Automatically Extracting ERP Component Latencies Using a Template Matching Algorithm

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

Automatically Extracting ERP Component Latencies Using a Template Matching Algorithm

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 template matching algorithm that attempts to locate ERP components using templates generated from the grand average waveform.

Prior Algorithms

Peak Latency

Automatically extracting component latencies has long been a goal in ERP research. 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.

Issues with peak latency.

1. Choice of measurement window Peak latency approaches are blind to the general structure within the ERP signal. They locate 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, even though the signal increases in amplitude just following that timepoint. [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 will not meet these criteria. This method is more accurate in finding the *local* peak latency [REF] but still vulnerable to other issues.

2. Superimposition of other components Later components may also already influence the amplitude on the outer ranges of the measurement window. *peak latency* algorithms can thus fail to pick the *local maximum* of a component of interest 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?]. Researchers 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 process underlying 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 later ERP components, as the broader measurement window most commonly applied will result in an increased likelihood of high frequency noise impacting the maximum voltage deflection [REF, luck? or liesemeier?, clayson 2013 (aus sadus)].

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. This technique revolves around the area under the ERP signal in a given measurement window. The goal is finding the point in time that divides that area under the signal into a given fraction to the left and right of it. The time-point 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 have a strong impact on the area under the signal. 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 mitigate 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 averaging procedure results in N ERPs with higher signal-to-noise ratios. 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 readily be associated with any single subject, preventing this method from generating individual-level latency estimates needed for individual differences research. This problem was addressed 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 simulated 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 decreases. 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.

[Kiesel 2008] were concerned with extracting condition differences in ERP research. While this is important, techniques applied in individual differences research have to extract latencies from subject-level ERPs and produce both reliable and valid estimates. [Kiesel 2008] did not assess psychometric properties of latency estimates. Addressing this gap in research, [Sadus 2023] investigated the influence of different preprocessing strategies and latency extraction techniques on psychometric qualities of the latency values generated and their ability to detect an age-related effect in P3 latency. They varied the strength of the low-pass filter applied to the data, used both single-participant and jackknife approaches and extracted latencies either automatically or manually, using either a peak latency or area latency approach. Both the size of the effect and the psychometric properties, such as reliability or homogeneity of the latency values, varied between the

different analysis strategies. No combination of preprocessing steps and extraction method proved best across all tasks and groups. A total of 7 out of 40 possible pipelines generated consistently desirable reliabilites ($r_{tt} \geq .70$), homogeneities ($r_h \geq .50$) and effect sizes ($.03 \leq \omega^2 < .80$). All of those seven pipelines used manual extraction methods either based on peak or area latencies. While automated extraction methods would improve both efficiency and objectivity, fully automated approaches failed to generate consistently reliable and valid latency measures [Sadus 2023; Schubert 2022 (aus Sadus)]. Yet, manual extraction methods are highly time-consuming and impede reproducibility. We hope to show that our algorithm can match the performance of manual extraction while providing a more efficient and objective approach for extracting individual component latency values.

Our algorithm

Motivation

The shortcomings of prior algorithms may be due to none of them aiming to replicate human behavior and focus on generating similar decisions rather than similar decision processes. We propose an algorithm that more closely resembles the process expert ERP researches employ.

template matching

Introduction

To investigate where in some noisy signal the component of interest is found, most researchers have some mental representation of what the component *should* look like and where it *should* generally appear. When visually inspecting ERP signals, their goal is to identify a pattern within the signal that resembles the component of interest in shape, size and location.

Finding a given pattern inside a noisy signal is not a novel task. Algorithms aiming to detect the appearance of a pattern - a *template* -inside audio-, video- or radio signals have been around for over 50 years [REFS]. And a large amount of research has gone into

optimizing these *template matching* algorithms [REF].

No matter the implementation details of a particular template matching algorithm, 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 they want to search the signal for. And second, the *similarity measure* which quantifies how well the template fits in a given spot of the signal.

Similarity measures

Specifying the template is mostly a philosophical question dependent on the specific task and type of signal. Choosing a similarity measure on the other hand is much more methodological. Across a number of papers, several different similarity measures have been proposed. They follow one of two general lines of thought [Brunelli & Poggiot 1997 says that these are the two common approaches; so does Goshtasby, 1984]. The first type of similarity measure aims to minimizes some value reflecting the distance between template and the signal. The second type aims to maximize some form of correlation between signal and template. [[mahalakshmi2012image]] [REFs; Brunelli1997, Brunelli2009, Goshtasby1984]. We have chosen to implement algorithms based on two of the possible measures, one minimizing the Sum of Squared Differences (MINSQ) and one maximizing the correlation (CORR) between the template and the signal. We wanted to implement both a similarity measure following a traditional distance-minimization approach and a correlational approach in order to gauge the efficacy of these approaches when applied to our field of research.

Template generation

Finding an appropriate template is more challenging. Depending on the field, the template to search for is easily specified. If you are looking to extract a particular audio-signal from a recording or some specific object in an image you can easily use that object as a template[REF]. The difficulty increases if 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 [REF]. You cannot just use "the ideal set of eyes" to identify a face. Each person comes with their own set of eyes, different from all others in some quantifiable way. A similar issue accompanies attempts of template matching approaches in ERP research. The variance in ERP signals introduced by the task or the sample of participants hinders a successful implementation of template matching algorithms using only one idealized template over all types of studies. Finding a template that reflects the influence of the task or the sample, equivalent to finding the person who's eyes you are looking for, will significantly improve performance.

Prior attempts

There have been some attempts at using an idealized signal structure as a template to identify ocular artifacts in noisy subject-level data [Li 2006] or to predict subject behavior on a single trial level [William 2020]. However, these approaches were not concerned with estimating the timing of components, but only interested in detection of a specific signal.

[Borst & Anderson 2015; Anderson et al. 2016] developed a machine-learning approach that aims to discover cognitive processing stages on a single trial level. In a first step, their algorithm makes use of multivariate pattern analysis to detect "bumps" in the EEG signal representing the onset of a new cognitive state. They assumed that entry into a new state would be accompanied by a spike in all electrodes similar to a 50 ms half-sine. This 50 ms half-sine then serves as a template with which their algorithm tries to detect those "bumps" in activity. However, their assumptions regarding the template and the location of activity are somewhat crude generalizations made necessary by noisy single trial data. Using template matching to extract component latencies from ERPs requires a more informative template.

our solution

A simple approach towards designing a more informative template would be to generate an idealized component structure. Prior knowledge about the shape, size and

location about the component of interest could then be made use of. One could draw up "the perfect P3" and attempt to use this as a template. Nonetheless, this would neglect the experiment-specific and task-specific variance in the morphology of ERP components, resulting in a template that does not optimally reflect the data. We addressed this problem by using the grand average as an experiment-specific template of the component of interest that will reflect influences of the task and sample on the morphology of the 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 [I think Kiesel 2008 recommends this]. The grand average and the specific window in which the component of interest occurs is used as a template and matched to subject-level ERPs.

Differences to other template matching studies

The grand average is the mean of all subject-level ERPs and thus minimizes the sum of squared deviations between each subject-level ERP and itself. Thus, across all subjects, it is the best approximation for each subject-level ERP. However, individual differences in the shape, size and location of the component of interest remain. The goal of our algorithm is to quantify these individual differences. This is similar to facial recognition not only detecting the presence of eyes, but also determining the person the eyes belong to. We do not only aim to detect the presence of a component, but also aim to measure individual differences in shape, size and location of the component.

To achieve this, we introduce variability into the template reflecting individual differences in the morphology of the component. Determining which transformation of the template best fits a given signal will allow us to investigate individual differences.

Subject-level transformations as the goal

We introduce variability in the template through two sources, amplitude and latency. Variability in amplitude is controlled by the parameter a, variability in latency by

the parameter b. Amplitude of the template is varied by multiplying the whole template-signal by parameter a. Latency is varied by "stretching" or "squishing" the template along the x-axis [See Figure?]. The parameter b controls the strength of this transformation. Possible transformations of the template are then compared to the signal and their similarity is evaluated. Recovering the transformation parameters $[a, b]$ that lead to the best match between template and signal thus allows us to describe individual differences in the latency of a component through the transformation parameter b_j for the participant j. In this manner, component latencies can be recovered. Researchers need to specify a time-point of the grand average denoting the latency of the component of interest. The optimal transformation parameters are then applied to the grand average latency to recover the subject-level latency of the component.

Why our algorithm may perform better

** makes use of the structure in the signal ** ** is not as influenced by peaks measurement window only controls the size of the template, should have less of an impact the decisions are similarly understandable, it is not some "black box" ML algorithm, you can plot the decision and understand the reasoning **

The present study

In order to compare the quality of our proposed algorithm we will reanalyze the same data analyzed by [Sadus 2023]. We can compare the psychometric properties of our algorithm to those of previously established algorithms, investigate the impact of different preprocessing steps and evaluate the correlation between latencies extract by our algorithm and those extracted manually be an expert ERP researcher.

In their study, [Sadus 2023] was extracted latencies of the P3 component. The P3, more specifically, the P3b, is a centroparietal positive-going component, peaking around 300 ms after stimulus onset. It is a late higher-order cognitive component [REFS] associated with [Processes]. A number of studies have demonstrated a large effect of age on the latency of the P3 across a number of tasks [REFs]. In a multiverse approach [Sadus

2023] tested several extraction methods with varying preprocessing steps in their ability to detect this age effect. They used three tasks, each measuring one of the executive functions proposed by [Miyake 2000]. To measure the functions *updating*, *shifting* and *inhibition*, they employed an Nback, a Switching and a Flanker Task, respectively. Studying three different tasks allows us to gain insight into a larger variety of higher-order cognitive processing, improving the generalizability of our findings.

Nonetheless, we will restrict our analysis to extracting P3 latencies. It usually has a broad and isolated structure with comparatively low influence of surrounding components [REF]. This makes the P3 one of the easier components for automated latency extraction approaches. After we can demonstrate proof-of-concept for P3 latency extraction, we will evaluate the algorithms ability to be extended to other ERP components.

We hope to show that a template matching approach using the grand average as a variable template can successfully extract P3 component latencies. Ideally, use of this algorithm will improve psychometric properties in comparison to prior algorithms, show high correlations with manually extracted data and present an objective and efficient way to extract ERP latencies.

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 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 $R^2 \geq 0.999$. 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. The template is described by a function $f(x, a, b)$ with n sine terms and their

respective amplitude A_i, frequency f_i and phase ϕ_i.

$$f(x, a, b) = \sum_{i=1}^{n} a * (A_i sin(b * f_i x + \phi_i)$$

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 $[a_j, b_j]$ that lead to the transformation of ERP of subject j most similar to the grand average.

MINSQ

The MINSQ algorithm minimizes the weighted sum S of squared differences d between the transformed subject signal and the grand average.

$$S = \sum_{i=1}^{n} \omega_i d_i^2$$

This weighting vector ω is computed as follows

$$\omega_i = 1 + \mathbb{1}_{[t_{start}, t_{end}]}(x_i)(10 * |\frac{y_i}{y_{max}}|)^2$$

Where

$$\mathbb{1}_{[t_{start}, t_{end}]}(x_i) = \begin{cases} 1 & \text{if } t_{start} \leq x_i \leq t_{end} \\ 0 & \text{otherwise} \end{cases}$$

$[t_{start}, t_{end}]$ denotes the measurement window, y_i the signal strength of the ith element, y_{max} the maximum voltage deflection inside the measurement window and x_i the time of the ith element.

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.

We use MATLABs *fit* function to find optimal parameters $[a_j, b_j]$ with upper and lower bounds such that $a_j \in [0, 50]; b_j \in [0.5, 3]$. As this function may be prone to

converging on local minima, we initialize 5 different start points. The solution with the best correlation between transformed template and signal that multiple start points converged on is selected.

In cases where the subject-level ERP only has signal with deflections opposite of the deflection of the component of interest, it may occur that the parameter $a_j \leq 0$. In these cases, we attempt to re-match the signal with an added parameter d shifting the entire template up or down.

$$f(x, a, b, d) = d + \sum_{i=1}^{n} a * (A_i sin(b * f_i x + \phi_i)$$

Should the algorithm again converge on a solution with $a_j \leq 0$, the latency value is set to NA.

CORR

The CORR algorithm optimizes the parameters to produce the maximal correlation r_{st} between the transformed subject-level signal s and the grand average t for values in the measurement window.

$$r_{st} = \frac{\sum(s_i - \bar{s})(t_i - \bar{t})}{\sqrt{\sum(s_i - \bar{s})^2(t_i - \bar{t})^2}}$$

t represents the vector of values of the transformed subject ERP that are in the measurement window, s the vector of values of the grand average that are in the measurement window.

MATLABs *fminbnd* function finds the optimal transformation parameter b_j maximizing r_{st}. This function will estimate the objective function for all values inside the given bounds $b_j \in [0.5, 2]$ and converge on the global optimum. We do not need to initialize a number of different starting points here.

Both algorithms then return the transformation parameters $[a_j, b_j]$ that result in optimal similarity of transformed template and signal. We use the returned value of the parameter b_j 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_j.

$$l_j = l_{GA} * b_j$$

Review methods

Researchers can manually review all choices the algorithm has made in a custom-built user interface. For both algorithms, we used the correlation between transformed template and signal as a fit-index. This can be used to only review those cases where the correlation between template and signal dips below a certain value, indicating low similarity between matched template and signal. 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

The data used here were first published by [Sadus 2023] and are a subset of the data collected by [Löffler REF]. ## Participants The data is comprised of 30 young participants (18-21 years old, mean age = 19.37, SD age = 0.76) and 30 old participants (50-60 years old, mean age = 55.83, SD age = 2.87). This sample was part of a larger study [Löffler REF] of which the 30 youngest and 30 oldest participants were selected. All participants had normal or corrected to normal vision. None of the participants had neurological or mental disorders, used psychotropic drugs, wore a pacemaker or suffered from red-green color vision deficiency. The participants provided informed consent prior to participation and received 75€ or course credit for participation. **ethics stuff?**

Tasks

Participants completed a set of 3 tasks: a Flanker Task, an Nback Task and a Switching Task. These tasks each measure one of the executive functions proposed by [Miyake, 2000]. [Löffler] programmed all tasks in MATLAB [ref] using the software package Psychtoolbox [version 3-0.13, REF]. Stimuli were presented centrally on a black background. We instructed participants to respond as quickly and accurately as possible.

Flanker Task

We administered a standard Arrow Flanker task [Eriksen & Eriksen 1974] to measure participants' *inhibition* ability. A central arrow pointing either to the left or to the right is flanked by two additional arrows to each side. These flanking arrows either point in the same or in the opposite direction as the central arrow. Participants have to respond by button press in which direction the central arrow pointed, disregarding the congruent or incongruent flanking arrows. Participants completed a set of practice trials and a total of 100 congruent and 100 incongruent trials.

Nback Task

We administered an adapted version of the Nback task from [Scharinger 2015] to measure participants' *updating* abilities. A stream of letters is shown to the participant. In the 0-back condition, participants have to indicate by keypress whether the presented letter is equivalent to a target letter. In the 1-back condition, participants have to indicate whether the currently presented letter is the same as the letter presented one trial before or not. [Löffler] also had participants complete a 2-back condition. We excluded this condition from our analysis as it did not produce clear ERPs. Participants completed a set of practice trials and a total of 96 trials per condition.

Switching Task

We administered a Switching task to measure participants' *shifting* ability. A stream of colored digits ranging from 1 to 9 was presented. Participants had to indicate whether the digit was greater than or less than 5 or whether the digit was odd or even depending on the color of the stimulus. A colored fixation cross just prior to stimulus presentation cued the rule participants had to follow in the upcoming trial. Participants had to either follow the same rule as in the trial before or switch to the other rule. Participants completed a set of practice trials and 192 trials each in the repeat and in the switch condition.

Procedure

The original study [Löffler] consisted of three test sessions. The three tasks this study focuses on were all administered in the first session. The second session also included EEG measurement with 3 additional tasks. The third session was used to measure intelligence and working memory capacity. No EEG measurements were taken here. In sessions including EEG measurements, participants were seated approximately 140cm away from a monitor in a sound-attenuated room.

EEG recording and processing

EEG was recorded using 32 Ag/AgCl scalp electrodes placed in the 10-20 system. Additional electrooculogram (EOG) measures were taken by two electrode placed above and below the left eye to correct for ocular artifacts. All impedances were kept below 5 kΩ. The signal was recorded with a sampling rate of 1000 Hz (band-pass 0.1 Hz - 100 Hz) and online-referenced to Cz. Following data acquisition, the raw data was down-sampled to 250 Hz. To remove artifacts we conducted an ICA on a cloned version of the dataset down-sampled to 100 Hz and passed through an additional high-pass filter of 1 Hz. Both the original down-sampled data as well as the ICA-dataset were cleaned by removing channels with unusually long flatlines, artifact-rates or line-noise. Channels removed were interpolated following this procedure and the data was re-referenced to the average across electrodes. ICA was conducted using the InfoMax algorithm and the resulting decomposition applied to the original dataset. ICs were labelled using the ICLabel Algorithm [Pio-Tonachini et al. 2019, aus sadus] and removed if the IC was less than 50% likely to be brain activity. We then applied Butterworth low-pass filters with varying cutoff frequencies (8 Hz, 16 Hz, 32 Hz) and a roll-off of 12 dB/octave. Data was segmented into 1200ms long segments starting 200ms before stimulus onset. Segments containing artifacts were automatically detected and removed. As a last step, we conducted a baseline correction using the 200ms prior to stimulus onset.

ERP analysis

Analyses were conducted in MATLAB [Version, citation]. We only included correct trials into analysis. We investigated the ERPs containing the P3b at the electrode Pz, similar to previous literature [REFS].

Latency extraction

To evaluate the impact of the specified measurement window, we extracted latencies three separate times using either a narrow (250-600ms), medium (200-700ms) or wide (150-900ms) measurement window. We used the peak latency approach to determine the latency of the P3 in the grand averages. We applied our algorithms using both the distance-based (MINSQ) and correlation-based (CORR) similarity measures to the data and obtained transformation parameters and fit values. Subject-level latencies were recovered from the respective grand average latency and the subject-specific transformation parameters. We also ran a traditional simple peak latency algorithm and a fractional area latency algorithm determining the 50% area latency. This was either done in an uninformed fashion, using the measurement window specified beforehand or with an adapted measurement window, based on the transformation parameters returned by the template matching algorithms. If the CORR algorithm returns the parameter $b_j = 1.1$ for participant j, for example, the medium measurement window of this participant would be transformed from 200-700ms to 245-745ms. To investigate the benefits of manually reviewing the decisions of the algorithm, we chose to review all matches that resulted in fit values $r_{st} \leq .60$. We then inspected the subject-level ERP with the matched template displayed over it and either accepted, rejected or manually determined the P3 latency of these ERPs. We also explored the impact of automatically excluding all those matches with fit values $r_{st} \leq .20$.

In our dataset, each of the 60 participants contributed 6 ERPs per task to the data. One ERP averaged over all trials of each of the two conditions and two more ERPs generated from an odd-even split on a trial level of that condition. These 360 ERPs each

from the 3 different tasks were passed through 3 different low-pass filters and subjected to analyses with 3 separate measurement windows. We applied both the correlation-based (CORR) and distance-based (MINSQ) algorithm and either reviewed the results manually, discarded bad matches automatically or accepted the results regardless of fit. We also applied both a peak latency and area latency algorithm with either uninformed measurement windows or measurement windows transformed by the transformation parameters recovered from the CORR or MINSQ algorithm. This results in $3\ tasks \times 3\ filters \times 3\ windows \times 12\ algorithms = 324$ different extraction pipelines.

Validation Techniques

We investigated the impact of latency extraction method on several psychometric properties as well as the effect size of the age effect. We also compared the methods in their correlation with manually extracted latencies, which can be seen as a benchmark for proper latency extraction [Sadus].

Reliability

We estimated reliability r_{tt} by computing Spearman-Brown corrected split-half correlations of ERPs generated from an odd-even split at the trial level.

Homogeneity

To compute a methods homogeneity r_h, we calculated its correlation with all other methods and took the mean of the Fisher-Z transformed correlation coefficients. Correlation coefficients of 1 cannot be transformed. Thus, we set all correlations $r = 1.00$ to $r = .99$. This mean correlation with other methods indicates the extent to which a particular method reflects the total of all other measures.

Effect size?

To investigate the effect of age on P3 latencies, we ran a repeated measures ANOVA with the between factor age (young vs. old) and the within factor task (Flanker, Nback, Switching).

Correlation with manual extraction

To quantify the extent to which an extraction method is able to replicate the benchmark of manual extraction, we correlated the latencies extracted by a particular method with those extracted by an expert researcher in the same task and filter condition

Impact of Filters / Measurement Window

We also quantified the impact preprocessing steps and the choice of measurement window have on the reliability and validity of extraction methods. We ran a repeated measures ANOVA with the dependent variable being either the estimated reliability of a method or its correlation with manually extracted data and the within factors filter (8 Hz vs. 16 Hz vs. 32 Hz) and measurement window (narrow vs. medium vs. wide).

Results

All data wrangling and statistical analyses were conducted in R [Version, Citation]. We used the packages [packages].

Review process

We reviewed results of the CORR and MINSQ algorithms if their fit was below $r_{st} \leq .60$ or if $0.65 \leq b_j \leq 1.7$. For the CORR algorithm, out of 9720 ERPs evaluated by the algorithm, we inspected 1045 (10.75 %) of ERPs. Of those ERPs, we rejected 23.35 % and accepted 64.21 % of the results despite their fit. We manually corrected the decisions in 12.44 % of cases. Automatically rejecting fits $r_{st} \leq .20$ discards 2.09 % of latencies. For the MINSQ algorithm, out of 9720 ERPs evaluated by the algorithm, we inspected 1063 (10.94 %) of ERPs. Of those ERPs, we rejected 28.22 % of ERPs and accepted 62.65 % of the results despite their fit. We manually corrected the decisions in 9.13 % of cases. Automatically rejecting fits $r_{st} \leq .20$ discards 1.43 % of latencies. Because the MINSQ algorithm may fail to find a valid solution if an amplitude parameter of $a = 0$ fits the signal best, we discarded 7.13 % of cases. This did not occur in the CORR algorithm.

When reporting the psychometric properties of the algorithms, we will focus on

those values passed through manual inspection. Values passed through the automatic rejection filter reported in parenthesis. Properties of uninspected pipelines can be found in the respective tables.

Reliability

An overview of Spearman-Brown corrected split-half correlations r_{tt} split by task, measurement window and filter setting can be found in [TABLE]. Across tasks, measurement windows and filter settings the CORR algorithm had mean split-half correlations of $\bar{r_{tt}} = .89$ for manually reviewed latencies ($\bar{r_{tt}} = .87$ for automatically reviewed latencies), the MINSQ algorithm $\bar{r_{tt}} = .90$ ($\bar{r_{tt}} = .86$). Uninformed area latency measures presented even better mean split-correlations $\bar{r_{tt}} = .92$. The average split-half correlation for values extracted by an expert ERP researcher was $\bar{r_{tt}} = .92$ for area latency measures and $\bar{r_{tt}} = .93$ for peak latency measures [Sadus 2023].

Homogeneity

An overview of a method's mean correlation with other methods r_h split by task, measurement window and filter setting can be found in [TABLE]. Across tasks, measurement windows and filter settings the CORR algorithm had a homogeneity of $\bar{r_h} = .81$ ($\bar{r_h} = .80$), the MINSQ algorithm $\bar{r_h} = .87$ ($\bar{r_h} = .86$). Uninformed area latency measures presented worse homogeneities. $\bar{r_h} = .78$.

Effect size

An overview of the effect size of the age effect detect by a particular method split by task, measurement window and filter setting can be found in [TABLE]. Across tasks, measurement windows and filter settings the CORR algorithm had a mean effect size of $\bar{\omega^2} = .18$ ($\bar{\omega^2} = .18$), the MINSQ algorithm $\bar{\omega^2} = .23$ ($\bar{\omega^2} = .23$). Uninformed area latency measures presented similar average effect sizes $\bar{\omega^2} = .15$. The average effect size for values extracted by an expert ERP researcher [Sadus 2023] was $\bar{\omega^2} = .18$ for area latency measures and $\bar{\omega^2} = .17$ for peak latency measures.

Correlation with manual rater

An overview of the correlation with latency values extracted by an expert ERP researcher [Sadus 2023] split by task, measurement window and filter setting can be found in [TABLE]. Across tasks, measurement windows and filter settings the CORR algorithm had mean correlations of $\bar{r} = .89$ ($\bar{r} = .88$), the MINSQ algorithm $\bar{r} = .91$ ($\bar{r} = .91$). Uninformed area latency measures had a mean correlation of $\bar{r} = .85$.

Influence of researcher degrees of freedom

The repeated measures ANOVA with split-half correlation of a particular method as a dependent variable, the between factor filter (8 Hz vs. 16 Hz vs. 32 Hz) and the within factor measurement window (narrow vs. medium vs. wide) showed no effect of filter, $\omega^2 = .00$, for the CORR algorithm, no effect, $\omega^2 = .00$, for the MINSQ algorithm and no effect, $\omega^2 = .00$, for uninformed area latency measures. The measurement windows had an effect of $\omega^2 = .17$ for the CORR algorithm, $\omega^2 = .00$ for the MINSQ algorithm and $\omega^2 = .31$ for uninformed area latency measures. Consult [TABLE] for a full overview.

When the correlation with manually extracted latencies was used as a dependent variable, the effect of filter settings for the CORR algorithm were $\omega^2 = .00$, for the MINSQ algorithm $\omega^2 = .00$ and $\omega^2 = .00$ for uninformed area latency measures. The measurement windows had an effect of $\omega^2 = .30$ for the CORR algorithm, $\omega^2 = .27$ for the MINSQ algorithm and $\omega^2 = .70$ for uninformed area latency measures. Consult [TABLE] for a full overview.

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