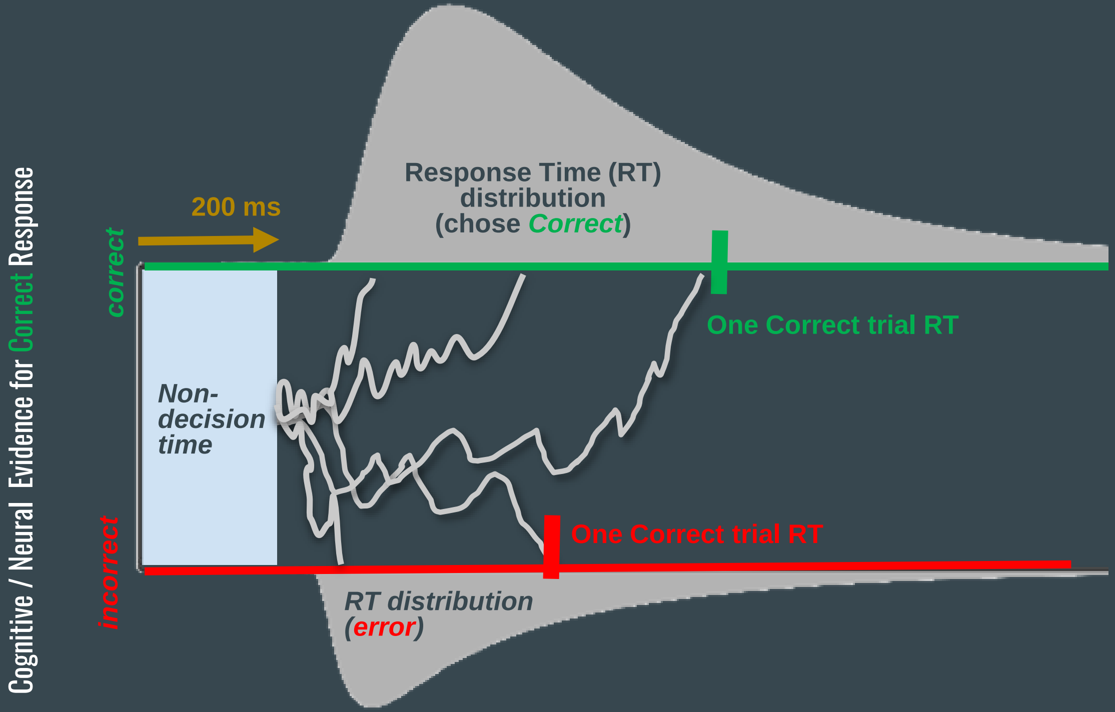
- Change the standard deviation of the random walk to 0.001, what happens? How would you describe this graph?
- Change the standard deviation of the random walk to 1, what happens? What has changed compared to the original figure?

Answers

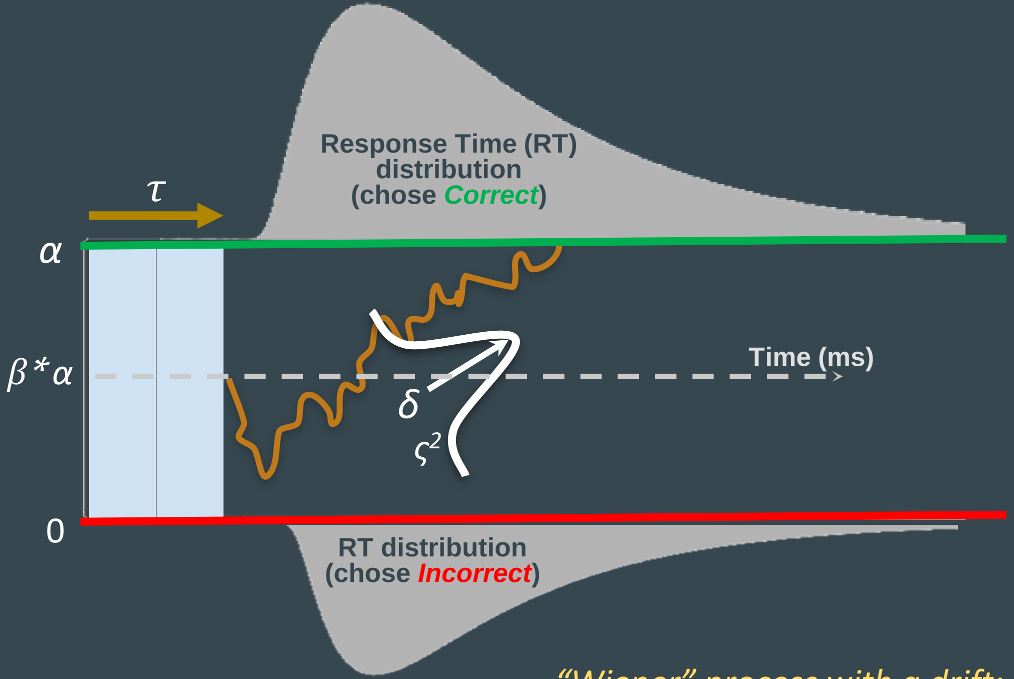
- Answer: All random walks trend upwards in almost a linear equation. These result in ballistic accumulation
- Answer: The scale of the y-axis is now about 10 times larger than the original graph

Simulating one accumulator with a random walk…





Simulating Drift-Diffusion Models (DDMs)


Response Time (RT) distribution (chose Correct)
τ
α
$\beta * \alpha$
Time (ms)
δ
ς^2
0
RT distribution (chose Incorrect)

"Wiener" process with a drift:
$dE(t) \sim N(\delta t, \varsigma^2 t)$

- α : boundary (speed-accuracy tradeoff)
- δ: drift rate (ability parameter)
- τ: non-decision time
- β: relative start point
  - Fixed at 0.5 in our simulation
- ς : Diffusion coefficient
  - Fixed at 1 in our simulation.
  - Almost all model *fitting* procedures assume this is fixed

Simulating a Drift Diffusion Model (DDM)

```
ntrials = 200; nsteps = 300; step_length = .01
time = seq(0, nsteps*step_length, length.out=nsteps)
boundary = 1.2; drift = 1.5; ndt = 0.2; dc = 1
random_walks = matrix(0, nrow=nsteps, ncol=ntrials)
rts = rep(0, ntrials); correct = rep(1, ntrials)
set.seed(1)
for(n in 1:ntrials)
{
    random_walks[1, n] = 0.5*boundary  # relative start point = 0.5
    for(s in 2:nsteps)
    {
        random_walks[s, n] = random_walks[s-1, n] + rnorm(1, drift*step_length, dc*sqrt(step_length))
        if(random_walks[s, n] >= boundary)
        {
            random_walks[-(1:(s-1)), n] = boundary
            rts[n] = s*step_length + ndt
            correct[n] = 1
            break
        } else if (random_walks[s, n] <= 0) {
            random_walks[-(1:(s-1)), n] = 0
            rts[n] = s*step_length + ndt
            correct[n] = 0
            break
        } else if (s == nsteps) {
            correct[n] = NA; rts[n] = NA
            break
        }
    }
}
x11(); matplot(time, random_walks,type='l')
x11(); hist((correct * 2 - 1) * rts, breaks=20)
simulate_data = data.frame(RT = rts, accuracy = correct)
```

Fitting DDMs to data

- To *fit* a cognitive model is to discover a set of parameter estimates that best describe the *data* given the model architecture

- Fitting models to data is a useful method to test hypotheses by either:

    - (1) by directly estimating and then evaluating parameters, like comparing **ability parameter** estimates across participants

    - (2) comparing multiple models' ability to describe data (e.g. a DDM vs. Linear Ballistic Accumulator)

Fitting DDMs to data using EZ Diffusion

See:
[Wagenmakers, van der Maas, Grasman (2007)]

```
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(simulate_data$RT, simulate_data$accuracy);  est_params[1]; est_params[2]; est_params[3]
```

Fitting DDMs to data using brms

For more information see this blog post by Henrik Singmann: http://singmann.org/wiener-model-analysis-with-brms-part-i/

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

our_prior <- c(set_prior("normal(0, 2)", class = "Intercept"),
set_prior("normal(1, 0.25)", class = "Intercept", dpar = "bs"),
set_prior("exponential(4)", class = "Intercept", dpar = "ndt"))

# Define the initial values for each Markov chain, that Stan does not have good default start values for DDMs.
mcmc_initials <- function() {
  list(Intercept = runif(1, -4, 4), Intercept_bs = runif(1, 0.5, 2), Intercept_ndt = runif(1, 0,
        min(simulate_data$RT)) ) }

bayes_ddm <- brm(ddm_formula,
                 family=wiener(link_bs ="identity",link_ndt="identity"),
                 init=mcmc_initials, prior=our_prior, data=simulate_data)
summary(bayes_ddm)
x11(); plot(bayes_ddm)
```

Do it yourself: Fitting DDMs

- Fit this cognitive model to the simulated data using both EZ Diffusion and a Bayesian analysis with brms.

- Do both fitted models recover the true simulated cognitive parameters?

- What do you learn from the Bayesian DDM that you do not learn from the EZ Diffusion?

Do it yourself: Comparing DDM *fitting* procedures

- Fit the EZ Diffusion and brms DDM to one participant from the real data set in previous sessions.

- Note any differences between the two procedures.

- Which procedure is more trustworthy?

Warning!

Stan (and thus brms) may underestimate non-decision time in real data