For this homework assignment, I want you to use and implement some of the code necessary to fit the diffusion model to observed data.

homework9.m opens with code to simulate the diffusion model using these parameters:

\[
\begin{align*}
a &= 0.100; & \text{upper boundary (lower boundary is 0)} \\
z &= 0.050; & \text{starting point} \\
\text{sz} &= 0.01; & \text{variability in starting point from trial to trial} \\
\text{eta} &= 0.050; & \text{variability in drift rate from trial to trial} \\
\text{Ter} &= 0.200; & \text{non-decisional time} \\
\text{st} &= 0.05; & \text{variability in TR from trial to trial} \\
\text{s2} &= .01; & \text{diffusion coefficient is the amount of within-trial noise} \\
\text{drifts} &= [.1 .2 .3]; 
\end{align*}
\]

This simulates three “conditions” where each has its own drift rate.

**QUESTION 1**

Fit the EZ Diffusion Model to the simulated data (see the week9.m code from class).

Plot the resulting Ter, a, and drift as a function of the condition.

Does the EZ Diffusion adequately capture the parameters that vary in the simulations?

**QUESTION 2**

homework9.m has the code we looked at in class that fits the diffusion model to observed data (here it’s the simulated data we’re using as observed data for this exercise). This code uses a version of the diffusion model that uses a simulation.

I would like you to adapt this code to instead use the CDFDif.m routine. This means adapting the mymodel_diff.m routine to use CDFDif.m instead of the simulation.

This will mean eliminating the part of the code that does the simulation and replacing the part that figures out the predicted CDF from the simulation with code that instead calculates the predicted CDF using CDFDif.m.

Recall that CDFDif.m basically implements the complicated equations behind the diffusion model. So instead of simulating 500 or 5000 times per condition, you only need to make one call to CDFDif.m per percentile of the RT distribution.
The calling convention for CDFDif is
\[ y = \text{CDFDif}(t, x, \text{Par}) \]
t is the time to evaluate the CDF
x=0 returns the CDF for hitting the 0 (lower) boundary (correct)
x=1 returns the CDF for hitting the a (upper) boundary (error)
Par is an array:
\[ \text{Par} = [ a \ \text{Ter} \ \text{eta} \ z \ sZ \ st \ nu ] \]
- a = boundary separation
- Ter = mean non-decisional component time
- eta = standard deviation of normal drift distribution
- z = mean starting point
- sZ = spread of starting point distribution
- st = spread of non-decisional component time distribution
- nu = mean drift rate

Recall that CDFDif returns what is known as a degenerate CDF. Whereas a regular CDF goes between 0 and 1, the degenerate CDF goes between 0 and the probability of hitting that boundary (i.e., \( P(\text{Correct}) \) or \( P(\text{Error}) \) if that’s what the boundaries correspond to).

So \( p_a = \text{CDFDif}(999,1,\text{Par}) \) would return the probability of hitting upper boundary a.

So \( \text{CDFDif}(t,1,\text{Par})/p_a \) will give the probability that the RT is less than t, \( P(\text{RT}<t \mid \text{par}) \). In other words, this would give the predicted CDF for correct responses (boundary a).

Similarity, \( p_0 = \text{CDFDif}(999,0,\text{Par}) \) will give the probability of hitting lower boundary 0.

\( \text{CDFDif}(t,0,\text{Par})/p_0 \) will give the probability that the RT is less than t, \( P(\text{RT}<t \mid \text{par}) \). In other words, this would give the predicted CDF for error responses (boundary 0).

Try running this from the same starting point you used with the diffusion simulation.