Last compiled: Tuesday 21st February, 2023 at 16:54

Chapter 1

bdots

Abstract

The Bootstrapped Differences of Timeseries (bdots) was first introduced by Oleson (and others) as a method for controlling type I error in a composite of serially correlated tests of differences between two time series curves in the context of eye tracking data. This methodology was originally implemented in R by Seedorff 2018. Here, we revist the original package, both expanding the underlying theoretical components and creating a more robust implementation.

1.1 Introduction

i hate introductions we will do this part last

In 2017, Oleson et al. introduced a method for detecting time-specific differences in the trajectory of outcomes between experimental groups. Particularly in the case of a densely sampled time series, the construction of evaluating differences at each point in time results in a series of highly correlated test statistics expanding the family-wise error rate, accommodated with an adjustment to the nominal alpha based on this autocorrelation. This was followed up with in 2018 with the introduction of the `bdots` package to CRAN [10]. Here, we introduce the second version of `bdots`, an update to the package that broadly expands the capabilities of the original.

This manuscript is not intended to serve as a complete guide for using the `bdots` package. Instead, the purpose is to showcase major changes and improvements, with those seeking a more comprehensive treatment directed to the package vignettes. Rather than taking a "compare and contrast" approach, we will first enumerate the major changes, followed by a general demonstration of the package use:

1. Major changes to underlying methodology with implications for prior users of the package
2. Simplified user interface
3. Introduction of user defined curves
4. Permit fitting for arbitrary number of groups
5. Automatic detection of paired tests based on subject identifier
6. Allows for non-homogeneous sampling of data across subjects and groups
7. Introduce formula syntax for bootstrapping difference function
8. Interactive refitting process

First, a quick review of framing the problem at hand.

We start by clearly delineating the type of problem that `bdots` has been created to solve.

Bootstrapped differences in time series We begin by assuming two or more experimental groups in which a subject response is measured over time. This may include the growth of tumors in mice or the change in the proportion of fixations over time in the context of the VWP. In either case, we assume that each of the subjects in the groups being considered has observed data of the following form:

$$y_{it} = f_{\theta_i}(t) + \epsilon_{it} \tag{1.1}$$

OR (as was written in the original)

$$y_{it} = f(\theta_{it}) + \epsilon_{it} \tag{1.2}$$

where f represents a functional mean structure, while the error structure of ϵ_{it} is open to be either IID or possess an AR(1) structure. At present, `bdots` requires that each of the subjects being compared have the same parametric function f, this is not strictly necessary and future directions of the package include accommodating non-parametric functions. While each of the subjects are required to be of the same parameteric form f_θ, each differs in their instance of their own subject-specific parameters, θ_i.

The collection of subjects' θ within a group constitutes a group distribution – bootstrapping parameters from this distribution and evaluating functions f at these values gives us an estimate of the distribution of functions. As these functions are *in time*, this in turn gives a representation of temporal changes in group characteristics. It is precisely the identification of how these temporal changes differ between groups that `bdots` seeks to [do?]

This differs from the original iteration of `bdots` (or Oleson 2017) in a critical way, however. Idk how to say this, really, but the original assumed that there was no variability between subject parameters (which I am calling the homogeneity of means assumption), in which $\theta_i = \theta_j$ for all i, j in an experimental group. The primary consequence of this was the absence of any need to estimate the variability within a group, taking into consideration only subject-specific variability (don't want to go into a lot of details because I kind of treat this mathematically in the next section). Anyways, see Chapter 3 for all the deets on this. Finally, a full review of the context in which it was introduced is available in Seedorff et. al., 2018 [10].

[transition paragraph]

I really need to decide how much detail to go into here. I don't want to repeat everything from chapter 3, and even though this doesn't have to be a stand alone paper now, it will be and when it is this will be pretty damn important to convey, especially to those who used bdots previously.

[also note: however we decide to deal with dissemination of this information, and while I think 2.0.0 should definitely be released with all of this, I almost htink we should address the fitcode awkwardness before this is presented to the public]

Alrightly, on to taking a looksee at how bdots works

1.2 Methodology and Overview

A standard analysis using `bdots` consists of two steps: fitting the observed data to a specified parametric function, f_θ, and then using the observed variability to construct estimates of the distributions of each groups' curves. Here, we briefly detail how this is implemented in practice and introduce the new methodologies in `bdots v2`. A more comprehensive treatment of these new methods, along with their justifications, is offered in Chapter 3.

1.2.1 Establishing subject-level curves

We begin with the assumption that for subject i, the observed data is of the form

$$y_{it} = f_{\theta_i}(t) + \epsilon_{it}, \tag{1.3}$$

where ϵ_{it} can be specified to be either independent or have an AR(1) auto-correlation structure. Each subject is fit in `bdots` via `gnls`, returning an estimated set of parameters and their associated standard errors. Assuming large sample normality, we are able to construct a sampling distribution for each subject, accounting for within-subject variability. This gives us for each subject a distribution

$$\hat{\theta}_i \sim N(\theta_i, s_i^2). \tag{1.4}$$

1.2.2 Estimating Group Distributions

Once sampling distributions are created for each subject, we are prepared to begin estimating group distributions. We assume that the mean parameters for each subject come from a group level distribution, where for each subject i,

$$\theta_i \sim N(\mu_\theta, V_\theta). \tag{1.5}$$

This is notably in contrast to the original implementation of `bdots`; there, the authors proceeded with an assumption of homogeneity of means, with $\theta_i = \theta_j$ for each i, j. A more thorough investigation of this, along with justification for the current method, is presented in Chapter 3.

This in hand, we go about estimating the group distributions according to the following procedure: (not in love with the notation here).

1. For a group of size n, select n subjects from the group *with replacement*. This allows us to construct an estimate of V_θ.
2. For each selected subject i, draw a set of parameters from the distribution $\theta_i^* \sim N(\hat{\theta}_i, s_i^2)$. This gives us an accounting of the observed within-subject variability
3. For each resampled θ_i^*, find the bth bootstrap estimate for the group,.

$$\theta_b^* = \frac{1}{n}\sum_{i=1}^{n} \theta_i^*, \quad \text{where} \quad \theta_b^* \sim N\left(\mu_\theta, \frac{1}{n}V_\theta + \frac{1}{n^2}\sum s_i^2\right). \tag{1.6}$$

4. Perform steps (1.)-(3.) B times, using each θ_b^* to construct a distribution of population curves, $f_\theta(t)$.

The final population curves from (4) can be used to create estimates of the mean response and an associated standard deviation at each time point for each of the groups bootstrapped. These estimates are used both for plotting and in the construction of confidence intervals. They also can be, but do not necessarily have to be, used to construct a test statistic, which is the topic of our next section.

1.2.3 Hypothesis testing for statistically significant differences

We now turn our attention to the primary goal of an analysis in `bdots`, identifying time windows in which the distribution of curves of two groups differ significantly. A problem unique to the ones address by `bdots` is that of multiple testing; and especially in densely sampled time series, we must account for multiple testing while controlling the family-wise error rate (FWER). Version 2 of `bdots` provides two ways with which this can be accomplished.

Oleson Adjustment

Just as in the original iteration of `bdots`, we are able to construct test statistics from the bootstrapped estimates described in the previous section. These test statistics $T(t)$ can be written as

$$T(t) = \frac{(\overline{f}_1(t) - \overline{f}_2(t))}{\sqrt{\frac{1}{n_1}\text{Var}(f_1(t)) + \frac{1}{n_2}\text{Var}(f_2(t))}}, \tag{1.7}$$

where [. . .] I actually don't like that notation at all, but I can't use $\overline{p}_{1t}$ and s_{1t}^2, really, because I already used s_i^2 for within-subject variance. Whatever, pin in this for now, pretend I carefully described what all of the components of this test statistic is for now.

To account for the multiple testing problem with autocorrelated test statics, we construct a nominal significance level α^* to produce the desired effective FWER α. For now I won't give any more details, it basically finds some autocorrelation parameter and uses this to create a new alpha. We then use this to determine which $T(t) < z1 - \alpha^*/2$, giving us our significant regions. neato.

Permutation testing

Alternatively, `bdots` provides a modified permutation test for controlling the FWER without any additional assumptions on the autocorrelated status of the errors or test statistics.

We begin by creating test statistics at each time point, similar to Equation 1.7. Using the fitted parameters $\hat{\theta}_i$ for each subject i, we construct subject-specific curves $f_{\hat{\theta}_i}$ and use *these* to construct population mean and standard deviations at each time point, giving population curves and standard deviations [...] (i still don't have notation i really like for this, especially in contrast to the bootstrapping step). We use these to create the observed test statistic

$$T_p(t) = [\dots] \tag{1.8}$$

(this is basically the same formula as Equation 6 but with absolute value and having $\overline{f}$ mean different things. whatever notation is used)

When then going about using permutations to construct a null distribution against which to compare the observed statics. We do so with the following algorithm:

1. Assign to each subject a label indicating group membership
2. Randomly shuffle the labels assigned in (1.), creating two new groups
3. Recalculate the test statistic $T_p(t)$, recording the maximum value
4. Repeat (2.)-(3.) P times

The collection of maximum values for $T(t)$ will serve as the null distribution against which to compare our observed $T(t)$. Regions in which the observed t statistic are beyond the specified α in the null distribution are then considered significant.

Odds and Ends

Both of the methods presented are able to accommodate paired assumptions with minor adjustments to their algorithms. In the case of the bootstrap, we simply must take care to ensure that for each iteration, the collection of subjects sampled in one group with replacement are also sampled in the other, ensuring that each bootstrapped estimate comes from the same distribution. Likewise, paired testing is implemented through permutation testing by modifying the shuffling process so that each subject has one set of observations in each of the permuted groups.

Finally, it is of note that other adjustments for FWER are offered here as were in the original implementation of `bdots`, including all of the adjustments present in the `p.adjust` function from the `stats` R package. I say this for completeness, but idk that anybody cares. Okie dokie, then, thems the methods! On to an example analysis.

```
> head(mouse, n = 10)
      Volume Day Treatment ID
 1:   47.432   0         A  1
 2:   98.315   5         A  1
 3:  593.028  15         A  1
 4:  565.000  19         A  1
 5: 1041.880  26         A  1
 6: 1555.200  30         A  1
 7:   36.000   0         B  2
 8:   34.222   4         B  2
 9:   45.600  10         B  2
10:   87.500  16         B  2
```

Figure 1.1: Illustration of Mouse data

1.3 Example Analysis

In this next section we are going to review a worked example of a typical use of the `bdots` package. We will use as our illustration a study (source?) comparing tumor growth for the 451LuBr cell line in mice data with repeated measures in five treatment groups.

Note that in Figure 1.1, the `Day` variable contains different values between two mice (indicated by `ID`). This reflects a new feature of `bdots`, which is the ability to fit and analyze subjects with non-homogenous time samples. Details on how to adjust how non-homogenous times are handled in a later section (they're not – I'm not sure what to do abou this yet when bootstrapping functions that are FAR outside of their observed range. I almost think the default should be a mini-max approach, creating an arbitrary range of values in the intersection of the range of all of the observed data, but with ability to specify time range directly in bootstrap function).

There are two primary functions in the `bdots` package: one for fitting the observed data to a parametric function and another for estimating group distributions and identifying time windows where they differ significantly. The first of these, `bfit`, is addressed in the next section.

1.3.1 Curve Fitting

The curve fitting process is performed with the `bfit` function (previously `bdotsFit`), taking the following arguments:

```
bfit(data, subject, time, y, group, curveType,, ...)
```

Figure 1.2: Main arguments to `bfit`, though see `help(bfit)` for additional options

(need to add/talk about an AR1 argument that fits data with this assumption or not. Really this AR1 thing is a mess. I think for the dissertation, outside of new methodology we just pretend that chapter 3 doesn't exist)

The `data` argument takes the name of the dataset being used, which should be stored in long format. `subject` is the subject identifier column in the data and should be passed as a character. It is important to note here that the identification of paired data is done automatically; in determining if two experimental groups are paired, `bdots` checks that the intersection of subjects in each of the groups are identical with the subjects in each of the groups individually. The `time` and `y` arguments are column names of the time variable and outcome, respectively. Similarly, `group` takes as an argument a character vector of each of the group columns that are meant to be fit, accommodating the fact that `bdots` is now able to fit an arbitrary number of groups at once, provided that the outcomes in each group adopt the same parametric form. This brings us to the `curveType` argument, which is addressed in the next section.

Curve functions Whereas the previous iteration of `bdots` had a separate fitting function for each parametric form (i.e., `logistic.fit` for fitting data to a four-parameter logistic), we are now able to specify the curves we wish to fit independent of the fitting function `bfit`. This is done with the `curveType` argument. Unlike the previous arguments which took either a `data.frame` or character vector, `curveType` takes as an argument a function call, for example, `logistic()`. The motivation for this is detailed elsewhere (the appendix, maybe?), but in short, it allows the user to pass additional arguments to further specify the curve. For example, among the parametric functions included in `bdots` is now the `polynomial` function, taking as an additional argument the number of degrees we wish to use. The fit the observed data with a five parameter polynomial in `bfit`, one would then pass the argument `curveType = polynomial(degree = 5)`. Literal magic takes care of the rest. Curve functions currently included in `bdots` include `logistic()`, `doubleGauss()`, `expCurve()`, and `polynomial()`. `bfit` can also accept user-created curves; detailed vignettes for writing your own can be found with `vignette("bdots")`. (maybe i'll show going from logistic to logistic with distribution for custom functions)

We fit the mouse data to an exponential curve with `expCurve()` and using the column names found in Figure 1.1:

```
mouse_fit <- bfit(data = mouse, subject = "ID", time = "Day",
                  y = "Volume", group = "Treatment", curveType = expCurve())
```

Return object and generics The function `bfit` returns an object of class `bdotsObj`, inheriting from class `data.table`. As such, each row uniquely identifies one permutation of subject and group values. Included in this row are the subject identifier, group classification, summary statistics regarding the curves, and a nested `gnls` object.

```
> class(mouse_fit)
[1] "bdotsObj"   "data.table" "data.frame"

> head(mouse_fit)
   ID Treatment        fit      R2   AR1 fitCode
1:  1         A <gnls[18]> 0.97349 FALSE       3
2:  2         B <gnls[18]> 0.83620 FALSE       4
3:  3         E <gnls[18]> 0.96249 FALSE       3
4:  4         C <gnls[18]> 0.96720 FALSE       3
5:  5         D <gnls[18]> 0.76156 FALSE       5
6:  7         B <gnls[18]> 0.96361 FALSE       3
```

Figure 1.3: A `bfit` object inheriting from `data.frame`

The number of columns will depend on the total number of groups specified, with the subject and group identifiers always being the first columns. Following this is the `fit` column, which contains the fitted object returned from `gnls`, as well as `R2` indicating the R^2 statistic. The `AR1` column indicates whether or not the observed data was able to be fit with an AR(1) error assumption. Finally, there is the `fitCode` column, which we will describe in more detail shortly.

Several methods exist for this object, including `plot`, `summary`, and `coef`, returning a matrix of fitted coefficients obtained from `gnls`.

Fit Codes The `bdots` package was originally introduced to address a very narrow scope of problems, and the `fitCode` designation is an artifact of this original intent. Specifically, it assumed that all of the observed data was of the form given in Equation B.1 where the observed time series was dense and the errors were autocorrelated. Autocorrelated errors can be specified in the `gnls` package (used internally by `bdots`) when generating subject fits, though there were times when the fitter would be incapable of converging on a solution. In that instance, the autocorrelation assumption was dropped and constructing a fit was reattempted.

R^2 proved a reliable metric for this kind of data, and preference was given to fits with an autocorrelated error structure over those without. From this, the hierarchy given in Table 1.1 was born. `fitCode` is a numeric summary statistic ranked from 0 to 6 detailing information about the quality of the fitted curve, constructed with the following pseudo-code:

```
AR1 <- # boolean, determines AR1 status of fit
fitCode <- 3L*(!AR1) + 1L*(R2 < 0.95)*(R2 > 0.8) + 2L*(R2 < 0.8)
```

A fit code of 6 indicates that `gnls` was unable to successfully fit the subject's data.

`bdots` today stands to accommodate a far broader range of data for which the original `fitCode` standard may no longer be reliable. The presence of autocorrelation cannot always be assumed, and users may opt for a metric other than R^2 for assessing the quality of the fits. Even the assessments of fits on a discretized scale may be something of only passing interest. Even then, however, this is how the current implementation of `bdots` categorizes the quality of its fits, with the creation of greater flexibility in this regard being a large priority for future directions. Outside of general summary information, the largest impact of this system is in the refitting process, which organizes the fits by `fitCode`. There is still flexibility in how this is handled, though we will reserve that for the relevant section. (i hate this whole section).

`fitCode`	AR(1)	R^2
0	TRUE	$R^2 > 0.95$
1	TRUE	$0.8 < R2 < 0.95$
2	TRUE	$R^2 < 0.8$
3	FALSE	$R^2 > 0.95$
4	FALSE	$0.8 < R2 < 0.95$
5	FALSE	$R^2 < 0.8$
6	NA	NA

Table 1.1: fit codes, though less relevant for other types of data so idk really what to do about it. nothing for the dissertation, at least

Plots and summaries Users are able to quickly summarize the quality of the fits with the `summary` method now provided.

```
> summary(mouse_fit)

bdotsFit Summary

Curve Type: expCurve
Formula: Volume ~ x0 * exp(Day * k)
Time Range: (0, 106) [31 points]

Treatment: A
Num Obs:  10
Parameter Values:
        x0          k
172.232953   0.056843
########################################
############### FITS ###################
########################################
AR1,       0.95 <= R2        -- 2
AR1,       0.80 < R2 <= 0.95 -- 1
AR1,       R2 < 0.8          -- 0
Non-AR1,   0.95 <= R2        -- 0
Non-AR1,   0.8 < R2 <= 0.95  -- 3
Non-AR1,   R2 < 0.8          -- 4
No Fit                       -- 0

[...]

All Fits
Num Obs:  42
Parameter Values:
        x0          k
102.487118   0.053662
########################################
############### FITS ###################
########################################
AR1,       0.95 <= R2        -- 4
AR1,       0.80 < R2 <= 0.95 -- 2
AR1,       R2 < 0.8          -- 0
Non-AR1,   0.95 <= R2        -- 9
Non-AR1,   0.8 < R2 <= 0.95  -- 16
Non-AR1,   R2 < 0.8          -- 11
No Fit                       -- 0
```

Figure 1.4: Abridged output from the summary function (missing summary information for groups B-E. Note that this includes data on the formula used, the quality of fits and mean parameter estimates by group, and a summary of all fits combined

It is also recommended that users visually inspect the quality of fits for their subjects, which includes a plot of both the observed and fit data. There are a number of options available in `?plot.bdotsObj` (they can call it with just plot but you look for help with this idk), including the option to fit the plots in base R rather than `ggplot2`. This is especially helpful when looking to quickly assess the quality of its (rather than reporting) because `ggplot2` is notoriously slow, and especially when you have 400 subjects. I haven't *actually* done this yet but will shortly. Anyways, check out Figure 1.5 for a plot of the first four fitted subjects.

```
plot(mouse_fit[1:4, ])
```

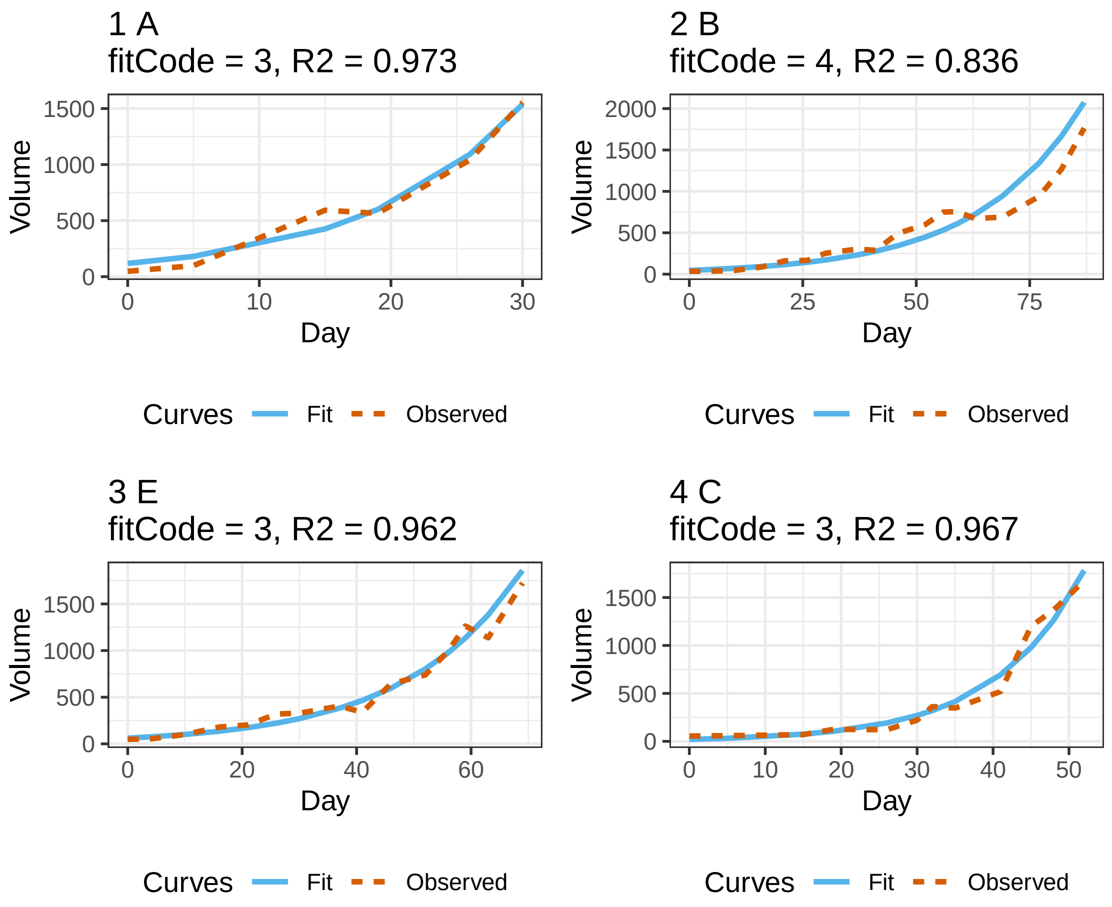


Figure 1.5: Plot of `mouse_fit`, using `data.table` syntax to subset to only the first four observations

1.3.2 Bootstrapping

Once fits have been made, we are ready to begin estimating the group distributions. This is done with the bootstrapping function, `bboot`. The number of options included in the `bboot` function have expanded to include a new formula syntax for specifying the analysis of interest as well as to include options for permutation testing. A call to `bboot` takes the following form

```
bboot(formula, bdObj, alpha, permutation = TRUE, padj = "oleson", ...)
```

Figure 1.6: should i caption this? It feels naked without and it feels too ritzy with

The `formula` argument is new to version 2 of `bdots` and will be discussed in the next section. As for the remaining arguments, `bdObj` is simply the object returned from `bfit` that we wish to investigate, and `B` serves the dual role of indicating the number of bootstraps/permutations we wish to perform. `alpha` is the rate at which we wish to control the FWER. `permutation` and `padj` work in contrast to one another; when `permutation = TRUE` (the default?), the argument to `padj` is ignored. Otherwise, `padj` indicates the method to be used in adjusting the nominal `alpha` to control the FWER. By default, `padj = "oleson"`. Finally, as previously mentioned, there is no longer a need to specify if the groups are paired, and `bboot` determines this automatically based on the subject identifiers in each of the groups.

Formula

As the `bfit` function is now able to create fits for an arbitrary number of groups at once, we rely on a formula syntax in `bboot` to specify precisely which groups differences we wish to compare. Let`y` designate the outcome variable indicated in the `bfit` function and let `group` be one of the group column names to which our functions were fit. Further, let `val1` and `val2` be values of two of the groups in that same column. The general syntax for the `bboot` function takes the following form:

```
y ~ group(val1, val2)
```

Note that this is an *expression* in R and is written without quotation marks as in a character vector. To give a more concrete example, suppose we wished to compare the difference in tumor growth curves for A and B from the `Treatment` column in our mouse data (Figure 1.1). We would do so with the following syntax:

```
Volume ~ Treatment(A, B)
```

There are two special cases to consider when writing this syntax. The first is the situation that arises in the case of multiple or nested groups, the second when a difference of difference analysis is conducted. Details on both of these cases are handled in the appendix.

Summary and Analysis

[what gets a paragraph, what gets a subsubsection? compare this with ***subsection*** for fitting]

Let's begin first by running `bboot` using bootstrapping to compare the difference in tumor growth between treatment groups A and E in our mouse data using permutations to test for regions of significant difference.

```
mouse_boot <- bboot(Volume ~ Treatment(A, E), bdObj = mouse_fit, permutation = TRUE)
```

This returns an object of class `bdotsBootObj`. A summary method is included to display relevant information:

```
> summary(mouse_boot)

bdotsBoot Summary

Curve Type: expCurve
Formula: Volume ~ x0 * exp(Day * k)
Time Range: (0, 59) [21 points]

Difference of difference: FALSE
Paired t-test: FALSE
Difference: Treatment

FWER adjust method: Permutation
Alpha: 0.05
Significant Intervals:
     [,1] [,2]
[1,]   15   32
```

There are a few components of the summary that are worth identifying when reporting the results. In particular, note the time range provided, an indicator of if the test was paired, and which groups were being considered (noticing now it only has `Treatment`, not `A` or `E`). The last section of the summary indicates the method used, an adjusted `alphastar` if `padj` was used, and then a matrix of regions identified as being significantly different. This matrix is `NULL` is no differences were identified at the specified alpha; otherwise there is one row included for each disjointed region of significant difference.

In addition to the provided summary output, a `plot` method is available, with a list of additional options included in `help(plot.bdotsBootObj)`.

```
plot(mouse_boot)
```

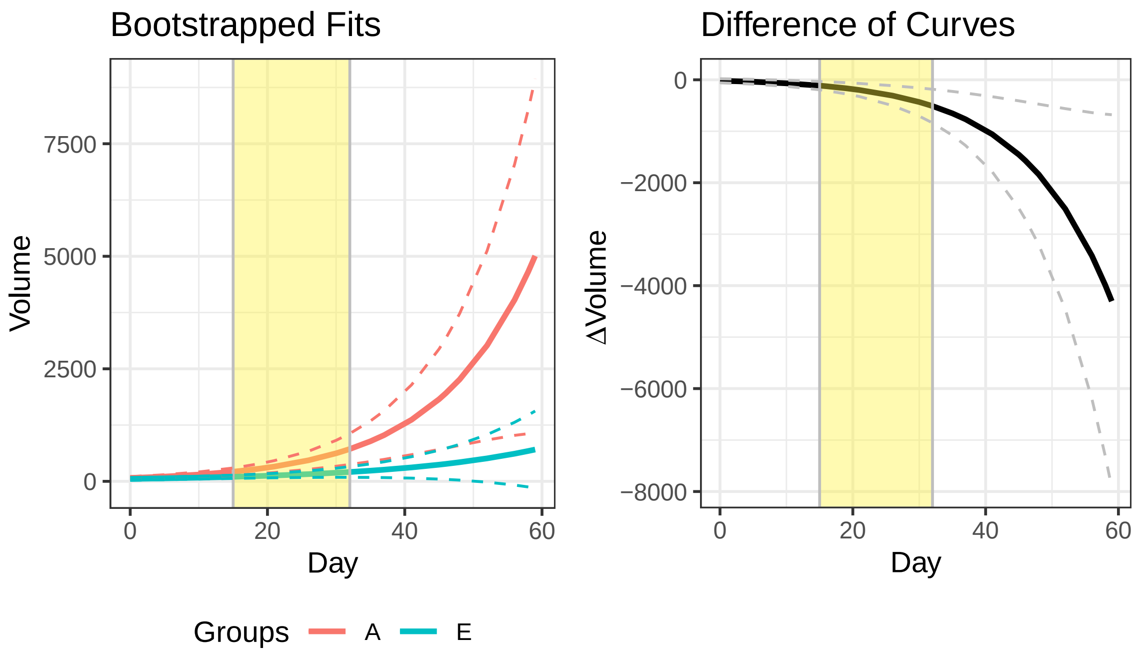


Figure 1.7: There are some obvious issues with time for non-homogenous samples, namely, what do we use for bootstrapping? It will be quick fix, whatever we decide, but I don't think "union of all observed times" is going to work. Here, I artificially cut it back to only 0-60

Depending on user needs, these plots can be recreated both without confidence bands or without the additional difference curve

```
plot(mouse_boot, ciBands = FALSE, plotDiffs = FALSE)
```

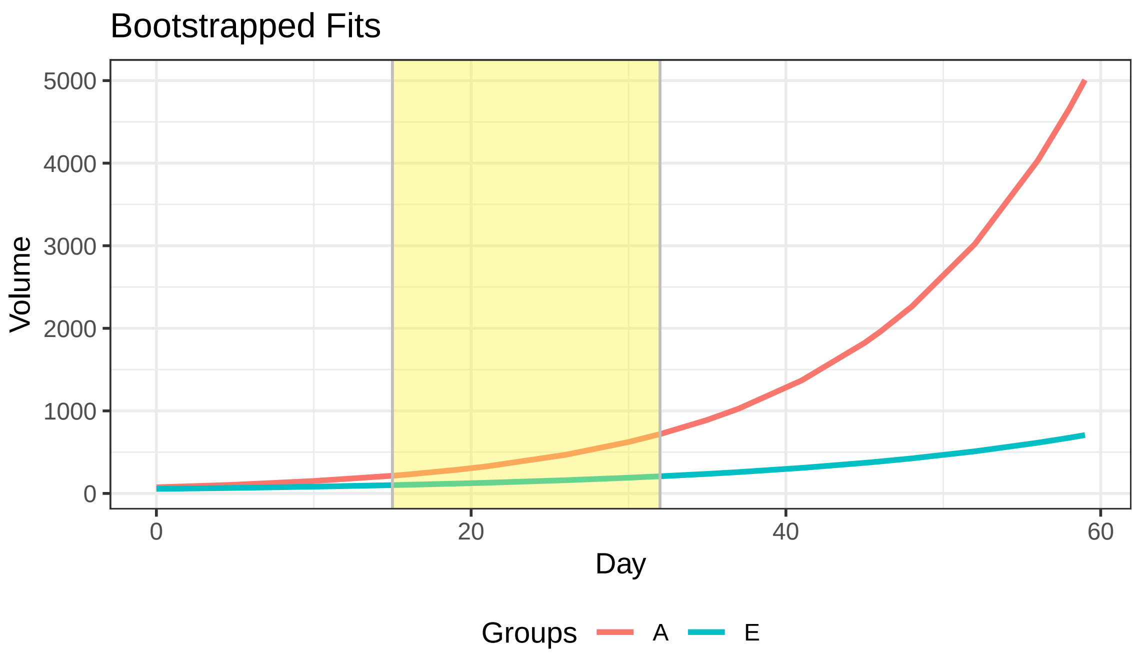


Figure 1.8: I think I'm going to actually not include this because who cares. I specified that they can find more options with the help function.

1.4 Extras

Let's do a brief tour of some of the other additions to bdots that probably doesn't warrant its own section for use

1.4.1 Refitting

["Better intro; I have an idea" - Patrick Breheny 1/18/31, what does better intro mean?]

There are sometimes situations in which the fitted function returned by `bfit` is a poor fit. This is largely a consequence of the sensitivity of the `nlme::gnls` function used in `bfit` for fitting the non-linear curves. Sensible starting parameters are computed as a part of the curve fitting functions (i.e., `logistic()`, but see the vignettes for more details), though these can often be improved upon. The quality of the fit can be evidenced by the `fitCode` or via a visual inspection of the fitted functions against the observations for each subject. When this occurs, there are several options available to the user, all of which are provided through the function `brefit` (previously `bdotsRefit`). `brefit` takes the following arguments:

```
brefit(bdObj, fitCode = 1L, subset = NULL, quickRefit = FALSE, paramDT = NULL)
```

The first of these arguments outside of the `bdObj` is `fitCode`, indicating the minimum fit code to be included in the refitting process. As discussed in Section 1.3.1, this can be sub-optimal. To add flexibility to which subjects are fit there is now the `subset` argument taking either a logical expression or collection of indices that would be used to subset an object of class `data.table` (am I explaining this clearly?) or a numeric vector with indices that the user wishes to refit.

To assist with the refitting process is the argument `quickRefit`. When set to `TRUE`, `brefit` will take the average coefficients of accepted fits within a group and use those as new starting parameters for poor fits. The new fits will be retained if they have a larger R^2 value by default (and really the only option and *technically* this isn't actually true yet but will be). When set to `FALSE`, the user will be guided through a set of prompts to refit each of the curves manually.

Finally, the `paramDT` argument allows for a `data.table` with columns for subject, group identifiers, and parameters to be passed in as a new set of starting parameters. This `data.table` requires the same format as that returned by `bdots::coefWriteout`. The use of this functionality is covered in more detail in the `bdots` vignettes and is a useful way for reproducing a `bdotsObj` from a plain text file.

When `quickRefit = FALSE`, the user is put through a series of prompts along with a series of diagnostics for each of the subjects to be refit:

```
Subject: 11
R2: 0.837
AR1: FALSE
rho: 0.9
fitCode: 4

 Model Parameters:
      x0          k
53.186497  0.051749

Actions:
1) Keep original fit
2) Jitter parameters
3) Adjust starting parameters manually
4) Remove AR1 assumption
5) See original fit metrics
6) Delete subject
99) Save and exit refitter
Choose (1-6):
```

Along with this is given a plot of the original fit, side-by-side with the suggested alternative, Figure 1.9.

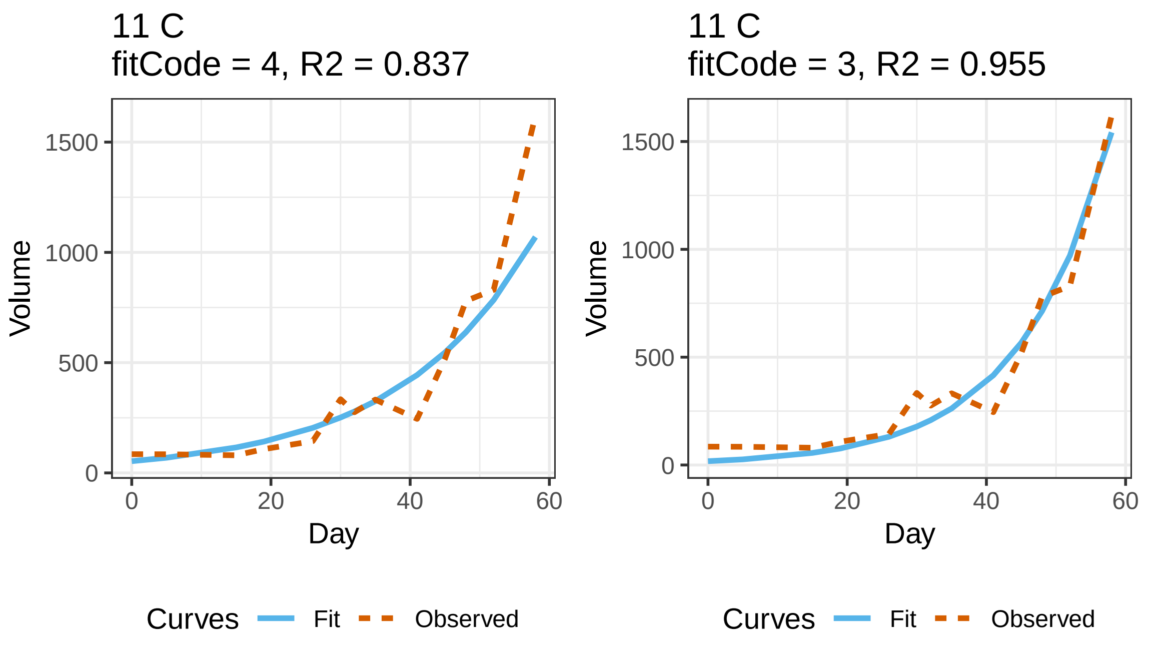


Figure 1.9: before and after refit: man i am good at picking new parameters

As the menu item suggests, users have the ability to end the manually refitting process early and save where they had left off. To retain previously refit items and start again at a later time, pass the first refitted object back into the refitter as such:

```
refit <- brefit(fit, ...)
refit <- brefit(refit, ...) # pass in the refitted object
```

A final note should be said regarding the option to delete a subject. As `bdots` now automatically determines if subjects are paired based on subject identifiers (necessary for calculations in significance testing), it is critical that if a subject has a poor fit in one group and must be removed that he or she is also removed from all additional groups in order to retain paired status. This can be overwritten with a final prompt in the `brefit` function before they are removed. The removal of subjects can also be done with the ancillary function, `bdRemove`, useful for removing subjects without undergoing the entire refitting process. See `help(bdRemove)` for details.

1.4.2 User created curves

Continue to ponder if this worth elaborating on here or appendix (or at all)

1.4.3 Correlations

There are sometimes cases in which we are interested in determining the correlation of a fixed attribute with group outcome responses across time . This can be done with the `bcorr` function (previously `bdotsCorr`), which takes as an argument an object of class `bdotsObj` as well as a character vector representing a column from the original dataset used in `bfit`

```
bcorr(fit, "value", ciBands, method = "pearson")
```

And I can't give an example of this with the mouse data atm because it doesn't gracefully handle nonhomogenous time samples.

This returns a thing that can be plotted. Idk, I'll see what bob wants me to say about this because truly im like so whatever

1.4.4 α Adjustment

Finally, we consider an extension to the `p.adjust` function, `p_adjust`, identical to `p.adjust` except that it accepts method `"oleson"` and takes additional arguments `rho`, `df`, and `cores`. `rho` determines the autocorrelation estimate for the oleson adjustment while `df` returns the degrees of freedom used to compute the original vector of t-statistics. If an estimate of `rho` isn't available, one can be computed on a vector of t-statistics using the `ar1Solver` function in `bdots`:

```
t       <- diffinv(rnorm(5))
rho     <- ar1Solver(t)
unadj_p <- pt(t, df = 10)
adj_p   <- p_adjust(unadj_p, method = "oleson",
                    df = 10, rho = rho, alpha = 0.05)
```

Doing so returns both adjusted p-values, which can be compared against the specified alpha (in this case, 0.05). Alternatively the result will also print an alphastar, a nominal alpha at which one can compare the original p-values:

```
> unadjp
[1] 0.5000000 0.0849965 0.0381715 0.1601033 0.0247453 0.0013016
> adjp
[1] 0.9201915 0.1564261 0.0702501 0.2946514 0.0455408 0.0023954
attr(,"alphastar")
[1] 0.027168
```

Here, for example, we see that the last two positions of `unadjp` have values less than `alphastar`, identifying them as significant; alternatively, we see these same two indices in `adjp` significant when compared to

```
alpha = 0.05
```

1.5 Discussion

[short and sweet? I hate unnecessary words]

The original implementation of `bdots` set out to address a very narrow set of problems. Previous solutions beget new opportunities, however, and it is in this space that the second iteration of `bdots` has sought to expand. Since then, the interface between user and application has been significantly revamped, creating a intuitive, reproducible workflow that is able to quickly and simply address a broader range of problems. The underlying methodology has also been improved and expanded upon, offering better control of the family-wise error rate.

While significant improvements have been made, there is room for further expansion. The most obvious of these is the need to include support for non-parametric functions, the utility of which cannot be overstated. Not only would this alleviate the need for the researcher to specify in advance a functional form for the data, it would implicitly accommodate more heterogeneity of functional forms within a group. Along with this, the current implementation is also limited in the quality-of-fit statistics used in the fitting steps to assess performance. R^2 and the presence of autocorrelation are relevant to only a subset of the types of data that can be fit, and allowing users more flexibility in specifying this metric is an active goal for future work. In all, future directions of this package will be primarily focused on user interface, non-parametric functions, and greater flexibility in fit metrics (this last sentence kind of sucks).

Appendix A - custom curves

From an R programming perspective, this is perhaps the most novel and interesting portion of the new package update. Worked use-case examples are included in the pacakge vignettes, so here we will limit discussion to the theoretical considerations when implementing it since it's actually pretty neat (I think).

Appendix X

Copy and paste hard data example here

We will illustrate use of the updated `bdots` package with a worked example, using an artificial dataset to help detail some of the newer aspects of the package. The dataset will consist of outcomes for a collection

Origin	Class	Color
foreign	car	red
		blue
	truck	red
		blue
domestic	car	red
		blue
	truck	red
		blue

Table 2: table of stuff

of vehicles, consisting of eight distinct groups. These groups will be nested in order of vehicle origin (foreign or domesetic), vehicle class (car or truck), and vehicle color (red or blue). Further, vehicles of different color but within the same origin and class groups will be considered paired observations. A table detailing the relationship of the groups is shown here:

The outcome here is simply `y` due to a lack of creativity, but the functional form assumed (and used in data generation) follows the four parameter logistic,

$$f_\theta(t) = b + \frac{p-b}{1+\exp\left(\frac{4s}{p-b}(x-t)\right)}, \tag{9}$$

where b, p, s, and x represent the baseline, peak, slope, and crossover points, respectively

The formula argument serves two functions in `bboot`: first, it specifies the collection of curves we wish to investigate the difference between, and second, it determines if we are interested in directly comparing the differences or the difference of differences between curves.

To begin, let's reintroduce the structure of the groups we have in our dataset. Recall that we have foreign and domestic cars and trucks, and each of these vehicles comes in red and blue. Recall also that the different colors of each vehicle are considered paired observations.

Beginning with a simple case, suppose we want to investigate the difference in outcome between foreign and domestic vehicles. Notionally, we would write

`y ∼ Origin(foreign, domestic).`

Note that this involves the grouping variable, `Origin`, with the two values we are interested in comparing, `domestic` and `foreign`. With this specification, the distribution of functions considered in `domestic` include all red and blue domestic cars and trucks.

If we wanted to limit our investigation to only foreign and domestic *trucks*, we would do this by including an extra term specifying the group and the desired value. In this case,

```
y ~ Origin(foreign, domestic) + Class(truck).
```

To compare only foreign and domestic *red* trucks, we would add an additional term for color:

```
y ~ Origin(foreign, domestic) + Class(truck) + Color(red).
```

There are also instances in which we might be considered in the interaction of two groups. Although there is no native way to handle interactions in `bdots`, this can be done indirectly through the difference of differences (McMurray et al 2019, though truthfully I still don't understand why). To illustrate, suppose we are interested in understanding how the color of the vehicle differentially impacts outcome based on the vehicle class. In such a case, we might look at the difference in outcome between red cars and red trucks, and then again the difference between blue cars and blue trucks. Any difference between these two differences would give information regarding the differential impact of color between each of the two classes. This is done in `bdots` using the `diffs` synatx in the formula:

```
diffs(y, Class(car, truck)) ~ Color(red, blue)
```

Here, the *outcome* that we are considering is the difference between vehicle classes, with the outcome of interest being color. This is helpful in remembering which term goes on the LHS of the formula.

Similar as to the case before, if we wanted to limit this difference of differences investigation to only include domestic vehicles, we can do so by including an additional term:

```
diffs(y, Class(car, truck)) ~ Color(red, blue) + Origin(domestic).
```

The formula syntax was originally contrived to make comparisons within groups or within nested groups. Conceivably, however, one could be interested in making the comparison between domestic red trucks and foreign blue cars. Doing so requires a bit of a work around. Examples detailing how one might go about doing this are included in appendix B.

Appendix B - Fitting non-nested groups

(currently just copy pasted from the body of document, not edited so no need to really review)

First, there would be some function of sorts, something like `makeUniqueGroups` which would create a new group column with each permutation of previous groups being given a unique identifier. Doing this on the vehicle example would look something like `fit <- makeuniquewhatever` resulting in the following grouping structure (for example) (and maybe you could specify group name and values who knows, kinda like factor this is just a working thought example)

Origin	Class	Color	bgroup
foreign	car	red	A
		blue	B
	truck	red	C
		blue	D
domestic	car	red	E
		blue	F
	truck	red	G
		blue	H

To then investigate differences in outcome between a foreign red car and a domestic blue truck would simply then be

```
y ~ bgroup(A, H)
```

yeah not like sexy or anything but whatever it would work.

.1 Appendix 2

```
> logistic
function (dat, y, time, params = NULL, ...)
{
    logisticPars <- function(dat, y, time, ...) {
        time <- dat[[time]]
        y <- dat[[y]]
        if (var(y) == 0) {
            return(NULL)
        }
        mini <- min(y)
        peak <- max(y)
        r <- (peak - mini)
        cross <- time[which.min(abs(0.5 * r - y))]
        q75 <- 0.75 * r + mini
        q25 <- 0.25 * r + mini
        time75 <- time[which.min(abs(q75 - y))]
        time25 <- time[which.min(abs(q25 - y))]
        tr <- time75 - time25
        tr <- ifelse(tr == 0, median(time), tr)
        slope <- (q75 - q25)/tr
        return(c(mini = mini, peak = peak, slope = slope, cross = cross))
    }
    if (is.null(params)) {
        params <- logisticPars(dat, y, time)
    }
    else {
        if (length(params) != 4)
            stop("logistic requires 4 parameters be specified for refitting")
        if (!all(names(params) %in% c("mini", "peak", "slope",
            "cross"))) {
            stop("logistic parameters for refitting must be correctly labeled")
        }
    }
    if (is.null(params)) {
        return(NULL)
    }
    y <- str2lang(y)
    time <- str2lang(time)
    ff <- bquote(.(y) ~ mini + (peak - mini)/(1 + exp(4 * slope *
        (cross - (.(time)))/(peak - mini))))
    attr(ff, "parnames") <- names(params)
    return(list(formula = ff, params = params))
}
<bytecode: 0x5591d0f862c0>
<environment: namespace:bdots>
```

```
function (dat, y, time, params = NULL, startSamp = 8, ...) {
    logisticPars <- function(dat, y, time, startSamp, ...) {
        time <- dat[[time]]
        y <- dat[[y]]
        if (var(y) == 0) {
            return(NULL)
        }
        spars <- data.table(param = c("mini", "peak", "slope", "cross"),
                    mean = c(0.115, 0.885, 0.0016, 765),
                    sd = c(0.12, 0.12, 0.00075, 85),
                    min = c(0, 0.5, 0.0009, 300),
                    max = c(0.3, 1, 0.01, 1100))
        fn <- function(p, t) {
            p[1] + (p[2] - p[1])/(1 + exp(4 * p[3] * ((p[4] - t)/(p[2] - p[1]))))
        }
        tryPars <- vector("list", length = startSamp)
        for (i in seq_len(startSamp)) {
            maxFix <- Inf
            while (maxFix > 1 | maxFix < 0.6) {
                tryPars[[i]] <- Inf
                while (any(spars[, tryPars[[i]] <= min | tryPars[[i]] >= max])) {
                  tryPars[[i]] <- spars[, rnorm(length(tryPars[[i]])) * sd + mean]
                }
                maxFix <- max(fn(tryPars[[i]], time))
            }
        }
        r2 <- vector("numeric", length = startSamp)
        for (i in seq_len(startSamp)) {
            yhat <- fn(tryPars[[i]], time)
            r2[i] <- mean((y - yhat)^2)
        }
        finalPars <- tryPars[[which.min(r2)]]
        names(finalPars) <- c("mini", "peak", "slope", "cross")
        return(finalPars)
    }
    if (is.null(params)) {
        params <- logisticPars(dat, y, time, startSamp)
    }
    # Was var(y) == 0?
    if (is.null(params)) {
        return(NULL)
    }
    y <- str2lang(y)
    time <- str2lang(time)
    ff <- bquote(.(y) ~ mini + (peak - mini)/(1 + exp(4 * slope *
        (cross - (.(time)))/(peak - mini))))
    attr(ff, "parnames") <- names(params)
    return(list(formula = ff, params = params))
}
<bytecode: 0x5591d39da1f0>
<environment: namespace:bdots>
```

Appendix A

onset

Abstract

In 1995 the Visual World Paradigm (VWP) was introduced blah blah (can I have citations in abstract or should it be more general?)

A.1 Introduction

Spoken words create analog signals that are processed by the brain in real time. That is, as spoken word unfolds, a collection of candidate words are considered until the target word is recognized. The degree to which a particular candidate word is being considered is known as activation. An important part of this process involves not only correctly identifying the word but also eliminating competitors. For example, we might consider a discrete unfolding of the word "elephant" as "el-e-phant". At the onset of "el", a listener may activate a cohort of potential resolutions such as "elephant", "electricity", or "elder", all of which may be considered competitors. With the subsequent "el-e", words consistent with the received signal, such as "elephant" and "electricity" remain active competitors, while incompatible words, such as "elder", are eliminated. Such is a rough description of this process, continuing until the ambiguity is resolved and a single word remains.

[start more broadly, there are a number of ways to do this, we use activation]Our interest is in measuring the degree of activation of a target, relative to competitors. Activation, however, is not measured directly, and we instead rely on what can be observed with eye-tracking data, collected in the context of the Visual World Paradigm (VWP) (Tannenhaus 1995)[11]. In the last few years, researchers have begun to reexamine some of the underlying assumptions associated with the VWP, calling into question the validity or interpretation of current methods. We present here a brief history of word recognition in the context of the VWP, along with an examination of contemporary concerns. We address some of these concerns directly, presenting an alternate method for relating eye-tracking data to lexical activation.

This section needs work but it mostly covers the gist of what I am trying to convey, namely we are about to go from history $\rightarrow$ current state of the world $\rightarrow$ proposal and comparison $\rightarrow$ results.

A.2 A brief history

We begin with a brief history to give context to later discussion. In particular, we will consider one of the leading theoretical models in speech perception, TRACE, followed by the introduction of the leading experimental paradigm, the VWP. We examine empirical evidence for the relation between these, and relevant theoretical advancements that have been made. Topics here are presented only briefly and limited to those directly relevant to the present work. For a fuller discussion of the history and uses of VWP, use google. (Or Huettig 2011b?)

An outline of the presentation (for internal use only):

1. VWP by Tannenhaus 1995 [11]

2. VWP + TRACE, Allopenna 1996 (trace aspect no longer relevant, just an aside) [1]
3. As far as I can tell, it's Bob's 2010 paper that was among first to [6]
   (a) Look at individual differences in word recognition (not counting the ortho polynomial fits) (also relevant for the "group distribution of curves" hypothesis) and
   (b) Introduce parametric forms to be fit to the data (the assumption we continue to run with), or at very least, introduce ones that are interpretable

All of the paragraphs in this section are narrative and not mission critical. Need to be fleshed out

Visual World Paradigm The Visual World Paradigm (VWP) was first introduced in 1995, making the initial link between the mental processes associated with language comprehension and eye movements [11]. A typical experiment in the VWP involves situating a subject in front of a "visual world", commonly a computer screen today, and asking them to identify and select an object corresponding to a spoken word. The initiation of eye movements and subsequent fixations are recorded as this process unfolds, with the location of the participants' eyes serving as a proxy for which words or images are being considered. This association was first demonstrated by comparing how the mean time to initiate an eye movement to the correct object was mediated by the presence of phonological competitors ("candy" and "candle", sharing auditory signal at word onset) and situations containing syntactic ambiguity ("Put the apple on the towel in the box" and "Put the apple *that's* on the towel in the box" in ambiguous scenarios with one or more apples). It is by comparing the trajectory of these mechanics across trials in the presence of auditory or semantic competitors that researchers have used the VWP in their investigation of spoken word recognition.

["We find that eye movements to objects in the workspace are closely time-locked to referring expression s in the unfolding speech stream, providing a sensitive and nondistruptive measure of spoken language comprehension during continuous speech" [1]]

Proportion of fixation It was against simulated TRACE data that Allopenna (1998) found a tractable way of analyzing eye tracking data. By coding the period of a fixation as a 0 or 1 for each referent and taking the average of fixations towards a referent at each time point, Allopenna was able to create a "fixation proportion" curve that largely reflected the shape and competitive dynamics of word activation suggested by TRACE, both for the target object, as well as competitors. This also served to establish a simple linking hypothesis, specifically, "We made the general assumption that the probability of initiating an eye movement to fixate on a target object o at time t is a direct function of the probability that o is the target given the speech input and where the probability of fixating o is determined by the activation level of its lexical entry relative to the activation of other potential targets." Further of note is what this linking hypothesis does not include, namely:

1. No assumption that scanning patterns in and of themselves reveal underlying cognitive processes
2. No assumption that the fixation location at time t necessarily reveals where attention is directed (only probabilistically related to attention)

Other assumptions included here include that language processing proceeds independent of vision (Magnuson 2019), and that visual objects are not automatically activated. Or, more succinctly, it assumes that fixation proportions over time provide an essentially direct index of lexical activation, whereby the probability of fixating an object increases as the likelihood that it has been referred to increases.

While other linking hypotheses have been presented (Magnuson 2019) [3], that there is *some* link between the function of fixation proportions and activation has guided the last 25 years of VWP research.

Parametric Methods and Individual Curves While there have most certainly been advancements to the use of the VWP for speech perception and recognition (and expanded into related domains, such as sentence processing and characterizing language disorders (according to Bob)), we limit ourselves here to one in particular. In 2010, McMurray et al expanded the domain of the VWP by introducing emphasis on individual differences in participant activation curves. Two aspects of this paper are relevant here. First, although they were not the first to introduce non-linear functions to be fit to observed data, they did introduce a number of important parametric functions in use today, namely the four (or five) parameter logistic and the double-gauss (asymmetrical gauss), the primary benefit being that the parameters of these functions are interpretable, that is, they "describe readily observable properties." Second, which I suppose was also introduced by Mirman (2008) [8] to some degree (though I have not read it yet, just pulling from Bob) is specifying individual subject curves across participants. This has been critical in that:

1. The parameters of the functions describe interpretable properties
2. This made the idea of distributions of parameters for a particular group a relevant construct

Though not stated directly (given it predates bdots by 8 years), this also served as the impetus for investigating group differences in word activation through the use of bootstrapped differences in time series [9] and the subsequent development of the `bdots` software in R for analyzing such differences. (A history of exploring differences in group curves can be found in [10]).

This brings us to the current day, where the state of things is such that TRACE-validated VWP data is widely used to measure word recognition by collecting data on individual subjects and fitting to them non-linear parametric curves with interpretable parameters. Context in hand, we are now able to introduce some of the main characters of our story, specifically how data in the VWP is understood and used.

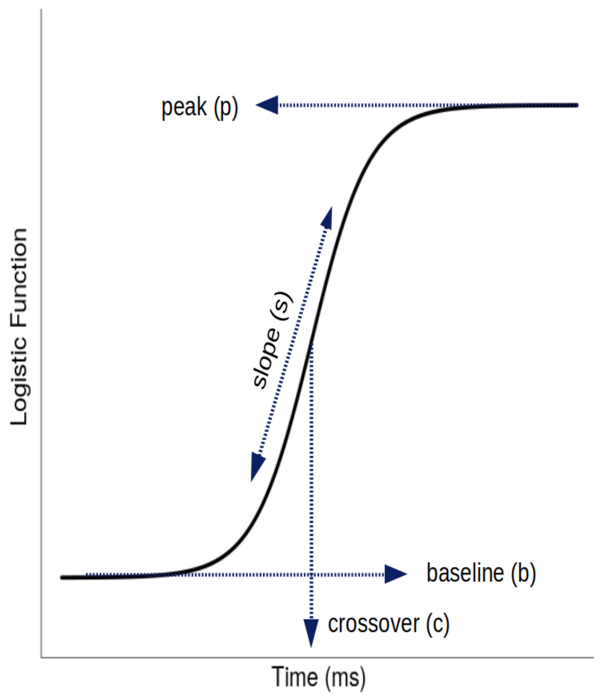


Figure A.1: An illustration of the four-parameter logistic and its associated parameters, introduced as a parametric function for fixations to target objects in McMurray 2010. Can describe the parameters in detail, but should also have the formula itself somewhere to be referenced. (Equation B.10)

A.3 Where we are now

The following section goes into more detail on the specifics

This section includes the finer points of the VWP, eye tracking data, and how allopena's introduction ties in with bob's parametric proposition.

A.3.1 Anatomy of Eye Mechanics

In the context of eye tracking data and word recognition, there are a few mechanics with which we are concerned. The first of these is activation [which i need to learn a little bit more about first]. Even with the immediacy and (fullness? some word they use to describe dense time series here being better than yes/no response), what we observe with any eye movement is not a direct readout of the underlying activation. Rather, there is a period of latency between the decision to launch an eye movement and the physiological response, a period known as oculomotor delay. And finally, there are they physical mechanics of the eye movements themselves, the saccade and the fixation which, together, make up a “look”. We will briefly address each of these in the reverse of the order in which they were introduced.

Saccades and fixations: Rather than acting in a continuous sweeping motion as our perceived vision might suggest, our eyes themselves move about in a series of short, ballistic movements, followed by brief periods of stagnation. These, respectively, are the saccades and fixations.

Saccades are short, ballistic movements lasting between 20ms-60ms, during which time we are effectively blind. Once in motion, saccades are unable to change trajectory from their intended destination. Following this movement is a period known as a fixation, itself made up of a necessary refraction period (during which time the eye is incapable of movement) followed by a period of voluntary fixation which may include planning time for deciding the destination of the next eye movement; the duration of fixations are typically (some length). It will be convenient to follow previous convention and consider a saccade followed by its adjacent fixation as a single concept called a “look” [7]. We take particular care here to note that the beginning of a look, or “look onset”, starts the instance that a previous look ends or, said another way, the instant an eye movement is launched. A visual description of these is provided in Figure A.2.

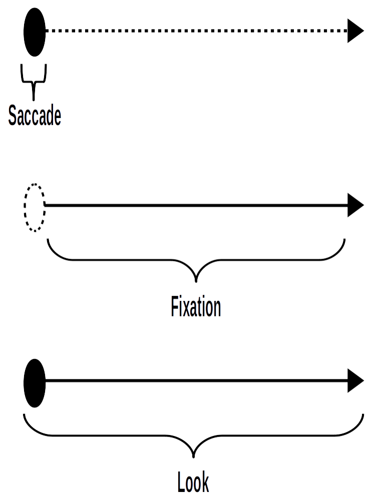


Figure A.2: redo this image to match anataomy of look image, also for size

Oculomotor delay: While the physiological responses are what we can measure, they are not themselves what we are interested in. Rather, we are interested in determining word activation, itself governing the cognitive mechanism facilitating movements in the eye. Between the decision to launch an eye movement (a cognitive mechanism governed by the activation, next section) and the movement itself is a period known as oculmotor delay. It is typically estimated to take around 200ms to plan and launch an eye movement, and this is usually accounted for by subtracting 200ms from any observed behavior. [12]. As oculomotor delay

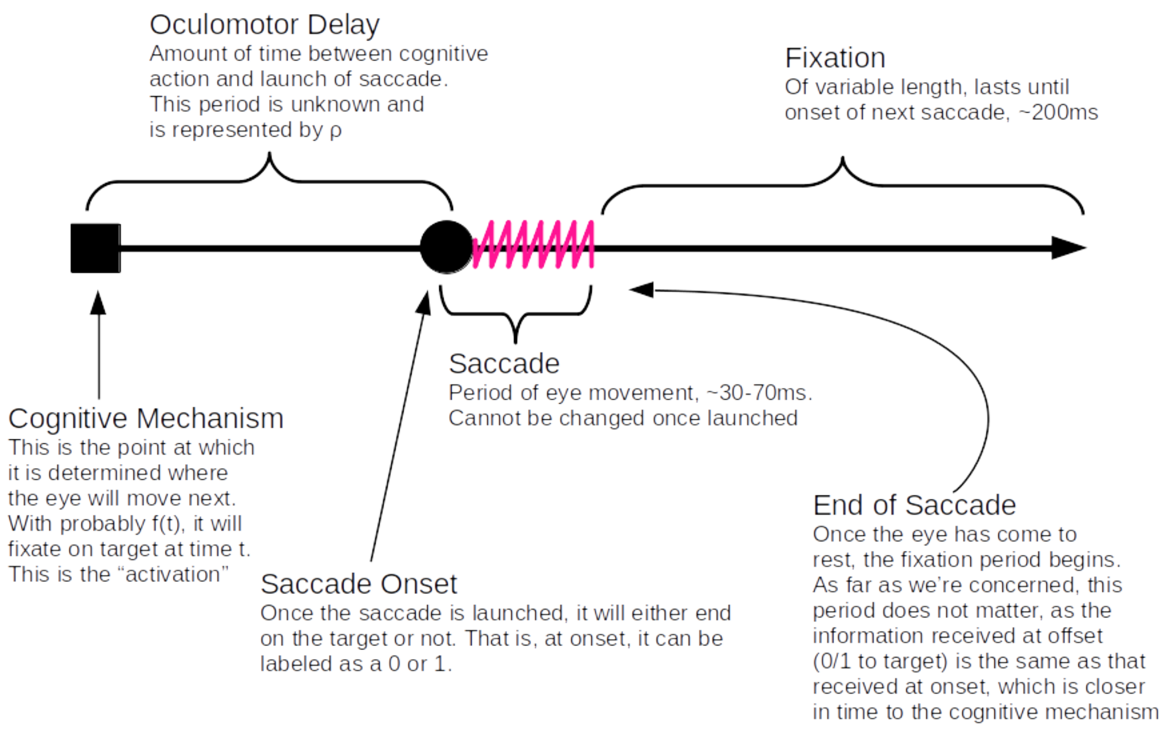

Oculomotor Delay
Amount of time between cognitive action and launch of saccade. This period is unknown and is represented by ρ
Fixation
Of variable length, lasts until onset of next saccade, ~200ms
Saccade
Period of eye movement, ~30-70ms. Cannot be changed once launched
Cognitive Mechanism
This is the point at which it is determined where the eye will move next. With probably f(t), it will fixate on target at time t. This is the "activation"
Saccade Onset
Once the saccade is launched, it will either end on the target or not. That is, at onset, it can be labeled as a 0 or 1.
End of Saccade
Once the eye has come to rest, the fixation period begins. As far as we're concerned, this period does not matter, as the information received at offset (0/1 to target) is the same as that received at onset, which is closer in time to the cognitive mechanism

Figure A.4: I want to do this figure again but differently. have saccade be two bars matching anatomy of look, include refractory period of fixation, noting that that and saccade are identical, followed by period of time of voluntary fixation (theoretically relevant) followed by next CM

is only roughly estimated to be around 200ms, we suggest that accounting for randomness will be critical in correctly recovering the the cognitive mechanism of interest or at very least in identifying possible sources of bias. How this phenomenon relates to saccades and fixations is demonstrated in Figure A.3.

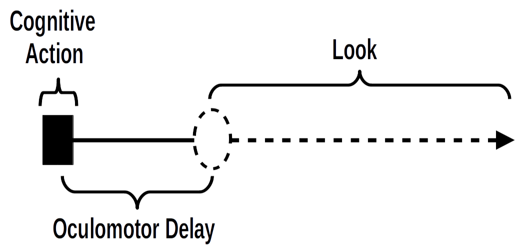


Figure A.3: redo this image

A.3.2 Activation

Here tie in idea of activation, though need to be more concise about what we mean than we are currently. Good source for framework being (McClelland and Rumelhard 1981? Rumelhart and McClelland 81 and 82, and mcclelland/elman 86 with trace). They seem to all mention the "interactive activation framework" which may be worthwhile to elaborate on further. For now, assume that we have adequately stated *what it is*.

While a number of experimental methods are used as real-time indices of lexical access (Spivey mouse trials), we concern ourselves here with the use of eyetracking as it relates to activation as first suggested by [1]. Whereas the initial treatment of eyetracking data made no attempt identify or model subject-specific trends, more recent work has made strides in making subject analysis more tractable. Specifically, we adopt the idea that each participant's results can be fit to non-linear functions who's parameters describe clinically relevant properties [6]. We will denote this activation function f with parameters θ as a function in time, giving $f(t|\theta)$

For example, the four parameter logistic function in Figure A.1 is often used to model fixations to the target object in the VWP with functional form

$$f(t|\theta) = \frac{p-b}{1+\exp\left(\frac{4s}{\mathrm{p}-b}(x-t)\right)} + b. \tag{A.1}$$

Similarly, a six parameter asymmetric Gaussian function,

$$f(t|\theta) = \begin{cases} \exp\left(\frac{(t-\mu)^2}{-2\sigma_1^2}\right)(p-b_1)+b_1 & \text{if } t \leq \mu \\ \exp\left(\frac{(t-\mu)^2}{-2\sigma_2^2}\right)(p-b_2)+b_2 & \text{if } t > \mu \end{cases} \tag{A.2}$$

(I didn't make a nice graph/label for this).

While both functions are commonly used in the VWP for modeling eye fixations, for simplicity we will limit the primary focus of our discussion, though ultimately our argument is agnostic to the modeling function used, parametric or otherwise. Discussion related to the asymmetric Gauss is treated in the appendix.

A.3.3 VWP data

We now consider how the aforementioned mechanics relate to the visual world paradigm. In a typical instantiation of the VWP, a participant is asked to complete a series of trials, during each of which they are presented with a number of competing images on screen (typically four). A verbal cue is given, and the participants are asked to select the image corresponding to the spoken word. All the while, participants are wearing (generally) a head-mounted eye tracking system recording where on screen they were fixated.

An individual trial of the VWP may be short, lasting anywhere from 1000ms to 2500ms before the correct image is selected. Prior to selecting the correct image, the participant's eyes scan the environment, considering images as potential candidates to the spoken word. As this process unfolds, a snapshot of the eye is taken at a series of discrete steps (typically every 4ms) indicating where on the screen the participant is fixated. A single trial of the VWP typically contains no more than four to eight total "looks" before the correct image is clicked, resulting in a paucity of data in any given trial.

To be clear, eye trackers themselves only record x and y coordinates of the eye at any given time, and it is only after the fact that "psychophysical" attributes are mapped onto the data (saccades, fixations, blinks, etc.,). We adopt the strategy of prior work in discussing eye tracking data in terms of their physiological mapping, as this will be crucial in constructing a physiologically relevant understanding of the problem at hand [7].

To create a visual summary of this process aggregated over all of the trials, a la Allopena, a "proportion of fixations" curve is created, aggregating at each discrete time point the average of indicators of whether or not a participant is fixated on a particular image. A resulting curve is created for each of the competing categories (target, cohort, rhyme, unrelated), creating an empirical estimate of the activation curve, $f(t|\theta)$. See Figure A.5. For any subject $i = 1, \ldots, n$, across times $t = 0, \ldots, T$ and trials $j = 1, \ldots, J$, a construction of this curves can be expressed as:

$$y_{it} = \frac{1}{J} \sum z_{ijt} \tag{A.3}$$

where z_{ijt} is an indicator $\{0, 1\}$ in trial j at time t and such that we have an empirical estimate of the activation curve,

$$f(t|\theta_i) \equiv y_{it}. \tag{A.4}$$

For our discussions here, we will call this the proportion of fixation method.

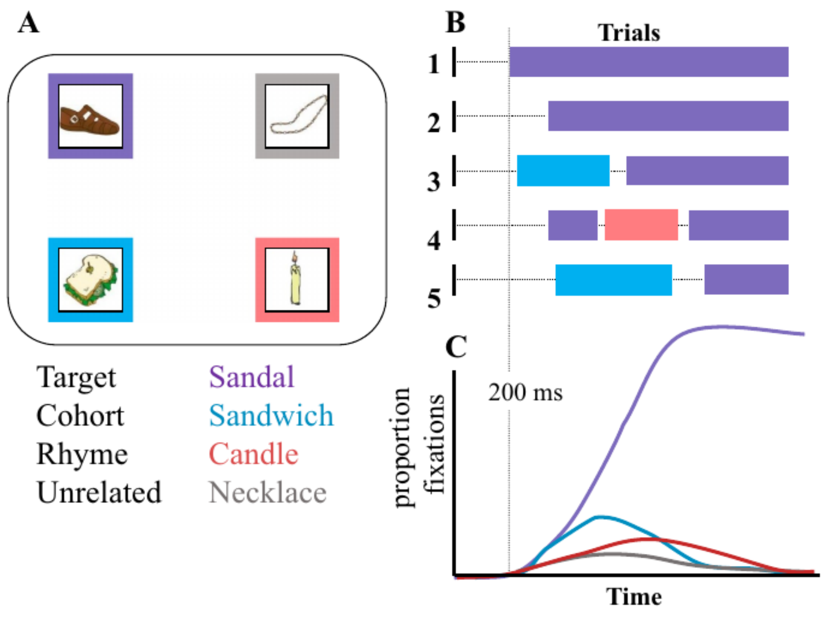


Figure A.5: Stole this from Bob (who apparently stole it from richard aslin), plan on making my own

As each individual trial is only made up of a few ballistic movements, the aggregation across trials allows for these otherwise discrete measurements to more closely represent a continuous curve. Curve fitting methods, such as those employed by `bdots`, are then used to construct estimates of function parameters fitted to this curve.

A.4 Where are we going?

Having given due consideration to the state of things are they are, we find ourselves in a time of moral reflection, reexamining the underlying relationship between lexical activation (the mechanism of interest) and the physiological behavior we are able to observe (here, specifically eye-tracking. This is referred to in the literature as the linking hypothesis. And while there are a number of competing hypothesis, they each share a collection of implicit assumptions relating what is observed to what is being studied [3].

The simplest version of a linking hypothesis in the context of the VWP is the “general assumption that the probability of initiating an eye movement to fixate on a target object o at time t is a direct function of the probability that o is the target given the speech input and where the probability of fixating o is determined by the activation level of its lexical entry relative to the activations of other potential targets (i.e., the other visible objects” [1]. It is from this assumption that we justify the relation in Equation A.4. To a degree, this assumption is shared by most linking hypothesis in that the probabilistic nature of the proportions of fixations is assumed to be related in time to the strength of the underlying activation. Primary differences in linking hypotheses tend to revolve around the particulars of the mechanics involved, including the duration of fixations, eye scanning behavior, the impacts of priming, or the relation between visual processing acting in conjunction with lexical activation.

We consider a particular meta contribution to this debate presented by McMurray in which he probed the relationship between the observed dynamics in the fixations and the underlying dynamics of activation under a variety of assumptions [5]. In short, he showed that curves reconstructed using the standard proportion of fixations analysis in the VWP were poor estimates of the underlying system, with the magnitude of bias increasing on the complexity of the mechanisms involved. Though this made no specific claims as to what the underlying mechanics may be, it did demonstrate the inherent difficulty in relating observable behavior to the underlying cognitive process.

An important contribution made there, however, and one that we adopt here is an explicit definition of the underlying activation function. Given the relation in Equation A.4, it is reasonable to assume that the underlying activation of any of the objects with the VWP could be modeled with a nonlinear function $f(t|\theta)$. The goal of a VWP analysis, then, is the recovery of this underlying function.

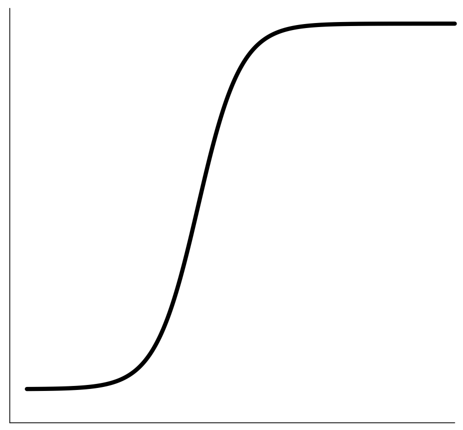
From this assumption we propose an alternative model of the relation between the underlying activation and the observed behavior, with a careful delineation of the psycho-physical components of a look in conjunction with its generating behavior. In particular, we consider the cognitive mechanism associated with initiating an eye movement, which is probabilistically associated with lexical activation, the delay between this and the onset of its associated look, and finally how the different components of the look are related to fundamentally different mechanisms. From this and what we ultimately argue is that observed bias in the recovery of the activation curve under the proportion of fixations method can be partitioned into two distinct components:

[i would like to maybe go into more detail here or have a picture idk]

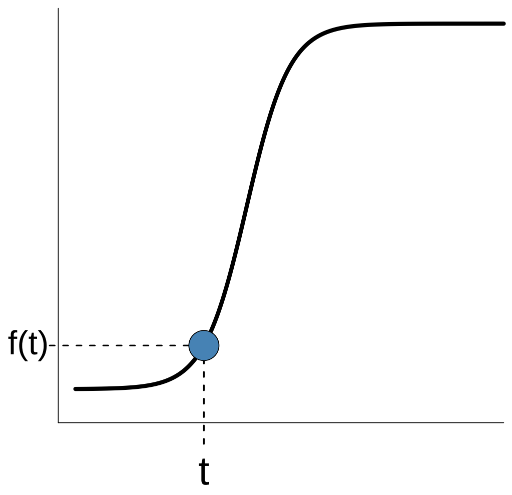
1. The first source of bias, which is the primary emphasis of my proposal, is what I call the "added observation" bias. This involves the fact that in a standard analysis of VWP data is, the entire duration of a fixation is indicated with a $\{0, 1\}$ at any time, t, without having observed any behavior associated with the initiation of an eye movement at that time.

2. The second source of bias is "delayed observation bias". This bias arises from the fact that an eye movement launched at some time t was planned at some time prior. This is primarily a consequence of the delay

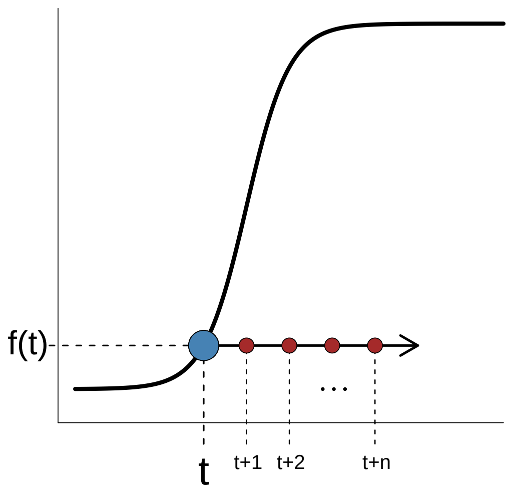
The first source of bias, the "added observation" bias, arises singularly from the fact that the destination of a look, which is observed at look onset, has a fundamentally *different* generating mechanism than what determines the duration of a look, never minding such mechanics as the duration of a saccade or the refractory period of a fixation. Nonetheless, a standard analysis of VWP data does not differentiate between the initial onset and the period of subsequent fixation; both are recorded as either 0 or 1 according to it's location. A look onset at time t is probabilistically determined by by its lexical activation $f(t|\theta)$ whereas the period of fixation is governed by a separate mechanism altogether. Treating the subsequent fixation as indistinguishable as the effect of not only "adding" observations to the data, but adding observations that necessarily biased. The result is a distorted estimation of the underlying activation. A depiction of this phenomenon is given in Figure A.6.



(a)



(b)



(c)

Figure A.6: **(a.)** Example of a nonlinear activation curve $f(t|\theta)$ **(b.)** At some time, t, a saccade is launched with its destination probabilistically determined by $f(t|\theta)$ **(c.)** For a look persisting over n time points, $t+1,\ldots,t+n$, we are recording "observed" data, adding to the proportion of fixations at each time but without having gathered any additional observed data at $f(t+1|\theta),\ldots,f(t+n|\theta)$, thus inflating (or in the case of a monotonically increasing function like the logistic, deflating) the true probability.

The second source of bias is the "delayed observation" bias. It is well established in the literature that the time it takes to plan and launch a saccade is around 200ms [12], which is typically accounted for by subtracting 200ms off of the observed data. There are two aspects of this that are worth considering further.

First, if the mean duration of this oculomotor delay is not 200ms, bias will be observed as the difference between the true time and the 200ms adjustment. And although not bias in the technical sense, there has been no accounting for what effect randomness in this delay has on the recovery of the underlying activation. It will be worthwhile in investigating this as the potential magnitude will determine if this delay is worth considering in any more detail in future research.

—

While we present no immediate solution to the effects of randomness in the delayed observation bias, we argue that the added observation bias can be rectified by using *only* the times observed with look onset in the recovery of the underlying dynamics. We call this the the "look onset" method, which we explain in more detail.

Look Onset Method: The look onset method differ in the proportion of fixation method only in determining which observed data should be considered relevant in the estimation of lexical activation. A particularly compelling argument to made in favor of the look onset method, a corollary of the added observation bias, is that it has a readily defensible mathematical description description. A saccade launched at time t (marking the onset of a look) is assumed to be probabilistically determined by its lexical activation (relative to competitors) at time t, giving us

$$s_t \sim Bin(f(t|\theta)) \tag{A.5}$$

(it may be that l_t for look onset is better notation, but my concern is that it doesn't capture the "onset" nature that we are concerned with and may instead suggest the entire saccade + fixation).

The utility of this is evident when tasked with stating the distribution of y_t in Equation A.3 as it relates to $f(t|\theta)$, where given the overlap of fixations within a particular trial, it is unclear what relation y_t may have to y_{t+1}.

Two further comments are made about this method here. First, in anticipation of the observation that the look onset method discards relevant information regarding the strength of activation (VOT studies, others from Magnuson review), we acknowledge this and reserve further comment for the discussion. Second given the difference in structure of the observed data, we confirm that the current iteration of `bdots` is capable of fitting nonlinear curves to data both under the proportion of fixation and look onset methods.

A.5 Simulations

Simulations were conducted to replicate the mechanics of a look combined with oculomotor delay, detailed in Figure A.7. This section only address Target fixations with a four parameter logistic as given in Equation B.10; simulations according to looks to competitors is treated in the appendix. We will begin by describing the process of simulating a single subject.

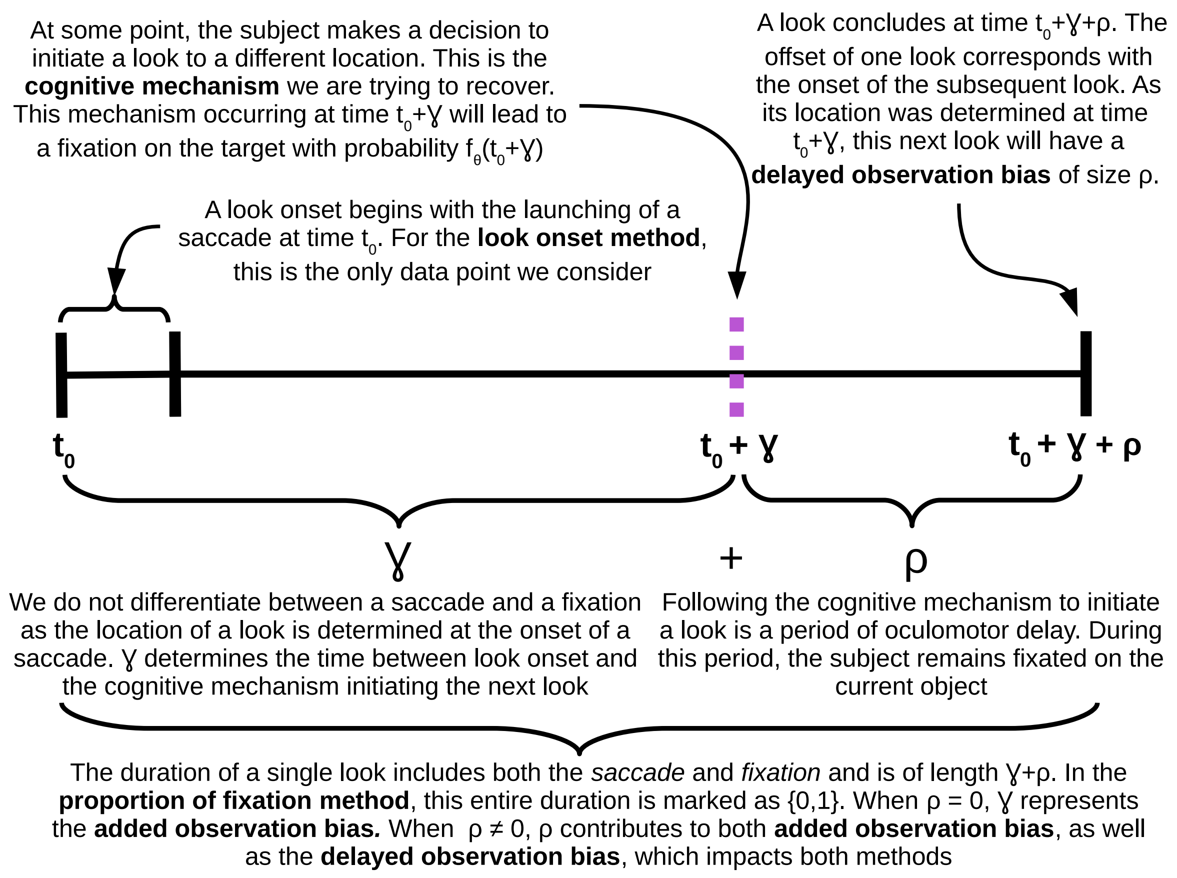

At some point, the subject makes a decision to initiate a look to a different location. This is the **cognitive mechanism** we are trying to recover. This mechanism occurring at time t_0+ɣ will lead to a fixation on the target with probability $f_\theta(t_0$+ɣ)
A look concludes at time t_0+ɣ+ρ. The offset of one look corresponds with the onset of the subsequent look. As its location was determined at time t_0+ɣ, this next look will have a **delayed observation bias** of size ρ.
A look onset begins with the launching of a saccade at time t_0. For the **look onset method**, this is the only data point we consider
t_0
t_0 + ɣ
t_0 + ɣ + ρ
ɣ
+
ρ
We do not differentiate between a saccade and a fixation as the location of a look is determined at the onset of a saccade. ɣ determines the time between look onset and the cognitive mechanism initiating the next look
Following the cognitive mechanism to initiate a look is a period of oculomotor delay. During this period, the subject remains fixated on the current object
The duration of a single look includes both the *saccade* and *fixation* and is of length ɣ+ρ. In the **proportion of fixation method**, this entire duration is marked as {0,1}. When ρ = 0, ɣ represents the **added observation bias.** When ρ ≠ 0, ρ contributes to both **added observation bias**, as well as the **delayed observation bias**, which impacts both methods

Figure A.7: Anatomy of a look – a key thing to discuss somewhere is the OM delay, refractory period, and planning time. The latter two go in γ. Worth noting also that while we do need to be able to control for ρ, *information* regarding strength of consideration will be in γ - refractory period

First, each subject randomly draws a set of parameters θ_i from an empirically determined distribution based on normal hearing participants in the VWP [2] to construct a subject specific generating curve, $f(t|\theta_i)$. It is according to this function that the decision to initiate a look at time t will subsequently direct itself to the Target with probability $f(t|\theta_i)$. We then go about simulating trials according to the following method:

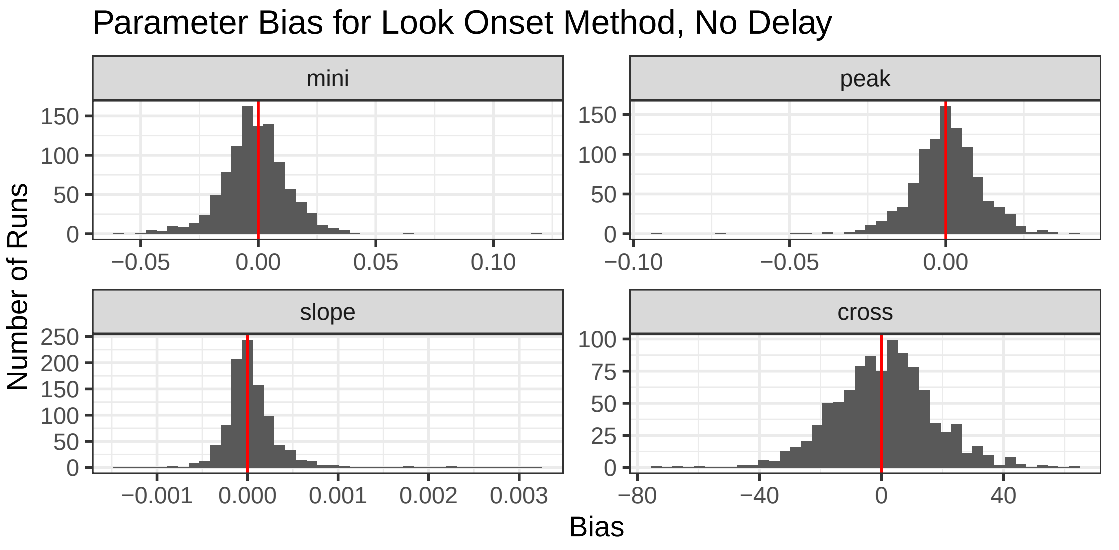
at some time t_0, a subject initiates a look. This look persists for at least a duration of γ, drawn from a gamma distribution with mean and standard deviation independent of time and previous fixations. At time $t_0 + \gamma$, the subject determines the location of its next look, with the next look being directed towards the target with probability $f(t + \gamma|\theta_i)$. The decision to initiate a look is followed by a period of oculomotor delay, ρ, during which time the subject remains fixated in the current location. Finally, at time $t_0 + \gamma + \rho$, the subject ends the look initiated at t_0 and immediately begins its second look to the location determined at time $t_0 + \gamma$. For the look onset method, the only data recorded are the times of a look onset and their location: in this case, at times t_0 and $t_0 + \gamma + \rho$. By contrast, the proportion of fixation method records the object of fixation at 4ms intervals for the entire period of length $\gamma + \rho$. A single trial begins at $t = 0$ and continues constructing looks as described until the total duration of looks exceeds 2000ms. Each subject undergoes 300 trials, and 1,000 subjects are included in each simulation.

Three total simulations were performed to investigate the biases identified in the previous section, each differing only in the random distribution of the oculomotor delay parameter, ρ. In the first simulation, we set $\rho = 0$ to remove any oculomotor delay. In this scenario, a look initiated at time t by subject i will be directed towards the target with probability $f(t|\theta_i)$. Doing so removes any potential bias from delayed observation and allows us to identify the effects of the added observation bias in isolation. In the remaining simulations we probe the effects of randomness in oculomotor delay, investigating what effect uncertainty may have in our recovery of the generating function. We did this assigning ρ to follow either a normal or Weibull distribution, each with a mean value of 200ms. As is standard in a VWP analysis, we subtracted 200ms from each observed point prior to fitting the data. Note that a consequence of this is that in these simulations, the bias itself is accurately accounted for by subtracting the correct mean, with the resulting error in the curve fitting process the result of the inherent variability. This does not detract from the argument being made, however, and any true bias in the mean of the oculomotor delay would asymptotically result in a horizontal shift of the observed data according to the direction and magnitude of the bias.

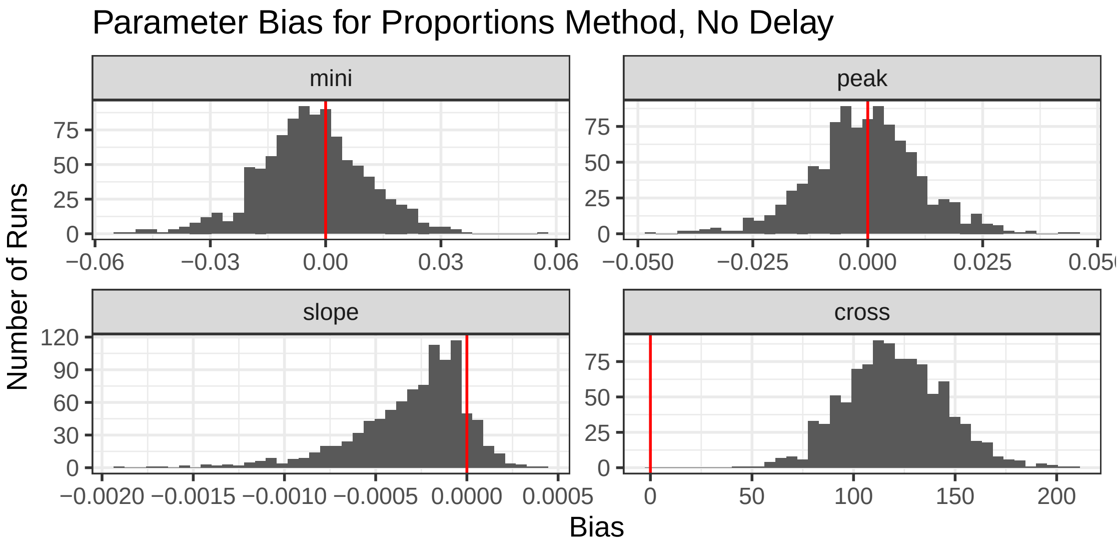
As all of the data could not be individually inspected prior to being included in the analysis, subjects were excluded from consideration if fitted parameters from either the look onset method or the proportion of fixation method resulted in a peak less than the slope, or if the crossover or slope were negative. In the settings in which there was no delay, normally distributed delay, or Weibull distributed delay, 981, 973, and 981 subjects were retained, respectively.

The simulations are performed in R, with the simulation code available on the author's Github page (link?). Simulated data was fit to the four parameter logistic function using `bdots v2.0.0`.

A.5.1 No Delay



(a)



(b)

Figure A.8: Parameter bias for no oculmotor delay.

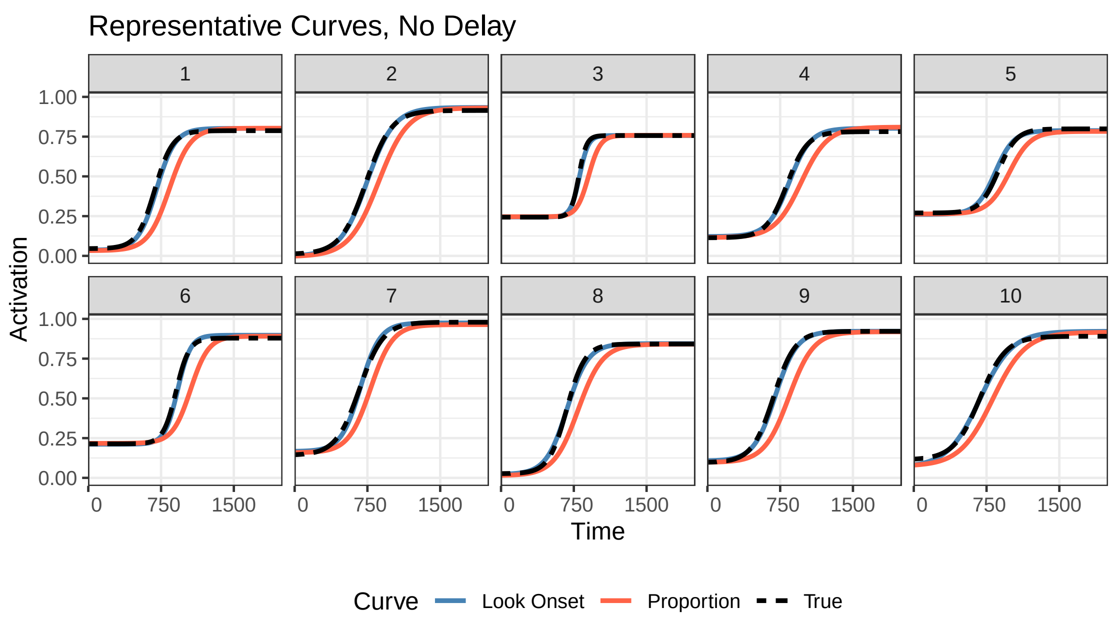
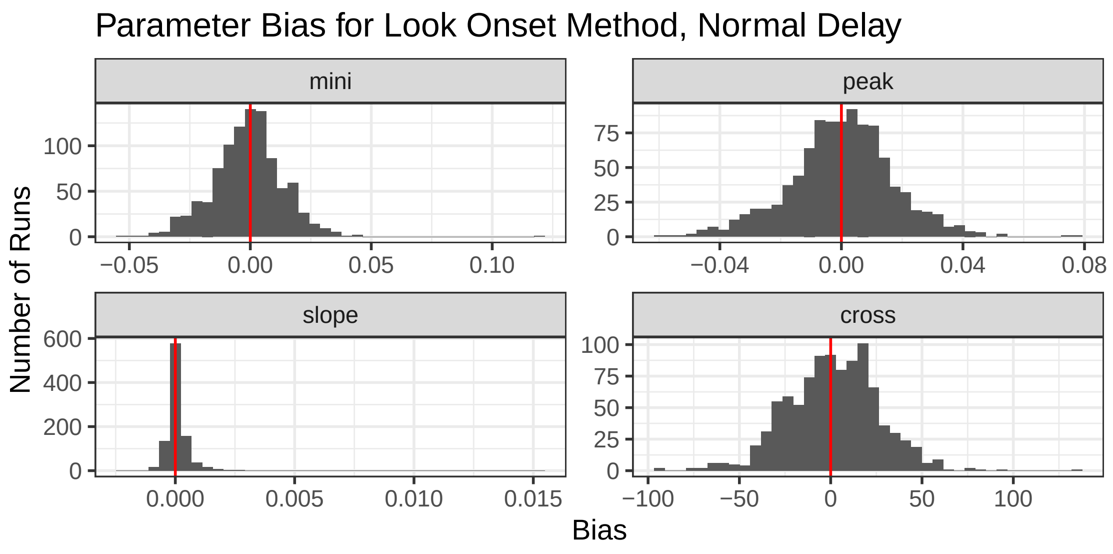
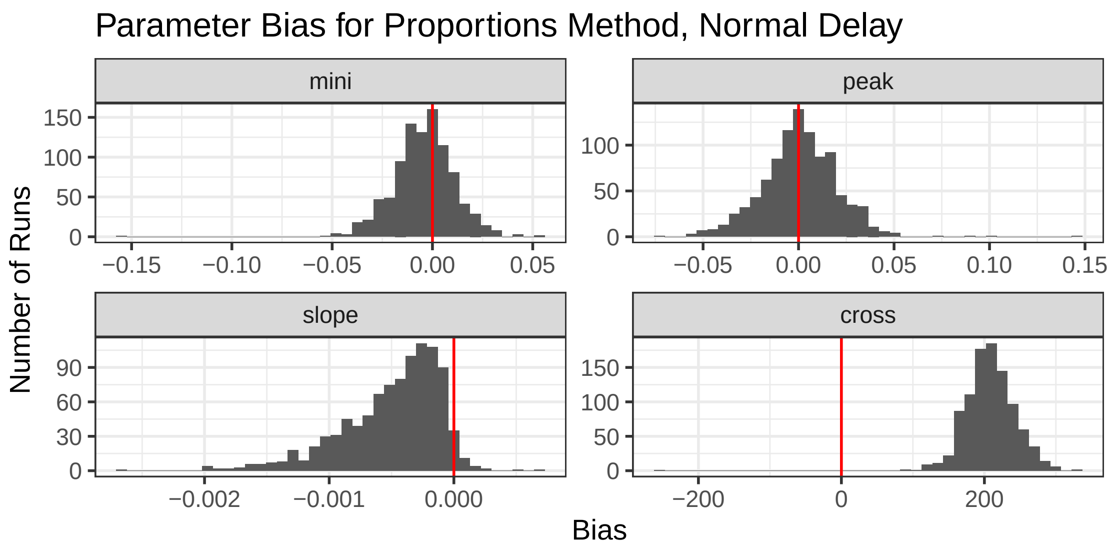


Figure A.9: Representative curves for no oculmotor delay

A.5.2 Normal Delay



(a)



(b)

Figure A.10: Parameter bias for normal OM delay

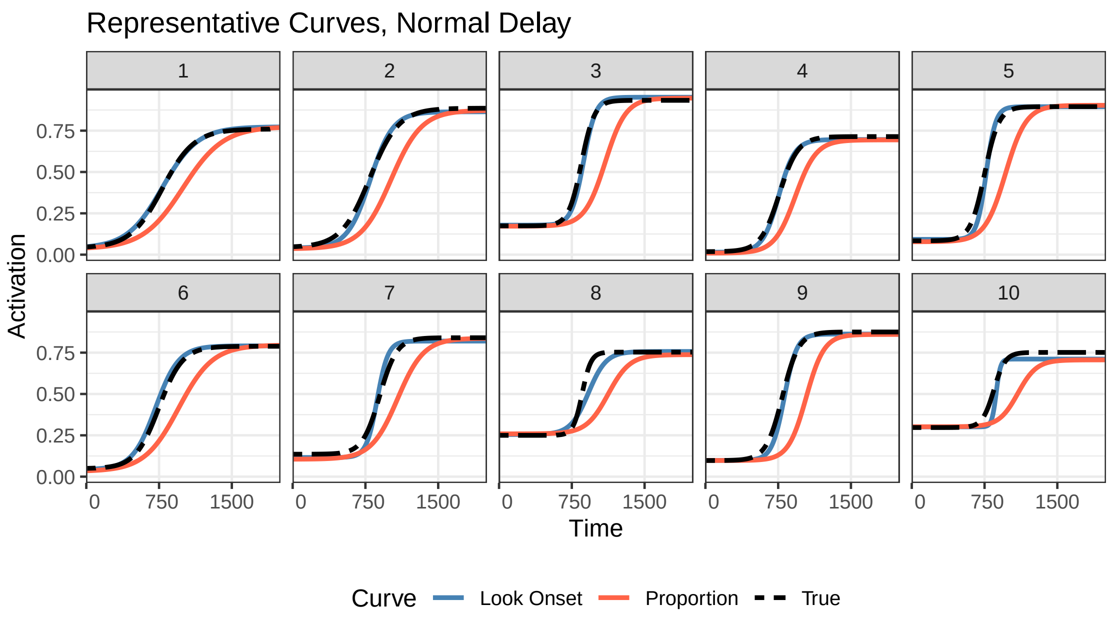
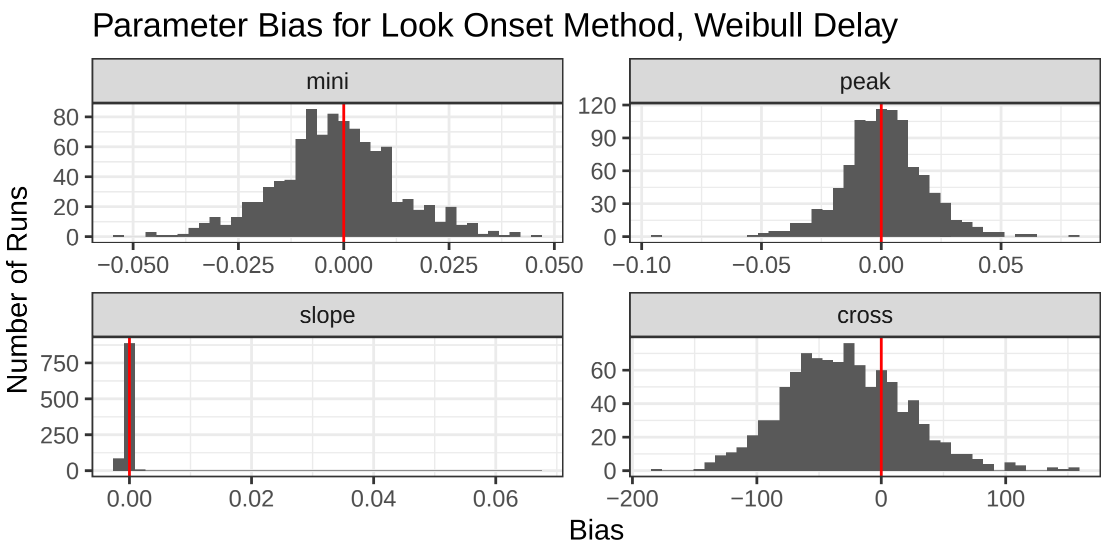
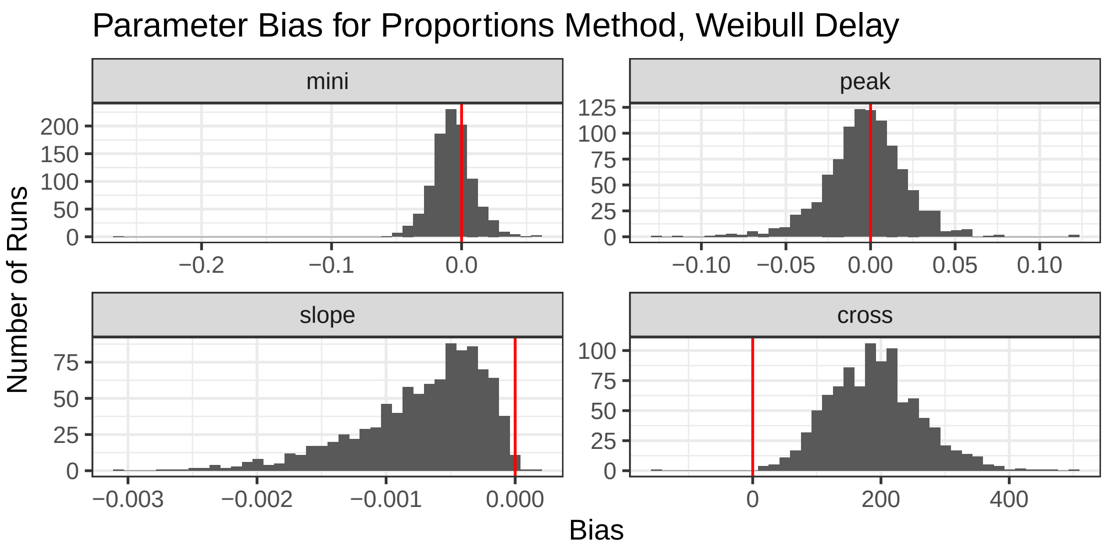


Figure A.11: Representative curves for normal oculomotor delay

A.5.3 Weibull Delay



(a)



(b)

Figure A.12: Parameter bias for weibull OM delay

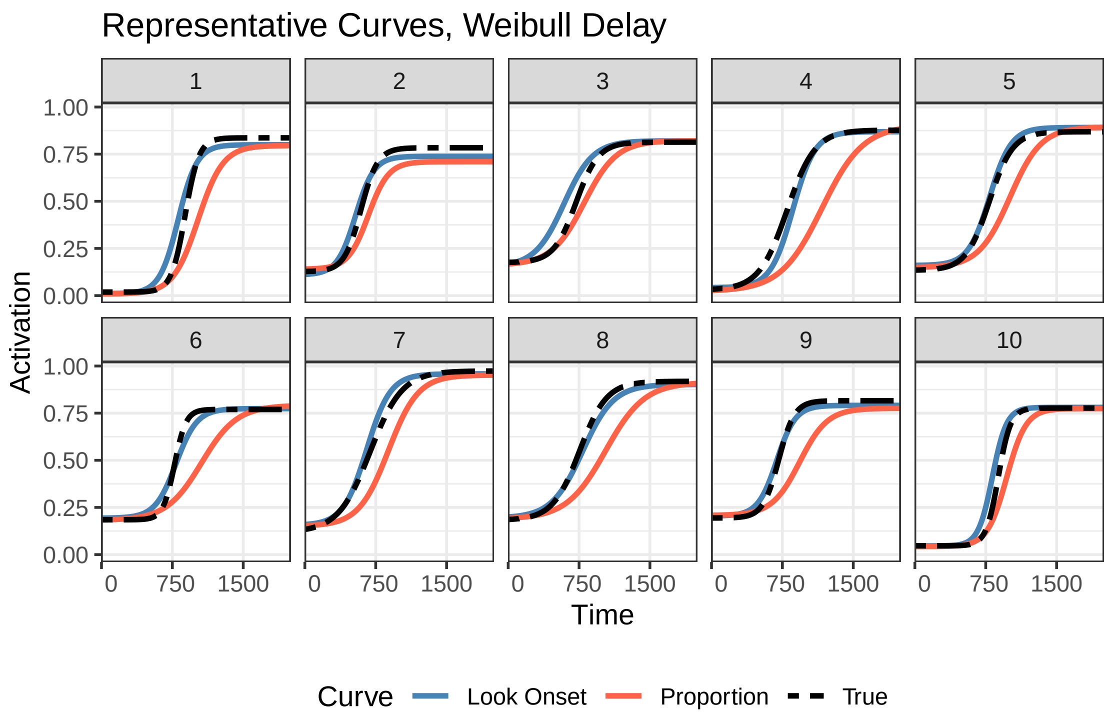


Figure A.13: Representative curves for weibull oculmotor delay

A.5.4 Results

Curve	Delay	1st Qu.	Median	3rd Qu.
Look Onset	No Delay	0.17	0.32	0.56
Look Onset	Normal Delay	0.37	0.71	1.24
Look Onset	Weibull Delay	1.05	2.16	4.23
Proportion	No Delay	8.21	11.33	16.01
Proportion	Normal Delay	22.90	30.65	39.37
Proportion	Weibull Delay	15.27	24.75	38.14

Table A.1: Summary of MISE across simulations. I don't think I necessarily need (or want) all of those summary stats (min/max specifically, cleaner without)

Unexplored is *where* the delay occurs also important

some concluding remarks about how terrible the proportion of fixation method is

A.6 Discussion

This is now "the" discussion

This section needs to be tightened and I have said some things elsewhere. Instead, let this be a general collection of thoughts for now.

I would like to speak a little bit more on the concept of "information gathering behavior". One of the primary benefits of the proportion method is that it indirectly captures the duration of fixations, with longer times being associated with stronger activation. This also becomes important when differentiating fixations associated with searching patterns (i.e., what images exist on screen?) against those associated with consideration (is this the image I've just heard?). There seems to be a general consensus also that longer fixations correspond to a stronger degree of activation, but a crucially overlooked aspect of this is the implicit assumption that fixation length and activation share a linear relationship. Specifically, insofar as the construction of the fixation curves is considered, a fixation persisting at 20ms after onset (and well within the refraction period) is considered identical to a fixation persisting at 400ms. More likely it seems this would be more of an exponential relationship, with longer fixations offering increasingly more evidence of lexical activation. By separating saccades and fixations at the mathematical level, we are able to construct far more nuanced models (one proposal, for example, might be weighting the saccades by the length of their subsequent fixation, or perhaps constructing a modified activation curve $f_{\theta(t)}(t)$ whereby the parameters themselves can accelerate based on previous information. But this is neither here nor there).

Speaking to the mathematical treatment, there is a wonderful simplicity in letting the saccades themselves follow a specific distribution, namely

$$s_t \sim Bin(f_\theta(t)) \tag{A.6}$$

or, with random oculomotor delay $\rho(t)$ (which I haven't really elaborated on as a separate mechanism),

$$s_t \sim Bin(f_\theta(t - \rho(t))) \tag{A.7}$$

This is in contrast to the fixation method, where the proportion of fixation curves can be described

$$y_t = \frac{1}{J} \sum z_{jt}. \tag{A.8}$$

Here, is there a clear distribution for what y_t follows? Under independence it may be the sum of binomials, but then what can be said about the relation of y_t to y_{t+1}, given that they may or may not share overlapping fixations from different trials? This is addressed to some degree in Oleson 2017, but this seems more of an ad hoc adjustment to account for this in retrospect. In contrast, the proposed saccade method makes no assumption of trial-level relationship and instead considers all saccades over all trials as binomial samples from the same generating curve in time.

This of course does ignore trial/word/speaker variability, but then perhaps it is time that we shift our language to speaking about a distribution of generating curves for a subject rather than a particular level of activation (note too that this utility is also reflected in the conversation regarding p-values against confidence intervals).

The arguments presented here has hoped to satisfy two goals, agnostic to the linking hypothesis or functions ultimately decided upon. Foremost is the recognition that saccades and fixations are governed by separate mechanisms, and treating them as such allows for fewer assumptions. For example, reconsider again the quote from Allopenna 1996:

"We made the general assumption that the probability of initiating an eye movement ot fixate on a target object o at time t is a direct function of the probability that o is the target given the speech input and where the probability of fixating o is determined by the activation level of its lexical entry relative to the activation of the other potential targets."

Under the saccade method, we omit the entirety of "and where the probability of fixating o is determined by the activation level of its lexical entry relative to the activation of the other potential targets" while still retaining the entirety of the utility in fitting *the same non-linear curves* to less of the data. This decoupling allows the typical time-course utility of the VWP to be used in conjunction with other methods treating aspects of the fixations separately.

Second to this, we have put a name to two important sources of potential bias in recovering generating curves in such a way as to be generalizable beyond the specifics of the assumptions of the simulation (both here and in McMurray 2022). The first, of course, addresses what was just discussed in the decoupling of saccade and fixation data. The utility of the second comes in that it makes no assumptions as to the source of the delayed observation, removing (possibly) unnecessary specifications between oculomotor delay and general mechanics when the goal is to simply recover the generating function. This may be less relevant when the goal of a study is to specifically address the mechanics of decision making (which itself seems to be difficult to pin down).

In short, what we have hoped to accomplish here is not to drastically change the original assumptions presented in Allopenna (1996) and elaborated upon in Magnuson (2019), but rather to qualify them in statistically sound ways. And really, that is pretty much it. Saccade method is neat, works the same way as the proportion of fixation method, has a more justifiable model while reducing assumptions and allowing room for others.

As a not really conclusion, I am sometimes left to wonder to what degree the proportion of fixation method was a "local minimum" is the pursuit of utilizing eye-tracking data. The proportion of fixations created an ostensible curve, prompting McMurray to establish theoretically grounded non-linear functions to model them. These, in turn, where shown to be suitable functions with which to model saccade data over a period of trials. Had saccades lent themselves so naturally to visualizing as the proportion of fixations, perhaps that is where we may have started.

A.7 Discussion

what have we learned?

No new contributions were added to the linking hypothesis, but introduced novel technique for identifying components of look in VWP and making a standard analysis more consistent with the original

Here are really the main takeaways.

1. We are all revisiting question of linking hypothesis
2. In the process of doing so, Bob identified some critical issues, revealing two distinct sources of bias
3. By introducing saccade method, we remove one source of bias and clearly delineate two separate but likely correlated mechanisms
4. This effectively keeps the assumptions from Allopenna and all of the benefits of constructing a function in time for activation, but also allowing room now for fixations to be used separately in a number of ways (length of fixation, latency to look, total fixations, etc.,)

A.8 limitations

probably good idea to keep running list of these all in one place

1. linking hypothesis/cognition curve
2. adding parametric form (necessity for saccade method)
3. oculomotor delay, where to discuss
4. Specific results consequence of values chosen and relationship of γ to ρ (in size, they are already uncorrelated)

A.9 appendices

Here I am just including more or less random sections that either do not have a definite place yet in the main body of the paper, are part of what might be considered future work, or truly are things that belong in the appendix. Presented in no particular order (commented out, input from other tex files)

Appendix B

method

Abstract

Something something high density time series collected to estimate group population curves and compare those for temporal differences while controlling FWER. Original `bdots` made strict assumptions which are likely to not hold in general, resulting in truly disastrous type I error rates. My modify the original 2017 algorithm to introduce an alternative bootstrapping scheme, along with a modified permutation test to examine differences between groups. Our results demonstrate comparable control of FWER and power under the original assumptions while also proving far more robust to divergences in both the mean and error structures of the observed data.

B.1 Introduction

A standard problem in psycholinguistics, and the cognitive sciences in general, is that of statistically analyzing a process unfolding in time. Particularly in the case of comparing a process between experimental groups in time, techniques appropriate for demonstrating showing that differences *exist* offer little towards the goal of identifying *when* they exist when the time windows are not specified in advance, with the area under a curve (AUC) representing an example of this. One may consider instead testing two time series at each point in time, using a method such a cluster based permutation testing [4] in which test-statistics are computed at each time, with adjacent significant tests being combined into single clusters. This is in an effort to address the problem of multiple comparisons and, by extension, control for the family wise error rate: by reducing adjacent test statistics into a single cluster, researchers work to control the FWER by simply reducing the number of tests. More recently has been the bootstrapped differences of time series (BDOTS) [9], using bootstrapped fits of subject-level curves in conjunction with a modified Bonferroni correction to the significance level to control the family wise error rate in the presence of highly correlated test statistics. A version of this was introduced in the R package `bdots` in 2018 [10].

A closer look at the specific conditions under which the original iteration of `bdots` was presented raises concerns, however, involving quite restrictive assumptions that are unlikely to be met in many, if not most, situations. This includes data typically collected in the Visual World Paradigm (VWP), context in which the underlying methodology was first proposed. Specifically, it assumed a homogeneous mean structure within each of the groups being compared, with no between-subject variability to be accounted for. Empirical data collected in a variety of (VWP) contexts can be used to show that such an assumption is unlikely to be true, with the resulting type I error rate being unacceptably high.

What we present instead is two alternatives, accommodating flexibility in two of the assumptions made in the original bdots. First, we propose a modified bootstrapping procedure that adequately accounts for observed between subject variability while retaining the FWER adjustment method presented for autocorrelated errors. In addition, we offer a permutation test between the groups, borrowing from the insight of the original bdots in that it also captures within-subject variability as demonstrated in the standard errors in the model fits. We begin by describing the two proposed alternatives to the original bdots bootstrap. We then outline the details of the simulations in demonstrating the type I error rate across a number of experimental conditions, and finally we conclude with a demonstration of power in the competing methods.

B.2 Detail on the original

Most generally the original bdots algorithm, which we will call the *homogeneous bootstrap*, proposed that we have empirically observed data in time resulting from a parametric function f with an associated error:

$$y_t = f(t|\theta) + \epsilon_{it} \tag{B.1}$$

where

$$\epsilon_{it} = \phi\epsilon_{i,t-1} + w_{it}, \quad w_{it} \sim N(0, \sigma). \tag{B.2}$$

Under this paradigm, the errors could be iid normal (with $\phi = 0$) or have an AR(1) structure, with $0 < \phi < 1$. It further assumes homogeneity of the mean structure, meaning that two subjects from the same group would have parameters $\theta_{it} = \theta_{jt}$ for all i, j. In other words, it was assumed that there was no variability in the mean structure between subjects in the same group.

1. For each subject, fit the nonlinear function, specifying AR(1) autocorrelation structure for model errors. Assuming large sample normality, the sampling distribution of each estimator can be approximated by a normal distribution with mean corresponding to the point estimate and standard deviation corresponding to the standard error

2. Using the approximate sampling distributions in (1.), randomly draw one bootstrap estimate for each of the model parameters on every subject

3. Once a bootstrap estimate has been collected for each parameter and for every subject, for each parameter, find the mean of the bootstrap estimates across individuals

4. Use the mean estimates to determine the predicted population level curve, which provides the average population response at each time point

5. Perform steps (2)-(4) 1000 times to obtain estimates of the population curves. Use these to create estimates of the mean response and standard deviation at each of the time points.

Population means and standard deviations at each time point for each of the groups were used to construct a series of (correlated) test statistics, where the family wise error rate was controlled by using the modified Bonferonni correction introduced in Oleson et. al., 2017 to test for significance.

B.3 Proposed Methods

As is more typically the case, subjects within a group demonstrate considerable variability in their mean parameter estimates. In such a case, there is no presumption that $\theta_i = \theta_j$, and accounting for between-subject variability within a group will be critical for obtaining a reasonable distribution of the population curves.

B.3.1 Modified Bootstrap

The distribution of parameters for subjects $i = 1, \ldots, n_g$ in group $g = 1, \ldots, G$ follows the distribution

$$\theta_i \sim N(\mu_g, V_g). \tag{B.3}$$

The uncertainty present in our estimation of θ_i can be accounted for by treating the standard errors derived when fitting the observed subject data to Equation B.1 as estimates of their standard deviations. This gives us a distribution for the subject's estimated parameter,

$$\hat{\theta}_i \sim N(\theta_i, s_i^2). \tag{B.4}$$

When obtaining reasonable estimates of the population curve, it is necessary to estimate both the observed within-subject variability found in each of the $\{s_i^2\}$ terms, *as well as* the between-subject variability present in V_g. For example, let θ_{ib}^* represent a bootstrapped sample for subject i in bootstrap $b = 1, \ldots, B$, where

$$\theta_{ib}^* \sim N(\hat{\theta}_i, s_i^2). \tag{B.5}$$

If we were to sample *without replacement*, we would obtain a homogeneous mean value from the bth bootstrap for group g, $\theta_{bg}^{(hom)}$, where

$$\theta_{bg}^{(hom)} = \frac{1}{n_g} \sum \theta_{ib}^*, \quad \theta_{bg}^{(hom)} \sim N\left(\mu_g, \frac{1}{n_g^2} \sum s_i^2\right). \tag{B.6}$$

Such an estimate captures the totality of the within-subject variability with each draw but fails to account for the variability in the group overall. For this reason, we sample the subjects *with* replacement, creating the heterogeneous bootstrap mean $\theta_{bg}^{(het)}$, where again each θ_{ib}^* follows the distribution in Equation B.5, but the heterogeneous bootstrapped group mean now follows

$$\theta_{bg}^{(het)} \sim N\left(\mu_g, \frac{1}{n_g} V_g + \frac{1}{n_g^2} \sum s_i^2\right). \tag{B.7}$$

The estimated mean value remains unchanged, but the variability is now fully accounted for. We therefore present a modified version of the bootstrap which we call the *heterogeneous bootstrap*, making the following changes to the original:

1. In step (1.), the specification of AR(1) structure is *optional* and can be modified with arguments to functions in `bdots`. Our simulations show that while failing to include it slightly inflates the type I error in the v2 bootstrap when the data truly is autocorrelated, specifying an AR(1) structure can lead to overly conservative estimates when it is not.
2. In step (2.), we sample subjects *with replacement* and then for each drawn subject, randomly draw one bootstrap estimate for each of their model parameters based on the mean and standard errors derived from the `gnls` estimate.

Just as with the homogeneous bootstrap, these bootstrap estimates are used to create test statistics T_t at each time point, written

$$T_t^{(b)} = \frac{(\overline{p}_{1t} - \overline{p}_{2t})}{\sqrt{s_{1t}^2 + s_{2t}^2}}, \tag{B.8}$$

where $\overline{p}_{gt}$ and s_{gt}^2 are mean and standard deviation estimates at each time point for groups 1 and 2, respectively. Finally, just as in Oleson 2017, one can use the autocorrelation of the $T_t^{(b)}$ statistics to create a modified α for controlling the FWER.

B.3.2 Permutation Testing

In addition to the heterogeneous bootstrap, we also introduce a permutation method for hypothesis testing. The permutation method proposed is analogous to a traditional permutation method, but with an added step mirroring that of the previous in capturing the within-subject variability. For a specified FWER of α, the proposed permutation algorithm is as follows:

1. For each subject, fit the nonlinear function with *optional* AR(1) autocorrelation structure for model errors. Assuming large sample normality, the sampling distribution of each estimator can be approximated by a normal distribution with mean corresponding to the point estimate and standard deviation corresponding to the standard error

2. Using the mean parameter estimates derived in (1.), find each subject's corresponding fixation curve. Within each group, use these to derive the mean and standard deviations of the population level curves at each time point, denoted $\overline{p}_{jt}$ and s_{jt}^2 for $j = 1, 2$. Use these values to compute a test statistic T_t at each time point,

$$T_t^{(p)} = \frac{|\overline{p}_{1t} - \overline{p}_{2t}|}{\sqrt{s_{1t}^2 + s_{2t}^2}}. \tag{B.9}$$

This will be our observed test statistic.

3. Repeat (2) P additional times, each time shuffling the group membership between subjects. This time, when constructing each subject's corresponding fixation curve, draw a new set of parameter estimates using the distribution found in (1). Recalculate the test statistics $T_t^{(p)}$, retaining the maximum value from each permutation. This collection of P statistics will serve as our null distribution which we denote $\widetilde{T}$. Let $\widetilde{T}_\alpha$ be the 1 - α quantile of $\widetilde{T}$

4. Compare each of the observed $T_t^{(p)}$ with $\widetilde{T}_\alpha$. Areas where $T_t^{(p)} > \widetilde{T}_\alpha$ are designated significant.

Paired permutation testing is implemented with a minor adjustment to step (3). Instead of permuting all of the labels between groups, randomly assign each subject to either retain their current group membership or to change groups. This ensures that each permuted group contains one observation from each subject.

B.4 Type I Error Simulations

We now go about comparing the type I error rate of the three methods just described. In doing so, we will consider several conditions under which the observed subject data may have been generated or fit. This includes two conditions for the mean structure, two conditions for the error structure, paired and unpaired data, and data fit with and without an AR(1) assumption. Considering each permutation of this arrangement results in sixteen different settings. Each simulation will then be examined for type I error using each of the three methods described.

B.4.1 Data Generation

Data was generated according to Equation B.1, with the parametric function $f(t|\theta)$ belonging to the family of four-parameter logistic curves defined:

$$f_\theta(t) = \frac{p-b}{1+\exp\left(\frac{4s}{\mathrm{p}-b}(x-t)\right)} + b \tag{B.10}$$

where $\theta = (p, b, s, x)$, the peak, baseline, slope, and crossover parameters, respectively.

We further assume that each group drew subject-specific parameters from a normal distribution, with subject $i = 1, \ldots, N$ in group $g = 1, \ldots, G$ following the distribution in Equation B.3.

Mean Structure In all of the simulations presented, the distribution used in Equation **??** was empirically determined from data on normal hearing subjects in the VWP (Farris-Trimble et al., 2014 [2]). Parameters

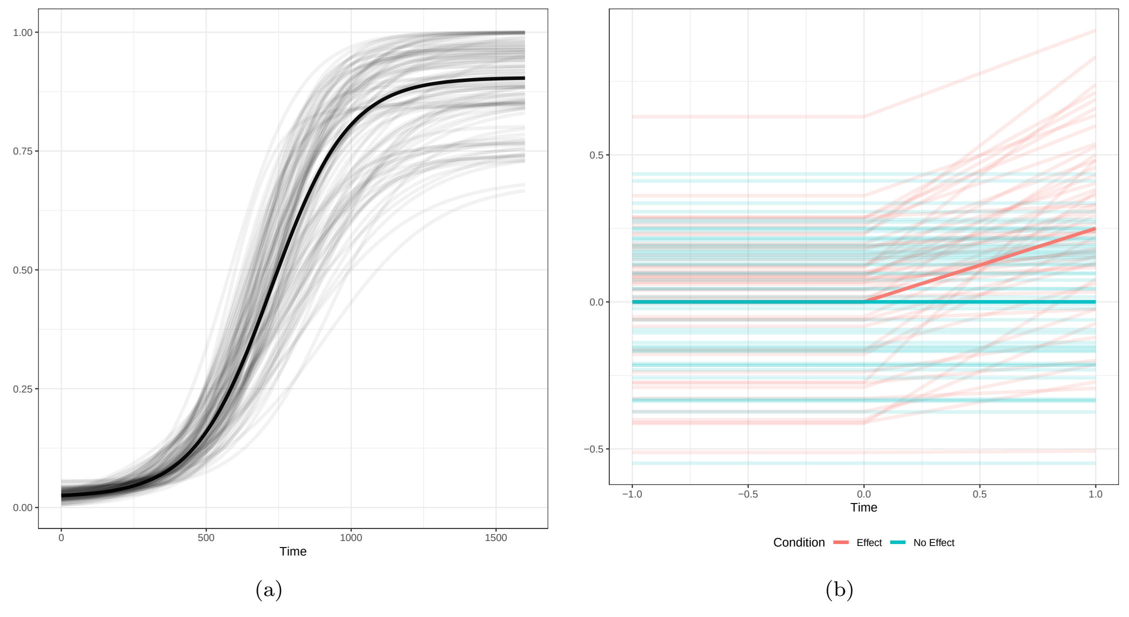


Figure B.1: Distributions of various group with the mean curve in bold (a.) 50 samples from the generating distribution of the four-parameter logistic in Equation B.10 used for testing FWER. (b.) 50 samples from the generating distributions of each group in Equation B.12. The legend makes size weird, might just explain what they are here. Also need to see if I can change size of the mean lines

used were those fit to fixations on the Target, following the functional form of Equation B.10. Under the assumption of between-subject homogeneity, we set $\theta_i = \theta_j$ for all subjects i, j, assuring that each of the subjects' observations is derived from the same mean structure, differing only in their observed error. We will call this the homogeneous means assumption.

Error Structure The error structure is of the form

$$e_{it} = \phi e_{i,t-1} + w_{it}, \quad w_{it} \sim N(0, \sigma) \tag{B.11}$$

where the w_{it} are iid with $\sigma = 0.025$. ϕ corresponds to an autocorrelation parameter and is set to $\phi = 0.8$ when the generated data is to be autocorrelated and set to $\phi = 0$ when we assume the errors are all independent and identically distributed.

Paired Data Finally we considered paired data, though this is only a sensible condition under the assumption of heterogeneous means. In order to construct paired data, we simply used the same parameter estimates for each subject between groups, with the observed data between subjects differing only in the observed error.

Each set of conditions generates two groups, with $n = 25$ subjects in each group, with timepoints

$t = 0, 4, 8, \ldots, 1600$ in each trial and with 100 simulated trials for each subject. Columns in the tables indicate homogeneity of means assumption, whether or not an AR(1) error structure was used in constructing the data, and if autocorrelation was specified in the fitting function. The last conditions helps assess the impact of correctly identifying the type of error when conducting an analysis in `bdots`. Each simulation was conducted 100 times to determine the rate of type I error (time intensive and will launch 1000 for final dissertation).

B.4.2 Results

We consider the efficacy the methods under each of the simulation settings with an analysis of the family wise error rate (FWER) and the median per-comparison error rate. The first of these details the proportion of simulations under each condition that marked *at least* one time point as being significantly different between the two groups. This is critical is understanding each method's ability to correct adjust for the multiple testing problem associated with testing each of the observed time points. These are presented in Table B.1 and Table B.2 for unpaired and paired data, respectively.

Complimenting the FWER estimate is an estimate of the median per-comparison rate. For each time point across each of the simulations, we computed the proportion of times in which that time was determined significant. The median of these values across all time points is what is considered. This metric gives a sense of magnitude to the binary FWER; for example, a situation in which there was a high FWER and low median per-comparison rate would indicate that the type I error within a particular time series would be sporadic and impact limited regions. Large median per-comparison rates indicate that large swaths of a time series frequently sustain type I errors. The median per-comparison rates for unpaired and paired simulations are presented in Table B.3 and Table B.4.

FWER

There are a few things of immediate note when considering the results of Table B.1. First, we see from the first two settings of the unpaired simulations that the type I error rates for the homogenous bootstrap are consistent with those presented in [9], confirming the importance of specifying the existence of autocorrelation in the `bdots` fitting function when autocorrelated error is present. By contrast, this is far less of a concern when using the heterogeneous bootstrap or permutation testing, both of which maintain a FWER near the nominal alpha, regardless of whether or not the error structure was correctly identified. This continues to be true under the homogenous mean assumption when the true error structure is not autocorrelated. Interestingly, the performance of the homogeneous bootstrap falters here despite theoretical consistency with the simulation settings.

The most striking results of this, however, appear when the data generation assumes a heterogeneous mean structure. While both the heterogeneous bootstrap and the permutation test maintain a FWER near the nominal alpha, the homogeneous bootstrap fails entirely, with a FWER > 0.9 in all cases.

Heterogeneity Assumption	Autocorrelated Error	AR(1) Specified	Homogeneous Bootstrap	Heterogeneous Bootstrap	Permutation
No	Yes	Yes	0.06	0.01	0.08
No	Yes	No	0.87	0.08	0.00
No	No	Yes	0.08	0.00	0.06
No	No	No	0.15	0.02	0.01
Yes	Yes	Yes	0.92	0.03	0.05
Yes	Yes	No	0.96	0.02	0.08
Yes	No	Yes	0.99	0.05	0.03
Yes	No	No	1.00	0.05	0.06

Table B.1: FWER for empirical parameters (unpaired)

Paired data is given in Table B.2. Matching the conclusions drawn from Table B.1, we only note here that both the permutation test and heterogeneous bootstraps maintain a valid FWER under the assumption of paired data.

Heterogeneity Assumption	Autocorrelated Error	AR(1) Specified	Homogeneous Bootstrap	Heterogeneous Bootstrap	Permutation
Yes	Yes	Yes	0.49	0.02	0.01
Yes	Yes	No	0.94	0.03	0.02
Yes	No	Yes	0.72	0.02	0.00
Yes	No	No	0.74	0.04	0.00

Table B.2: FWER for empirical parameters (paired)

Median per comparison error rate

We next consider the median comparison rate, which offers some insight into the FWER. In particular, consider the situation in which in Table B.3, in the fourth row we see a median per-comparison error rate of 0.00 for the homogeneous bootstrap, despite Table B.1 indicating a FWER of 0.15. This is a consequence of the majority of the type I errors occuring in a relatively limited region. In contrast, the median per-comparison error rate of the homogeneous bootstrap under the assumption of heterogeneity suggests that the type I errors are widespread and not limited to any particular area.

It is also worth commenting on the permutation test median per-comparison error rate in Table B.3; combined with a FWER near the nominal 0.05, these values suggest that errors are likely distributed across the entire range rather than limited to a small area (which would result in a MPCR of 0).

Heterogeneity Assumption	Autocorrelated Error	AR(1) Specified	Homogeneous Bootstrap	Heterogeneous Bootstrap	Permutation
No	Yes	Yes	0.01	0.00	0.04
No	Yes	No	0.31	0.00	0.04
No	No	Yes	0.00	0.00	0.02
No	No	No	0.00	0.00	0.03
Yes	Yes	Yes	0.51	0.01	0.01
Yes	Yes	No	0.76	0.01	0.00
Yes	No	Yes	0.86	0.01	0.00
Yes	No	No	0.81	0.01	0.00

Table B.3: median per comparison error rate (unpaired)

Heterogeneity Assumption	Autocorrelated Error	AR(1) Specified	Homogeneous Bootstrap	Heterogeneous Bootstrap	Permutation
Yes	Yes	Yes	0.13	0.00	0.00
Yes	Yes	No	0.52	0.02	0.00
Yes	No	Yes	0.38	0.01	0.00
Yes	No	No	0.44	0.01	0.00

Table B.4: median per comparison error rate (paired)

B.4.3 Discussion

Table B.1 and Table B.2 demonstrate that under a variety of settings, both the heterogeneous bootstrap and permutation offer a FWER near the nominal alpha, making their performance similar to that of the homogeneous bootstrap in the best of cases. This includes less restrictive assumptions in which they continue to perform well while the homogenous bootstrap maintains an unacceptably high FWER.

Transition sentence to power simulations

B.5 Power Simulations

To determine power, two experimental groups were simulated with mean structures of the following form:

$$y = \begin{cases} b & x < 0 \\ mx + b & x \geq 0 \end{cases} \tag{B.12}$$

The first simulated group was "No Effect", with intercept and slope parameters normally distributed and standard deviation $\sigma = 0.05$. The second group, the "Effect" group, was similarly distributed, but with the slope parameter having mean value of $\mu = 0.25$. The error structure was identical to that in the FWER simulations, with both an AR(1) error structure and independent noise included. 100 simulations were conducted for each scenario.

We limited consideration to three possible scenarios: first, we assumed the conditions presented in Oleson 2017, assuming homogeneity between subject parameters and an AR(1) error structure, with the model fitting performed assuming autocorrelated errors. For the remaining scenarios, we assumed heterogeneity in the distribution of subject parameters, simulated with and without an AR(1) error structure. In both of these last two scenarios, we elected to *not* fit the model assuming autocorrelated errors. This was for two reasons: first, simulations exploring the type I error rate suggested that models fit with the autocorrelation assumption tended to be conservative. Second, and given the results of the first, this makes setting the assumption of autocorrelation to FALSE in `bdots` seem like a sensible default, and as such, it would be of interest to see how the model performs in cases in which their is autocorrelated error that is not accounted for.

For each subject, parameters for their mean structure given in Equation B.12 were drawn according to their group membership and fit using `bdots` on the interval (-1,1). Time windows in which the groups differed were identified using each the homogenous bootstrap, heterogeneous bootstrap, and permutation testing. By including the interval (-1,0) in which the null hypothesis was true, we are able to mitigate the effects of over-zealous methods in determining power, and we present the results in the following way: any tests in which a difference was detected in (-1,0) was marked as having a type I error, and the proportion of simulations in which this occurred for each method is reported as the FWER in the column labeled α. The next column, β, is the type II error rate, indicating the proportion of trials in which no differences were identified over the entire region. The last Greek-letter column is $1 - \beta - \alpha$, a modified power statistic indicating the proportion of tests in which a difference was correctly identified. The remaining columns relate to this modified power columns, giving a partial summary of the earliest onset of detection. As a true difference occurs on the interval $t > 0$, smaller values indicate greater power in detecting differences. Finally, a (currently base-R) plot giving the power at each time point is given in Figure B.2. This plot represents the true power, though note that it does not take into account the rate at which these regions were identified in conjunction with a type I error rate.

B.5.1 Results

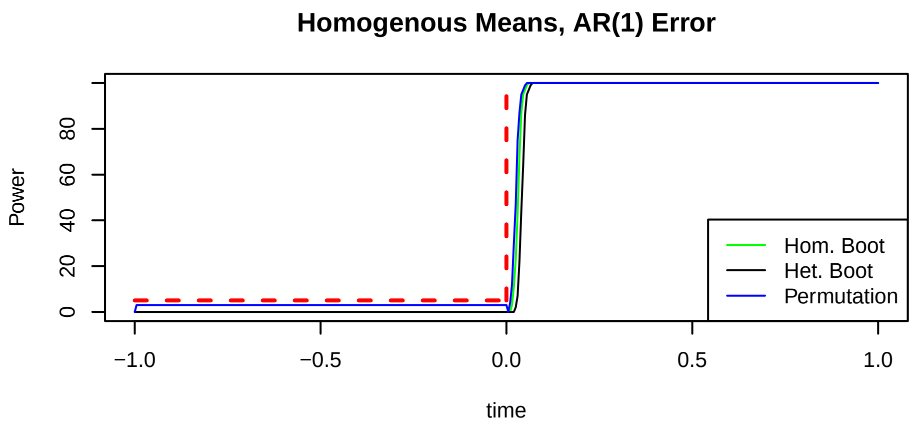
Method	Heterogeneity	AR(1)	α	β	1 - α - β	1st Qu.	Median	3rd Qu.
Hom. Boot	No	Yes	0.00	0.00	1.00	0.025	0.030	0.035
Hom. Boot	Yes	No	0.96	0.00	0.04	0.005	0.008	0.010
Hom. Boot	Yes	Yes	0.97	0.00	0.03	0.008	0.010	0.010
Het. Boot	No	Yes	0.00	0.00	1.00	0.035	0.040	0.045
Het. Boot	Yes	No	0.00	0.10	0.90	0.403	0.513	0.690
Het. Boot	Yes	Yes	0.01	0.10	0.89	0.420	0.525	0.690
Perm	No	Yes	0.03	0.00	0.97	0.015	0.025	0.025
Perm	Yes	No	0.03	0.05	0.92	0.378	0.515	0.681
Perm	Yes	Yes	0.08	0.03	0.89	0.360	0.540	0.705

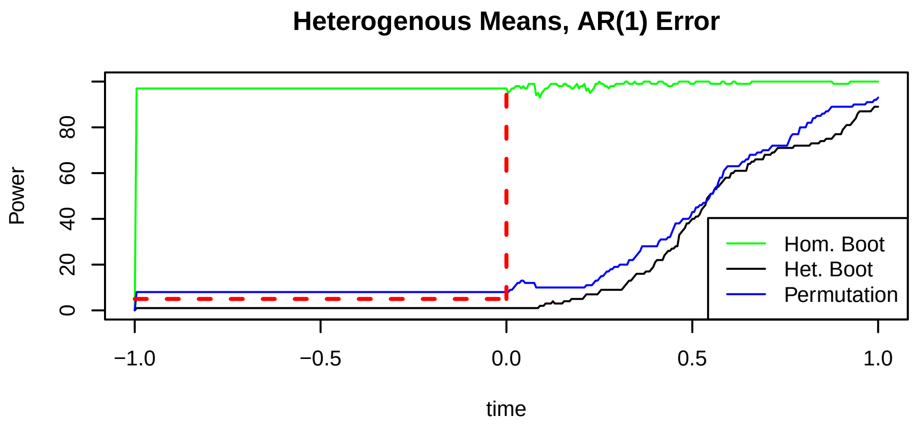
Table B.5: Power for methods (possibly reorder?)

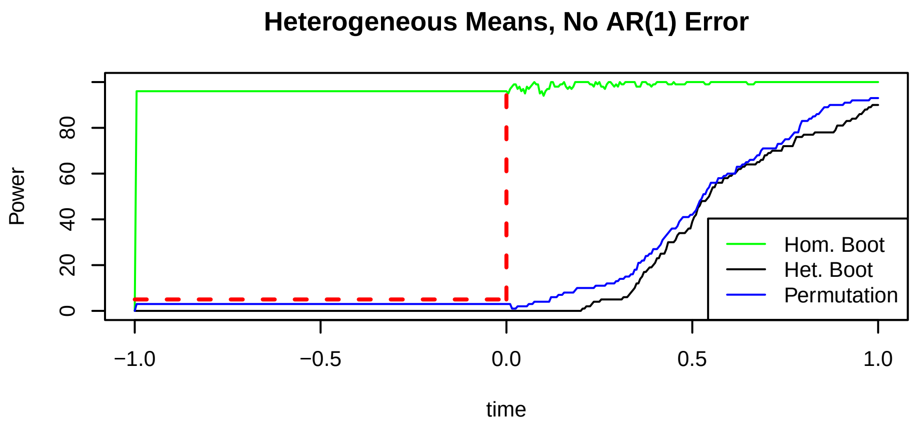
Method	Type I Error	Type II Error	Power	1st Qu.	Median	3rd Qu.
Hom. Bootstrap	0.643	0.000	0.357	0.013	0.016	0.018
Het. Bootstrap	0.003	0.067	0.930	0.286	0.359	0.475
Permtuation	0.047	0.027	0.927	0.251	0.360	0.470

Table B.6: Summary of methods for Type II error

Summary of methods







.

Figure B.2: Observed power of each of the methods at each time in (-1,1)

B.6 Discussion and concluding remarks

We set out to interrogate the validity of the v1 bootstrap assumptions and to propose two alternative methods that would be more robust under a greater variety of assumptions. In doing so, we demonstrated conclusively the utility of the v2 bootstrap and permutation tests while also highlighting a major shortcoming of the v1 bootstrap. It's worth noting, however, that the FWER adjustment proposed in [9] is still valid, if not slightly conservative, and with power similar to that of the permutation method, and this will remain an option in version 2.0 of `bdots`.

There are a few limitations to the current paper that are worthy of investigation. First, limited consideration was given to the effect of sample density on the observed type I error rate or power. As the fitting function in `bdots` simply returns a set of parameters, one could conceivably perform any of the methods presented on any arbitrary collection of points, whether or not any data were observed there. This extends itself to the condition in which subjects were sampled at heterogeneous time points, as may be the case in many clinical settings. What impact this may have or how to best handle these cases remains investigated. The current implementation of `bdots` takes the union of observed time points, though this runs the risk of extrapolating many subjects past what they were ever observed. It would be of interest to know if either the permutation or v2 bootstrap perform better in these situations, and if both retain their validity under increasingly suspect conditions. Finally, in noting the rather conservative FWER estimates for both the v2 bootstrap and the permutation test, it would be worthwhile investigating if *not* resampling subject-specific parameters from the distribution provided by `gnls` would retain an acceptable FWER while increasing power.

We conclude pretty much by noting that even in the best case presented in Oleson 2017, these other methods do an identical job, and in situations in which these assumptions are wrong, it is a veritable train wreck. It seems that the $1 - \beta - \alpha$ (whatever this is called) is nearly identical between the v2 bootstrap and permutation, whereas the type II error is much greater in the bootstrap. This seems justification enough for making the permutation method the new default in `bdots` or, should i say PDOTS. It is conceivable that the assumptions presented in Oleson 2017 would have their place, say repeated observation from the same mechanism (i.e., not vwp data), in which case v1 bootstrap has optimal performance. Still, users of `bdots` will need to go out of their way in order to do so, possibly with a warning. Because really, even in that case, it hardly does any better than permutation, perhaps with a bit smaller of a type I error.

Appendix

Could include oleson 2017 parameters just to say we did it and verifying the two results that they had previously found (i.e., we have implemented this correctly). Commented out for now

Bibliography

[1] Paul D Allopenna, James S Magnuson, and Michael K Tanenhaus. Tracking the time course of spoken word recognition using eye movements: Evidence for continuous mapping models. *Journal of memory and language*, 38(4):419–439, 1998.

[2] Ashley Farris-Trimble, Bob McMurray, Nicole Cigrand, and J. Bruce Tomblin. The process of spoken word recognition in the face of signal degradation. *Journal of Experimental Psychology: Human Perception and Performance*, 40(1):308–327, feb 2014.

[3] James S. Magnuson. Fixations in the visual world paradigm: where, when, why? *Journal of Cultural Cognitive Science*, 3(2):113–139, sep 2019.

[4] Eric Maris and Robert Oostenveld. Nonparametric statistical testing of EEG- and MEG-data. *Journal of Neuroscience Methods*, 164(1):177–190, aug 2007.

[5] Bob McMurray. I'm not sure that curve means what you think it means: Toward a [more] realistic understanding of the role of eye-movement generation in the visual world paradigm. *Psychonomic Bulletin & Review*, pages 1–45, 2022.

[6] Bob McMurray, Vicki M Samelson, Sung Hee Lee, and J Bruce Tomblin. Individual differences in online spoken word recognition: Implications for sli. *Cognitive psychology*, 60(1):1–39, 2010.

[7] Bob McMurray, Michael K. Tanenhaus, and Richard N. Aslin. Gradient effects of within-category phonetic variation on lexical access. *Cognition*, 86(2):B33–B42, dec 2002.

[8] Daniel Mirman, James A. Dixon, and James S. Magnuson. Statistical and computational models of the visual world paradigm: Growth curves and individual differences. *Journal of Memory and Language*, 59(4):475–494, nov 2008.

[9] Jacob J Oleson, Joseph E Cavanaugh, Bob McMurray, and Grant Brown. Detecting time-specific differences between temporal nonlinear curves: Analyzing data from the visual world paradigm. *Statistical methods in medical research*, 26(6):2708–2725, 2017.

[10] Michael Seedorff, Jacob Oleson, and Bob McMurray. Detecting when timeseries differ: Using the bootstrapped differences of timeseries (bdots) to analyze visual world paradigm data (and more). *Journal of memory and language*, 102:55–67, 2018.

[11] Michael K Tanenhaus, Michael J Spivey-Knowlton, Kathleen M Eberhard, and Julie C Sedivy. Integration of visual and linguistic information in spoken language comprehension. *Science*, 268(5217):1632–1634, 1995.

[12] P. Viviani. Eye movements in visual search: cognitive, perceptual and motor control aspects. *Eye movements and their role in visual and cognitive processes*, pages 353–393, 1990.