Week 5. Experiments with PsyScript

Topics in Linguistics

What we’re doing
- The driver who my neighbor who I trust suggested took me to the airport.
- The driver who my neighbor who my boss trusts suggested took me to the airport.
- Overarching hypothesis: Sentence difficulty has to do with holding onto several unsatisfied dependencies. Longer ones are harder to hold.
- Question: What measures length?
- Hypothesis: New referents.

How do we see if that’s right?
- Center-embedded sentences are the most taxing, several started dependencies, center-most element triple-counted.
- The driver who my neighbor who I trust …
- That’s the most sensitive point, seems to be near critical point of processability.

Experimenting
- Does it matter whether we have a known referent (I, you) or a new referent (my neighbor)?
- To know for sure, we try holding everything constant except the most embedded subject and see if there are differences (which can then be attributed to the only thing that’s different, the properties of the most embedded subject).

Building the items
- The driver who my neighbor who I trust suggested took me to the airport.
- The driver who my neighbor who John trusts suggested took me to the airport.
- The driver who my neighbor who the housekeeper trusts suggested took me to the airport.
- The driver who my neighbor who they trust suggested took me to the airport.

Planning the experiment
- Each set of four sentences constitutes a token set (a.k.a. item)
- Each item are four conditions (1/2 pronoun, name, definite description, 3 pronoun).
- Counterbalancing rules:
  - Each subject will judge no more than one sentence from each token set.
  - Each subject will judge all conditions and will see equal numbers of sentences from each condition.
  - Every sentence in every token set will be judged by some subject.
Trial lists

- We have four conditions, so we need:
  - Four different “scripts” (versions of the lists)
  - Some number of fourples of token sets.
- E.g., items 1-4, each with conds a-d
  - Subj W: 1a, 2b, 3c, 4d  (script 1)
  - Subj X: 1b, 2c, 3d, 4a  (script 2)
  - Subj Y: 1c, 2d, 3a, 4b  (script 3)
  - Subj Z: 1d, 2a, 3b, 4c  (script 4)

Our experiment

- We will have 20 items (picked from the ones you submitted) and 20 fillers.
  - (Note: That’s on the small side for a real experiment)
- Next steps:
  - Create the lists of test sentences for the four different scripts.
  - Spec out and pseudocode our experiment
  - Investigate PsyScript
  - Run the experiment
  - Deal with the data

Creating the scripts

- Our sentences are made of very predictable components:
  - The X who/that the Y who/that Z VP1 VP2 VP3
  - The only thing that changes across conditions is Z, while the rest changes across token sets.
  - We can use Excel to build these from their pieces, to avoid unnecessary errors.

Worksheets

- Components
  - Subj
  - Rel1
  - Subj2
  - Rel2
  - Subj3a
  - Subj3b
  - Subj3c
  - VP3
  - VP2
  - VP1
  - Answer
  - Question

- Fillers
  - Question
  - Answer
  - Regions…

- The way I’ve set it up, everything needs to be exactly 8 regions long (even the fillers)

Worksheets

- Constructed
  - Computes item (token group) and condition based on row number, comes up with a code like 15V2 (fifth token group, version 2). Builds the sentence region by region based on the condition number.

- Tables
  - Keeps track of what will be on each script.
  - Scripts are divided into “blocks”, and each block has one of each condition and four fillers, randomized.
  - Sort column is 2*block plus a random number (to order the blocks, but randomly within)

- Script
  - The master script sheet
  - This generates a script based on the columns you put into I1 and J1. (The column refer to the tables sheet, where the item and condition numbers will be found)
    - B and C for script 1
    - D and E for script 2
    - F and G for script 3
    - H and I for script 4

- Script a, …, script d
  - Actual scripts.
  - Select the part of script sheet that has data (A1:O41) and copy.
  - Go to script a sheet
  - Paste special… and choose Value (so we don’t copy formulas, only results).
  - Delete column B-D (item, cond, row), select rows 2-41, hit sort button, delete column A (sort), and row 1 (labels)
  - Save as tab-delimited text.
The scripts are ready

- So, we have the data that we’re going to use.
- The next thing is to figure out how we’re going to test these.
- The goal is to test reading time on each region of the sentence by presenting the sentence region by region.

Thinking through the experiment

- What do we want to have happen?
  - Display some instructions
  - Do some practice trials
  - Display “practice is over” message
  - Do some real trials
  - Display “thanks!”
- The trials:
  - Show fully obscured sentence, wait for a key
  - Reveal next word, wait for a key, until done
  - Ask question, wait for response
  - Give sound feedback about correctness

PsyScript

- To do this, we’ll use PsyScript, an environment for creating psychology experiments on the Mac.

- (It’s basically the only freely available software of this type that has promise for working in the future – if PsyScope had not become commercial as E-Prime, we’d be learning that instead).

AppleScript

- The underlying machinery behind PsyScript is something called AppleScript.
  - This has been part of the Mac OS for about the past 10 years, although it is gaining power and popularity recently.
  - AppleScript is a means by which you can tell other programs what to do.
    - For example, tell Internet Explorer to go to a particular web page, tell Word to create a new document and type the date, …
    - Until you have an actual need for this, it doesn’t seem very exciting…

AppleScript

- AppleScript is a sophisticated high-level programming language designed to be human readable (and kind of human writable). It’s supposed to look a lot like English.

- PsyScript itself is an application that can be bossed around by AppleScript, and has the features that are useful in psycholinguistic experiments, such as timing, drawing, input, data recording functions.

Getting started

- To write (and use) AppleScript, we use Script Editor.

- Easiest way to do this: Find the end experiment script and double-click on it.
  
  ```
  tell application "PsyScript"
  end experiment
  end tell
  ```
Note about PsyScript

- PsyScript runs faster from the Script Editor
- If you run PsyScript from the Script Editor you have to manually tell it where your script is.
  - To do this, find the line that says `tell fileHelper to setContainer` and change the thing in parentheses to what you see when you Command-click on the name of the script in the title bar of the Script Editor Window, bottom to top, each separated by : and not including the actual name of the script. E.g.,
  - `setContainer("Station 5:Desktop Folder:PsyScript:`")

Movingwindow

- I wrote a script called `movingwindow` to do what we’re going to do today.
  - The stimuli and instructions files are in a folder called “resources” in the same folder as the script is. The names of these files are set at the top of the script, in mine, they are:
    - `Mwstimuli.txt`: sentence list as exported from Excel (tab-delimited text, exporting e.g., `script a`)
    - `Mwpractice.txt`: sentence list for the practice items
    - `Mwinstruc.txt`: initial instructions
    - `Mwready.txt`: post-practice instructions
    - `Mwthanks.txt`: end of experiment debriefing.
  - Results are stored in “results” folder.

Sentence lists

- To generate the sentence lists in the right format for `movingwindow`, go to one of the script a-d pages, do Save As… from Excel, and choose tab-delimited text.
  - Columns should be code, question, answer, sentence (in eight columns)
- The end results will come out in a file that you can load back into Excel (a tab-delimited file):
  - Columns are: code, region number, time for region, correct answer 1/0, text of region