**Automatically Extracting ERP Component Latencies Using a Dynamic Template Matching Algorithm**

Sven Lesche1

1 Ruprecht-Karls-University Heidelberg

**Author note**

This work represents the author’s master-thesis. It was completed in R-Markdown with the code for data preparation, analysis, and communication integrated into the scripts. All code needed to replicate this work can be found at: <https://github.com/SLesche/master>.

Correspondence concerning this article should be addressed to Sven Lesche, Kurfürsten-Anlage 57, 69115 Heidelberg. E-mail: [sven.lesche@psychologie.uni-heidelberg.de](mailto:sven.lesche@psychologie.uni-heidelberg.de)

**Abstract**

Investigating the latency of event-related potentials (ERPs) in EEG data allows insight into the timing of cognitive processes. This work introduces a novel algorithm for ERP latency extraction based on template matching. The algorithm uses the grand average to generate an experiment-specific template of the component of interest and then matches transformations of that template to subject-level ERPs. These transformations enable the algorithm to measure individual differences in latency. I compared the new algorithm to peak latency and area latency algorithms by extracting P3 latencies from the same data as Sadus et al. (2023). The new algorithm displayed superior psychometric properties and correlated highly with latency values extracted manually by an expert ERP researcher. The algorithm provides a fit statistic for each subject-level ERP, indicating the degree of certainty the algorithm has in its decision. This allows researchers to automatically discard or manually review choices the algorithm has made based on an informative fit statistic. While manual review slightly improved the results, the algorithm was able to generate reliable latency values that correlate highly with decisions made manually by an expert even when the algorithm is applied in a fully automatic fashion. Overall, application of this template matching algorithm improves psychometric qualities and provides a more objective, efficient, and robust way to extract latencies of ERP components.

*Keywords:* event-related potentials, latency extraction, P3, template matching

**Automatically Extracting ERP Component Latencies Using a Dynamic Template Matching Algorithm**

In cognitive neuroscience, the accurate extraction of latencies of event-related potentials (ERPs) stands as a crucial step in understanding the timing of neural processes that underlie cognitive functions ([Luck, 2014](#ref-luck2014introduction); [Meyer et al., 1988](#ref-meyer1988modern); [Posner, 2005](#ref-posner2005timing)). Especially in studies focusing on individual differences an objective, efficient, valid, and reliable extraction process is paramount. Algorithms allow efficient and objective approaches to latency extraction, but often prove to be unreliable and invalid due to ERPs with low signal-to-noise ratios ([Clayson et al., 2013](#ref-clayson2013noise); [Kiesel et al., 2008](#ref-kiesel2008measurement); [Sadus et al., 2023](#ref-sadus2023multiverse); [Schubert et al., 2023](#ref-schubert2023robust)). To deal with this issue, researchers often forgo the use of algorithms and manually inspect each ERP, identifying patterns reflecting the component of interest by hand ([Sadus et al., 2023](#ref-sadus2023multiverse)). Manual extraction is a labor-intensive and time-consuming endeavor, improving reliability and validity at the cost of objectivity and efficiency. Low objectivity endangers replicability, it is therefore recommendable that all processing steps be automated ([Cowley et al., 2017](#ref-cowley2017computational); [Rodrigues et al., 2021](#ref-rodrigues2021epos)). This paper introduces a novel algorithm for the automatic extraction of ERP latencies using template matching. I aim to show that this new algorithm improves on existing approaches and enables more efficient, objective, reliable, and valid extraction of ERP latencies.

## Peak Latency Algorithms

Automatically extracting component latencies has long been a goal in ERP research. *Peak latency* algorithms present the earliest attempt at automating the latency extraction process. The peak latency approach involves finding the point in time within a fixed measurement window that displays the largest voltage deflection in the appropriate direction. This approach remains the most common way of latency extraction ([Kiesel et al., 2008](#ref-kiesel2008measurement); [Liesefeld, 2018](#ref-liesefeld2018estimating)), partially due to its ease of implementation and low computational requirements ([Donchin & Heffley, 1978](#ref-donchin1978multivariate)). However, ease of implementation is accompanied by several drawbacks.

Firstly, peak latency approaches are blind to the general structure of 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 time-point. The simple peak latency algorithm can be improved by only considering those maxima inside the measurement window which are also larger than the surrounding data-points ([Luck, 2014](#ref-luck2014introduction)). A spike just on the edge of the measurement window will not meet these criteria. This approach protects against the influence of noise or surrounding components on the edges of the measurement window, but remains vulnerable to high frequency noise inducing peaks inside the measurement window.

Due to high frequency noise, the maximum voltage deflection may not reflect the true point in time at which the process underlying the component reaches its maximum. Rather, the maximum voltage deflection 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. Here, the broader measurement window that most researchers use increases the likelihood of high frequency noise inducing the maximum voltage deflection ([Clayson et al., 2013](#ref-clayson2013noise)).

The sensitivity of the peak latency approach to the size of the measurement window is further increased by the *superimposition problem* ([Luck, 2014](#ref-luck2014introduction)). In larger measurement windows, later components can already influence the amplitude on the beginning and end of the signal in the measurement window. 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.

Lastly, as Luck ([2014](#ref-luck2014introduction)) so aptly states: “There is nothing special about the point at which the voltage reaches a local maximum” (p. 286). 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. Luck ([2005](#ref-luck2005ten)) 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 Algorithms

*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 to find the point in time that divides the area under the signal into a given fraction to the left and right of it. The time-point splitting the area under the signal in half, 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 ([Liesefeld, 2018](#ref-liesefeld2018estimating)). Nonetheless, area latency approaches remain highly dependent on the measurement window ([Luck, 2014](#ref-luck2014introduction)). 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 include the influence of surrounding components. Fractional Area measures thus work best for investigating an isolated component ([Luck, 2014](#ref-luck2014introduction)), limiting the applicability of area latency algorithms.

## Jackknifing

Another approach towards dealing with noisy subject-level ERPs is to mitigate that noise by averaging multiple subject-level ERPs. This technique is referred to as *jackknifing* ([Miller et al., 1998](#ref-miller1998jackknife)). A total of subject-level ERPs are averaged, each subject-level signal being left out once. This results in 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 percent of the subject-level signals that are averaged with each other, each jackknifed sub-average is quite similar to all others. This artificially decreases the error variance, which needs to be corrected for when testing for differences between groups ([Ulrich & Miller, 2001](#ref-ulrich2001using)). 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](#ref-smulders2010simplifying)) who introduced a transformation that is able to generate individual-level latency estimates.

## Comparison of Algorithms

In order to test which of these approaches is most effective, Kiesel et al. ([2008](#ref-kiesel2008measurement)) applied these different algorithms, as well as a few additional approaches not mentioned here, to a variety of ERP components. They simulated latency differences in the visual and auditory N1, the N2pc, 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 not to be the best method to detect latency effects. Its effectiveness decreases even further 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.

This finding was further corroborated by Wascher et al. ([2022](#ref-wascher2022mental)), who investigated the ability of peak latency and area latency measures combined with jackknifing to generate reliable latency measures. Area latency measures combined with jackknifing generated the most reliable ERP latencies across a variety of components. However, even the best automated approach did not lead to consistently high reliabilities.

Investigating this further, Sadus et al. ([2023](#ref-sadus2023multiverse)) assessed the influence of different preprocessing strategies and latency extraction techniques on psychometric properties of the latency values as well as the 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 and only 7 out of 40 possible pipelines showed consistently desirable reliabilities   
(), homogeneities (), and effect sizes (). 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 et al., 2023](#ref-sadus2023multiverse); [Schubert et al., 2023](#ref-schubert2023robust); [Wascher et al., 2022](#ref-wascher2022mental)). Yet, manual extraction methods are highly time-consuming and impede reproducibility. I hope to show that my algorithm can match the performance of manual extraction while providing a more efficient and objective approach for extracting individual component latency values.

## Template Matching

The algorithm proposed in this master thesis aims to resemble the process expert ERP researches employ during manual latency extraction. Most ERP researchers use the grand average to gather insight into what the component of interest *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, and a large amount of research has gone into optimizing these template matching algorithms ([Briechle & Hanebeck, 2001](#ref-briechle2001template); [Brunelli, 2009](#ref-brunelli2009template); [Lewis, 1995](#ref-lewis1995fast); [Mahalakshmi et al., 2012](#ref-mahalakshmi2012image)).

No matter what the implementation details of a particular template matching algorithm are, they all aim to answer the question “Does this (smaller) template appear in my (larger) signal?”. To achieve this, a researcher needs to specify two things. First, a template they want to search the signal for and second, a *similarity measure* that quantifies how well the template fits in a given spot of the signal.

### Similarity measures.

Specifying the template is mostly a substantive question that depends on the specific task and type of signal. Choosing a similarity measure 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](#ref-brunelli1997template); [Goshtasby et al., 1984](#ref-goshtasby1984two)). The first type of similarity measure aims to minimize some value reflecting the distance between template and the signal. The second type aims to maximize some form of correlation between signal and template. I have chosen to implement the algorithm based on one of each of the two of the possible types, the first minimizing the sum of squared differences and the second maximizing the correlation between the template and the signal. I wanted to implement both a similarity measure following a traditional distance-minimization approach and a correlation-based approach in order to gauge the efficacy of these approaches when applied to ERP research. I call the algorithm based on minimization of the squared differences *MINSQ* and the algorithm based on maximization of the correlation *MAXCOR*.

### Template generation.

Depending on the field of study, the template 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. The difficulty increases if it is not exactly certain what template you are looking for and what shape the template may take depending on various external factors. Recent research in image processing has attempted to use template matching to process faces ([Brunelli, 2009](#ref-brunelli2009template)). 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 template matching algorithms

There have been some attempts at using an idealized signal structure as a template to identify ocular artifacts in noisy subject-level data ([Li et al., 2006](#ref-li2006automatic)) or to predict subject behavior on a single-trial level ([William et al., 2020](#ref-william2020erp)). These studies were able to successfully implement template matching algorithms. However, they were not concerned with estimating the timing of components, but rather only interested in detection of a specific signal.

Borst and Anderson ([2015](#ref-borst2015discovery)) and Anderson et al. ([2016](#ref-anderson2016discovery)) 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 “bump” in the signal of 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 than a half-sine.

## Using the grand average as a template

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 used. One could draw up “the perfect P3” and attempt to use this as a template. However, this neglects the experiment-specific, task-specific, and subject-sample-specific variance in the morphology of ERP components, resulting in a template that does not optimally reflect the data. I addressed this problem by using the grand average as an experiment-specific template of the component of interest. The grand average reflects influences of the task and sample on the morphology of the idealized ERP component while retaining a high signal-to-noise ratio.

This makes the grand average a prime candidate for an experiment-specific template. It is by definition the average 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, i.e. the best predictor, for each subject-level ERP. Using the grand average to gain insight into subject-level ERPs is already quite common. For example, researchers often use the grand average to gather insight into the time window in which the component of interest occurs ([Kiesel et al., 2008](#ref-kiesel2008measurement); [Luck, 2014](#ref-luck2014introduction)).

Importantly, the goal of this algorithm is to quantify individual differences in the latency of ERP components, not only the presence of a particular ERP component. This is similar to facial recognition not only detecting the presence of eyes, but also determining the person the eyes belong to. The algorithm does not only aim to detect the presence of a component, but also aims to measure individual differences in shape, size, and location of the component.

Due to these individual differences, there may be a mismatch between a particular subject-level ERP and the grand average. For example, a specific signal may have higher amplitudes than the grand average or the component may appear earlier. Quantifying this deviation of a particular subject-level ERP from the grand average in amplitude and time-course is precisely the goal of this algorithm.

To measure the deviation in amplitude and time-course, I introduce free parameters that transform the template in its amplitude and time-course. Because of this transformation, the algorithm does not attempt to match a specific, static template. Rather, it attempts to match a variable template and then determines which transformation fits the particular subject-level ERP best. Crudely, each version of the variable template reflects an adaptation of the idealized template that I obtained using the grand average that has higher vs. lower overall amplitudes and earlier vs. later latencies of the component of interest. Determining which transformation of the template has the best fit to the subject-level ERP allows the algorithm to quantify the deviation in amplitude and time-course of a specific subject-level ERP in relation to the grand average. This results in a measure quantifying individual differences in amplitude and time-course.

## Mapping individual differences

For this master thesis, I only allow linear transformations of the template to reflect changes in amplitude and time-course. This was done in an effort to limit the complexity and increase the traceability of the algorithm. One key characteristic of baseline-corrected ERP signals is that the signal at is always equal to . Therefore, I chose to restrict the algorithm to linear transformations of the template that also do not disturb this property. Horizontal “shifts” of the template’s signal, for example, would lead to non-zero signal at . Therefore, the algorithm is not allowed to introduce variability into the template through horizontal shifts. For this master thesis, I only implemented two transformations, controlled by one free parameter each.

Variability in amplitude is controlled by the transformation parameter , variability in latency by the parameter . Amplitude of the template is varied by multiplying the whole template-signal by parameter . Latency is varied by “stretching” or “compressing” the template along the x-axis (see Figure 1). Importantly, this does not “shift” the signal, which would lead to a non-zero signal at the origin. The parameter controls the strength of this transformation. Possible transformations of the template are then compared to the signal and their similarity is evaluated. Recovering the set of transformation parameters that lead to the best match between template and signal thus allows me to quantify individual differences in the latency of a component. Whereas the matching procedure is based on the entire template, I can also apply the transformation parameters to a specific time-point. Researchers can extract subject-level component latencies by specifying 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 and result in the subject-level latency of the component.

**Figure 1**

*Scaling Templates Horizontally*



## Measurement windows

The two key ingredients for template matching are present. I use the grand average to construct a variable template that is then matched to a particular subject-level ERP by optimizing either the distance-criterion *MINSQ* or the correlation-criterion *MAXCOR*. However, during manual inspection of ERP signals, not all signal is considered equal. Depending on the component of interest, different points in time of the signal are viewed as more or less relevant. When an early component, e.g. the P1, is the target, the activity after 300ms becomes less relevant. Similarly, when a late component is of interest, early activity becomes less relevant than activity at those times where the component typically occurs.

I extended the template matching algorithm to incorporate this information. A time window can be supplied to the algorithm that specifies at which time-points the signal of the template is more important. Similar to the measurement windows used in previous approaches, this time window should be constructed based on visual inspection of the grand average ([Kiesel et al., 2008](#ref-kiesel2008measurement)). The similarity between transformed template and signal is weighted to be more important in that time window than outside of it. For example, if a late component, like the P3, is the target and the grand average shows that the activity of this component occurs mostly between 200 - 700 ms, the algorithm can take that information into account by weighting the similarity measure during that time with a higher weight than signal that lies outside of 200 - 700 ms.

## Why the algorithm may perform better

This algorithm aims to address some of the issues faced by other algorithms. It makes use of the entire component structure to construct the template that is matched to subject-level ERPs. This reflects the decision process of expert ERP researchers and enables an intuitive understanding of the decisions made by the algorithm. Because the similarity measures take the whole component structure into account, they are robust to peaks introduced by high frequency noise. Furthermore, the measurement window set by the researcher only impacts the size and shape of the template. It has no direct connection to the subject-level ERP. The influence of measurement windows on the extracted latencies should thus be lower than in peak latency or area latency algorithms. Lastly, it is important to note that this is not a machine learning algorithm with a neural net representing some “black box” decision making algorithm. Simplicity and traceability of the decision process was an important goal, allowing more insight into the benefits and drawbacks of the algorithm.

## The present study

In order to compare the quality of my proposed algorithm with the quality of previously proposed algorithms, I will reanalyze the same data analyzed by Sadus et al. ([2023](#ref-sadus2023multiverse)). I compare the psychometric properties of the algorithm to those of previously established algorithms, investigate the impact of different preprocessing steps, and evaluate the correlation between latencies extracted by my algorithm and those extracted manually by an expert ERP researcher.

In their study, Sadus et al. ([2023](#ref-sadus2023multiverse)) extracted latencies of the P3b component, henceforth simply referred to as P3. The P3 is a centro-parietal positive-going component, peaking around 300 ms after stimulus onset. It is often associated with higher-order cognitive processes ([Donchin, 1981](#ref-donchin1981surprise); [Duncan-Johnson, 1981](#ref-duncan1981young); [McCarthy & Donchin, 1981](#ref-mccarthy1981metric); [Polich, 2007](#ref-polich2007updating), [2012](#ref-polich2012neuropsychology); [Verleger, 2020](#ref-verleger2020effects)). A number of studies have demonstrated a large effect of age on the latency of the P3 across a number of tasks, with older participants displaying systematically later P3 peaks than their younger counterparts ([Friedman, 2011](#ref-friedman2012components); [Scrivano & Kieffaber, 2022](#ref-scrivano2022behavioral)). In a multiverse approach, Sadus et al. ([2023](#ref-sadus2023multiverse)) tested several extraction methods with varying preprocessing steps in their ability to detect this age effect. They also used three tasks, each measuring one of the executive functions proposed by Miyake et al. ([2000](#ref-miyake2000unity)). To measure the functions *updating*, *shifting*, and *inhibition*, they employed an Nback Task, a Switching Task, and a Flanker Task, respectively. Studying three different tasks allows insight into a larger variety of higher-order cognitive processing, improving the generalizability of my findings.

For the present work, I will restrict the analysis to extracting P3 latencies, as the P3 usually has a broad and isolated structure with comparatively low influence of surrounding components ([Luck, 2014](#ref-luck2014introduction)). This makes it one of the easier components to extract using automated latency extraction approaches. After I can demonstrate proof-of-concept for P3 latency extraction, I will evaluate whether I can apply the algorithm to other ERP components in future work.

To investigate the impact of choices made by the researcher during preprocessing and analysis, I will vary the low-pass filter frequency as well as the measurement window used during template matching. This will allow me to gain insight into which combination of preprocessing steps and size of measurement window leads to most optimal results.

I hope to show that a template matching algorithm using the grand average as a variable template can successfully extract subject-level 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.

# Implementation

I implemented the algorithm in MATLAB (Version 2022b) ([The Math Works, 2022](#ref-matlab2022b)).

Let the vector denote the time-points at which a subject-level signal of subject is taken and the signal of the template. In order to transform the template, I use MATLABs Curve Fitting Toolbox ([The Math Works, 2022](#ref-matlab2022b)) to find a *sum of sines* function with sine terms

where denotes a time-point, represents the amplitude-scaling parameter, represents the latency-scaling parameter and , , and denote amplitude, frequency, phase. The sum of sines parameter vectors , , and are optimized such that predicts with . The number of sine terms is increased until this fit is achieved. After the sum of sines function is found, the variable template is given by the function with the parameter allowing scaling of the amplitude and the parameter allowing “compressing” or “stretching” the template along the x-Axis.

As these transformations also change the measurement window, I 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, I use different functions to find the set of optimal parameters that lead to the optimal transformation for a given subject .

## MINSQ

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

The weighting vector that I use to place emphasis on those time-points of the signal that were specified by the researcher beforehand is computed as follows:

denotes the measurement window, the signal strength of the th element, the maximum voltage deflection inside the measurement window and the time of the th element.

The resulting weighting vector places more emphasis on fitting the template within the specified measurement window and to places in the signal where the voltage deflection is high.

I use MATLABs *fit* function to find the set of optimal parameters with upper and lower bounds such that . As this function may be prone to converging on local minima, I initialize 5 different start points. Only solutions that at least two of the five starting points converged on are considered. Of those, the algorithm selects the solution with the best correlation between transformed template and signal.

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 . In these cases, I attempt to re-match the signal with the parameter added to the variable template shifting the entire template up or down.

Should the algorithm again converge on a solution with , the latency value is set to NA.

## MAXCOR

The MAXCOR algorithm optimizes the parameters to produce the maximum correlation between the signal and the transformed template for values where . Time-points outside the measurement window are not allowed to influence the correlation.

where

and

represents the vector of values of the transformed template that are in the measurement window, the vector of values of the signal are in the measurement window. As the correlation-coefficient is independent of translation and scaling, varying the parameter will not impact the correlation . I therefore set and only optimize .

Because I only need to optimize one parameter, I use MATLABs *fminbnd* function to find the optimal transformation parameter maximizing . This function will estimate the correlation for all values inside the given bounds and converge on the global optimum. Hence, I do not need to initialize a number of different starting points here.

### Recovering subject-level latencies.

For both approaches, I use the returned value of the parameter to transform the component latency specified by the researcher in the grand average to the component latency of the subject-level ERP signal .

## Review methods

Researchers can manually review all choices the algorithm has made in a custom-built user interface (see Figure 2). For both approaches, I used the correlation between transformed template and signal as a fit-index. I chose to use the correlation as the final fit-index because it is scale-invariant and provides an intuitive quantification of how strongly the structure of the matched template resembles the structure of the subject-level ERP. The fit index 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. I will investigate the additional benefits that a manual review process provides over accepting the choices as-is or automatically discarding those matches with correlations .

**Figure 2**

*User Interface for Manual Review Process*



# Method

All analyses are based on data that were first published by Sadus et al. ([2023](#ref-sadus2023multiverse)) and are a subset of the data collected by Löffler et al. ([2022](#ref-loffler2022common)).

## Participants

The present sample consists 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), representing the 30 youngest and 30 oldest participants from the overall study ([Löffler et al., 2022](#ref-loffler2022common)). 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. All participants provided informed consent prior to participation and received 75€ or course credit for participation.

## Tasks

All participants completed a set of 3 tasks: a Flanker Task, an Nback Task, and a Switching Task. Each task measures one of the executive functions proposed by Miyake et al. ([2000](#ref-miyake2000unity)). Löffler et al. ([2022](#ref-loffler2022common)) programmed all tasks in MATLAB ([The Math Works, 2022](#ref-matlab2022b)) using the software package Psychtoolbox (Version 3-0.13) ([Brainard & Vision, 1997](#ref-brainard1997psychophysics); [Kleiner et al., 2007](#ref-kleiner2007psychtoolbox); [Pelli & Vision, 1997](#ref-pelli1997videotoolbox)). Stimuli were presented centrally on a black background. All participants were instructed to respond as quickly and accurately as possible.

### Flanker Task.

A standard Arrow Flanker task ([Eriksen & Eriksen, 1974](#ref-eriksen1974effects)) was administered 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. All participants have to indicate by button press in which direction the central arrow pointed, disregarding the congruent or incongruent flanking arrows. All participants completed a set of practice trials and a total of 100 congruent and 100 incongruent trials.

### Nback Task.

An adapted version of the Nback task from ([Scharinger et al., 2015](#ref-scharinger2015flanker)) was administered to measure participants’ *updating* abilities. A stream of letters is presented. 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 et al. ([2022](#ref-loffler2022common)) also had participants complete a 2-back condition. Following Sadus et al. ([2023](#ref-sadus2023multiverse)), I excluded this condition from the analysis as it did not produce clear ERPs. In total, all participants completed a set of practice trials and 96 trials per condition.

### Switching Task.

A Switching task was administered to measure participants’ *shifting* ability. A stream of colored digits ranging from 1 to 9 was presented. All 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 consisted of three test sessions. The three tasks analyzed here 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 continuously using 32 equidistant Ag/AgCl electrodes. Additional electrooculogram (EOG) measures were taken by two electrodes 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 and online-referenced to Cz. To remove artifacts, an ICA was conducted on a cloned version of the dataset that was down-sampled to 200 Hz and passed through an additional high-pass filter of 1 Hz. Both the original data as well as the ICA-dataset were cleaned by removing line-noise using the CleanLine function ([Mullen, 2012](#ref-mullen2012cleanline)). A critical z-value of 3.29 was used for z-value based bad channels detection as recommended in the EPOS pipeline ([Rodrigues et al., 2021](#ref-rodrigues2021epos)). Channels that were removed following this procedure were interpolated and the data was re-referenced to the average across electrodes. The threshold for large fluctuations was set and data had to be more than 5 SDs from the mean to exceed the probability threshold. Based on these settings, segments containing artifacts were automatically detected and removed in the ICA-dataset up to a maximum of 5% of segments per iteration. ICA was conducted using the InfoMax algorithm and the resulting decomposition applied to the original dataset. ICs were labelled using the ICLabel Algorithm ([Pion-Tonachini et al., 2019](#ref-pion2019iclabel)) and removed if the IC was less than 50% likely to be brain activity. A Butterworth low-pass filter with varying cut-off frequencies (8 Hz, 16 Hz, 32 Hz) and a roll-off of 12 dB/octave was then applied and data were segmented into 1200ms long segments starting 200ms before stimulus onset. Again, segments containing artifacts were automatically detected and removed. As a last step, segments were baseline corrected using the 200ms prior to stimulus onset.

## ERP analysis

ERP analyses were conducted in MATLAB (Version 2022b) ([The Math Works, 2022](#ref-matlab2022b)). I only included correct trials into the analysis and investigated the P3 at the electrode Pz ([Polich, 2012](#ref-polich2012neuropsychology); [Verleger, 2020](#ref-verleger2020effects)).

### Latency extraction.

To evaluate the impact of the specified measurement window, I extracted latencies three separate times using either a narrow (250-600ms), medium (200-700ms) or wide (150-900ms) measurement window. These measurement windows are based on the grand average of each task and capture either only the central part of the positive-going peak (narrow window), the onset and offset of the P3 (medium window), or the full P3 and some surrounding signal (wide window). I used the peak latency approach to determine the latency of the P3 in the grand averages that is then used to recover subject-level latencies in the template matching algorithm. I applied my algorithm using both the distance-based (MINSQ) and correlation-based (MAXCOR) similarity measures to the data and obtained transformation parameters and fit values.

To investigate the benefits of manually reviewing the decisions of the algorithm, I chose to review all matches that resulted in fit values . I then inspected the subject-level ERP with the matched template superimposed on the subject-level ERP to allow visual confirmation of the fit. Using the interface, I either accepted, rejected, or manually determined the P3 peak latency of these ERPs. I also explored the impact of automatically excluding all those matches with fit values .

In the present dataset, each of the 60 participants contributed 6 ERPs per task to the data. All participants contributed one ERP averaged over all trials of each of the two conditions and two more ERPs that were generated by 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. I applied both the correlation-based (MAXCOR) and distance-based (MINSQ) algorithm and either reviewed the results manually, discarded bad matches automatically or accepted the results regardless of fit. I also applied both a peak latency and 50% area latency algorithm. For the area latency algorithm, I set all values below zero to be equal to zero to combat low frequency noise ([Liesefeld, 2018](#ref-liesefeld2018estimating); [Luck, 2014](#ref-luck2014introduction)) and determined the exact latency by linear interpolation between data points. This results in 3 tasks 3 filters 3 windows algorithms = 216 different extraction pipelines.

## Validation Techniques

I investigated the impact of the latency extraction method on several measures of psychometric quality. I estimated reliability by computing Spearman-Brown corrected split-half correlations of latencies that were extracted from subject-level ERPs based on even trials with those extracted from subject-level ERPs based on odd trials. I assessed the validity of my algorithm through measures of homogeneity, the effect sizes of the age effect, and the intraclass correlation of latencies that were extracted by the algorithm with latencies that were extracted by an expert ERP researcher in the same task and filter condition. The intraclass correlation used a two-way random-effects model focusing on absolute agreement.

To compute a methods homogeneity , I calculated the correlation of latencies that were extracted using that method with all other methods and took the mean of the Fisher-Z transformed correlation coefficients. Correlation coefficients of 1 cannot be transformed. Thus, I set all correlations to . This mean correlation with other methods indicates the extent to which a particular method reflects the total of all other measures ([Kline, 1986](#ref-kline1986handbook)).

To investigate the effect of age on P3 latencies, I ran a repeated measures ANOVA with the between factor age (young vs. old) and the within factor condition. The condition factor depends on the task analyzed. In the Flanker task, incongruent trials are compared to congruent trials. In the Nback task, 0-back trials are compared to 1-back trials. In the Switching task, repeat trials are compared to switch trials.

# Results

All data preprocessing and statistical analyses were conducted using R [Version 4.1.3; R Core Team ([2022](#ref-R-base))][[1]](#footnote-1).

## Review process

I reviewed results of the MINSQ and MAXCOR approaches if their fit was below   
 or if or . For the MINSQ algorithm, out of 9720 ERPs evaluated by the algorithm, I inspected 1063 (10.94 %). Of those ERPs, I rejected 28.22 % of ERPs and accepted 62.65 % of the results despite their fit. I manually corrected the decisions in 9.13 % of ERPs I reviewed. Automatically rejecting fits with discards 1.43 % of latencies. Because the MINSQ algorithm may fail to find a valid solution if an amplitude parameter of fits the signal best, I discarded 7.13 % of the 9720 total cases. This did not occur in the MAXCOR algorithm. For the MAXCOR algorithm, out of 9720 ERPs evaluated by the algorithm, I inspected 1045 (10.75 %). Of those ERPs, I rejected 23.35 % and accepted 64.21 % of the results despite their fit. I manually corrected the decisions in 12.44 % of ERPs I reviewed. Automatically rejecting fits with discards 2.09 % of latencies.

When reporting the psychometric properties of the algorithm, I will focus on those values passed through manual inspection. Values that were gained from a pipeline ending with the automatic rejection filter are reported in parentheses. Properties of uninspected pipelines can be found in the respective tables.

## Reliability

I estimated reliability using Spearman-Brown corrected split-half correlations. An overview of reliability split by task, measurement window, and filter setting can be found in Tables 1 - 3. Across tasks, measurement windows, and filter settings the MAXCOR algorithm had a mean reliability of .85 for manually reviewed latencies ( .83 for automatically reviewed latencies). The MINSQ algorithm had a mean reliability of .88 ( .82). Area latency measures showed a mean reliability of .91. Peak latency measures had a mean reliability of .79. The average reliability for values extracted by an expert ERP researcher was for area latency measures and for peak latency measures ([Sadus et al., 2023](#ref-sadus2023multiverse)).

**Table 1**

Reliability of different algorithms - Flanker Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.96 | 0.95 | 0.95 | 0.93 | 0.93 | 0.93 | 0.96 | 0.91 |
| medium | 0.97 | 0.97 | 0.97 | 0.87 | 0.87 | 0.95 | 0.95 | 0.84 |
| wide | 0.88 | 0.88 | 0.96 | 0.95 | 0.97 | 0.94 | 0.94 | 0.82 |
| 16 | narrow | 0.95 | 0.94 | 0.94 | 0.90 | 0.90 | 0.92 | 0.96 | 0.85 |
| medium | 0.97 | 0.97 | 0.97 | 0.80 | 0.80 | 0.86 | 0.95 | 0.81 |
| wide | 0.88 | 0.87 | 0.96 | 0.93 | 0.95 | 0.97 | 0.94 | 0.81 |
| 32 | narrow | 0.96 | 0.96 | 0.95 | 0.92 | 0.92 | 0.96 | 0.96 | 0.84 |
| medium | 0.95 | 0.95 | 0.97 | 0.92 | 0.92 | 0.97 | 0.95 | 0.80 |
| wide | 0.94 | 0.96 | 0.96 | 0.91 | 0.91 | 0.97 | 0.94 | 0.63 |
| *Note.* Values represent the Spearman-Brown corrected split-half correlation of a particular extraction method; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 2**

Reliability of different algorithms - Nback Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.81 | 0.83 | 0.82 | 0.95 | 0.95 | 0.95 | 0.91 | 0.80 |
| medium | 0.82 | 0.88 | 0.83 | 0.75 | 0.76 | 0.89 | 0.93 | 0.79 |
| wide | 0.65 | 0.69 | 0.90 | 0.90 | 0.93 | 0.93 | 0.88 | 0.64 |
| 16 | narrow | 0.75 | 0.75 | 0.76 | 0.80 | 0.84 | 0.87 | 0.88 | 0.76 |
| medium | 0.78 | 0.78 | 0.81 | 0.86 | 0.86 | 0.93 | 0.92 | 0.76 |
| wide | 0.66 | 0.74 | 0.75 | 0.87 | 0.88 | 0.91 | 0.86 | 0.69 |
| 32 | narrow | 0.75 | 0.77 | 0.85 | 0.80 | 0.80 | 0.78 | 0.84 | 0.76 |
| medium | 0.64 | 0.71 | 0.71 | 0.69 | 0.69 | 0.88 | 0.92 | 0.80 |
| wide | 0.54 | 0.61 | 0.76 | 0.82 | 0.85 | 0.93 | 0.88 | 0.68 |
| *Note.* Values represent the Spearman-Brown corrected split-half correlation of a particular extraction method; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 3**

Reliability of different algorithms - Switching Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.70 | 0.82 | 0.79 | 0.65 | 0.71 | 0.84 | 0.94 | 0.90 |
| medium | 0.76 | 0.92 | 0.93 | 0.80 | 0.84 | 0.90 | 0.91 | 0.89 |
| wide | 0.65 | 0.69 | 0.66 | 0.71 | 0.72 | 0.84 | 0.86 | 0.81 |
| 16 | narrow | 0.61 | 0.77 | 0.80 | 0.40 | 0.70 | 0.86 | 0.95 | 0.84 |
| medium | 0.89 | 0.94 | 0.93 | 0.86 | 0.89 | 0.86 | 0.91 | 0.86 |
| wide | 0.65 | 0.67 | 0.66 | 0.81 | 0.81 | 0.81 | 0.88 | 0.82 |
| 32 | narrow | 0.76 | 0.92 | 0.87 | 0.63 | 0.82 | 0.80 | 0.89 | 0.86 |
| medium | 0.85 | 0.91 | 0.90 | 0.78 | 0.82 | 0.77 | 0.92 | 0.86 |
| wide | 0.68 | 0.73 | 0.72 | 0.65 | 0.64 | 0.63 | 0.86 | 0.83 |
| *Note.* Values represent the Spearman-Brown corrected split-half correlation of a particular extraction method; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

## 

## Homogeneity

An overview of a method’s mean correlation with other methods split by task, measurement window, and filter setting can be found in Tables 4 - 6. Across tasks, measurement windows, and filter settings the MAXCOR algorithm had a mean homogeneity of .81   
( .80), and the MINSQ algorithm a mean homogeneity of .87 ( .86). The mean homogeneity of area latency measures was .78. The mean homogeneity of peak latency measures was .71. Homogeneity is larger when a medium-sized measurement window is employed compared to the other measurement windows for both the MINSQ ( 0.87 vs.  0.82) and the MAXCOR approach ( 0.84 vs.  0.75).

**Table 4**

Homogeneity of different algorithms - Flanker Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.77 | 0.90 | 0.89 | 0.92 | 0.92 | 0.92 | 0.83 | 0.85 |
| medium | 0.90 | 0.90 | 0.90 | 0.91 | 0.91 | 0.91 | 0.88 | 0.85 |
| wide | 0.88 | 0.89 | 0.88 | 0.91 | 0.97 | 0.91 | 0.85 | 0.73 |
| 16 | narrow | 0.77 | 0.88 | 0.88 | 0.93 | 0.93 | 0.93 | 0.83 | 0.79 |
| medium | 0.90 | 0.90 | 0.90 | 0.93 | 0.93 | 0.93 | 0.88 | 0.74 |
| wide | 0.83 | 0.89 | 0.88 | 0.93 | 0.95 | 0.95 | 0.84 | 0.65 |
| 32 | narrow | 0.73 | 0.86 | 0.87 | 0.92 | 0.92 | 0.90 | 0.83 | 0.78 |
| medium | 0.90 | 0.90 | 0.90 | 0.91 | 0.91 | 0.97 | 0.88 | 0.73 |
| wide | 0.83 | 0.84 | 0.87 | 0.79 | 0.87 | 0.97 | 0.85 | 0.70 |
| *Note.* Values represent the average correlation of a particular extraction method with other extraction methods; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 5**

Homogeneity of different algorithms - Nback Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.76 | 0.77 | 0.77 | 0.82 | 0.83 | 0.80 | 0.76 | 0.76 |
| medium | 0.76 | 0.79 | 0.78 | 0.87 | 0.88 | 0.87 | 0.80 | 0.71 |
| wide | 0.69 | 0.70 | 0.75 | 0.82 | 0.83 | 0.85 | 0.77 | 0.67 |
| 16 | narrow | 0.69 | 0.70 | 0.77 | 0.86 | 0.86 | 0.86 | 0.76 | 0.74 |
| medium | 0.81 | 0.84 | 0.82 | 0.84 | 0.85 | 0.85 | 0.81 | 0.70 |
| wide | 0.67 | 0.71 | 0.78 | 0.81 | 0.83 | 0.85 | 0.77 | 0.67 |
| 32 | narrow | 0.70 | 0.70 | 0.76 | 0.84 | 0.84 | 0.87 | 0.76 | 0.74 |
| medium | 0.77 | 0.80 | 0.81 | 0.86 | 0.87 | 0.86 | 0.80 | 0.69 |
| wide | 0.71 | 0.75 | 0.78 | 0.81 | 0.82 | 0.82 | 0.77 | 0.64 |
| *Note.* Values represent the average correlation of a particular extraction method with other extraction methods; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 6**

Homogeneity of different algorithms - Switching Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.66 | 0.76 | 0.75 | 0.79 | 0.80 | 0.80 | 0.71 | 0.71 |
| medium | 0.79 | 0.80 | 0.80 | 0.82 | 0.83 | 0.82 | 0.75 | 0.67 |
| wide | 0.51 | 0.55 | 0.64 | 0.55 | 0.56 | 0.73 | 0.67 | 0.57 |
| 16 | narrow | 0.62 | 0.75 | 0.72 | 0.77 | 0.77 | 0.76 | 0.67 | 0.69 |
| medium | 0.80 | 0.81 | 0.80 | 0.81 | 0.82 | 0.81 | 0.74 | 0.67 |
| wide | 0.48 | 0.53 | 0.57 | 0.65 | 0.65 | 0.65 | 0.63 | 0.58 |
| 32 | narrow | 0.50 | 0.71 | 0.71 | 0.72 | 0.73 | 0.73 | 0.67 | 0.67 |
| medium | 0.79 | 0.82 | 0.80 | 0.82 | 0.83 | 0.81 | 0.74 | 0.67 |
| wide | 0.48 | 0.57 | 0.62 | 0.62 | 0.63 | 0.70 | 0.63 | 0.53 |
| *Note.* Values represent the average correlation of a particular extraction method with other extraction methods; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

## 

## Effect size

An overview of the effect size of the age effect estimated by a particular method and split by task, measurement window, and filter setting can be found in Tables 7 - 9. Across tasks, measurement windows, and filter settings, the MAXCOR algorithm had a mean effect size of .18 ( .18). The MINSQ algorithm had a mean effect size of .22 ( .22). Area latency measures showed average effect sizes of .14. Peak latency measures showed a mean effect size of .08. The average effect size for values extracted by an expert ERP researcher was for area latency measures and for peak latency measures ([Sadus et al., 2023](#ref-sadus2023multiverse)).

In the Flanker task data, the MINSQ algorithm with a 32 Hz low-pass filter and a narrow measurement window yielded the largest effect sizes, while peak latency algorithms yielded the lowest effect sizes. In the Switching task, the MINSQ algorithm combined with a wide measurement window yielded the largest effect size estimates, while area latency and peak latency algorithms showed effect sizes of in some conditions. Similarly, in the Nback task, the MAXCOR algorithm combined with a wide measurement window showed the largest effect sizes while area latency and peak latency algorithms showed effect sizes of in some conditions.

**Table 7**

Effect size for the age effect - Flanker Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.39 | 0.41 | 0.41 | 0.51 | 0.51 | 0.51 | 0.43 | 0.38 |
| medium | 0.28 | 0.28 | 0.28 | 0.47 | 0.47 | 0.47 | 0.35 | 0.24 |
| wide | 0.30 | 0.29 | 0.30 | 0.45 | 0.44 | 0.45 | 0.24 | 0.11 |
| 16 | narrow | 0.38 | 0.39 | 0.41 | 0.48 | 0.48 | 0.48 | 0.43 | 0.37 |
| medium | 0.30 | 0.30 | 0.30 | 0.43 | 0.43 | 0.43 | 0.35 | 0.12 |
| wide | 0.33 | 0.31 | 0.32 | 0.45 | 0.49 | 0.46 | 0.25 | 0.06 |
| 32 | narrow | 0.34 | 0.36 | 0.38 | 0.55 | 0.55 | 0.55 | 0.44 | 0.29 |
| medium | 0.31 | 0.31 | 0.31 | 0.48 | 0.48 | 0.47 | 0.35 | 0.11 |
| wide | 0.32 | 0.31 | 0.33 | 0.41 | 0.40 | 0.50 | 0.24 | 0.02 |
| *Note.* Values represent the effect-size of the age effect; values greater than 0.03 are colored in green, less than 0.03 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 8**

Effect size for the age effect - Nback Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.07 | 0.07 | 0.06 | 0.08 | 0.10 | 0.06 | 0.11 | 0.09 |
| medium | 0.06 | 0.05 | 0.05 | 0.08 | 0.09 | 0.08 | 0.07 | 0.01 |
| wide | 0.00 | 0.00 | 0.03 | 0.04 | 0.04 | 0.04 | 0.00 | 0.00 |
| 16 | narrow | 0.07 | 0.07 | 0.05 | 0.10 | 0.10 | 0.09 | 0.12 | 0.10 |
| medium | 0.11 | 0.10 | 0.05 | 0.08 | 0.07 | 0.05 | 0.08 | 0.02 |
| wide | 0.08 | 0.11 | 0.10 | 0.08 | 0.09 | 0.08 | 0.00 | 0.00 |
| 32 | narrow | 0.05 | 0.05 | 0.03 | 0.06 | 0.06 | 0.07 | 0.11 | 0.10 |
| medium | 0.09 | 0.09 | 0.07 | 0.06 | 0.06 | 0.06 | 0.06 | 0.04 |
| wide | 0.12 | 0.15 | 0.13 | 0.08 | 0.09 | 0.06 | 0.00 | 0.00 |
| *Note.* Values represent the effect-size of the age effect; values greater than 0.03 are colored in green, less than 0.03 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

**Table 9**

Effect size for the age effect - Switching Task

| filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| 8 | narrow | 0.11 | 0.08 | 0.09 | 0.04 | 0.02 | 0.08 | 0.06 | 0.04 |
| medium | 0.12 | 0.11 | 0.11 | 0.03 | 0.01 | 0.07 | 0.03 | 0.01 |
| wide | 0.16 | 0.23 | 0.19 | 0.24 | 0.27 | 0.22 | 0.00 | 0.00 |
| 16 | narrow | 0.10 | 0.08 | 0.09 | 0.05 | 0.03 | 0.05 | 0.05 | 0.03 |
| medium | 0.11 | 0.11 | 0.11 | 0.04 | 0.02 | 0.04 | 0.01 | 0.00 |
| wide | 0.13 | 0.17 | 0.18 | 0.22 | 0.22 | 0.21 | 0.00 | 0.00 |
| 32 | narrow | 0.10 | 0.12 | 0.12 | 0.09 | 0.17 | 0.09 | 0.06 | 0.03 |
| medium | 0.12 | 0.11 | 0.11 | 0.08 | 0.06 | 0.15 | 0.01 | 0.03 |
| wide | 0.11 | 0.12 | 0.13 | 0.27 | 0.26 | 0.23 | 0.00 | 0.00 |
| *Note.* Values represent the effect-size of the age effect; values greater than 0.03 are colored in green, less than 0.03 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | |

## Correlation with manual rater

An overview of the intraclass correlation of latencies that were extracted by the algorithm with latency values extracted by an expert ERP researcher ([Sadus et al., 2023](#ref-sadus2023multiverse)) split by task, measurement window, and filter settings can be found in Tables 10 - 12. Across tasks, measurement windows, and filter settings, the MAXCOR algorithm had mean correlations of   
 .82 ( .80) with manually extracted latencies. The MINSQ algorithm had mean intraclass correlations of .85 ( .83). Area latency measures had a mean intraclass correlation of   
 .75, peak latency measures .75.

In the Flanker task data, the MINSQ approach combined with a medium measurement window led to intraclass correlations with manually extracted latencies consistently above , even in the fully automatic approach. The area latency approach showed intraclass correlations between and . A similar pattern emerged for the Nback and Switching tasks. The MINSQ approach combined with a medium measurement window led to the highest intraclass correlations, showing values consistently above . Other measures or other measurement windows displayed less consistency, showing intraclass correlations in some conditions.

**Table 10**

Intraclass correlation with manually extracted latencies - Flanker Task

| manual | filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| area | 8 | narrow | 0.83 | 0.95 | 0.95 | 0.97 | 0.97 | 0.97 | 0.82 | 0.92 |
| 8 | medium | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.97 | 0.93 | 0.93 |
| 8 | wide | 0.96 | 0.96 | 0.96 | 0.97 | 0.97 | 0.97 | 0.91 | 0.79 |
| 16 | narrow | 0.80 | 0.93 | 0.94 | 0.96 | 0.96 | 0.96 | 0.82 | 0.88 |
| 16 | medium | 0.97 | 0.97 | 0.97 | 0.96 | 0.96 | 0.96 | 0.93 | 0.89 |
| 16 | wide | 0.90 | 0.96 | 0.96 | 0.96 | 0.96 | 0.96 | 0.90 | 0.78 |
| 32 | narrow | 0.81 | 0.91 | 0.92 | 0.94 | 0.94 | 0.93 | 0.81 | 0.84 |
| 32 | medium | 0.95 | 0.95 | 0.95 | 0.94 | 0.94 | 0.94 | 0.92 | 0.85 |
| 32 | wide | 0.94 | 0.94 | 0.94 | 0.62 | 0.62 | 0.94 | 0.92 | 0.80 |
| peak | 8 | narrow | 0.64 | 0.96 | 0.88 | 0.91 | 0.91 | 0.91 | 0.72 | 0.89 |
| 8 | medium | 0.89 | 0.89 | 0.89 | 0.91 | 0.91 | 0.91 | 0.79 | 0.91 |
| 8 | wide | 0.86 | 0.86 | 0.86 | 0.91 | 0.91 | 0.91 | 0.75 | 0.74 |
| 16 | narrow | 0.77 | 0.94 | 0.95 | 0.96 | 0.96 | 0.96 | 0.79 | 0.94 |
| 16 | medium | 0.95 | 0.95 | 0.95 | 0.96 | 0.96 | 0.96 | 0.87 | 0.89 |
| 16 | wide | 0.86 | 0.93 | 0.93 | 0.96 | 0.96 | 0.96 | 0.81 | 0.75 |
| 32 | narrow | 0.84 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.80 | 0.88 |
| 32 | medium | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.93 | 0.89 | 0.85 |
| 32 | wide | 0.90 | 0.90 | 0.90 | 0.62 | 0.62 | 0.93 | 0.85 | 0.81 |
| *Note.* Values represent the intraclass-correlation of latency values extracted by a certain algorithm with latencies extracted by an expert ERP researcher; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); manual = expert researcher either used a peak latency or area latency approach; filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | | |

**Table 11**

Intraclass correlation with manually extracted latencies - Nback Task

| manual | filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| area | 8 | narrow | 0.78 | 0.84 | 0.78 | 0.78 | 0.79 | 0.72 | 0.78 | 0.86 |
| 8 | medium | 0.78 | 0.78 | 0.76 | 0.83 | 0.83 | 0.83 | 0.89 | 0.78 |
| 8 | wide | 0.70 | 0.70 | 0.73 | 0.78 | 0.78 | 0.82 | 0.85 | 0.62 |
| 16 | narrow | 0.65 | 0.70 | 0.70 | 0.86 | 0.86 | 0.85 | 0.77 | 0.84 |
| 16 | medium | 0.84 | 0.84 | 0.83 | 0.84 | 0.84 | 0.84 | 0.88 | 0.72 |
| 16 | wide | 0.70 | 0.70 | 0.82 | 0.75 | 0.75 | 0.79 | 0.79 | 0.62 |
| 32 | narrow | 0.75 | 0.75 | 0.82 | 0.85 | 0.85 | 0.86 | 0.78 | 0.84 |
| 32 | medium | 0.85 | 0.85 | 0.83 | 0.85 | 0.85 | 0.85 | 0.90 | 0.69 |
| 32 | wide | 0.78 | 0.78 | 0.82 | 0.81 | 0.81 | 0.78 | 0.86 | 0.62 |
| peak | 8 | narrow | 0.76 | 0.82 | 0.76 | 0.83 | 0.83 | 0.78 | 0.78 | 0.90 |
| 8 | medium | 0.74 | 0.75 | 0.73 | 0.85 | 0.85 | 0.85 | 0.83 | 0.80 |
| 8 | wide | 0.64 | 0.64 | 0.69 | 0.75 | 0.75 | 0.79 | 0.74 | 0.61 |
| 16 | narrow | 0.71 | 0.76 | 0.77 | 0.90 | 0.90 | 0.89 | 0.72 | 0.90 |
| 16 | medium | 0.84 | 0.84 | 0.82 | 0.86 | 0.86 | 0.86 | 0.72 | 0.68 |
| 16 | wide | 0.74 | 0.74 | 0.83 | 0.82 | 0.81 | 0.82 | 0.68 | 0.65 |
| 32 | narrow | 0.73 | 0.73 | 0.82 | 0.83 | 0.83 | 0.89 | 0.78 | 0.90 |
| 32 | medium | 0.85 | 0.86 | 0.85 | 0.88 | 0.88 | 0.88 | 0.78 | 0.68 |
| 32 | wide | 0.81 | 0.82 | 0.73 | 0.84 | 0.83 | 0.81 | 0.72 | 0.65 |
| *Note.* Values represent the intraclass-correlation of latency values extracted by a certain algorithm with latencies extracted by an expert ERP researcher; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); manual = expert researcher either used a peak latency or area latency approach; filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | | |

**Table 12**

Intraclass correlation with manually extracted latencies - Switching Task

| manual | filter | window | maxcor | | | minsq | | | autoarea | autopeak |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| none | auto | manual | none | auto | manual | none | none |
| area | 8 | narrow | 0.68 | 0.84 | 0.83 | 0.85 | 0.84 | 0.85 | 0.62 | 0.82 |
| 8 | medium | 0.90 | 0.92 | 0.92 | 0.96 | 0.96 | 0.94 | 0.84 | 0.74 |
| 8 | wide | 0.39 | 0.46 | 0.66 | 0.55 | 0.54 | 0.79 | 0.77 | 0.48 |
| 16 | narrow | 0.56 | 0.74 | 0.71 | 0.75 | 0.80 | 0.75 | 0.54 | 0.71 |
| 16 | medium | 0.80 | 0.81 | 0.80 | 0.84 | 0.92 | 0.84 | 0.66 | 0.64 |
| 16 | wide | 0.32 | 0.38 | 0.46 | 0.56 | 0.56 | 0.56 | 0.57 | 0.39 |
| 32 | narrow | 0.45 | 0.75 | 0.76 | 0.73 | 0.73 | 0.73 | 0.56 | 0.71 |
| 32 | medium | 0.87 | 0.89 | 0.88 | 0.88 | 0.88 | 0.89 | 0.72 | 0.66 |
| 32 | wide | 0.38 | 0.52 | 0.64 | 0.55 | 0.55 | 0.75 | 0.60 | 0.34 |
| peak | 8 | narrow | 0.81 | 0.82 | 0.81 | 0.82 | 0.81 | 0.82 | 0.58 | 0.87 |
| 8 | medium | 0.83 | 0.84 | 0.84 | 0.88 | 0.88 | 0.88 | 0.69 | 0.77 |
| 8 | wide | 0.47 | 0.54 | 0.69 | 0.55 | 0.55 | 0.76 | 0.60 | 0.50 |
| 16 | narrow | 0.62 | 0.78 | 0.76 | 0.78 | 0.77 | 0.77 | 0.55 | 0.86 |
| 16 | medium | 0.82 | 0.83 | 0.82 | 0.86 | 0.86 | 0.86 | 0.65 | 0.77 |
| 16 | wide | 0.38 | 0.44 | 0.52 | 0.58 | 0.59 | 0.58 | 0.56 | 0.43 |
| 32 | narrow | 0.52 | 0.73 | 0.73 | 0.70 | 0.69 | 0.71 | 0.53 | 0.81 |
| 32 | medium | 0.81 | 0.82 | 0.81 | 0.84 | 0.83 | 0.83 | 0.65 | 0.75 |
| 32 | wide | 0.41 | 0.52 | 0.66 | 0.54 | 0.54 | 0.67 | 0.57 | 0.38 |
| *Note.* Values represent the intraclass-correlation of latency values extracted by a certain algorithm with latencies extracted by an expert ERP researcher; values greater than 0.8 are colored in green, less than 0.8 in orange; maxcor = MAXCOR-based algorithm; minsq = MINSQ-based algorithm; autoarea = Area latency algorithm; autopeak = Peak latency algorithm; results of the algorithms either not reviewed (none), automatically reviewed based on the fit statistic (auto), or reviewed manually (manual); manual = expert researcher either used a peak latency or area latency approach; filter = low-pass filter used in preprocessing (in Hz); window = measurement window used for latency extraction (narrow = 250 - 600 ms; medium = 200 - 700 ms; wide = 150 - 900ms) | | | | | | | | | | |

# 

# Discussion

The newly proposed template matching algorithm displayed consistently good psychometric properties and showed an improved ability to replicate human extraction behavior over previously established approaches like peak latency or area latency algorithms. Manual extraction has so far proven superior to algorithmic approaches ([Sadus et al., 2023](#ref-sadus2023multiverse)) but presents a time- and resource-intensive process. The template matching algorithm based on minimizing the weighted squared distance between transformed template and signal (MINSQ) showed a mean intraclass correlation with manually extracted P3 latencies of .85 across tasks and preprocessing steps. This indicates that the new algorithm was able to replicate manual extraction accurately while presenting a more objective and efficient approach to latency extraction. Across three different tasks, using the MINSQ algorithm in combination with a medium-sized (200ms - 700ms) measurement window generated the best results. Application of my algorithm would increase both replicability and scalability as well as significantly reduce the time and resources researchers need to spend on latency extraction.

Because previous algorithms have been proposed, I considered it prudent to compare the effectiveness of my new algorithm against already established algorithms.

## Reliability

Regarding the reliability of extracted latencies across tasks and preprocessing steps, my algorithm did not prove superior to the area latency approach. Both the MINSQ and MAXCOR approaches led to slightly lower average Spearman-Brown corrected split-half correlations than in area latency approaches. Following manual inspection, the MINSQ algorithm showed mean reliabilities of .88, only slightly lower than the mean reliability of the area latency approach ( .91). These differences in reliability are quite small and only carry low practical implications. If the researcher uses latent-variable approaches in their analysis of ERP latencies like structural equation modeling, the error variance may be controlled and resulting latent variables may be considered free of measurement error ([Bollen, 1989](#ref-bollen1989structural)). Especially in this case, slightly lower reliabilities carry negligible practical implications.

## Homogeneity

Latency values extracted by the MINSQ algorithm proved to have the highest average correlation with all other extraction methods across tasks and preprocessing steps ( .87). This indicates that this approach best reflects the total of all other measures. The MAXCOR algorithm also proved superior to previously established extraction methods. Across tasks and filter settings, the MINSQ algorithm combined with a medium measurement window showed the highest homogeneities compared to other extraction methods.

## Effect sizes

Across tasks, filter settings, and measurement windows, the average effect size of the age effect was highest for the MINSQ algorithm, average for the MAXCOR algorithm and lowest for the peak latency and area latency algorithms. The MINSQ algorithm also produced the highest effect size estimates in the Flanker and Switching tasks. This pattern of consistently higher effect sizes in the template matching approaches may indicate an overestimation of true effect sizes. Because the grand average is used extensively in the template matching procedure, the variance within groups may be artificially deflated as individual latency estimates are biased towards the group mean. This may inflate effect size estimates. I will investigate this potential issue in future work by using simulated data with a known true effect size.

Area latency and peak latency measures show consistently lower effect size estimates, sometimes yielding implausible estimates () of the age effect. This indicates that these approaches sometimes fail to detect the true age effect at all. Again, simulating data will help reveal the severity of this potential underestimation.

In the absence of a simulation study that can inform about the true effect size, I can only turn to the literature regarding the age effect. Effect sizes range from ([Wild-Wall et al., 2008](#ref-wild2008flanker)) to ([Kray et al., 2005](#ref-kray2005age)). In light of this information, I deem the failure to detect an age effect as more severe than estimating the age effect to be .

## Convergent Validity

Sadus et al. ([2023](#ref-sadus2023multiverse)) showed that manually extracting latency values is the best approach to ensure good psychometric properties and high power to detect experimental effects. The ability of an algorithm to extract latency values that correlate highly with those extracted by an expert ERP researcher was therefore of high importance to me. Again, the newly proposed algorithm proved to have a superior ability to replicate human behavior compared to previous approaches. The MINSQ algorithm, after manual inspection, had a mean intraclass correlation of .85 with manually extracted latencies across tasks and preprocessing steps.

The MAXCOR algorithm also outperformed previously established approaches in the ability to replicate human behavior, correlating very highly with manually extracted latencies ( .82). Area latency measures do correlate highly with manually extracted data ( .75) but failed to match the performance of my new algorithm.

Importantly, even the fully automated pipelines of my algorithm that rejected those matches with fits led to mean intraclass correlations of .83 for the MINSQ algorithm and .80 for the MAXCOR algorithm. This indicates that this new algorithm may be applied fully autonomously, eliminating the need for human intervention and increasing replicability, objectivity, and efficiency.

Sadus et al. ([2023](#ref-sadus2023multiverse)) investigated the interrater reliability using intraclass correlations for the Nback task. For the 3 filter settings further analyzed here (8 Hz, 16 Hz, 32 Hz), the mean intraclass correlation between two expert ERP researchers was .76. This indicates that the template matching algorithm correlates higher with an expert ERP researcher than two expert ERP researchers correlate with each other.

The superiority of the template matching algorithm further improved when I employed a medium measurement window. Both the average correlation and the minimum correlation with manually extracted latencies were highest across tasks and filter settings when using a MINSQ algorithm with a medium measurement window. Using a medium measurement window, the average intraclass correlation across tasks and filter settings between the fully autonomous MINSQ approach and an expert ERP researcher was 0.89. It was 0.87 between the fully autonomous MAXCOR approach and an expert ERP researcher.

This underscores a clear trend that can be observed in reliability, homogeneity, effect sizes, and validity of different extraction methods. Overall, the MINSQ algorithm shows the best average qualities, followed by the MAXCOR algorithm, and then followed by area latency approaches. More specifically, using a medium sized measurement window (200ms - 700ms) that aims to capture the onset and offset of the component as displayed in the grand average, leads to the best results, regardless of task and filter settings employed.

Inspecting the lowest reliabilities, homogeneities, and intraclass correlations illustrates the consistency of the MINSQ algorithm combined with a medium sized measurement window. It shows reliabilities of at least .77 ( .69 when applied fully automatically), homogeneities of at least .81 ( .82 when applied fully automatically), and intraclass correlations with latencies extracted by an expert ERP researcher of at least .83 ( .83 when applied fully automatically). Using the MINSQ approach and a measurement window that is constructed to include the onset and offset of the component visible in the grand average leads to consistently good qualities in P3 latency extraction.

## Objectivity

Any algorithmic approach to ERP latency extraction will be more objective than manually extracting ERP latencies. So I cannot crown any particular algorithm as more or less objective. The completely autonomous versions of peak latency, area latency or my algorithm with automatic rejection of bad fits are all equally objective. One strength of my approach is the ability for the researcher to inspect a subset of the ERPs based on the fit statistic of the matching procedure. This does introduce some subjectivity.

However, this ability of the algorithm to generate a fit statistic indicating the degree of certainty with which the match was made is a great strength of my new algorithm. Depending on the size of their data and the degree of certainty to which researchers want to manually inspect their data, one may choose any cut-off value for the fit statistic and inspect none, a subset, or all of the ERPs and the respective choices made by the algorithm by hand. This feature is not present in any of the previous algorithms.

## Comparing MINSQ and MAXCOR

I chose to implement two different approaches to quantifying the degree of similarity between template and signal. The first minimizes the weighted squared difference (MINSQ) and the second maximizes the correlation (MAXCOR). Both showed improvements over previous algorithms and I can recommend that both approaches be studied further. I did observe some differences between the two approaches in both procedural factors as well as outcome measures.

Procedurally, the largest difference between the two approaches is the optimization algorithm underlying them. Due to the invariance of the correlation of two vectors to scaling in amplitude of one vector, I can reduce the number of free parameters optimized during the MAXCOR approach to one. This allows me to use a more exhaustive optimization algorithm that will find the global optimum in some bounded parameter space without the possibility of converging on a local optimum. This is not the case for the multivariate optimization function needed for the MINSQ approach. Here, I initialize the optimization process at several different starting points and check for convergence on a common solution indicating that this solution represents the true global optimum. This is not ideal and could be improved in the future by implementing a more suitable optimization algorithm or improving on the one currently used.

The MINSQ algorithm may also converge on solutions where if the subject-level signal is largely of a polarity opposite to that of the component of interest (see Figure 3). Although I did extend the variability of the template by a parameter vertically shifting the template to account for these cases, sometimes even the extended version will converge on solutions with non-sensible parameter values. This leads to missing values and unreliable fit statistics in those cases. In my data, this happened for 7.13 % of all ERP signals analyzed by the algorithm. A large proportion of these cases may be considered unidentifiable even by an expert researcher due to particularly low signal-to-noise ratios. However, a subset of cases where the component can be identified by a human researcher or the MAXCOR algorithm may be classified as missing by the MINSQ algorithm. I will implement additional measures aiming to reduce the number of cases where the MINSQ algorithm fails to converge on a valid solution in future work.

**Figure 3**

*ERP with no positive-going signal*



This leads to the difference in the number of cases classified as missing by the MINSQ and MAXCOR approaches. While 2.52 % of all cases were set to NA after manual inspection of the MAXCOR algorithm (2.09 % after automatic inspection), 7.42 % of all cases were set to NA in the MINSQ algorithm following inspection (8.56 % after automatic inspection). This tradeoff between better properties of the MINSQ algorithm accompanied by more missing values must be taken into account when selecting which algorithm to use. Depending on the number of participants available and the means of analysis, missing values may be detrimental, leading to the MAXCOR algorithm being the preferable choice.

The weighting vector that is used in the MINSQ algorithm represents another difference between the two approaches. I used it to reflect the increased emphasis a human researcher places on those parts of the signal with the highest amplitude and signal appearing in the measurement window where the component of interest is expected to occur. The particular shape of the weighting function is somewhat arbitrary, but general aspects were chosen to reflect a few key considerations. For example, the maximum-normalization conducted before weights are calculated ensures that the weighting function is scale-invariant. Furthermore, I added larger weights to values inside the measurement window without completely discarding the impact of values outside the measurement window. I also chose to square the normalized amplitude in order to reflect a non-linear relationship between amplitude and importance. The exact shape of this weighting function may be argued and optimized further.

Regarding outcome measures, the MINSQ algorithm dominates the MAXCOR algorithm in almost all of the indices I inspected. It has better reliability, homogeneity, and validity. This provides evidence towards the argument that the MINSQ algorithm presents the better choice if one is limited to the application of just one algorithm.

## The impact of manual inspection

In order to choose a cut-off value for the fit statistic, I tested different cut-off values and checked whether a large enough proportion of them proved problematic enough to merit manual inspection. I set as my cut-off because more conservative cut-offs led to a situation where a large proportion of the matches I inspected had clearly correct results and manual inspection was not necessary. Considering the size of my data and the number of ERPs I applied the algorithm to, I wanted to test whether efficient latency extraction using the new algorithm was possible even in the face of a large dataset. I inspected around 10.75 % of ERPs of the MAXCOR algorithm and 10.94 % of ERPs in the MINSQ algorithm. Depending on how liberal or conservative the inspection is to be conducted, the cut-off value can be adjusted to increase or decrease the percentage of ERPs that have to be inspected manually.

This additional effort of manual inspection led to improved qualities of the extracted latencies over just automatically discarding fits with very bad fit statistics. Mean reliability and homogeneity improved and the values had slightly higher correlations with manually extracted latencies. However, using the template matching algorithm and forgoing manual inspection, but rather using the automatic rejection filter of still extracted latency values better than previously established algorithms. This fully automatic approach showed mean intraclass correlations across tasks, filter settings, and measurement windows with an expert researcher of .80 for the MAXCOR and .85 for the MINSQ algorithm. This improves even further when using a medium measurement window. The average intraclass correlation across tasks and filter settings between the fully autonomous MINSQ approach and an expert ERP researcher was 0.89. It was 0.87 between the fully autonomous MAXCOR approach and an expert ERP researcher.

Quantifying the certainty with which the template matching procedure chose a particular solution sets this algorithm apart from previous approaches. This enables the researcher to choose cut-off values for manual inspection and automatic rejection based on their particular needs in the current study. While more conservative inspection and rejection criteria will most likely improve the qualities of the extraction method, it also increases the time spent on inspection or the number of unidentifiable subject-level ERPs. This degree of control, especially using an objective criterion, is not available to researchers using other latency extraction algorithms.

Recently, Luck et al. ([2021](#ref-luck2021standardized)) introduced the Standardized Measurement Error (SME) quantifying the quality of an individual’s ERP data. This can be used to prune the dataset automatically, increasing the reliability of the latency values ([Wascher et al., 2022](#ref-wascher2022mental)). Future work will investigate the benefit of incorporating the SME into my review process.

## Limitations

This template matching algorithm is limited by the type of transformation I employ to introduce the variability in the template that allows quantifying individual differences. For example, I chose not to implement a parameter shifting the entire template along the x-axis. Thus, latency can only be shifted by scaling the entire component. Transformation of location and transformation shape of the component are confounded. Therefore, later peaks necessitate broader components. This is true in empirical ERPs, too, and thus not necessarily an issue. I could address this by introducing a parameter shifting the template without scaling it. However, this would also move the amplitude at 0 ms to some other time-point. As the origin is of special importance in ERP research, I decided against this shifting parameter. The origin is the only fix-point in a given ERP, resulting from the averaging and baselining procedures. Thus, I chose not to disturb this property. Future work may investigate the impact the introduction of this additional parameter in template transformations has on the template matching algorithm.

I also limited the algorithm to linear transformations of the template but could easily extend it to include non-linear transformations as well. Non-linear transformations, such as a transformation parameter , that depends on the time would enable the template transformations to capture the effect of some participants not displaying speed differences in early components (low scaling), but showing slow late components (higher scaling). This helps disentangle transformations of location and transformations of shape without disturbing the origin of the signal.

I only inspected one cut-off value for manual inspection and one for automatic rejection. These values were based on my experience in working with the algorithm, but this only provides limited insight into the impact of the cut-off value. Choosing a more conservative automatic rejection criterion may improve reliability and validity even further but come at a cost of a larger amount of missing values.

The generalizability of my findings is limited by the data I analyzed here. I inspected a limited sample of participants, narrow range of tasks and only one ERP component. Depending on the component of interest, the effectiveness of different algorithms can vary ([Kiesel et al., 2008](#ref-kiesel2008measurement); [Wascher et al., 2022](#ref-wascher2022mental)). I suspect that the effectiveness of all algorithms will decline when attempting to extract earlier components. The P3 is a broad, high-amplitude, and isolated component. This renders it ideal for algorithmic approaches, as the influence of surrounding components is comparatively low and the measurement window quite easily specified. Especially area latency approaches should diminish in quality due to the less isolated component structure of earlier components ([Luck, 2014](#ref-luck2014introduction)). I therefore expect that the benefits of my new algorithm relative to established algorithms will increase in earlier components.

## Future research

Future research should focus on applying template matching algorithms to earlier components. I also suggest simulating data, enabling future researchers to quantify the algorithm’s ability to recover true effect sizes or the true latency of a component. This master thesis serves largely as a proof-of-concept. The algorithm has yet to prove itself in a larger variety of tasks, samples, and different ERP components.

I further suggest improving the optimization processes that I used. The optimization function implemented for the MINSQ approach does not consistently converge on the global optimum. I compensated for this by initializing five different starting points and testing the solutions for convergence. However, this could be improved upon further. Finding an analytical solution would be ideal, but exceeds the scope of this master thesis.

Currently, the algorithm aims to identify the global optimum representing the absolute best similarity between transformed template and signal. It may be advantageous to use a linear combination of the best percentile of transformations as the solution of the optimization process ([Brunelli, 2009](#ref-brunelli2009template); [Brunelli & Poggiot, 1997](#ref-brunelli1997template)). Brunelli ([2009](#ref-brunelli2009template)) raised this issue in the context of multiclass pattern recognition. Correlation filters tend to result in broad peaks of optimality. I currently just choose the absolute peak and the algorithm returns the corresponding transformation parameters. Choosing the highest point in a broad peak is influenced by noise in the same manner as picking peaks of ERP components with peak latency algorithms. Future research should investigate a linear transformation like a weighted average when determining the optimal set of transformation parameters.

Aside from improvements in the implementation of the algorithm and extensions of the algorithm to earlier components, I will also improve the user interface that is used during manual inspection. Currently, the interface displays the matched template and informs the researcher about the latency and fit statistic this match would yield. I also display the choices a peak latency and an area latency algorithm would have made. The researcher can then either accept the result of the template matching algorithm, choose a result of the older algorithms, manually specify the component latency, or reject the ERP overall due to poor identifiability. I will aim to improve this by adding a slider controlling the transformation parameters, allowing the researcher to manually match the template to the subject-level ERP. The functionality of manual latency specification will also be improved by integrating already existing software like the Measurement Tool provided by ERPLAB ([Lopez-Calderon & Luck, 2014](#ref-lopez2014erplab)).

The particular cut-off values I chose for manual inspection or automatic rejection of the template matching solution allowed me to demonstrate both the algorithm’s ability to extract ERP latencies completely automatically and the improvements gained from manually inspecting a subset of the choices the algorithm made. However, I did not quantify how different cut-off values would impact the number of ERPs that would be inspected or rejected and the resulting quality of the extraction method. I will investigate this in further research, quantifying the impact of different cut-off values in order to gain insight into which cut-off values may be recommended depending on the context in which the algorithm is applied.

# Conclusion

This work provides proof-of-concept showing that a template matching algorithm using the grand average as a template can be feasibly used to extract P3 latencies. Latencies that were extracted by this algorithm correlate highly with latencies that were extracted by an expert human researcher in a variety of different tasks and preprocessing steps. The newly proposed algorithm is superior to previous algorithms like peak latency and area latency regarding the correlation with manually extracted latencies and homogeneity. Across three different tasks and three different low-pass filters, using the MINSQ algorithm and a medium measurement window (200ms - 700ms) led to the best outcomes regarding reliability, homogeneity, the detection of the age effect, and validity. A main benefit of my approach is the ability to quantify the algorithm’s confidence in a particular solution via a fit statistic. This allows researchers to inspect only the subset of ERPs with the worst fits and thus correct potential measurement error of the algorithm in a time-efficient manner. This fit statistic enables the researcher to specify a cut-off value for automatically rejecting template matches with bad fits, eliminating the need for human intervention. This fully automatic approach displays qualities superior to previous algorithms. When comparing the two similarity measures, the MINSQ algorithm displays better qualities than the MAXCOR algorithm. However, it results in a higher number of missing values. I will aim to improve the implementation of my algorithm and attempt to use it to extract earlier ERP components. Overall, the results obtained here leave me optimistic regarding the applicability of this template matching approach. It provides a more objective and efficient way to extract ERP latencies while maintaining consistently good psychometric quality and accurately replicating decisions made by an expert human researcher.

# References

Anderson, J. R., Zhang, Q., Borst, J. P., & Walsh, M. M. (2016). The discovery of processing stages: Extension of Sternberg’s method. *Psychological Review*, *123*(5), 481.

Aust, F., & Barth, M. (2022). *papaja: Prepare reproducible APA journal articles with R Markdown*. <https://github.com/crsh/papaja>

Bollen, K. A. (1989). *Structural equations with latent variables* (Vol. 210). John Wiley & Sons.

Borst, J. P., & Anderson, J. R. (2015). The discovery of processing stages: Analyzing EEG data with hidden semi-Markov models. *NeuroImage*, *108*, 60–73.

Brainard, D. H., & Vision, S. (1997). The psychophysics toolbox. *Spatial Vision*, *10*(4), 433–436.

Briechle, K., & Hanebeck, U. D. (2001). Template matching using fast normalized cross correlation. *Optical Pattern Recognition XII*, *4387*, 95–102.

Brunelli, R. (2009). *Template matching techniques in computer vision: Theory and practice*. John Wiley & Sons.

Brunelli, R., & Poggiot, T. (1997). Template matching: Matched spatial filters and beyond. *Pattern Recognition*, *30*(5), 751–768.

Clayson, P. E., Baldwin, S. A., & Larson, M. J. (2013). How does noise affect amplitude and latency measurement of event-related potentials (ERPs)? A methodological critique and simulation study. *Psychophysiology*, *50*(2), 174–186. <https://doi.org/10.1111/psyp.12001>

Cowley, B. U., Korpela, J., & Torniainen, J. (2017). Computational testing for automated preprocessing: A Matlab toolbox to enable large scale electroencephalography data processing. *PeerJ Computer Science*, *3*, e108. <https://doi.org/10.7717/peerj-cs.108>

Donchin, E. (1981). Surprise!… surprise? *Psychophysiology*, *18*(5), 493–513.

Donchin, E., & Heffley, E. F. (1978). *Multivariate analysis of event-related potential data: A tutorial review*.

Duncan-Johnson, C. C. (1981). Young psychophysiologist award address, 1980: P300 latency: A new metric of information processing. *Psychophysiology*, *18*(3), 207–215.

Eriksen, B. A., & Eriksen, C. W. (1974). Effects of noise letters upon the identification of a target letter in a nonsearch task. *Perception & Psychophysics*, *16*(1), 143–149. <https://doi.org/10.3758/BF03203267>

Friedman, D. (2011). The components of aging. In *The oxford handbook of event-related potential components*. Oxford University Press. <https://doi.org/10.1093/oxfordhb/9780195374148.013.0243>

Gohel, D., & Skintzos, P. (2023). *Flextable: Functions for tabular reporting*. <https://CRAN.R-project.org/package=flextable>

Goshtasby, A., Gage, S. H., & Bartholic, J. F. (1984). A two-stage cross correlation approach to template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *3*, 374–378.

Kiesel, A., Miller, J., Jolicœur, P., & Brisson, B. (2008). Measurement of ERP latency differences: A comparison of single-participant and jackknife-based scoring methods. *Psychophysiology*, *45*(2), 250–274.

Kleiner, M., Brainard, D., & Pelli, D. (2007). *What’s new in psychtoolbox-3?*

Kline, P. (1986). *A handbook of test construction: Introduction to psychometric design. New York: Methuen*. Inc.

Kray, J., Eppinger, B., & Mecklinger, A. (2005). Age differences in attentional control: An event-related potential approach. *Psychophysiology*, *42*(4), 407–416.

Lenth, R. V. (2023). *Emmeans: Estimated marginal means, aka least-squares means*. <https://CRAN.R-project.org/package=emmeans>

Lewis, J. P. (1995). Fast template matching. *Vision Interface*, *95*, 15–19.

Li, Y., Ma, Z., Lu, W., & Li, Y. (2006). Automatic removal of the eye blink artifact from EEG using an ICA-based template matching approach. *Physiological Measurement*, *27*(4), 425.

Liesefeld, H. R. (2018). Estimating the timing of cognitive operations with MEG/EEG latency measures: A primer, a brief tutorial, and an implementation of various methods. *Frontiers in Neuroscience*, *12*, 765.

Löffler, C., Frischkorn, G. T., Hagemann, D., Sadus, K., & Schubert, A.-L. (2022). *The common factor of executive functions measures nothing but speed of information uptake*.

Lopez-Calderon, J., & Luck, S. J. (2014). ERPLAB: An open-source toolbox for the analysis of event-related potentials. *Frontiers in Human Neuroscience*, *8*, 213.

Luck, S. J. (2005). Ten simple rules for designing and interpreting ERP experiments. *Event-Related Potentials: A Methods Handbook*, *4*.

Luck, S. J. (2014). *An introduction to the event-related potential technique*. MIT press.

Luck, S. J., Stewart, A. X., Simmons, A. M., & Rhemtulla, M. (2021). Standardized measurement error: A universal metric of data quality for averaged event-related potentials. *Psychophysiology*, *58*(6), e13793. <https://doi.org/10.1111/psyp.13793>

Mahalakshmi, T., Muthaiah, R., & Swaminathan, P. (2012). Image processing. *Research Journal of Applied Sciences, Engineering and Technology*, *4*(24), 5469–5473.

McCarthy, G., & Donchin, E. (1981). A metric for thought: A comparison of P300 latency and reaction time. *Science (New York, N.Y.)*, *211*(4477), 77–80.

Meyer, D. E., Osman, A. M., Irwin, D. E., & Yantis, S. (1988). Modern mental chronometry. *Biological Psychology*, *26*(1), 3–67. <https://doi.org/10.1016/0301-0511(88)90013-0>

Miller, J., Patterson, T., & Ulrich, R. (1998). Jackknife-based method for measuring LRP onset latency differences. *Psychophysiology*, *35*(1), 99–115. <https://doi.org/10.1111/1469-8986.3510099>

Miyake, A., Friedman, N. P., Emerson, M. J., Witzki, A. H., Howerter, A., & Wager, T. D. (2000). The unity and diversity of executive functions and their contributions to complex “frontal lobe” tasks: A latent variable analysis. *Cognitive Psychology*, *41*(1), 49–100.

Mullen, T. (2012). CleanLine EEGLAB plugin. *San Diego, CA: Neuroimaging Informatics Toolsand Resources Clearinghouse (NITRC)*.

Pelli, D. G., & Vision, S. (1997). The VideoToolbox software for visual psychophysics: Transforming numbers into movies. *Spatial Vision*, *10*, 437–442.

Pion-Tonachini, L., Kreutz-Delgado, K., & Makeig, S. (2019). ICLabel: An automated electroencephalographic independent component classifier, dataset, and website. *NeuroImage*, *198*, 181–197.

Polich, J. (2007). Updating P300: An integrative theory of P3a and P3b. *Clinical Neurophysiology*, *118*(10), 2128–2148.

Polich, J. (2012). Neuropsychology of P300. *The Oxford Handbook of Event-Related Potential Components*, 159–188.

Posner, M. I. (2005). Timing the Brain: Mental Chronometry as a Tool in Neuroscience. *PLOS Biology*, *3*(2), e51. <https://doi.org/10.1371/journal.pbio.0030051>

R Core Team. (2022). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. <https://www.R-project.org/>

Rodrigues, J., Weiß, M., Hewig, J., & Allen, J. J. (2021). EPOS: EEG processing open-source scripts. *Frontiers in Neuroscience*, *15*, 660449.

Sadus, K., Schubert, A.-L., Löffler, C., & Hagemann, D. (2023). *A multiverse study for extracting differences in P3 latencies between young and old adults*.

Scharinger, C., Soutschek, A., Schubert, T., & Gerjets, P. (2015). When flanker meets the n-back: What EEG and pupil dilation data reveal about the interplay between the two central-executive working memory functions inhibition and updating. *Psychophysiology*, *52*(10), 1293–1304. <https://doi.org/10.1111/psyp.12500>

Schubert, A.-L., Löffler, C., Hagemann, D., & Sadus, K. (2023). How robust is the relationship between neural processing speed and cognitive abilities? *Psychophysiology*, *60*(2), e14165.

Scrivano, R. M., & Kieffaber, P. D. (2022). Behavioral and electrophysiological correlates of Simon and flanker conflict interference in younger and older adults. *Aging, Neuropsychology, and Cognition*, *29*(2), 318–348.

Singmann, H., Bolker, B., Westfall, J., Aust, F., & Ben-Shachar, M. S. (2023). *Afex: Analysis of factorial experiments*. <https://CRAN.R-project.org/package=afex>

Smulders, F. T. Y. (2010). Simplifying jackknifing of ERPs and getting more out of it: Retrieving estimates of participants’ latencies. *Psychophysiology*, *47*(2), 387–392. <https://doi.org/10.1111/j.1469-8986.2009.00934.x>

The Math Works, Inc. (2022). *MATLAB version: 9.13.0 (r2022b)*. The MathWorks Inc. <https://www.mathworks.com>

Ulrich, R., & Miller, J. (2001). Using the jackknife-based scoring method for measuring LRP onset effects in factorial designs. *Psychophysiology*, *38*(5), 816–827.

Verleger, R. (2020). Effects of relevance and response frequency on P3b amplitudes: Review of findings and comparison of hypotheses about the process reflected by P3b. *Psychophysiology*, *57*(7), e13542.

Wascher, E., Sharifian, F., Gutberlet, M., Schneider, D., Getzmann, S., & Arnau, S. (2022). Mental chronometry in big noisy data. *PLOS ONE*, *17*(6), e0268916. <https://doi.org/10.1371/journal.pone.0268916>

Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., … Yutani, H. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. <https://doi.org/10.21105/joss.01686>

Wild-Wall, N., Falkenstein, M., & Hohnsbein, J. (2008). Flanker interference in young and older participants as reflected in event-related potentials. *Brain Research*, *1211*, 72–84.

William, F., Aygun, R., & Zhu, F. (2020). ERP template matching for EEG single trial classification. *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2876–2883.

Xie, Y. (2015). *Dynamic documents with R and knitr* (2nd ed.). Chapman; Hall/CRC. <https://yihui.org/knitr/>

Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R markdown: The definitive guide*. Chapman; Hall/CRC. <https://bookdown.org/yihui/rmarkdown>

Xie, Y., Dervieux, C., & Riederer, E. (2020). *R markdown cookbook*. Chapman; Hall/CRC. <https://bookdown.org/yihui/rmarkdown-cookbook>

1. We, furthermore, used the R-packages *afex* (Version 1.3.0; [Singmann et al., 2023](#ref-R-afex)), *emmeans* (Version 1.8.9; [Lenth, 2023](#ref-R-emmeans)), *flextable* (Version 0.9.4; [Gohel & Skintzos, 2023](#ref-R-flextable)), *knitr* (Version 1.45; [Xie, 2015](#ref-R-knitr)), *papaja* (Version 0.1.1.9001; [Aust & Barth, 2022](#ref-R-papaja)), *rmarkdown* (Version 2.25; [Xie et al., 2018](#ref-R-rmarkdown_a), [2020](#ref-R-rmarkdown_b)), and *tidyverse* (Version 2.0.0; [Wickham et al., 2019](#ref-R-tidyverse)). [↑](#footnote-ref-1)