A-maze of Natural Stories: Texts are comprehensible during the Maze task

Veronica Boyce[1] & Roger Levy[2]

[1] Stanford University
[2] Massachusetts Institute of Technology

Author Note

TODO

The authors made the following contributions. Veronica Boyce: Conceptualization, Formal Analysis, Investigation, Software, Writing - Original Draft Preparation, Writing - Review & Editing; Roger Levy: Conceptualization, Formal Analysis, Writing - Review & Editing.

Correspondence concerning this article should be addressed to Veronica Boyce, TODO. E-mail: vboyce@stanford.edu

Abstract

TODO Easy to use, reliable methods are important to science. In studying how people understand language in real time, we use incremental processing methods to measure word-by-word reading time. However, neither of the common methods (eye-tracking and self-paced reading) can run over the web and produce localized effects. Another method, called the Maze task, seems to be able to do this, especially since a technique called A-maze makes the task much faster to construct. Due to task limitations, Maze has not been used on long, naturalistic passages, only short targeted materials. Here we present an adaptation of the Maze method suitable for long materials; we test this method on the Natural Stories corpus and find that participants can comprehend what they read while doing the task and that the reading time patterns are broadly similar to those found with other methods. We find support for the localization of reading time effects during the Maze task, as well as extending the range of materials Maze is suitable for.

Keywords: TODO
Word count: X

A-maze of Natural Stories: Texts are comprehensible during the Maze task

Intro

We process language incrementally, building a model of the sentence as we hear or read word. When a word is unexpected and does not fit the model of the sentence so far, our understanding of the sentence may need to be revised. Incremental processing methods measure how long each word takes to read as a proxy for how hard it is to incoroporate into the mental model of the sentence. Fluent language uses process language rapidly and flexibly adapt to many unexpected words, which makes it hard to pinpoint slow downs related to difficult-to-process words.

TODO motivate why we care about localization

The two most commonly used methods are eye-tracking and self-paced reading. Both these methods are known to have spillover effects, which makes localization harder. CITE some things TODO might be worth bringing up RT/surprisal/spillover findings hereOur next question is what the relationship between a word's predictability and it's reading time is for Maze. It's well established for eye-tracking and SPR that RTs are roughly linear in terms of a word's surprisal (negative log probability). Due to spillover effects, on SPR and eye-tracking, there is also a positive linear relationship between the surprisal of a previous word and the RT on the current word – this is an indication of lack of localization. In addition to suprisal predicting RT, word length and word's overall frequency are also often found to be predictive. TODO many CITES

An alternative method that seems to have superior localization is the Maze task, which adopts an unnatural way of reading to force incremental processing (Forster, Guerrera, & Elliot, 2009). In the Maze task, participants see two words at a time, a correct word that continues the sentence, and a distractor which does not. Participants must choose the correct word, and their reaction time (RT) is the dependent measure. If participants make a mistake, the sentence discontinues. Theoretically, participants must fully integrate each word into the sentence in order to confidently select it. This idea is supported by studies finding strongly localized effects (Witzel, Witzel, & Forster, 2012).

The downside of Maze is that materials are effort-intensive to construct because of the need to select infelicitious words as distractors for each spot of each sentence; this may explain why the Maze task was not widely adopted. Boyce, Futrell, and Levy (2020) demonstrate a way to automatically generate Maze distractors by using NLP language models to find words that are high-surprisal in the context of the target sentence. The quality of these A-Maze distractors is not up to that of hand-generated distractors, but Boyce et al. (2020) found that materials with A-maze distractors had similar results to the hand-generated distractors from Witzel et al. (2012). Sloggett, Handel, and Rysling (n.d.) also found that A-maze and G-maze distractors yielded similar results on a disambiguation paradigm.

While new, A-maze seems like a potentially powerful addition to the psycholinguists toolkit. However, like the other Maze tasks it has been limited in its application to single-sentence items probing minimal comparisons in constructed sentences. While some

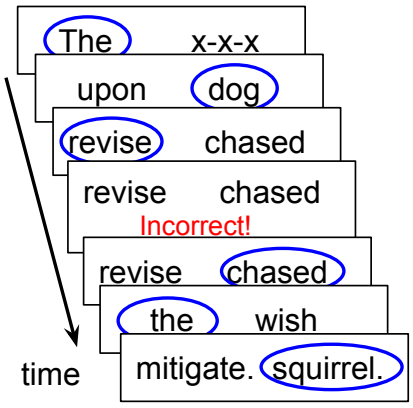

Figure 1. Schematic of error-correction Maze. A participant reads a sentence word by word, choosing the correct word at each time point (selections marked in blue). When they make a mistake, an error message is displayed, they try again, and continue with the sentence.

psycholinguistic phenomena are well-explored within individual targeted sentences, there's also a need for reading time data on longer passages, for instance in order to compare to language models or study discourse effects. Therefore, it's important to test whether the Maze task works on these types of materials.

While the issue of needing to generate distractors for long passages is solved with A-maze, another problem with Maze remains. In particular, because Maze tasks discontinue after participants make mistakes, the farther into an item a word is, the fewer participants see it. This makes it hard to run long materials using Maze, and prevents Maze from being used on long text passages where SPR or eye-tracking could be used.

We address this problem by creating a Maze task interface where participants correct their mistakes and continue with the sentence rather than terminating on mistakes. This allows for multi-sentence items to be run using Maze, which we verify by running Maze on the Natural Stories corpus. With this data, we are able to confirm that participants can comprehend what they read using Maze and investigate the effects of surprisal on RT in Maze.

Error-correction maze

One advantage of the Maze task is that it forces incremental processing and automatically excludes inattentive participants by terminating a sentence when a participant makes an error. Thus, only participants who have processed the sentence and are paying attention contribute data at critical regions later in the sentence. However, this means we don't have data after a participant makes a mistake in an item. In traditional G-maze tasks, with hand-crafted distractors and attentive participants, this is a small issue. However, this data loss is much worse with A-maze materials and crowd-sourced participants (Boyce et al., 2020). The high errors are likely from some combination of participants guessing randomly and from auto-generated distractors that in fact fit the

sentence; as Boyce et al. (2020) noted, some distractors, especially early in the sentence, were problematic and caused considerable data loss.

We could try to improve the situation by auto-generating better distractors or hand-replacing problematic ones. However, the inherent data losses poses a more fundamental problem: even with attentive participants and good distractors, the Maze paradigm will eventually run into problems as error rates compound over long materials where few participants will make it to the end. For instance, with a 1% error rate, 86% would complete each 15-word sentence, but only 61% of participants would complete a 50 word vignette, and 13% a 200 word passages. In order to run longer materials, we need something to do when participants make a mistake, other than terminate the entire item.

To resolve this, we introduce an *error-correction* variant of Maze, where we let participants continue on when they make a mistake. We present them with an error message and wait until they select the correct option, before continuing the sentence as normal. This *error-correction* variant is shown in Figure 1. We make this "error-correction" Maze available as an option in a modification of the Ibex Maze implementation introduced in Boyce et al. (2020) (TODO repo link). The code records both the RT to the first click and also the total RT until the correct answer is selected as separate values.

Our goal is that this error-correction will minimize the problems of having occasional poor auto-generated distractors by letting participants continue on and still read the whole item. It will also enable us to run long materials on Maze, thus expanding the range of questions that Maze can be used on.

Natural Stories corpus

We test this error-correction Maze on the Natural Stories corpus. The Natural Stories corpus (Futrell et al., 2020) consists of 10 passages each roughly 1000 words long which are designed to read fluently to native speakers. At the same time, the passages contain copious punctuation (including quoted speech), many proper nouns, and low frequency grammatical constructions. Taken together, these properties make this a severe test of our process. The materials are on the difficult side for our participants and for the language model we use to generate distractors. If participants can succeed at the Maze task on this set of materials, we think they are likely to succeed on basically any naturalistic text.

The corpus also comes with binary-choice comprehension questions, 6 per story, which we use to assess comprehension. Using this corpus on the A-maze task allows us to address, first whether participants will read and understand long passages using A-maze with error correction, and second, whether the resulting RTs profiles will show expected patterns such as a linear relationship with surprisal.

Methods

We constructed A-maze distractors for the Natural stories corpus (Futrell et al., 2020) and recruited 100 MTurkers to each read a story in the Maze paradigm.

Materials

We took the texts of the Natural Stories corpus (Futrell et al., 2020) and split it into sentences because the A-Maze process works on the sentence level. We used the original comprehension questions provided in the Natural Stories corpus. To familiarize participants with the task, we wrote a short practice passage and corresponding comprehension questions. We then ran the sentences of the items through the A-maze generation process. We updated the code of the generation process to fix some issues identified in that paper; the code used is available at REPO. The underlying method did not change, but features such as appropriately handling more types of punctuation (needed for the Natural Stories corpus) were added. We ran the materials using FOOBAR parameters. We used the auto-generated distractors as they were, without checking them for quality. Materials are available at REPO. TODO other notable changes to the codebase??

Participants

We recruited 100 participants from Amazon Mechanical Turk, and paid each participant 3.50 dollars. We excluded data from those who did not report English as their native language, leaving 95 participants.

Procedure

Participants first gave their informed consent and saw task instructions. Then they read a short practice story in the Maze paradigm and answered 2 binary-choice practice comprehension questions, before reading the main story in the A-maze task. After the story, they answered the 6 main comprehension questions, commented on their experience, answered optional demographic questions, saw an debriefing and were given a code to enter into Mturk for payment. The experiment was implemented in Ibex (CITE) and the experimental code is availabe at REPO.

Data analysis

Analyses were done in R (CITE), using the following CURATE packages. We used R (Version 4.0.3; R Core Team, 2020) and the R-packages *brms* (Version 2.14.4; Bürkner, 2017, 2018), *citr* (Version 0.3.2; Aust, 2019), *cowplot* (Version 1.1.1; Wilke, 2020), *dplyr* (Version 1.0.2; Wickham et al., 2020), *forcats* (Version 0.5.0; Wickham, 2020a), *ggplot2* (Version 3.3.3; Wickham, 2016), *gridExtra* (Version 2.3; Auguie, 2017), *here* (Version 1.0.1; Müller, 2020), *lme4* (Version 1.1.26; Bates, Mächler, Bolker, & Walker, 2015), *Matrix* (Version 1.3.2; Bates & Maechler, 2021), *mgcv* (Version 1.8.33; Wood, 2011, 2003, 2004; Wood, N., Pya, & S"afken, 2016), *nlme* (Version 3.1.151; Pinheiro, Bates, DebRoy, Sarkar, & R Core Team, 2020), *papaja* (Version 0.1.0.9997; Aust & Barth, 2020), *patchwork* (Version 1.1.1; Pedersen, 2020), *purrr* (Version 0.3.4; Henry & Wickham, 2020), *Rcpp* (Version 1.0.5; Eddelbuettel & François, 2011; Eddelbuettel & Balamuta, 2017), *readr* (Version 1.4.0; Wickham & Hester, 2020), *stringr* (Version 1.4.0; Wickham, 2019), *tibble*

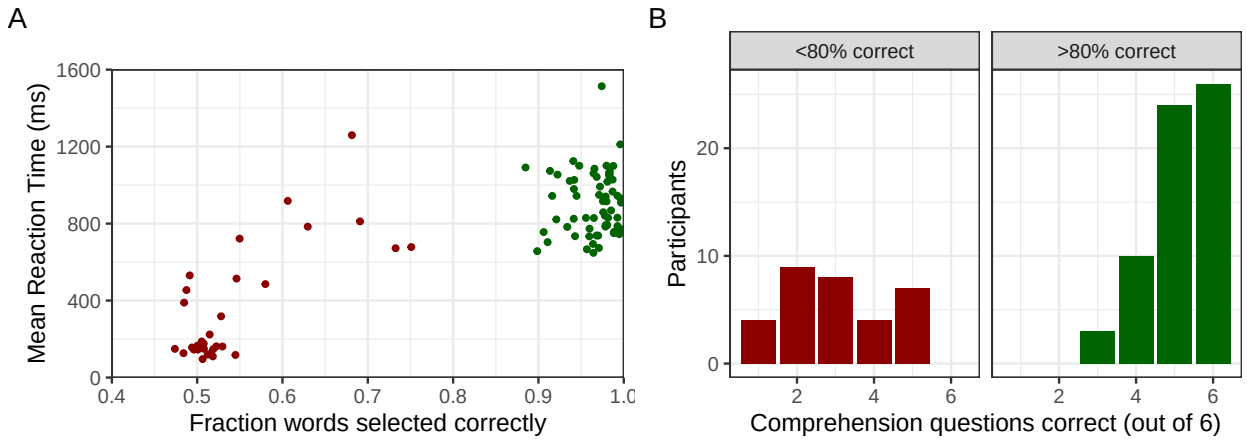

Figure 2. A. Correlation between a participant's accuracy on the Maze task (fraction of words selected correctly) and their average reaction time (in ms). Many participants (marked in green) chose the correct word >80% of the time; others (in red) appear to be randomly guessing and were excluded from further analysis. B. Performance on the comprehension questions. Participants who had >80% task accuracy tended to do well on comprehension questions; those who were at chance on the task were also at chance on comprehension questions.

(Version 3.0.4; Müller & Wickham, 2020), *tidybayes* (Version 2.3.1; Kay, 2020), *tidyr* (Version 1.1.2; Wickham, 2020b), and *tidyverse* (Version 1.3.0; Wickham, Averick, et al., 2019) for all our analyses.

Results

Reading stories in the Maze task

Our first concern was whether this error-correction paradigm on long passages would work. It did; some participants were able to complete the Maze task with high accuracy and understand the story while doing so.

Participant accuracy reflects how well participants can navigate the task and whether the auto-generated distractors are of acceptable quality. We calculated the per-word error rate for each participant and graphed it against their average reaction time. (To avoid biasing the average if a participant took a pause before returning to the task, RTs greater than 5 seconds were excluded.) As seen in Figure 2A, there is a cluster of participants who make relatively few errors (marked in green), with some reaching 99% accuracy. This indicates that the distractors were generally appropriate and that some participants can maintain focus on the task for the whole story. These careful participants tend to take around 1 second for each word selection, which is much slower than other paradigms CITE SPR something. We also have a cluster of participants (in red) who we infer sped through the task clicking randomly. This distribution is likely due to the mix of workers on Mturk; as we did not use qualification cutoffs.

Another check is whether participants comprehended the story. We counted how many of the binary-choice comprehension questions each participant got right (out of 6), and group them by their performance on the Maze task. As seen in Figure 2B, most participants were were accurate on the task also did well on comprehension questions, while participants who were at chance on the Maze task were also at chance on the comprehension questions. Participants usually answered quickly (within 10 seconds), so we do not believe they were looking up the answers on the internet. These are not especially hard questions, and we can't rule out that some participants may have been able to guess the answers without reading the story. Nonetheless, this provides preliminary evidence that people can understand and remember details of stories they read during the Maze task.

Given that Maze task accuracy and comprehension question accuracy are correlated, but task accuracy is finer-grained, we use task performance as our exclusion metric. This easy to calculate speed and accuracy comparision makes it easy to see who was paying attention to the task; we exclude the random guessers and only analyse data from participants with at least 80% accuracy (in the gap between high-performers and low-performers).

RT and surprisal

GAMs. We wanted to see if Maze also showed a linear relationship between surprisal and RT, and whether this was localized at the current word or whether previous word surprisals were also influential. To do this, we created a set of predictors of frequency, word length, and surprisals from several different langauge models, and fit generalized additive models to check for a linear relationship. For length, we use the length of characters (excluding end punctuation). For a measure of unigram frequency, we tokenize the training data from Gulordava, Bojanowski, Grave, Linzen, and Baroni (2018) (this tokenizes off punctuation, but preserves case) and tally up instances. For the models, we use the log2 frequency of the expected occurances in 1 billion words (thus frequency is log, but higher values indicate higher frequency words). For surprisals, we get per-word surprisals for each of 3 different language models: a Kneser-Ney smoothed 5-gram, GRNN (**???**), and Transformer-XL (Dai et al., 2019). We centered but did not rescale the length and frequency predictors; for interpretability, we kept surprisal uncentered. For all of these predictors, we consider both the predictor at the current word as well as lagged predictors from the previous word. CITE some papers that did the same thing . We used smooths for the surprisal terms and tensor effects for the frequency by length effects and interactions.

We only include words that were in the vocabularies of all three models as a single token and for which we have frequency information. This has the effect of excluding words that had punctuation as well as some uncommon or proper nouns. We also exclude the first word of every sentence (which had a dummy distractor). We do not analyse RTs for words where the RT was <100 or >5000 ms (<100 is likely a recording error, >5000 is likely the participant getting distracted). We also omit words where the participant made a mistake and had made a mistake earlier in the sentence (Although see supplement for results including post-mistake TODO).

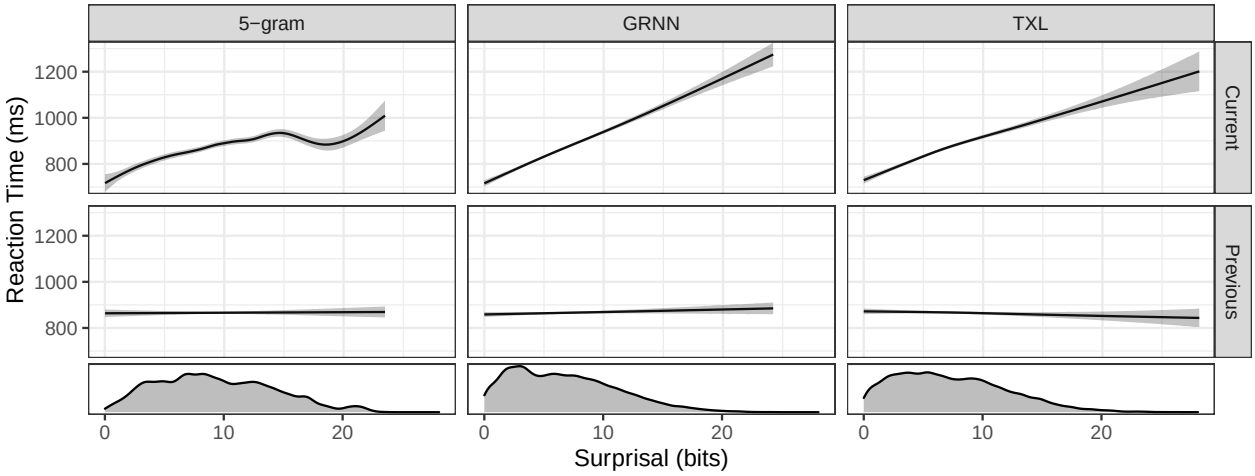

Figure 3. GAM predictions of reaction time (RT) as a function of either current word surprisal (top) or previous word surprisal (bottom). Density of data is shown along the x-axis. For each of the 3 language models used, there is a linear relationship between current word surprisal and RT (at least when there is enough data). There is no relationship between previous word surprisal and RT.

The smooths for the current and previous words surprisals are shown in Figure 3. Note that for each of the models, high-surprisal words at relatively rare, with much of the data for words between 0 and 15 bits of surprisal. All of the models show a roughly linear relationship between current word surprisal and RT, especially in the region with more data. All of the models show a flat relationship between previous word surprisal and RT. This is a sign of localization as the previous word's surprisal is not affecting RT, only the word's own suprisal is. The linear relationship matches that found with other methodologies.

Given the linear relationship, we next fit linear models to look at coefficients and model fit. We fit BRMS models.

BRM models. We built models using BRMS, with surprisal, frequency, length and surprisal x length and frequency x length effects. Predictors were as described above, except all were centered. We used full mixed effects, including by-subject effects of everything and a per-word random intercept. We used weak priors (normal(1000,1000) for intercept, normal(0,500) for beta and sd, and lkj(1) for correlations). TODO CITE also maybe this is too details-y methods.

As we can see in Table 1, we find large effects of surprisal and length, but minimal effects of frequency. These effects are larger than what is usually reported in other methods CITE, but this could be due to the overall slowness of the method; the intercepts are also much bigger. The lack of frequency effects is somewhat surprising, and some CITE argue that they aren't real anyway. Notably the coefficients for the lagged terms are smaller than surprisal and length of the current word.

Model comparison. We also ran a nested model comparison with models fit in lmer. For this, we fit models with only frequency and length as predictors, as well as

Table 1
Predictions from fitted Bayesian regression models. All terms were centered, but not rescaled. Units are in ms. Surprisal is per bit, length per character, and frequency per log_2 occurance per billion words.

Term	5-gram	GRNN	TXL
Intercept	865.3 [829.9, 902.9]	871.1 [837.9, 905.3]	870.8 [832.5, 907.8]
Surprisal	11.7 [9.3, 14.1]	23.7 [21, 26.5]	18.5 [16.1, 21.1]
Length	20.5 [15.4, 25.6]	18.5 [13.3, 23.7]	21.4 [16.2, 26.6]
Frequency	-2.9 [-6.3, 0.5]	2.9 [-0.2, 6]	0.4 [-2.7, 3.5]
Surp x Length	-2 [-3, -1]	-1.8 [-2.7, -0.9]	-1.4 [-2.2, -0.6]
Freq x Length	-1 [-2.5, 0.4]	-0.1 [-1.2, 1]	0.2 [-0.9, 1.2]
Past Surprisal	1.6 [-0.5, 3.6]	2.7 [0.8, 4.5]	0.8 [-0.9, 2.5]
Past Length	-4.8 [-9, -0.1]	-6.6 [-10.9, -2.1]	-5.2 [-9.3, -0.7]
Past Freq	2.6 [-0.1, 5.4]	1.9 [-0.2, 4.2]	1.2 [-1.1, 3.6]
Past Surp x Length	-0.2 [-1.2, 0.8]	-0.9 [-1.7, -0.2]	-0.6 [-1.3, 0.2]
Past Freq x Length	-1 [-2.3, 0.3]	-1.8 [-2.9, -0.8]	-1.5 [-2.6, -0.5]

models that also had one or more sources of surprisal. For all effects, we included both current and lagged versions of it. The model with all 3 surprisal predictors is better than any model with only one. Any one surprisal predictor is better than no surprisals predictors. However, adding ngram predictors to a model that already has txl & grnn does not help. In other cases, adding the 3rd surprisal source to the other two does help. Ngram+Grnn is not better than Grnn only. Otherwise, pairs are better than singletons. This suggests that Ngram's info is a subset of GRNN, but not a subset of TXL.

Discussion

[summarize results] - error-correction paradigm - test on natural stories - it works

We've shown another way to adapt the Maze task to make it suitable for a wider range of experiments. While we think that error-correction Maze is generally a good idea, especially when using A-maze, this is an orthagonal adjustment to the task. By using the Natural Stories corpus, we establish that Maze works for longer naturalistic tasks, with participants able to do the task and understand what they are reading. Additionally, their RTs reflect expected patterns in terms of length and surprisal. While there are some differences, especially in scale, to that found with other incremental processing paradigms, we think this is a reason to explore more. Could be a statistical thing (???) or due to task differences. Comparisons between methods could be very useful for identifying how task demands influence processing.

All the code is available, and we encourage researchers to consider trying out Maze if they think it's appropriate to their experiments.

While Maze has traditionally been limited to single sentence items; we innovate on how the materials are presented and show that it is possible for participants to read

naturalistic passages that are presented in the Maze paradigm. We confirm that participants can understand what they are reading, and that Maze shows linear surprisal effects. This opens up another area of research to being amenable to the new A-maze paradigm.

The Maze task might be espeically good for comparing with NNs because of the forced incrementality. While eventually we do want to figure out models of more natural reading, this might be a useful complement b/c we don't have to deal with nasty spillover. Might be extra good to do with pauses for mistakes to fix the incentives.

Secondly, this error-correction compensates for some of the shortcomings of A-Maze identified in Boyce et al. (2020); distractors might still cause participants to make unavoidable mistakes, but at least the still see the rest of the sentence and we get their data. Whether this also solves the data loss problem from the researcher perspective within a sentence depends on whether post-mistake data are high-quality and trustworthy; this is a hard-to-assess question of potential interest.

Even if researchers exclude all post-mistake data from analysis, this process can still address the question of whether errors are due to inattentive participants or bad distractors (as discussed in Boyce et al. (2020), section XX). When we have a participants selection for every word, we can easy calculate a per-word error rate for each participant, without having to address the censoring present in error-terminating Maze (where we can't know if a participant would have made more mistakes after the first; with error correction, we know how may more they did make). If participants are guessing, we'll see high per-word error rates, if the early distractors are bad, we'll see low per-word error rates (with errors concentrated at the start of the sentences). This per-word error rate metric measures participants task accuracy, and allows us to exclude data from inattentive participants. This provides a convenient and clear-cut way of controlling data quality after the fact.

It also starts to reduce the perverse incentives. With error-terminates Maze, the fastest way to get through the task is to select randomly, and it's quite quick because mistakes skip you to the next sentence. With error-correction Maze, randomly jamming buttons takes more effort, but is still an effective strategy. [Footnote: In discussing this work, we recieved the suggestion that one way to disincentivize random clicking is to add a pause when a participant makes a mistake, forcing them to wait some short period of time (ex 500ms or 1 sec) before being able to correct their mistake. This seems like a promising improvement that could be worth implementing and testing.]

References

Auguie, B. (2017). *GridExtra: Miscellaneous functions for "grid" graphics*. Retrieved from https://CRAN.R-project.org/package=gridExtra

Aust, F. (2019). *Citr: 'RStudio' add-in to insert markdown citations*. Retrieved from https://CRAN.R-project.org/package=citr

Aust, F., & Barth, M. (2020). *papaja: Create APA manuscripts with R Markdown*. Retrieved from https://github.com/crsh/papaja

Bates, D., & Maechler, M. (2021). *Matrix: Sparse and dense matrix classes and methods*. Retrieved from https://CRAN.R-project.org/package=Matrix

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, *67*(1), 1–48. https://doi.org/10.18637/jss.v067.i01

Boyce, V., Futrell, R., & Levy, R. P. (2020). Maze Made Easy: Better and easier measurement of incremental processing difficulty. *Journal of Memory and Language*, *111*, 104082. https://doi.org/10.1016/j.jml.2019.104082

Bürkner, P.-C. (2017). brms: An R package for Bayesian multilevel models using Stan. *Journal of Statistical Software*, *80*(1), 1–28. https://doi.org/10.18637/jss.v080.i01

Bürkner, P.-C. (2018). Advanced Bayesian multilevel modeling with the R package brms. *The R Journal*, *10*(1), 395–411. https://doi.org/10.32614/RJ-2018-017

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv:1901.02860 [Cs, Stat]*. Retrieved from http://arxiv.org/abs/1901.02860

Eddelbuettel, D., & Balamuta, J. J. (2017). Extending extitR with extitC++: A Brief Introduction to extitRcpp. *PeerJ Preprints*, *5*, e3188v1. https://doi.org/10.7287/peerj.preprints.3188v1

Eddelbuettel, D., & François, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, *40*(8), 1–18. https://doi.org/10.18637/jss.v040.i08

Forster, K. I., Guerrera, C., & Elliot, L. (2009). The maze task: Measuring forced incremental sentence processing time. *Behavior Research Methods*, *41*(1), 163–171. https://doi.org/10.3758/BRM.41.1.163

Futrell, R., Gibson, E., Tily, H. J., Blank, I., Vishnevetsky, A., Piantadosi, S. T., & Fedorenko, E. (2020). The Natural Stories corpus: A reading-time corpus of English texts containing rare syntactic constructions. *Lang Resources & Evaluation*. https://doi.org/10.1007/s10579-020-09503-7

Gulordava, K., Bojanowski, P., Grave, E., Linzen, T., & Baroni, M. (2018). Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 1195–1205).

Henry, L., & Wickham, H. (2020). *Purrr: Functional programming tools*. Retrieved from https://CRAN.R-project.org/package=purrr

Kay, M. (2020). *tidybayes: Tidy data and geoms for Bayesian models*. https://doi.org/10.5281/zenodo.1308151

Müller, K. (2020). *Here: A simpler way to find your files*. Retrieved from https://CRAN.R-project.org/package=here

Müller, K., & Wickham, H. (2020). *Tibble: Simple data frames*. Retrieved from https://CRAN.R-project.org/package=tibble

Pedersen, T. L. (2020). *Patchwork: The composer of plots*. Retrieved from https://CRAN.R-project.org/package=patchwork

Pinheiro, J., Bates, D., DebRoy, S., Sarkar, D., & R Core Team. (2020). *nlme: Linear and nonlinear mixed effects models*. Retrieved from https://CRAN.R-project.org/package=nlme

R Core Team. (2020). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Sloggett, S., Handel, N. V., & Rysling, A. (n.d.). A-maze by any other name, 1.

Wickham, H. (2016). *Ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York. Retrieved from https://ggplot2.tidyverse.org

Wickham, H. (2019). *Stringr: Simple, consistent wrappers for common string operations*. Retrieved from https://CRAN.R-project.org/package=stringr

Wickham, H. (2020a). *Forcats: Tools for working with categorical variables (factors)*. Retrieved from https://CRAN.R-project.org/package=forcats

Wickham, H. (2020b). *Tidyr: Tidy messy data*. Retrieved from https://CRAN.R-project.org/package=tidyr

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., ... Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Wickham, H., François, R., Henry, L., & Müller, K. (2020). *Dplyr: A grammar of data manipulation*. Retrieved from https://CRAN.R-project.org/package=dplyr

Wickham, H., & Hester, J. (2020). *Readr: Read rectangular text data*. Retrieved from https://CRAN.R-project.org/package=readr

Wilke, C. O. (2020). *Cowplot: Streamlined plot theme and plot annotations for 'ggplot2'*. Retrieved from https://CRAN.R-project.org/package=cowplot

Witzel, N., Witzel, J., & Forster, K. (2012). Comparisons of online reading paradigms: Eye tracking, moving-window, and maze. *Journal of Psycholinguistic Research*, *41*(2), 105–128.

385 Wood, S. N. (2003). Thin-plate regression splines. *Journal of the Royal Statistical Society*
386 *(B)*, *65*(1), 95–114.

387 Wood, S. N. (2004). Stable and efficient multiple smoothing parameter estimation for
388 generalized additive models. *Journal of the American Statistical Association*,
389 *99*(467), 673–686.

390 Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood
391 estimation of semiparametric generalized linear models. *Journal of the Royal*
392 *Statistical Society (B)*, *73*(1), 3–36.

393 Wood, S. N., N., Pya, & S"afken, B. (2016). Smoothing parameter and model selection for
394 general smooth models (with discussion). *Journal of the American Statistical*
395 *Association*, *111*, 1548–1575.