An algorithm for detecting ERP components using the grand average waveform as a template

Sven Lesche[1]

[1] Ruprecht-Karls-University Heidelberg

Author Note

Author Notes go here.

The authors made the following contributions. Sven Lesche: Conceptualization, Writing - Original Draft Preparation, Writing - Review & Editing.

Correspondence concerning this article should be addressed to Sven Lesche, Im Neuenheimer Feld 695, 69120 Heidelberg. E-mail: sven.lesche@psychologie.uni-heidelberg.de

Abstract

One or two sentences providing a **basic introduction** to the field, comprehensible to a scientist in any discipline.

Two to three sentences of **more detailed background**, comprehensible to scientists in related disciplines.

One sentence clearly stating the **general problem** being addressed by this particular study.

One sentence summarizing the main result (with the words "**here we show**" or their equivalent).

Two or three sentences explaining what the **main result** reveals in direct comparison to what was thought to be the case previously, or how the main result adds to previous knowledge.

One or two sentences to put the results into a more **general context**.

Two or three sentences to provide a **broader perspective**, readily comprehensible to a scientist in any discipline.

Keywords: keyword, more keyword, crazy keyword

Word count: X

An algorithm for detecting ERP components using the grand average waveform as a template

History of Extraction Methods

Importance of ERPs

Event-related potentials (ERPs) are a key tool used in a variety of fields of study. Clinical psychology [REF], cognitive psychology [REF], experimental psychology [REF] all make use of ERPs to investigate the neural processes. Both the amplitude as well as the latency of components are often used to gain insights into individual differences in cognitive processes [REF]. Accurate estimation of the latency of ERP components thus presents an important topic relevant to all applications of ERP research [REF]. This work will introduce a novel algorithm that attempts to locate ERP components using component-templates generated from the grand average waveform.

something about N2 and P3 processes here also write something about its relevance for individual differences research

the first paragraph should already illustrate how imporant accurate latency extraction methods are

Prior Algorithms

Peak Latency

A number of algorithms have been developed hoping to ease identification of components and their latencies. The earliest version of an algorithm to automatically extract ERP latencies is the *peak latency* approach [REF]. It involves finding the point in time within a set measurement window that has the largest voltage deflection in the appropriate direction. This method was developed as it was easy to implement **something about historical ease of use Luck, 2014, p. 286 cites Donchin & Heffley, 1978 as historical example**. Even though an algorithm detecting the *peak latency* inside some measurement window is easy to implement and run, it can be corrupted by several factors.

be aware that luck lists the following issues: peak != component; high-freq noise; time-window/point differences in subjects; biased by noise-level; is non-linear measure; influence of latency jitter; difficult to compare with RT

Issues with peak latency.

1. Choice of measurement window As the peak latency method is blind to the general structure within the ERP signal, it simply chooses the point in time with the largest voltage deflection within the measurement window. The "true" maximum signal may lay just outside of the measurement window, resulting in the algorithm picking an outside edge of the window to reflect the *peak latency*, even though the signal increases in amplitude just following that timepoint. This may be true even if there is a clear component-like structure fully contained in the measurement window. Later components may also already influence the amplitude on the outer ranges of the measurement window. Simple *peak latency* algorithms can thus fail to pick the *local maximum* of a component of interest fully contained inside the measurement window in favor of a peak possibly influenced by later or earlier components. This problem is often referred to as the *superimposition problem* [REF, just luck, 2014?].

To combat the influence of other components, [REF] have modified the simple *peak latency* algorithm to only consider maxima inside the measurement window which are also larger than the surrounding datapoints. A spike just on the edge of the measurement window induced by a later component will not meet these criteria, as the last datapoint inside the measurement window will be smaller than the following datapoint. This method is more accurate in finding the *local* peak latency [REF] but still vulnerable to other issues with peak latency approaches.

2. Superimposition of other components The *peak latency* approach is also especially vulnerable to influences of other surrounding components on the ERP signal. Peak

latency approaches may fail to detect the peak of the component of interest if the measurement window contains portions of non-relevant components. Researchers thus have to carefully choose the measurement window in order to include most of the signal related to the component of interest while simultaneously excluding influences of other components.

3. high frequency noise The influence of high frequency noise presents another issue for peak latency algorithms. The maximum voltage deflection may not reflect the true point in time at which the component reaches its maximum. Rather it may be the result of high frequency noise inducing a spike in the signal, independent of any cognitive process. This is problematic especially in the identification of the latency of later ERP components, as the larger measurement window most commonly applied here will increase the likelihood of high frequency noise impacting the maximum voltage deflection [REF, luck? or liesemeier?].

4. The peak latency is also just not that relevant Lastly, as [Luck, 2014] so aptly states: *There is nothing special about the point at which the voltage reaches a local maximum.* The largest deflection does not inherently relate to any physiological or psychological process and may not even reflect the true maximum of the component of interest. [REF luck rule paper] visually demonstrates how peak latency may be a result of the overlap of multiple components and not related to any single component.

Fractional Area Latency

Fractional Area Latency approaches hope to remedy some of the problems associated with *peak latency* algorithms. *Fractional Area Latency* revolves around the area under the ERP signal in a given measurement window. It's goal is finding the point in time that divides that area under the signal before and a after that point in time into a given fraction. The timepoint halving the area under the signal, for example, is referred to as the *50% area latency*. This approach is much less susceptible to the influence of high frequency

noise, as short spikes in the signal do not influence the area under the signal much. It remains highly dependent on the measurement window [Luck, 2005]. Choosing a shorter window may result in only a part of the area of the component of interest being captured. A wider measurement window on the other hand might be influenced by surrounding components. *Fractional Area* measures thus work best for investigating an isolated component [Luck, 2014], limiting the applicability of algorithms based on this. ### Jackknifing Another approach towards dealing with noisy subject-level ERPs is to try and remove that noise by averaging multiple subject-level ERPs. This technique is referred to as *jackknifing*. It uses the grand average of N subject-level ERPs and generates N sub-grand averages by removing one of the subject-level ERPs. This results in N ERPs with higher signal-to-noise ratios as they are the average of $N - 1$ subject-level ERPs. Both *peak* and *area* based measures can then be applied to the jackknifed data to extract latencies. As any two jackknifed signals share $\frac{N-2}{N} * 100$ percent of the subject-level signals that are averaged with each other, each jackknifed-subaverage is quite similar to all others. This artificially decreases the error variance, which needs to be corrected for when testing for significant differences between groups [(Ulrich & Miller, 2001), aus Sadus (2023)]. Because Latencies extracted from jackknifed ERPs are based on averaged waveforms, they can not be readily associated with any single subject, preventing this method from generating individual-level latency estimates needed for individual differences research. This problem was adressed by [Smulders 2010, aus Sadus (2023)] who introduced a transformation able to generate individual-level latency estimates.

Comparison of Algorithms

[Kiesel 2008] compared a number of extraction methods for a variety of ERP components. They induced latency differences of the visual and auditory N1, the N2pc and the P3 and frequency-related P3 and tested single-participant approaches and jackknife-approaches combined with peak latency, fractional area latency, relative criteria and baseline deviation methods on their ability to detect these effects. The most widely

used technique of single-participant approaches combined with peak latency extraction proved to not be the most efficient method to detect latency effects. This becomes especially true as the signal-to-noise ratio worsenes. Overall, jackknifing ERPs and using the relative criterion technique or the fractional area latency technique was shown to be the best approach across the components and datasets they analyzed. However, they note that their results are not necessarily applicable to any dataset. ERP researchers should first visually inspect the ERP waveform in order to inform their decision on an appropriate measurement window and extraction technique. Recently, [Sadus 2023] used a multiverse approach to investigate P3 stuff and utilizied **these** algorithms as well as manual extraction of ERP latencies ...

Look at comparison studies report what they thought is best. Sadus et al. as last

the conclusion from this needs to be made clear: no algorithm provides a sufficiently reliable method of extracting component latencies. - Manual inspection seems to reign supreme, **talk about the practical problems associated with this**, **also talk about the methods problems, low objectivity**

Our algorithm

Motivation

When visually inspecting ERP signals, the goal is to identify a pattern within the signal that resembles the component of interest in shape, size and location. None of the previously described algorithms used in most ERP research aim to replicate this human behavior but rather focus on generating similar decisions, not similar decision processes. The goal of this work is to introduce a new method for extracting component latencies that more closely resembles the manual extraction process.

template matching

Introduction

The general question each ERP-researcher asks themselves when investigating ERP signals is: "Where in this ERP signal is the component of interest?" Most researchers have some mental representation of what the component *should* look like and where it *should* generally appear that helps them identify a specific component in some noisy ERP. Finding a given pattern inside a noisy signal is not a novel task in the field of signal processing. Algorithms aiming to detect the appearance of a pattern - a *template* -inside audio-, video- or radio signals have been around since **HISTORY** [REF] and a large amount of research has gone into optimizing these *template matching* algorithms [REF].

No matter the implementation of the template matching algorithm, they all aim to answer the question "Does this (shorter) template appear in my (larger) signal?". To achieve this, a researcher needs to specify two things. First, the template that they want to search the signal for. And second, the *similarity measure* by which the algorithm can figure out how well the template fits in a given spot of the signal.

** bessere Überleitung und Erklärung warum 2. Punkt zuerst ### **Similarity measures A number of similarity measures have been proposed to capture the degree of similarity between a template and the signal. These include, but are not limited to: the Sum of Absolute Differences (SAD), the Mean Square Error (MSE), the Geometric Distance, Mutual Information or Cross-Correlation (CC)[[mahalakshmi2012image]] [REFs]. We have chosen to implement algorithms based on two of these measures, one minimizing the Sum of Squared Differences (MINSQ) and one maximizing the correlation between the template and the signal (CORR). ### Template generation Depending on the field, the template to search for is easily specified. If you are looking to extract a particular audio-signal or some image [REF]. . . . It becomes more challenging when it is not exactly certain what template you are looking for.**

Recent research in image processing has attempted to use template matching to process faces, for example. You cannot just use "the ideal nose" describe more here... [**REF**] more about history of pattern matching here... **A similar issue has plagued template matching approaches in ERP research. The variance in ERP signals introduced by the task used or the type of participants prevents a successful implementation of template matching algorithms using one idealized template over all types of studies. ## Prior attempts There have been some attempts at using an idealized component structure, like the COMPONENT[depending on citation] as a template to identify the component in noisy subject-level data. The variance of ERP signal structure within and especially between ERP studies has presented a challenge to this approach [REF]. The morphology of the P3, for example, varies depending on the task used and the participants studied [REF]. This compromises the ability to use a heavily idealized P3-like component as a template to search for in subject-level ERPs [Does it work with GA?].** want to talk about the Hidden-semi-markov-chain approach?**

Brain interfaces have used ERPs as a template for single-trial EEG-data to detect the presence or absence of some kind of response. But not done for ERP-data [[william2020erp]] (Seen Face or no?) [[shahlaei2019detection]] (Brain-Computer-Interfaces)

our solution

We addressed this problem by using using an experiment-specific template of the component of interest that will reflect influences of the task and participant sample on the morphology of the idealized ERP component. The grand average enables us to generate this sort of idealized ERP component. It has a higher signal-to-noise ratio than any individual subject-level ERP while still being influenced by experiment-specific changes in component morphology. Researchers can (and already often do) use the grand average to gather insight into the time window in which the component of interest occurs. This part

of the grand average's signal can then be used as a template and matched to subject-level ERPs.

Differences to other template matching studies

Template Matching in ERP studies differs from classical approaches in the sense that the template may change in shape, location or size based on individual differences. Some participant's P3 occurs much later or with larger amplitudes than those of others participants. The question shifts from "Does the template appear in the signal" to "Which transformation of the template appears in the signal". Of course some aspects of both questions are present in traditional approaches and our algorithm should also be able to detect cases in which no clear match is found. But the goal of this algorithm is to accurately measure individual differences in the latency of ERP components. Since not all subjects' ERP components have the same size, shape and location as the grand averages', transformations of the template allow for a more realistic representation of individual differences in ERP signals. Finding the transformation of the template that best fits each subject-level ERP thus allows us to identify how the subject-level signal differs from the grand average. This is precisely the issue that individual differences research using ERPs is faced with and that we hope to provide an algorithm for.

Subject-level transformations as the goal

All ERP signals can be transformed into a sum of sines function with the properties amplitude, frequency and phase for each sine term. ERP signals can thus be modified by multiplying all amplitude-factors **this is terrible, do math notation!** by a factor a. This effectively transforms the amplitudes of the overall signal by that factor. Multiplying all frequency factors by a factor b can change the latency of peaks in the signal. It effectively squishes or stretches the signal along the x axis. We have limited our transformations to these to operations, with future studies inspecting the impact of manipulating the phase parameter. This would effectively "shift" the signal. ** Explain why this was done show examples of the effect of this**

Figuring out the optimal transformation

We can produce an infinite number of transformed grand averages. Each transformation represents one version of our "template". The algorithm then finds the one template that best fits the subject-level ERP signal and returns the parameters used to generate this template. A subject whos P3 peaks at 400ms, while the grand average's P3 is located around 600ms would most likely receive a parameter around $b = 1,5$. All frequency parameters in the sum of sine functions would be multiplied by 1,5, effectively "speeding up" the signal and thus "squishing" it more closely to the origin.

Component latencies can also be recovered from this method. Researchers need to specify a time-point of the grand average denoting the latency of the component of interest in the grand average. The transformation parameters resulting in the best fit of the grand average to the subject level signal can then just be applied to the grand average latency to recover the subject-level latency of the component.

The algorithm is naive to the method used to generate the component latency of the grand average. It simply applies the optimal transformation of the whole template to a given latency value. Researchers can use whatever tool they choose to identify component latencies in the grand average waveform. Both peak latency and area latency measures, 50% area or onset / offset latencies can be supplied to the algorithm. It will simply return the transformed latency based on the transformation parameters.

Even though the algorithm allows for the use of peak latency measures when specifying the grand average component latency, it is by no means a peak based algorithm. Approaches based on both similarity measures, Squared Sum of Differences and Cross-Correlation are not impacted by high frequency noise.

** Talk about how this matches the entire component and is also similar to area measures. Talk about how this is more like z-Standardizing the Dataset. You can extract peak latencies, but you can also just leave it be defined by the stretching parameter ** ## Validation - this is in method To evaluate the efficacy of our proposed algorithm, we will

compare its psychometric properties to other commonly used algorithms. We hope to show that our algorithm produces more reliable estimates of component latency. We will also compare our algorithms and previous algorithms in their ability to produce latencies that correlate with latencies generated by a trained ERP researcher.

Our analysis will be conducted on data from [Sadus (2023)] that contains ERPs on Flanker, Nback and Switching tasks each preprocessed with a variety low-pass filters (4hz, 8hz, 16hz, 32hz, 64hz). We will also investigate the impact of using a moving average to create a signal with higher signal-to-noise ratio [REF for moving average]. The only degree of freedom this method offers to the researcher is the definition of the measurement window, so we will investigate the impact of 3 different measurement windows with small, medium and large sizes around the component of interest.

Our hopes

We hope to show that **explain how we hope to validate this algorithm what are we investigating: measurement window, filtering, sub-GAs** explain how we hope to make ERP extraction more objective and easier**

** you can say something about moving averages clearing up signal-to-noise ration (in [[william2020erp]] cited: “Stein’s Unbiased Risk [21]. This thresholding technique has better signal to noise ratio and outperforms other methods such as moving average and Savitzky Golay”) **

** in methods, talk about how actually the subject-signal is transformed also talk about the weighting function of the minsq algorithm **

Method

Algorithm

Implementation in Matlab with details

We implemented the algorithm in MATLAB [VERSION, CITATION]. Data must be in ERPLAB - shape **better way**. The researcher then specifies the name and polarity

of the component of interest as well as the measurement window. This window is then applied to the grand average and used to extract the template. In order to transform the template, we use MATLABs Curve Fitting Toolbox [Citation] to generate *sum of sines* functions that fit to the data with $R^2 >= 0.999$. We then add an amplitude parameter a to the overall function in order to allow for scaling of the amplitude of the template and a frequency parameter b to all frequency terms in the *sum of sines* function to allow for "squishing" or "stretching" the template along the x-Axis. As these transformations also change the measurement window, we chose to use the subject-level ERP as a template and keep the grand average untransformed as a signal. This reverse matching approach is only an implementation detail and does not affect any decisions made by the algorithm.

Depending on the similarity measure employed, we use different functions to find the set of optimal parameters [a, b] that lead to the transformation most similar to the signal.

The MINSQ algorithm minimizes the sum of squared differences between the transformed subject signal and the grand average. We also added a vector weighting the differences. This weighting vector ω is computed by the following function:

$$\omega_i = 1 + (10 * |\frac{y_i}{y_{max}}|)^2$$

All values outside of the measurement window receive a weight of 1. This places more emphasis on fitting the template within the measurement window specified and to places in the signal where the voltage deflection is high.

The CORR algorithm optimizes the parameters to produce the maximum correlation between the transformed subject-level signal and the grand average for values in the measurement window. It only optimizes the parameter b as amplitude changes would not affect the correlation.

Both algorithms then return the transformation parameters that result in optimal similarity of template and signal. We use the returned value of the parameter b_i to transform the component latency specified by the researcher in the grand average l_{GA} to the component latency of the subject-level ERP signal l_i.

$$l_i = l_{GA} * b_i$$

Fit index For both algorithms, we use the correlation of the transformed subject level signal and the grand average template as a fit indicator. ## Review methods Researchers can manually review all choices the algorithm has made in a custom-built user interface. They can also make use of the fit index to only review those cases where the correlation between template and signal dips below a certain value. We will investigate the additional benefits a manual review process provides over accepting the choices as-is or only automatically discarding those matches with correlations $r < 0.2$. # Data We made use of data published by [Sadus 2023]. They conducted a multiverse study to investigate the impact of data preparation techniques on various methods of extracting P3 latency differences. We will compare our algorithm to the methods presented in their paper. ## Participants

The data is comprised of 30 young (AGE) and 30 old (AGE) participants. ## Tasks Participants completed a set of 3 tasks, a Flanker Task, a Nback Task and a Switching Task. ** The flanker task has people... (short) the Nback task requires... the switching task requires ** Details regarding the procedure can be found in [Sadus 2003].

Preparation

** say that EEG procedure is described in Sadus et al. different filter settings and data preparation techniques **

Components

All algorithms estimated the latency of the P3 component. Based on signals from **which electrode**

Exact analysis plan

Which algorithms are run based on which data?

something about conditions, the bins used. Number of erpsets we looked at filter settings used; **measurement windows deployed**; **algorithms used**

Validation Techniques

** make clear that it will be done just as in Sadus' paper ## **Reliability** estimate reliability based on split-half correlation between odd-even splitted bins in the different conditions. Good method should have reliability... ** ## Homogeneity Estimated the mean correlation between the method and other methods, good homogeneity should be.. ## Effect size? We also investigated the effect of age on P3 latency with different methods. They should not lead to misclassifications ## Correlation with manual extraction [Sadus 2023] found manual extraction methods to be the best indicator. We will assess correlations of different algorithms with latencies extracted by expert raters.

Results

Discussion

Here you can discuss your results.

References

Appendix A

Talking about appendices

First-level headers create appendix-sections labelled A-Z. You can print tables here as well and refer to them in your main part. They will receive a prefix to their Table/Figure Number based on the appendix section they are in.

Appendix B

Another section

this creates another appendix section