**The Stop Signal Task**

**The task.**

* Participants are told that a circle will appear on the screen, after which an arrow will appear in the circle. They should press the button under their left and right index fingers corresponding to the direction the arrow points (L arrow = <, R arrow = >). They should be as quick and as accurate as they can. When they hear a beep, they should try to stop. They are told that they can’t stop all the time, and that going and stopping are equally important.
* The arrow is up for about 1000ms. The average RT is about 600ms (aka, the Go process), and the stop signal (the beep) delay adjusts so that each participant can stop correctly about 50% of the time (say that’s at about 400ms). Therefore, the leftover time (600ms – 400ms = 200ms) is the stop signal reaction time (SSRT), or the time your Stop process takes.

**Running the task.**

* Each subject needs a ladder file, which should be created before the study begins. These ladder files are fine to reuse from other studies, since there are only a limited number of unique ladders.
  + Ladder files are in: StopSignal/experiment/ladderFiles. You should have enough, but if you need more you can just copy and rename them by subject number, starting at the beginning.
  + Each subject has 5 runs
* To run the task:
  + Make sure Psychtoolbox is installed. (Can do this by typing “PsychtoolboxVersion” into the command line; if it doesn’t return a version, you don’t have it installed.)
  + Make sure the sound is on.
  + Open Matlab.
  + Add /StopSignal to path with subfolders.
  + Navigate to /StopSignal/experiment.
  + Check to make sure their fingers are on the right buttons and that they are responding with their L and R index fingers. Cleaning up the data for incorrect button use is possible, but work.
  + Type: SSRT
  + Enter in subject number and session number
  + Type 1 if you are scanning, 0 if not
  + Instructions on a gray background should appear, along with a test beep. Ask the participant if they can hear the beep.
  + Behaviorally, press any key to start. In the MRI, it’ll wait for a trigger.
    - To change the trigger and left/right keys, change the values called out in SSRT.m lines 35-37.

**Analyzing the data.**

* Check the data files: all subject data files will be .mat files with a variable called “Seeker” in them.
  + Column 3 = trial type (2 = ITI, 0/1 = go/stop)
  + Column 4 = direction of arrow (0 = L, 1 = R)
  + Column 7 = button pressed
* Once the data files are checked, get the data ready to analyze
  + Open prep4analysis.m
  + Change parameters to get data ready for the following scripts (this includes the directory, the study prefix and the subject info)
  + Run this script to create analysis ready files:
    - It will rename the files to a consistent convention: studycode\_sub#\_run#\_SSRT.mat (e.g., “REV\_sub3\_r2\_SSRT.mat”)
* Now you can run the script that checks the data quality
  + Open initialCheck.m
    - Change the parameters for this data set (location of the data in the /analysisReady folder, the buttons used, the study prefix, and the subject numbers to exclude)
  + Output: this script creates a bunch of .txt files located in /compiledResults/upToSubNum/initialCheck
    - Check that all runs have 96 gos and 32 stops (sanity check)
    - Look at the number of NRs (no responses on Go trials).
      * Note that late responses are also considered NRs. Thus, for any run where there are lots of NRs, there’s a good chance they were just responding LATE, and may have done the same on stop trials as well.
      * Lots of late responses on STOP trials will make the SSRT crazy – in fact, it will be too small or negative, and meaningless. That run may need to be dropped.
    - Weird button file: look to see if there are bunch of trials per run that used weird buttons (used different buttons than they should have)
      * Use this info to investigate what buttons they DID use on each problem run, and set up systematicWrongButtons.txt accordingly.
    - Wrong go - look to see if there are a bunch of trials per run with wrong Go (if a run has a high number of wrong go but not weird button, then they most likely switched L/R fingers)
* Analyses: first, keep a clean raw copy, and work in “output/analysisReady"
  + Clean up weird button presses
    - Open extractAllSSTResults.m
    - Make sure systematicWrongButtons.txt has been created
    - Run extractAllSSTResults.m
    - Results will appear in two folders: singleVarTxts and varMats
  + output/compiledResults/uptoSUBNUM/singleVarTxts
    - Lots of .txt files containing information about the sample
    - PctInhib.txt = percent correct file - check to make sure that subjects and runs all had an acceptable number of % inhibitions
      * if <40%, question the SSRT
      * if <25%, drop the run
    - SSRTint.txt = integration file - takes into account the actual number of % inhibitions —> USE THIS ONE
      * notes: adults usually start at about 200-300ms stop time
      * if super high, usually matches with a low % inhib
      * if too small, usually they were too slow or stopped too much
    - SSRT.txt = regular file, when it tries to make everyone at 50% (not as accurate as the SSRTint file)
    - GRTmean.txt = average go RT —> check to see if it’s too long for runs where the SSRT is super short
* Imaging (onsets/durations)
  + Modeling:
    - makeVecs outputs the following columns:
      * CorrectGo, CorrectStop, FailedStop, Cue, Trash (sometimes)
    - model stop & go trials
      * correct go (button), correct stop (no button press), failed stop (button), failed go (trash)
      * contrasts:
        + correct stop > correct go
        + correct stop > failed stop
    - need to know which trials are stop and which are go
    - model: cue period (circle onset before arrow) for all trials, correct stop
  + Making multiple conditions files:
    - makeVecs: carries over button rules from previous scripts
      * correct go, correct stop, failed stop: all have a duration = 1
      * makes multiple condition file: names, onsets, durations in fx folder in subject directory
    - Check onsets: each should be - a lot (correct Gos - ideal is 96), 16, 16, 128