# Precise temporal localisation of M/EEG effects with Bayesian generalised additive multilevel models

Ladislas Nalborczyk[1] and Paul Bürkner[2]
[1]Aix Marseille Univ, CNRS, LPL
[2]TU Dortmund University, Department of Statistics

Abstract

Time-resolved electrophysiological measurements such as those obtained through magneto- or electro-encephalography (M/EEG) offer a unique window into the neural activity underlying cognitive processes. Researchers are often interested in determining whether and when these signals differ across experimental conditions or participant groups. The conventional approach involves mass-univariate statistical testing across time and space, followed by corrections for multiple comparisons such as cluster-based inference. While effective for controlling error rates at the cluster-level, cluster-based inference comes with a significant limitation: by shifting the focus of inference from individual time points to clusters, it makes difficult to draw precise conclusions about the onset or offset of observed effects. Here, we introduce a *model-based* approach for analysing M/EEG timeseries such as event-related potentials (ERPs) or decoding performance over time. Our method leverages Bayesian generalised additive multilevel models (GAMMs), providing posterior probabilities that an effect is above zero (or above chance) at each time point, while naturally accounting for temporal dependencies and between-subject variability. Using both simulated and actual M/EEG datasets, we demonstrate that this approach substantially outperforms conventional methods in estimating the onset and offset of neural effects, yielding more precise and reliable results. We provide an R package implementing the method and show how it can be seamlessly integrated into M/EEG analysis pipelines using MNE-Python.

*Keywords:* EEG, MEG, cluster-based inference, simulation, multiple comparisons, generalised additive models, mixed-effects models, multilevel models, Bayesian statistics, brms

**Table of contents**



Ladislas Nalborczyk https://orcid.org/0000-0002-7419-9855
Paul Bürkner https://orcid.org/0000-0001-5765-8995
The authors have no conflicts of interest to disclose.
Correspondence concerning this article should be addressed to Ladislas Nalborczyk, Aix Marseille Univ, CNRS, LPL, 5 avenue Pasteur, 13100 Aix-en-Provence, France, email: ladislas.nalborczyk@cnrs.fr

# Precise temporal localisation of M/EEG effects with Bayesian generalised additive multilevel models

## Introduction

### Problem statement

Understanding the temporal dynamics of cognitive processes requires methods that can capture fast-changing neural activity with high temporal resolution. Magnetoencephalography and electroencephalography (M/EEG) are two such methods, widely used in cognitive neuroscience for their ability to track brain activity at the millisecond scale. These techniques provide rich time series data that reflect how neural responses unfold in response to stimuli or tasks. A central goal in many M/EEG studies is to determine whether, when, and where neural responses differ across experimental conditions or participant groups.

The conventional approach involves mass-univariate statistical testing with a unique threshold, followed by corrections for multiple comparisons with the goal of maintaining the familywise error rate (FWER) or false discovery rate (FDR) at the nominal level. Cluster-based inference is the most common way of achieving this sort of error control in the M/EEG literature, being the default option in several software programs. While effective for controlling error rates, cluster-based inference comes with a significant limitation: they shift the focus of inference from individual time points to clusters, making it difficult to draw precise conclusions about the onset or offset of observed effects (Maris & Oostenveld, 2007; Sassenhagen & Draschkow, 2019). As pointed by Maris & Oostenveld (2007); "there is a conflict between this interest in localized effects and our choice for a global null hypothesis: by controlling the FA [false alarm] rate under this global null hypothesis, one cannot quantify the uncertainty in the spatiotemporal localization of the effect". Even worse, as noted by Rosenblatt et al. (2018), cluster-based inference suffers from low spatial resolution: "Since discovering a cluster means that 'there exists at least one voxel with an evoked response in the cluster', and not that 'all the voxels in the cluster have an evoked response', it follows that the larger the detected cluster, the less information we have on the location of the activation." As a consequence, cluster-based inferenc has been shown...

To overcome the limitations of cluster-based inference, we introduce a novel model-based approach for precisely localising M/EEG effects in time (space, and other dimensions). The proposed approach, based on Bayesian generalised additive multilevel models, allows quantifying the posterior probability of effects being above chance at the level of timesteps, sensors, voxels, etc., while naturally taking into account spatiotemporal dependencies present in M/EEG timeseries. We compare the performance of the proposed approach to well-established alternative methods using both simulated and actual M/EEG data and show that it significantly outperforms alternative methods in estimating the onset and offset of M/EEG effects.

### Cluster-based inference

Different methods exist to control the family-wiser error rate (FWER), defined as the type-1 error rate (false positive) over an ensemble (family) of tests... for instance the Bonferroni correction (Dunn, 1961), however this method is generally overconservative as it assumes statistical independence of tests, an assumptions that is clearly violated in the context of M/EEG timeseries with spatiotemporal dependencies... or the false discovery rate (FDR), defined as the proportion of false positive among positive tests (e.g., Benjamini & Hochberg, 1995; Benjamini & Yekutieli, 2001)...

On popular technique to account for spatiotemporal dependencies while controlling the FWER is cluster-based inference. Description of cluster-based approaches (see Sassenhagen & Draschkow, 2019)... A typical cluster-based inference consists of two successive steps. First, clusters are defined as sets of contiguous voxels, channels, and/or timesteps, whose intensity/activity exceeds some predefined threshold. Clusters are then characterised by their height (i.e.,

maximal value), extent (number of constituent elements), or some combination of both, such at its "mass", for instance by summing the statistics within a cluster, an approach refereed to as "cluster mass" (Maris & Oostenveld, 2007; Pernet et al., 2015). Then, the null hypothesis is tested by assessing the probability.. As different cluster-forming thresholds lead to clusters with different spatial or temporal extent, this initial threshold modulates the sensitivity of the subsequent permutation test. The threshold-free cluster enhancement method (TFCE) was introduced by S. Smith & Nichols (2009) to overcome this choice of an arbitrary threshold.

In brief, the TFCE method works as follows. Instead of picking an arbitrary cluster-forming threshold (e.g., $t = 2$), we try all (or many) possible thresholds in a given range and check whether a given timestep/voxel belongs to a significant cluster under any of the set of thresholds. Then, instead of using cluster mass (e.g., the sum of squared t-values within the cluster), we use a weighted average between the cluster extend ($e$, how broad is the cluster, that is, how many connected samples it contains) and the cluster height ($h$, how high is the cluster, that is, how large is the test statistic). The TFCE score at each timestep/voxel $t$ is given by:

$$\text{TFCE(t)} = \int_{h=h_0}^{h_t} e(h)^E h^H \mathrm{d}h$$

where $h_0$ is typically 0 and parameters $E$ and $H$ are set a priori (typically to 0.5 and 2, respectively) and control the influence of the extend and height on the TFCE. Then, p-value for timestep/voxel $t$ is computed by comparing it TFCE with the null distribution of TFCE values. For each permuted signal, we keep the maximal value over the whole signal for the null distribution of the TFCE. The TFCE combined with permutation (assuming a large enough number of permutations) has been shown to provide accurate type 1 FWER (e.g., Pernet et al., 2015). However, previous simulation work showed that cluster-based methods have indeed poor performance in localising the onset of M/EEG effects (e.g., Rousselet, 2025; Sassenhagen & Draschkow, 2019).

To sum up this section, cluster-based inference main limitations is that it provides inference at the cluster level only, not allowing inference at the level of timesteps, sensors, etc. Thus, it does not allow inferring the localisation of effects (Maris, 2011; for more details on cluster-based inference, see for instance Maris & Oostenveld, 2007; Sassenhagen & Draschkow, 2019). In the following, we briefly review previous modelling work. Then, we provide a brief introduction to Bayesian statistical modelling, before moving to generalised additive models (GAMs) and generalised additive multilevel models (GAMMs) to illustrate how they can be used to precisely localise the onset and offset of M/EEG effects.

## Previous work on modelling M/EEG data

Recent example of GLM for EEG (Fischer & Ullsperger, 2013; Wüllhorst et al., 2025)... See also (Hauk et al., 2006; Rousselet et al., 2008)... Example of two-stage regression analysis (i.e., individual-level then group-level, Dunagan et al., 2024)...

As put by Rousselet (2025), concluding on the onset of effect based on a series of univariate tests... commits to three fallacies... here, we want to avoid these by introducing a model-based approach, which naturally take into account the temporal dependencies in the data to output a series of posterior probabilities...

See also the rERP framework (N. J. Smith & Kutas, 2014a, 2014b) and Tremblay & Newman (2014)...

From Dimigen & Ehinger (2021): Recently, spline regression has been applied to ERPs (e.g., Hendrix et al., 2017; Kryuchkova et al., 2012)... GAMMs for EEG data (Abugaber et al., 2023; Meulman et al., 2015, 2023)...

Disentangling overlapping processes (Skukies et al., 2024; Skukies & Ehinger, 2021)... Weighting single trials (Pernet, 2022)... The LIMO toolbox (Pernet et al., 2011)...

Recently, Teichmann (2022) provided a detailed tutorial on using Bayes factors (BFs) to analyse the 1D or 2D output from MVPA, that is, for testing, at every timestep, whether decoding performance is above chance level. However, this approach provides timeseries of BFs that ignores temporal dependencies...

## Generalised additive models

See for instance these tutorials (Sóskuthy, 2017; Winter & Wieling, 2016) or application to phonetic data (Sóskuthy, 2021; Wieling, 2018) or this introduction (Baayen & Linke, 2020) or these reference books (Hastie & Tibshirani, 2017; Wood, 2017a)... application to pupillometry (Rij et al., 2019)... GAMLSS for neuroimaging data (Dinga et al., 2021)... Modelling autocorrelation in GAMMs + EEG example (Baayen et al., 2018)...

In generalised additive models (GAMs), the functional relationship between predictors and response variable is decomposed into a sum of low-dimensional non-parametric functions. A typical GAM has the following form:

$$y_i \sim \text{EF}\left(\mu_i, \phi\right)$$

$$g\left(\mu_i\right) = \underbrace{\mathbf{A}_i\gamma}_{\text{parametric part}} + \underbrace{\sum_{j=1}^{J} f_j\left(x_{ij}\right)}_{\text{non-parametric part}}$$

where $y_i \sim \text{EF}\left(\mu_i, \phi\right)$ denotes that the observations $y_i$ are distributed as some member of the exponential family of distributions (e.g., Gaussian, Gamma, Beta, Poisson) with mean $\mu_i$ and scale parameter $\phi$; $g(\cdot)$ is the link function, $\mathbf{A}_i$ is the $i$th row of a known parametric model matrix, $\gamma$ is a vector of parameters for the parametric terms (to be estimated), $f_j$ is a smooth function of covariate $x_j$ (to be estimated as well). The smooth functions $f_j$ are represented in the model via penalised splines basis expansions of the covariates, that are a weighted sum of $K$ simpler, basis functions:

$$f_j\left(x_{ij}\right) = \sum_{k=1}^{K} \beta_{jk} b_{jk}\left(x_{ij}\right)$$

where $\beta_{jk}$ is the weight (coefficient) associated with the $k$th basis function $b_{jk}()$ evaluated at the covariate value $x_{ij}$ for the $j$th smooth function $f_j$. To clarify the terminology at this stage: *splines* are functions composed of simpler functions. These simpler functions are basis functions (e.g., cubic polynomial, thin-plate) and the set of basis functions is a *basis*. Each basis function gets its coefficient and the resultant spline is the sum of these weighted basis functions (Figure 1). Splines coefficients are penalised (usually through the squared of the smooth functions' second derivative) in a way that can be interpreted, in Bayesian terms, as a prior on the "wiggliness" of the function. In other words, more complex (wiggly) basis function are penalised.

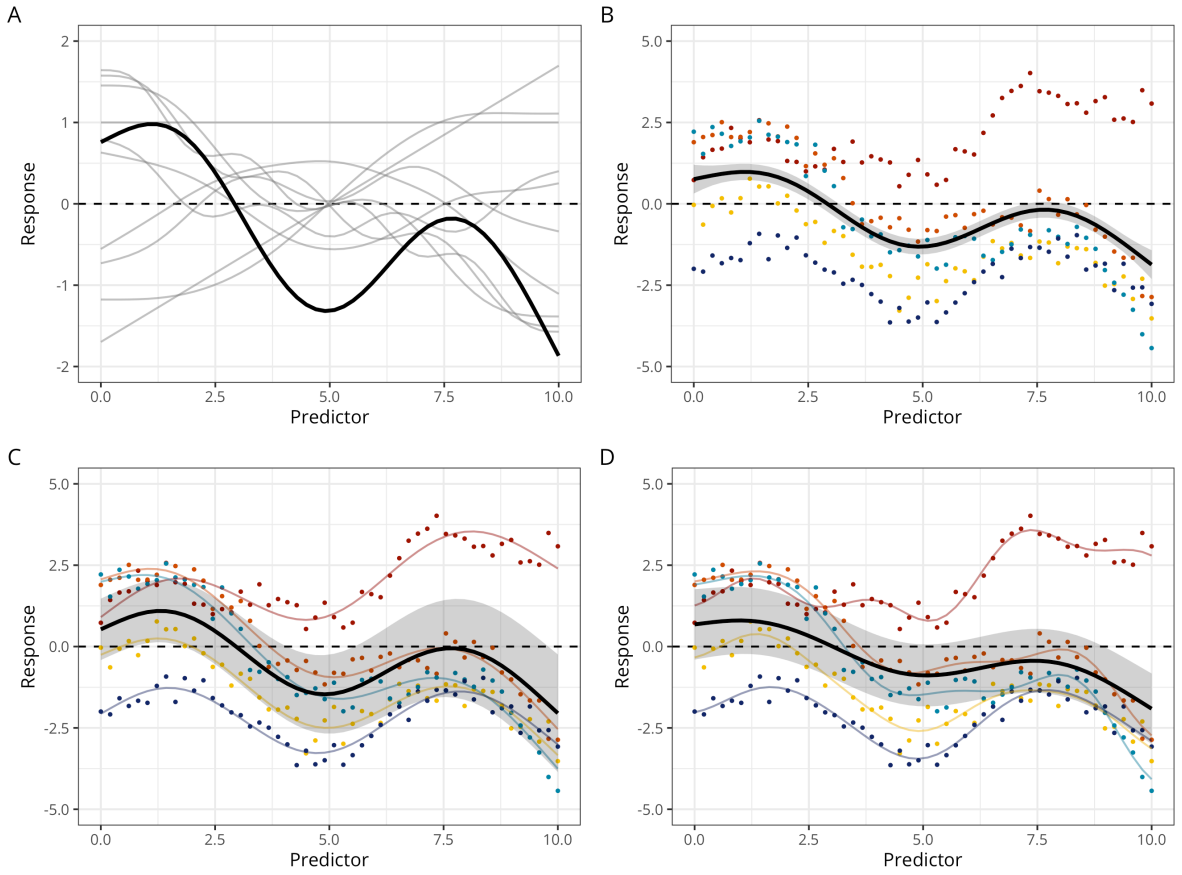
## Bayesian generalised additive multilevel models

The Bayesian approach to statistical modelling is characterised by its reliance on probability theory to (Gelman et al., 2020)... In this framework, all unknown entities are assigned probability distributions reflecting the uncertainty... These probability distributions are commonly referred to as "priors" and represent some state of knowledge about unknown quantities before seeing any data. There are debates among Bayesian practitioners as to whether prior distributions should encore subjective (personal) beliefs or... but these debates are outside the scope of the present paper and we therefore the interested reader to dedicated work (e.g., XX; YY)... In practice, weakly informative priors are often used as default priors in situations in

which subjective priors are difficult to define/elicit... Bayesian models are then fitted on empirical (actual or simulated) data to update prior states of knowledge to posterior states of knowledge using Bayes theorem, or in practise, sampling-based approximations of the posterior distribution...

**Figure 1**

*Different types of GAM(M)s.* ***A****: GAMs predictions are computed as the weigthed sum (in black) of basis functions (here thin-plate basis functions, in grey).* ***B****: Constant-effect GAM, with 5 participants in colours and the group-level prediction in black.* ***C****: Varying-intercept + varying-slope GAMM (with common smoother).* ***D****: Varying-intercept + varying-slope + varying-smoother GAMM. In this model, each participant gets its own intercept, slope, and degree of 'wiggliness' (smoother).*




See Figure 1... Introduction to multilevel GAMs (E. J. Pedersen et al., 2019)... Now describe the Bayesian GAMM (Miller, 2025)... Proper inclusion of varying/random effects in the model specification protects against overly wiggly curves (Baayen & Linke, 2020)... Generalising to scale and shape or "distributional GAMs" (Rigby & Stasinopoulos, 2005; Umlauf et al., 2018) and applied to neuroimaging data (Dinga et al., 2021)...

Instead of averaging, obtain the smooth ERP signal from multilevel GAM... less susceptible to outliers (Meulman et al., 2023)...

## Objectives

Given the previously reported limitations of conventional methods to precisely identify the onset and offset of M/EEG effects (e.g., ERPs, decoding performance), we developed a model-based approach for estimating the onset and offset of such effects. To achieve this, we

leveraged Bayesian generalised additive multilevel models (BGAMMs) fitted in `R` via the `brms` package and compared the performance of this approach to conventional methods on both simulated and actual M/EEG data.

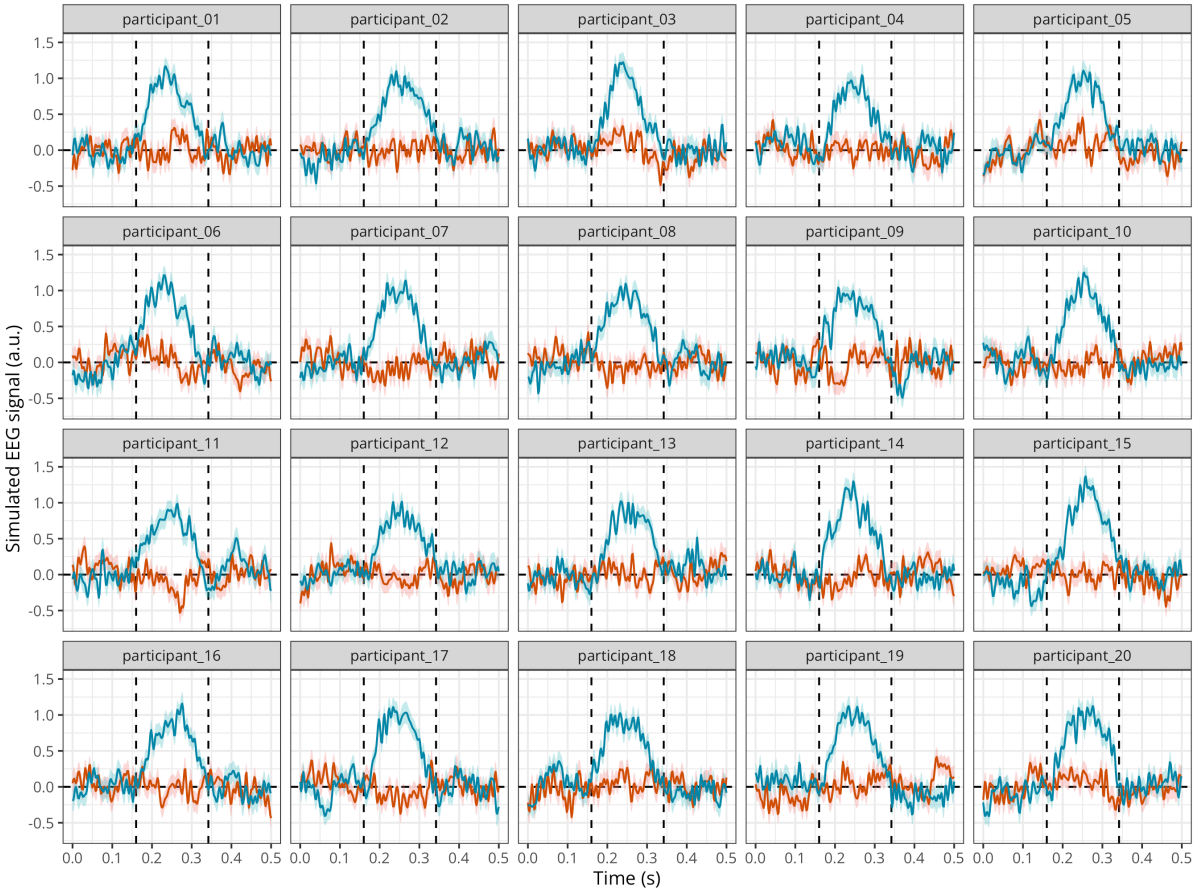
# Methods

## M/EEG data simulation

Following the approach of Sassenhagen & Draschkow (2019) and Rousselet (2025), we simulated EEG data stemming from two conditions, one with noise only, and the other with noise + signal. As in previous studies, the noise was generated by superimposing 50 sinusoids at different frequencies, following an EEG-like spectrum (see code in the online supplementary materials and details in Yeung et al., 2004). As in Rousselet (2025), the signal was generated from a truncated Gaussian distribution with an objective onset at 160 ms, a peak at 250 ms, and an offset at 342 ms. We simulated this signal for 250 timesteps between 0 and 0.5s, akin to a 500 Hz sampling rate. We simulated data for a group of 20 participants (with variable true onset) with 50 trials per participant and condition (Figure 2).

**Figure 2**

*Mean simulated EEG activity in two conditions with 50 trials each, for a group of 20 participants. The error band represents the mean +/- 1 standard error of the mean.*




## Model description and model fitting

We then fitted a Bayesian GAM (BGAM) using the `brms` package (Bürkner, 2017, 2018; Nalborczyk et al., 2019). We used the default priors in `brms` (i.e., weakly informative

priors). We ran eight Markov Chain Monte-Carlo (MCMC) to approximate the posterior distribution, including each 5000 iterations and a warmup of 2000 iterations, yielding a total of $8 \times (5000 - 2000) = 24000$ posterior samples to use for inference. Posterior convergence was assessed examining trace plots as well as the Gelman–Rubin statistic $\hat{R}$ (Gabry et al., 2019; Gelman et al., 2020). The `brms` package uses the same syntax as the `R` package `mgcv` v 1.9-1 (Wood, 2017b) for specifying smooth effects. Figure 3 shows the predictions of this model together with the raw data.

```
# averaging across participants
ppt_df <- raw_df %>%
    group_by(participant, condition, time) %>%
    summarise(eeg = mean(eeg) ) %>%
    ungroup()

# defining a contrast for condition
contrasts(ppt_df$condition) <- c(-0.5, 0.5)

# fitting the BGAM
gam <- brm(
    # cubic regression splines with k-1 basis functions
    # eeg ~ condition + s(time, bs = "cr", k = 20, by = condition),
    # thin-plate regression splines with k-1 basis functions
    eeg ~ condition + s(time, bs = "tp", k = 20, by = condition),
    data = ppt_df,
    family = gaussian(),
    warmup = 2000,
    iter = 5000,
    chains = 8,
    cores = 8,
    file = "models/gam.rds"
    )
```

However, the previous model only included constant (fixed) effects, thus not properly accounting for between-participant variability. We next fit a multilevel version of the BGAM (BGAMM). Although it is possible to fit a BGAMM at the single-trial level, we present a resource-efficient version of the model that is fitted directly on the by-participant summary statistics (mean and SD), similar to what is done in meta-analysis.

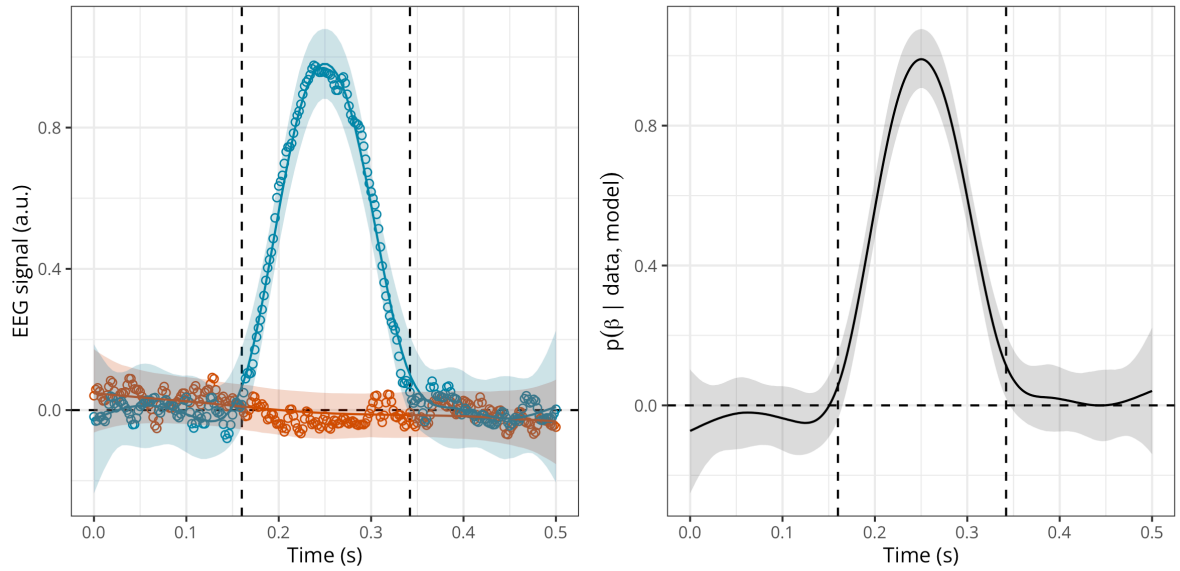
```
# averaging across participants
summary_df <- raw_df %>%
    summarise(
        eeg_mean = mean(eeg),
        eeg_sd = sd(eeg),
        .by = c(participant, condition, time)
        )

# defining a contrast for condition
contrasts(summary_df$condition) <- c(-0.5, 0.5)

# fitting the BGAMM
meta_gam <- brm(
    # using by-participant SD of ERPs across trials
```

```
    eeg_mean | se(eeg_sd) ~
        condition + s(time, bs = "cr", k = 20, by = condition) +
        (1 | participant),
    data = summary_df,
    family = gaussian(),
    warmup = 2000,
    iter = 5000,
    chains = 8,
    cores = 8,
    file = "models/meta_gam.rds"
    )
```

We depict the posterior predictions together with the posterior estimate of the slope for `condition` at each timestep (Figure 3). This figure suggests that the BGAMM provides an adequate description of the simulated data.

**Figure 3**

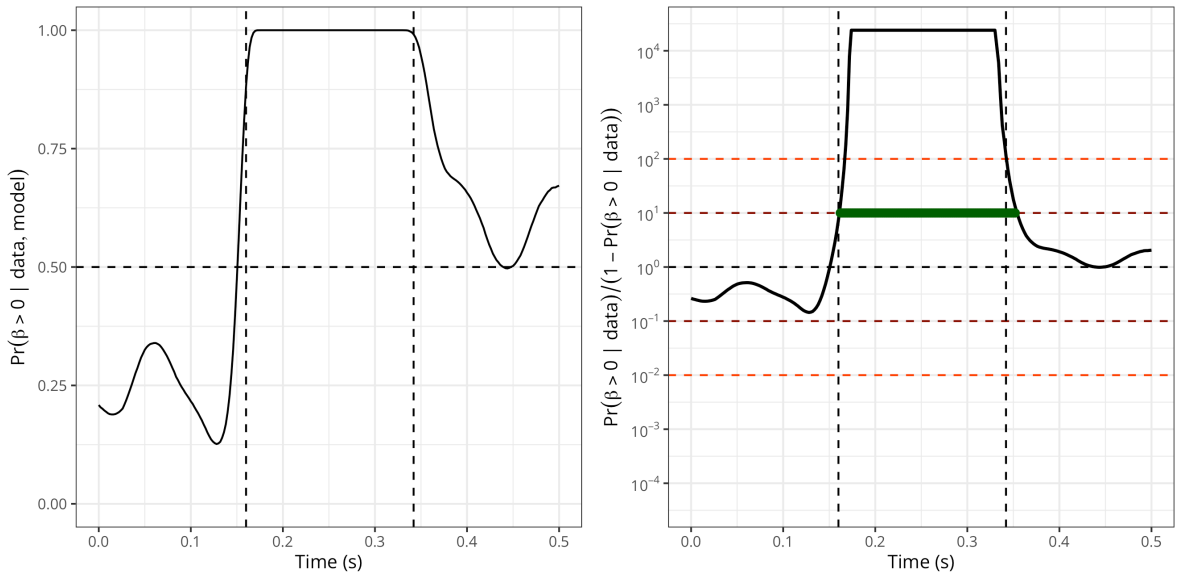
*Posterior estimate of the EEG activity in each condition (left) and posterior estimate of the difference in EEG activity (right) according to the BGAMM.*




We then compute the posterior probability of the slope for `condition` being above 0 (Figure 4, left). We can also express this as the ratio of posterior probabilities (i.e., $p/(1-p)$) and visualise the timecourse of this ratio superimposed with the conventional thresholds on evidence ratios (Figure 4, right). Note that a ratio of 10 means that the probability of the difference being above 0 is 10 times higher than the probability of the difference not being above 0, given the data, the priors, and other model's assumptions. Thresholding the posterior probability ratio thus provides a model-based approach for estimating the onset and offset of M/EEG effects.

**Figure 4**

*Left: Posterior probability of the EEG difference (slope) being above 0 according to the BGAMM. Right: Ratio of posterior probability according to the BGAMM (on a log10 scale). Timesteps above threshold (10) are highlighted in green.*

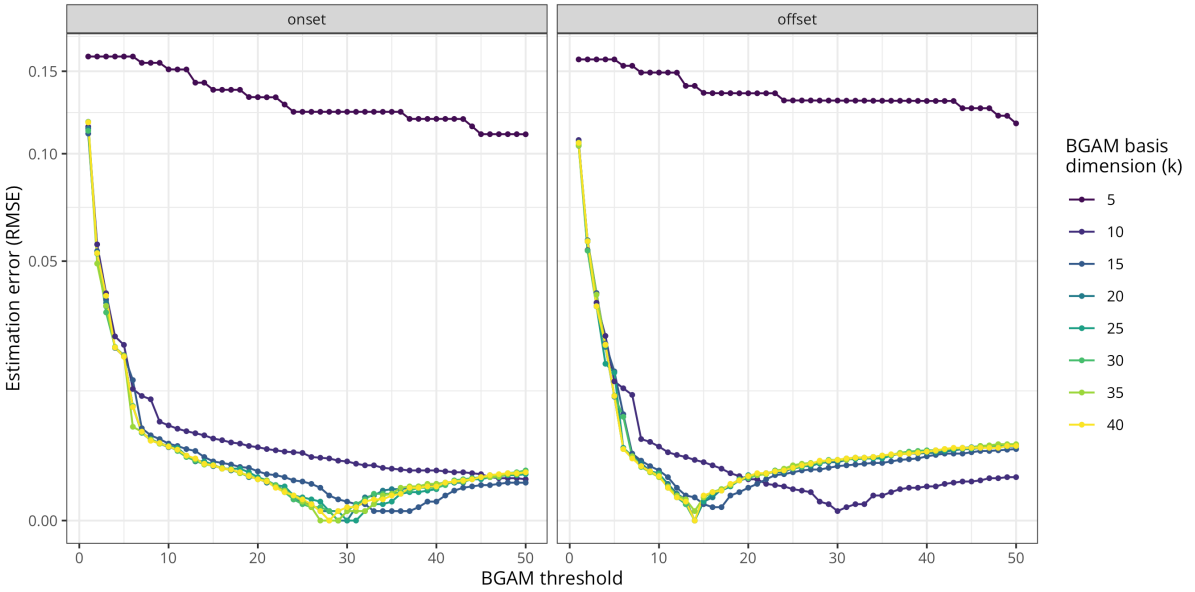



## Error properties of the proposed approach

We then assess the performance of the proposed approach by computing the difference between the true and estimated onset/offset of the EEG difference according to various `k` (BGAM basis dimension) and `threshold` values. Remember that the EEG signal was generated from a truncated Gaussian with an objective onset at 160 ms, a maximum at 250 ms, and an offset at 342 ms. Figure 5 shows that the multilevel GAM can almost exactly recover the true onset and offset values, given some reasonable choice of `k` and `threshold` values. We provide more detailed recommendations on how to set `k` in Section C. This figure further reveals that the optimal `k` and `threshold` values may differ for the onset and offset values, and there there seems to exist a trade-off between these two parameters: lower `k` values lead to poorer estimations, but these poor estimations can be compensated (only to some extent) by higher `threshold` values (and reciprocally).

**Figure 5**

*Average estimation error (RMSE) for the onset (left) and offset (right) according to various basis dimension and threshold values for the BGAM (computed from 100 simulated datasets).*

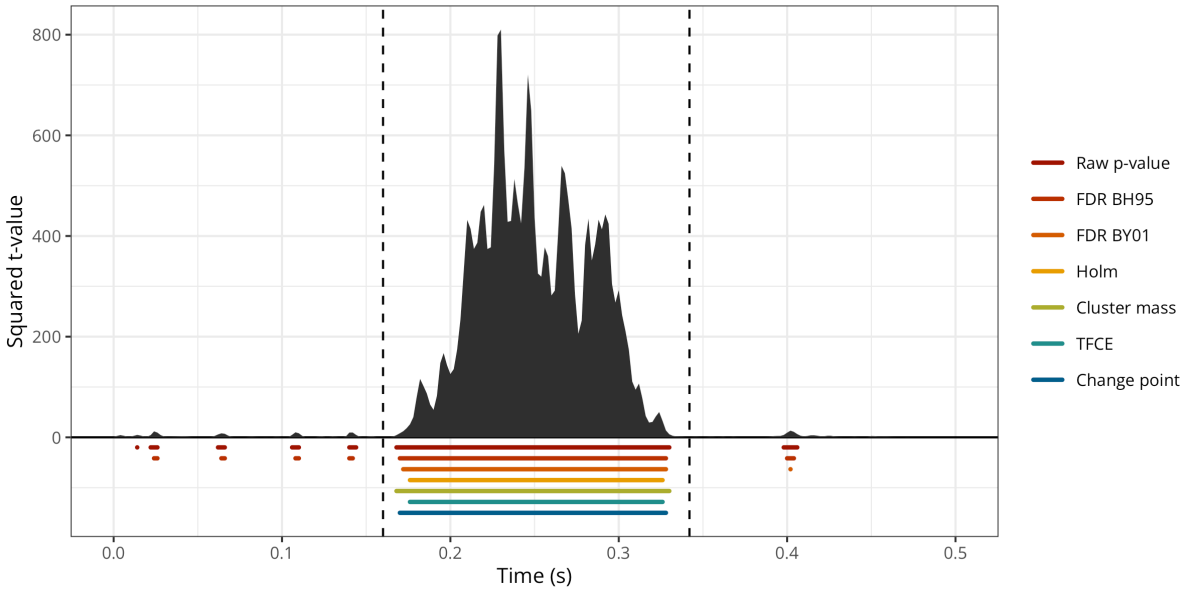



## Comparing the identified onsets/offsets to other approaches

We then compared the ability of the BGAMM to correctly estimate the onset and offset of the ERP difference to other widely-used methods. First, we conducted mass-univariate t-tests (thus treating each timestep independently) and identified the onset and offset of the ERP difference as the first and last values crossing an arbitrary significance threshold ($\alpha = 0.05$). We then followed the same approach but after applying different forms of multiplicity correction to the p-values. We compared two methods that control the false discovery rate (FDR) (i.e., BH95, Benjamini & Hochberg, 1995; and BY01, Benjamini & Yekutieli, 2001), one method that controls the family-wise error rate (FWER) (i.e., Holm–Bonferroni method, Holm, 1979), and two cluster-based permutation methods (permutation with a single cluster-forming threshold and threshold-free cluster enhancement, TFCE, S. Smith & Nichols, 2009). The BH95, BY01, and Holm corrections were applied to the p-values using the `p.adjust()` function in `R`. The cluster-based inference was implemented using a cluster-sum statistic of squared t-values, as implemented in `MNE-Python` (Gramfort, 2013), called via the `R` package `reticulate` v 1.42.0 (Ushey et al., 2024). We also compared these estimates to the onset and offset as estimated using the binary segmentation algorithm, as implemented in the `R` package `changepoint` v 2.3 (Killick et al., 2022), and applied directly to the squared t-values (as in Rousselet, 2025). Figure 6 illustrates the onsets and offsets estimated by each method on a single simulated dataset and shows that all methods systematically overestimate the true onset and underestimate the true offset.

**Figure 6**

*Exemplary timecourse of squared t-values with true onset and offset (vertical black dashed lines) and onsets/offsets identified using the raw p-values, the corrected p-values (BH95, BY01, Holm), the cluster-based methods (Cluster mass, TFCE), or using the binary segmentation method (Change point).*




## Simulation study

To assess the accuracy of group-level onset estimation, the various methods were compared using the bias (i.e., median(estimated-true)), median absolute error (MAE), root mean square error (RMSE), variance, and median absolute deviation (MAD) of onset/offset estimates computed on 1000 simulated datasets. As in Rousselet (2025), each participant was assigned a random onset betwee, 150 and 170ms.

## Application to actual MEG data

To complement the simulation study, we evaluated the performance of the various methods on actual MEG data (decoding results from Nalborczyk et al., in preparation). In this study, we conducted time-resolved multivariate pattern analysis (MVPA, also known as decoding) of MEG data during reading tasks. As a result, we obtain a timecourse of decoding performance (ROC AUC), bounded between 0 and 1, for each participant (for a total of 32 participants). Next, we wanted to *test* whether the group-level average decoding accuracy is above chance (i.e., 0.5) at each timestep (Figure 7). To achieve this, we fitted a BGAM as introduced previously, but we replaced the Normal likelihood function by a Beta one to account for the bounded nature of AUC values (between 0 and 1) (for a tutorial on Beta regression, see Coretta & Bürkner, 2025).

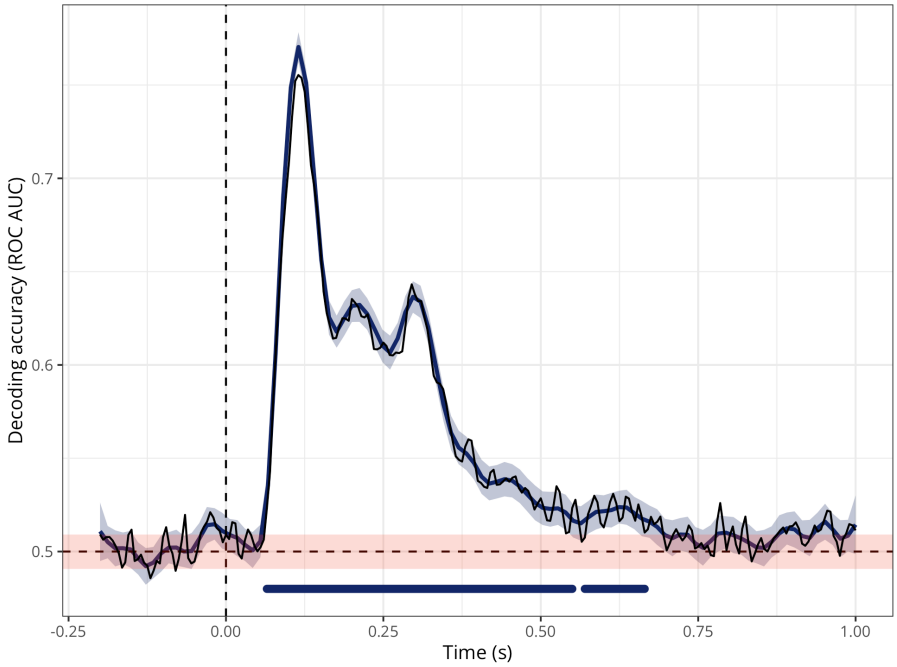
Note that although we chose a basis dimension of $k = 50$, which seems appropriate for the present data, this choice should be adapted according to the properties of the modelled data (e.g., signal-to-noise ratio, prior low-pass filtering, sampling rate, etc) and should be assessed by the usual model checking tools (e.g., posterior predictive checks, see also Section C). To better distinguish signal from noise, we also defined a region of practical equivalence (ROPE, Kruschke & Liddell, 2017), defined as the chance level plus the standard deviation of the (group-level average) decoding performance during the baseline period.

```
# fitting the Beta GAM
meg_decoding_gam <- brm(
    auc ~ s(time, bs = "cr", k = 50),
    data = decoding_df,
    family = Beta(),
    warmup = 2000,
    iter = 5000,
    chains = 4,
    cores = 4
    )
```

We assessed the reliability of the proposed approach using a form of permutation-based split-half reliability (as for instance in Rosenblatt et al., 2018), which consisted of the following steps. First, we created 1000 split halves of the data (i.e., with half the participants in the original data, that is, 16 participants). For each split, we estimated the onset/offset using all methods described previously. Third, we summarised the distribution of onset/offset estimates using the median "error" (i.e., difference between the split estimate and the estimate obtained using the full dataset) and the variance across splits. This approach allows assessing how similar the estimate of each half split is to the full dataset (thus acting as a proxy for the population) and how variable the estimates are (across split halves).

**Figure 7**

*Group-level average decoding performance (N=32) superimposed with the GAM predictions (in blue) and the region of practical equivalence (ROPE, in orange) computed from the baseline period (data from Nalborczyk et al., in preparation). The blue horizontal markers indicate the timesteps at which the posterior probability ratio exceeds 20.*



## Results

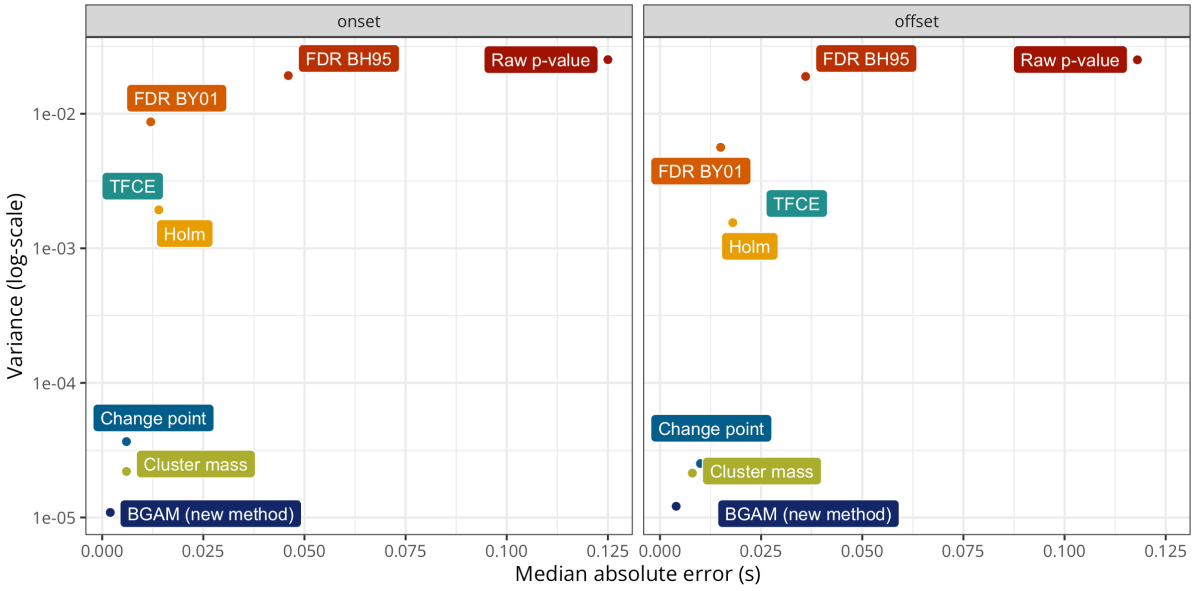
This section is divided in two parts. First, we present the results from the simulation study, assessing the bias and variance of each method when applied to simulated data in which the ground truth is known. Second, we present the results obtained when applying the different methods to actual MEG data (decoding performance through time), assessing the reliability of the estimates provided by each method.

### Simulation study (bias and variance)

Figure 8 shows a summary of the simulation results, revealing that the proposed approach (`BGAM`) has the lowest median absolute error (MAE) and variance for both the onset and offset estimates. The `Cluster mass` and `Change point` also have good performance, but surprisingly, the `TFCE` method has relatively bad performance for estimating the effect offset (similar performance to the `Holm` and `FDR BY01` methods). Unsurprisingly, the `Raw p-value` and `FDR BH95` methods show the worst performance.

**Figure 8**

*Median absolute error and variance of onset and offset estimates for each method. Variance is plotted on a log10 scale for visual purposes.*




Results are further summarised in Table 1, which shows that the `BGAM` is perfectly unbiased (i.e., it has a bias of 0s) for the onset and almost exactly unbiased for the onset (with a bias of approximately -2ms). The `Bias` column shows that all other methods tend to estimate the onset later than the true onset and to estimate the offet earlier than the true offset. As can be seen from this table, the `BGAM` has the best performance on all included metrics.

**Table 1**

*Summary statistics of the onset and offset estimates for each method (ordered by MAE).*

| | Bias | MAE | RMSE | Variance | MAD |
|---|---|---|---|---|---|
| onset | | | | | |
| BGAM (new method) | 0.0000 | 0.0020 | 0.0001 | 0.0000 | 0.0030 |
| Cluster mass | 0.0060 | 0.0060 | 0.0057 | 0.0000 | 0.0044 |
| Change point | 0.0060 | 0.0060 | 0.0051 | 0.0000 | 0.0059 |
| FDR BY01 | 0.0120 | 0.0120 | 0.0443 | 0.0087 | 0.0059 |
| TFCE | 0.0140 | 0.0140 | 0.0270 | 0.0019 | 0.0059 |
| Holm | 0.0140 | 0.0140 | 0.0270 | 0.0019 | 0.0059 |
| FDR BH95 | 0.0080 | 0.0460 | 0.0599 | 0.0192 | 0.0801 |
| Raw p-value | 0.0060 | 0.1250 | 0.0711 | 0.0252 | 0.1942 |
| offset | | | | | |
| BGAM (new method) | -0.0020 | 0.0040 | 0.0021 | 0.0000 | 0.0030 |
| Cluster mass | -0.0080 | 0.0080 | 0.0084 | 0.0000 | 0.0059 |
| Change point | -0.0100 | 0.0100 | 0.0096 | 0.0000 | 0.0059 |
| FDR BY01 | -0.0140 | 0.0150 | 0.0204 | 0.0056 | 0.0059 |
| TFCE | -0.0180 | 0.0180 | 0.0261 | 0.0016 | 0.0030 |
| Holm | -0.0180 | 0.0180 | 0.0262 | 0.0016 | 0.0030 |
| FDR BH95 | -0.0100 | 0.0360 | 0.0578 | 0.0189 | 0.0682 |
| Raw p-value | -0.0080 | 0.1180 | 0.0725 | 0.0252 | 0.1868 |

## Application to actual MEG data (reliability)

Figure 9 shows the group-level average decoding performance through time with onset and offset estimates from each method. Overall, this figure shows that both the `Raw p-value` and `FDR BH95` methods are extremely lenient, considering that the decoding performance is above chance before the onset of the stimulus (false positive) and until the end of the trial. The `Change point` and `Cluster mass` methods seem the most conservative methods, identifying a time window from approximately +60ms to +500ms. The `Holm`, `TFCE`, and `BGAM` methods produce similar estimates of onset and offset, ranging from approximately +60ms to +650ms, although the `BGAM` method seems to result in fewer clusters.[1]

Figure 10 shows the median difference between the onset and offset estimates from each data split and the onset and offset estimates from the full dataset (x-axis) along with the variance of its onset and offset estimates across data splits (error bar). This figure reveals that the `BGAM` *onset and offset* estimates on each split are the closest to the estimates from the full dataset on average (0ms difference for the onset estimate and 5ms difference for the offset estimate). The `Raw p-value` method has similar performance, but given the aberrant estimates it produces (cf. Figure 9), the fact that it is consistent between data splits and the full dataset is not convincing on its own. The `Change point` method also has a very good performance (i.e., very low difference between split estimates and full estimates), but produces too short cluster of significant decoding performance (cf. Figure 9).[2] Overall, the figure reveals that for all other methods, split datasets produce later onset estimates and earlier offset estimates (as compared to the estimates from the model fitted on the full dataset). These results highlight that the results of any method should be assessed using both i) good asymptotic properties on simulated data, ii) sensible identified clusters in actual data, and iii) reliable/stable estimates on actual data.

[1] It should be noted that although each method can produce several "clusters" of timesteps, we only considered the first (onset) and last (offset) timesteps identified by each method to compute the estimation error.

[2] As in Rousselet (2025), we fixed the number of expected change points to two in the binary segmentation algorithm, thus producing always one cluster.

**Figure 9**

*Group-level average decoding performance through time with onset and offset estimates for each method (data from Nalborczyk et al., in preparation).*

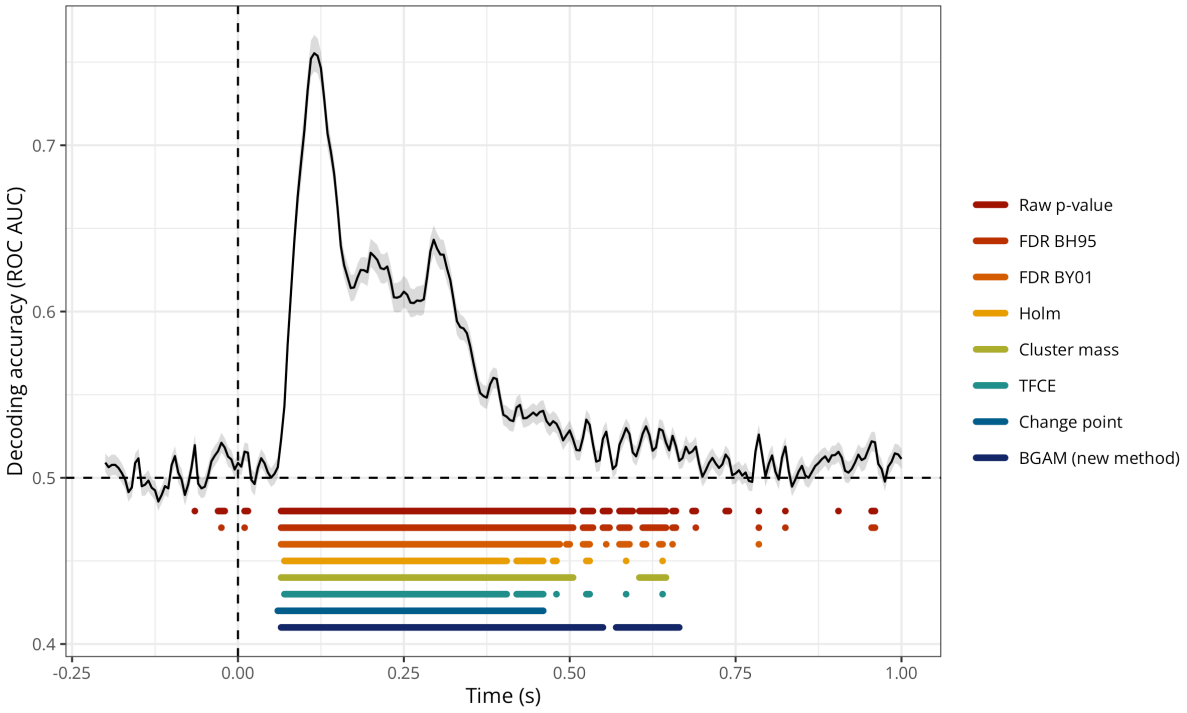
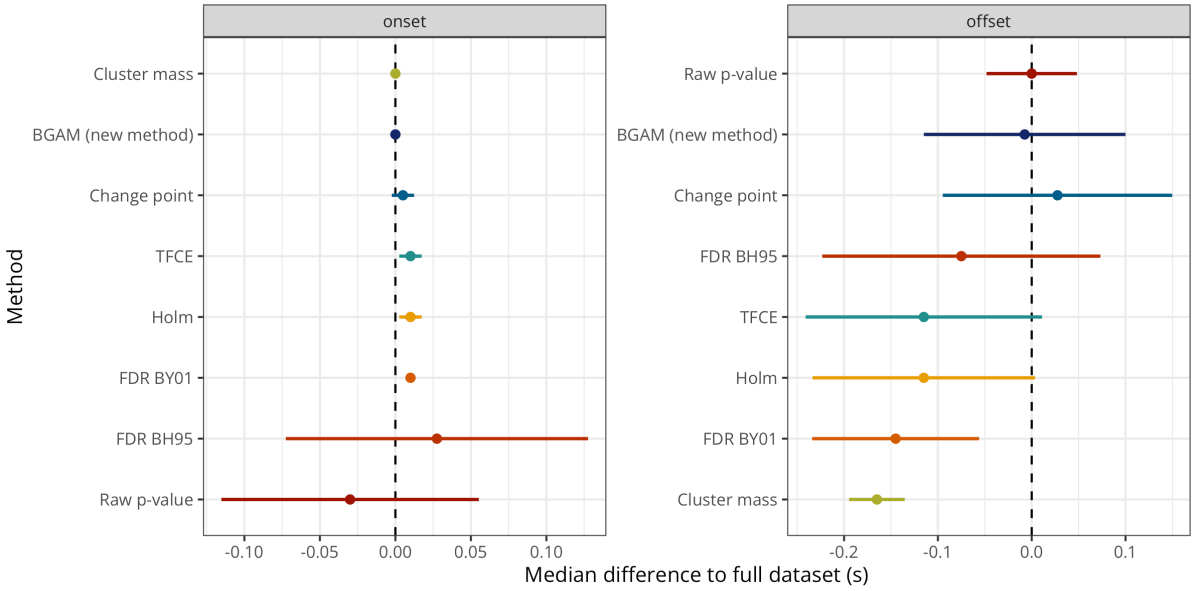



**Figure 10**

*Median error and median absolute deviation of the error for the onset (left) and offset (right) estimates according to each method. Methods are ordered from lowest (top) to highest (bottom) median absolute error (separately for the onset and offset estimates).*




## Discussion

### Summary of the proposed approach

Overall, before concluding on the onset/offset of effect based on the model, we need to ensure that the model provides a faithful description of the data-generating process (e.g., via

posterior predictive checks etc)...

The TFCE performs worse than the cluster-sum approach, which was anticipated by Rousselet (2025) based on the initial results of S. Smith & Nichols (2009)...

## Limitations and future directions

As in previous simulation work (e.g., Rousselet et al., 2008; Sassenhagen & Draschkow, 2019), the present simulation results depend on various choices such as the specific cluster-forming algorithm and threshold, signal-to-noise ratio, negative impact of preprocessing steps (e.g., low-pass filter) on temporal resolution... note however, that the same caveats apply to all methods...

The error properties depend on the threshold parameter, a value of 10 or 20 seems to be a reasonable default, but the optimal threshold parameter can be adjusted using split-half reliability assessment... also depends on `k`...

Can be applied to any 1D timeseries (e.g., pupillometry, electromyography)... Extending the approach to spatiotemporal data (i.e., time + sensors) or spatiotemporal time-frequency 4D data...

We kept the exemplary models simple, but can be extended by adding varying/random effects (intercept and slope) for item (e.g., word)... but also continuous predictors at the trial level?

## Data and code availability

The simulation results as well as the `R` code to reproduce the simulations are available on GitHub: https://github.com/lnalborczyk/brms_meeg. The `neurogam R` package is available at https://github.com/lnalborczyk/neurogam.

## Packages

We used R version 4.4.3 (R Core Team, 2025) and the following R packages: assertthat v. 0.2.1 (Wickham, 2019), brms v. 2.22.0 (Bürkner, 2017, 2018, 2021), changepoint v. 2.3 (Killick et al., 2024; Killick & Eckley, 2014), doParallel v. 1.0.17 (Corporation & Weston, 2022), easystats v. 0.7.4 (Lüdecke et al., 2022), foreach v. 1.5.2 (Microsoft & Weston, 2022), furrr v. 0.3.1 (Vaughan & Dancho, 2022), future v. 1.34.0 (Bengtsson, 2021), ggrepel v. 0.9.6 (Slowikowski, 2024), glue v. 1.8.0 (Hester & Bryan, 2024), grateful v. 0.2.11 (Rodriguez-Sanchez & Jackson, 2024), gt v. 1.0.0 (Iannone et al., 2025), knitr v. 1.50 (Xie, 2014, 2015, 2025), MetBrewer v. 0.2.0 (Mills, 2022), neurogam v. 0.0.1 (Nalborczyk, 2025), pakret v. 0.2.2 (Gallou, 2024), patchwork v. 1.3.0 (T. L. Pedersen, 2024), rmarkdown v. 2.29 (Allaire et al., 2024; Xie et al., 2018, 2020), scales v. 1.3.0 (Wickham et al., 2023), scico v. 1.5.0 (T. L. Pedersen & Crameri, 2023), tictoc v. 1.2.1 (Izrailev, 2024), tidybayes v. 3.0.7 (Kay, 2024), tidytext v. 0.4.2 (Silge & Robinson, 2016), tidyverse v. 2.0.0 (Wickham et al., 2019).

## Acknolwedgements

Centre de Calcul Intensif d'Aix-Marseille is acknowledged for granting access to its high performance computing resources.

## References


Abugaber, D., Finestrat, I., Luque, A., & Morgan-Short, K. (2023). Generalized additive mixed modeling of EEG supports dual-route accounts of morphosyntax in suggesting no word frequency effects on processing of regular grammatical forms. *Journal of Neurolinguistics*, *67*, 101137. https://doi.org/10.1016/j.jneuroling.2023.101137

Allaire, J., Xie, Y., Dervieux, C., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., & Iannone, R. (2024). *rmarkdown: Dynamic documents for r*. https://github.com/rstudio/rmarkdown

Baayen, R. H., & Linke, M. (2020). *Generalized Additive Mixed Models* (pp. 563–591). Springer International Publishing. https://doi.org/10.1007/978-3-030-46216-1_23

Baayen, R. H., Rij, J. van, Cat, C. de, & Wood, S. (2018). *Autocorrelated errors in experimental data in the language sciences: Some solutions offered by generalized additive mixed models* (pp. 49–69). Springer International Publishing. https://doi.org/10.1007/978-3-319-69830-4_4

Bengtsson, H. (2021). A unifying framework for parallel and distributed processing in r using futures. *The R Journal*, *13*(2), 208–227. https://doi.org/10.32614/RJ-2021-048

Benjamini, Y., & Hochberg, Y. (1995). Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *57*(1), 289–300. https://doi.org/10.1111/j.2517-6161.1995.tb02031.x

Benjamini, Y., & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics*, *29*(4). https://doi.org/10.1214/aos/1013699998

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

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

Bürkner, P.-C. (2021). Bayesian item response modeling in R with brms and Stan. *Journal of Statistical Software*, *100*(5), 1–54. https://doi.org/10.18637/jss.v100.i05

Coretta, S., & Bürkner, P.-. C. (2025). *Bayesian beta regressions with brms in r: A tutorial for phoneticians*. http://dx.doi.org/10.31219/osf.io/f9rqg_v1

Corporation, M., & Weston, S. (2022). *doParallel: Foreach parallel adaptor for the "parallel" package*. https://CRAN.R-project.org/package=doParallel

Dimigen, O., & Ehinger, B. V. (2021). Regression-based analysis of combined EEG and eye-tracking data: Theory and applications. *Journal of Vision*, *21*(1), 3. https://doi.org/10.1167/jov.21.1.3

Dinga, R., Fraza, C. J., Bayer, J. M. M., Kia, S. M., Beckmann, C. F., & Marquand, A. F. (2021). *Normative modeling of neuroimaging data using generalized additive models of location scale and shape*. http://dx.doi.org/10.1101/2021.06.14.448106

Dunagan, D., Jordan, T., Hale, J. T., Pylkkänen, L., & Chacón, D. A. (2024). *Evaluating the timecourses of morpho-orthographic, lexical, and grammatical processing following rapid parallel visual presentation: An EEG investigation in english*. http://dx.doi.org/10.1101/2024.04.10.588861

Dunn, O. J. (1961). Multiple Comparisons among Means. *Journal of the American Statistical Association*, *56*(293), 52–64. https://doi.org/10.1080/01621459.1961.10482090

Fischer, Adrian G., & Ullsperger, M. (2013). Real and Fictive Outcomes Are Processed Differently but Converge on a Common Adaptive Mechanism. *Neuron*, *79*(6), 1243–1255. https://doi.org/10.1016/j.neuron.2013.07.006

Gabry, J., Simpson, D., Vehtari, A., Betancourt, M., & Gelman, A. (2019). Visualization in Bayesian workflow. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *182*(2), 389–402. https://doi.org/10.1111/rssa.12378

Gallou, A. (2024). *pakret: Cite "R" packages on the fly in "R Markdown" and "Quarto"*. https://CRAN.R-project.org/package=pakret

Gelman, A., Vehtari, A., Simpson, D., Margossian, C. C., Carpenter, B., Yao, Y., Kennedy, L., Gabry, J., Bürkner, P.-C., & Modrák, M. (2020). Bayesian workflow. *arXiv:2011.01808 [Stat]*. http://arxiv.org/abs/2011.01808

Gramfort, A. (2013). MEG and EEG data analysis with MNE-python. *Frontiers in Neuroscience*, *7*. https://doi.org/10.3389/fnins.2013.00267

Hastie, T. J., & Tibshirani, R. J. (2017). *Generalized Additive Models*. Routledge. https://doi.org/10.1201/9780203753781

Hauk, O., Davis, M. H., Ford, M., Pulvermüller, F., & Marslen-Wilson, W. D. (2006). The time course of visual word recognition as revealed by linear regression analysis of ERP data. *NeuroImage*, *30*(4), 1383–1400. https://doi.org/10.1016/j.neuroimage.2005.11.048

Hendrix, P., Bolger, P., & Baayen, H. (2017). Distinct ERP signatures of word frequency, phrase frequency, and prototypicality in speech production. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *43*(1), 128–149. https://doi.org/10.1037/a0040332

Hester, J., & Bryan, J. (2024). *glue: Interpreted string literals*. https://CRAN.R-project.org/package=glue

Holm, S. (1979). A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, *6*(2), 65–70. http://www.jstor.org/stable/4615733

Iannone, R., Cheng, J., Schloerke, B., Hughes, E., Lauer, A., Seo, J., Brevoort, K., & Roy, O. (2025). *gt: Easily create presentation-ready display tables*. https://CRAN.R-project.org/package=gt

Izrailev, S. (2024). *tictoc: Functions for timing r scripts, as well as implementations of "Stack" and "StackList" structures*. https://CRAN.R-project.org/package=tictoc

Kay, M. (2024). *tidybayes: Tidy data and geoms for Bayesian models*. https://doi.org/10.5281/zenodo.1308151

Killick, R., & Eckley, I. A. (2014). changepoint: An R package for changepoint analysis. *Journal of Statistical Software*, *58*(3), 1–19. https://www.jstatsoft.org/article/view/v058i03

Killick, R., Haynes, K., & Eckley, I. A. (2022). *changepoint: An R package for changepoint analysis*. https://CRAN.R-project.org/package=changepoint

Killick, R., Haynes, K., & Eckley, I. A. (2024). *changepoint: An R package for changepoint analysis*. https://CRAN.R-project.org/package=changepoint

King, J.-R., & Dehaene, S. (2014). Characterizing the dynamics of mental representations: the temporal generalization method. *Trends in Cognitive Sciences*, *18*(4), 203–210. https://doi.org/10.1016/j.tics.2014.01.002

Kruschke, J. K., & Liddell, T. M. (2017). The Bayesian New Statistics: Hypothesis testing, estimation, meta-analysis, and power analysis from a Bayesian perspective. *Psychonomic Bulletin & Review*, *25*(1), 178–206. https://doi.org/10.3758/s13423-016-1221-4

Kryuchkova, T., Tucker, B. V., Wurm, L. H., & Baayen, R. H. (2012). Danger and usefulness are detected early in auditory lexical processing: Evidence from electroencephalography. *Brain and Language*, *122*(2), 81–91. https://doi.org/10.1016/j.bandl.2012.05.005

Lüdecke, D., Ben-Shachar, M. S., Patil, I., Wiernik, B. M., Bacher, E., Thériault, R., & Makowski, D. (2022). easystats: Framework for easy statistical modeling, visualization, and reporting. *CRAN*. https://doi.org/10.32614/CRAN.package.easystats

Maris, E. (2011). Statistical testing in electrophysiological studies. *Psychophysiology*, *49*(4), 549–565. https://doi.org/10.1111/j.1469-8986.2011.01320.x

Maris, E., & Oostenveld, R. (2007). Nonparametric statistical testing of EEG- and MEG-data. *Journal of Neuroscience Methods*, *164*(1), 177–190. https://doi.org/10.1016/j.jneumeth.2007.03.024

Meulman, N., Sprenger, S. A., Schmid, M. S., & Wieling, M. (2023). GAM-based individual difference measures for L2 ERP studies. *Research Methods in Applied Linguistics*, *2*(3),

100079. https://doi.org/10.1016/j.rmal.2023.100079
Meulman, N., Wieling, M., Sprenger, S. A., Stowe, L. A., & Schmid, M. S. (2015). Age Effects in L2 Grammar Processing as Revealed by ERPs and How (Not) to Study Them. *PLOS ONE*, *10*(12), e0143328. https://doi.org/10.1371/journal.pone.0143328
Microsoft, & Weston, S. (2022). *foreach: Provides foreach looping construct*. https://CRAN.R-project.org/package=foreach
Miller, D. L. (2025). Bayesian views of generalized additive modelling. *Methods in Ecology and Evolution*. https://doi.org/10.1111/2041-210x.14498
Mills, B. R. (2022). *MetBrewer: Color palettes inspired by works at the metropolitan museum of art*. https://CRAN.R-project.org/package=MetBrewer
Nalborczyk, L. (2025). *neurogam: Precise temporal localisation of m/EEG effects with bayesian generalised additive multilevel models*. https://github.com/lnalborczyk/neurogam
Nalborczyk, L., Batailler, C., Lœvenbruck, H., Vilain, A., & Bürkner, P.-C. (2019). An Introduction to Bayesian Multilevel Models Using brms: A Case Study of Gender Effects on Vowel Variability in Standard Indonesian. *Journal of Speech, Language, and Hearing Research*, *62*(5), 1225–1242. https://doi.org/10.1044/2018_jslhr-s-18-0006
Nalborczyk, L., Hauw, F., Torcy, H. de, Dehaene, S., & Cohen, L. (in preparation). *Neural and representational dynamics of tickertape synesthesia*.
Pedersen, E. J., Miller, D. L., Simpson, G. L., & Ross, N. (2019). Hierarchical generalized additive models in ecology: An introduction with mgcv. *PeerJ*, *7*, e6876. https://doi.org/10.7717/peerj.6876
Pedersen, T. L. (2024). *patchwork: The composer of plots*. https://CRAN.R-project.org/package=patchwork
Pedersen, T. L., & Crameri, F. (2023). *scico: Colour palettes based on the scientific colour-maps*. https://CRAN.R-project.org/package=scico
Pernet, C. R. (2022). Electroencephalography robust statistical linear modelling using a single weight per trial. *Aperture Neuro*, *2*, 1–22. https://doi.org/10.52294/apertureneuro.2022.2.seoo9435
Pernet, C. R., Chauveau, N., Gaspar, C., & Rousselet, G. A. (2011). LIMO EEG: A Toolbox for Hierarchical LInear MOdeling of ElectroEncephaloGraphic Data. *Computational Intelligence and Neuroscience*, *2011*, 1–11. https://doi.org/10.1155/2011/831409
Pernet, C. R., Latinus, M., Nichols, T. E., & Rousselet, G. A. (2015). Cluster-based computational methods for mass univariate analyses of event-related brain potentials/fields: A simulation study. *Journal of Neuroscience Methods*, *250*, 85–93. https://doi.org/10.1016/j.jneumeth.2014.08.003
R Core Team. (2025). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/
Rasmussen, C. E., & Williams, C. K. I. (2005). *Gaussian Processes for Machine Learning*. https://doi.org/10.7551/mitpress/3206.001.0001
Rigby, R. A., & Stasinopoulos, D. M. (2005). Generalized Additive Models for Location, Scale and Shape. *Journal of the Royal Statistical Society Series C: Applied Statistics*, *54*(3), 507–554. https://doi.org/10.1111/j.1467-9876.2005.00510.x
Rij, J. van, Hendriks, P., Rijn, H. van, Baayen, R. H., & Wood, S. N. (2019). Analyzing the Time Course of Pupillometric Data. *Trends in Hearing*, *23*. https://doi.org/10.1177/2331216519832483
Riutort-Mayol, G., Bürkner, P.-C., Andersen, M. R., Solin, A., & Vehtari, A. (2023). Practical Hilbert space approximate Bayesian Gaussian processes for probabilistic programming. *Statistics and Computing*, *33*(1), 17. https://doi.org/10.1007/s11222-022-10167-2
Rodriguez-Sanchez, F., & Jackson, C. P. (2024). *grateful: Facilitate citation of R packages*. https://pakillo.github.io/grateful/
Rosenblatt, J. D., Finos, L., Weeda, W. D., Solari, A., & Goeman, J. J. (2018). All-

Resolutions Inference for brain imaging. *NeuroImage*, *181*, 786–796. https://doi.org/10.1016/j.neuroimage.2018.07.060

Rousselet, G. A. (2025). Using cluster-based permutation tests to estimate MEG/EEG onsets: How bad is it? *European Journal of Neuroscience*, *61*(1), e16618. https://doi.org/10.1111/ejn.16618

Rousselet, G. A., Pernet, C. R., Bennett, P. J., & Sekuler, A. B. (2008). Parametric study of EEG sensitivity to phase noise during face processing. *BMC Neuroscience*, *9*(1). https://doi.org/10.1186/1471-2202-9-98

Sassenhagen, J., & Draschkow, D. (2019). Cluster-based permutation tests of MEG/EEG data do not establish significance of effect latency or location. *Psychophysiology*, *56*(6). https://doi.org/10.1111/psyp.13335

Silge, J., & Robinson, D. (2016). tidytext: Text mining and analysis using tidy data principles in r. *JOSS*, *1*(3). https://doi.org/10.21105/joss.00037

Skukies, R., & Ehinger, B. (2021). Modelling event duration and overlap during EEG analysis. *Journal of Vision*, *21*(9), 2037. https://doi.org/10.1167/jov.21.9.2037

Skukies, R., Schepers, J., & Ehinger, B. (2024, December 9). *Brain responses vary in duration - modeling strategies and challenges*. https://doi.org/10.1101/2024.12.05.626938

Slowikowski, K. (2024). *ggrepel: Automatically position non-overlapping text labels with "ggplot2"*. https://CRAN.R-project.org/package=ggrepel

Smith, N. J., & Kutas, M. (2014a). Regression-based estimation of ERP waveforms: I. The rERP framework. *Psychophysiology*, *52*(2), 157–168. https://doi.org/10.1111/psyp.12317

Smith, N. J., & Kutas, M. (2014b). Regression-based estimation of ERP waveforms: II. Nonlinear effects, overlap correction, and practical considerations. *Psychophysiology*, *52*(2), 169–181. https://doi.org/10.1111/psyp.12320

Smith, S., & Nichols, T. (2009). Threshold-free cluster enhancement: Addressing problems of smoothing, threshold dependence and localisation in cluster inference. *NeuroImage*, *44*(1), 83–98. https://doi.org/10.1016/j.neuroimage.2008.03.061

Sóskuthy, M. (2017). *Generalised additive mixed models for dynamic analysis in linguistics: A practical introduction*. https://doi.org/10.48550/ARXIV.1703.05339

Sóskuthy, M. (2021). Evaluating generalised additive mixed modelling strategies for dynamic speech analysis. *Journal of Phonetics*, *84*, 101017. https://doi.org/10.1016/j.wocn.2020.101017

Teichmann, L. (2022). An empirically driven guide on using bayes factors for m/EEG decoding. *Aperture Neuro*, *2*, 1–10. https://doi.org/10.52294/apertureneuro.2022.2.maoc6465

Tremblay, A., & Newman, A. J. (2014). Modeling nonlinear relationships in ERP data using mixed-effects regression with R examples. *Psychophysiology*, *52*(1), 124–139. https://doi.org/10.1111/psyp.12299

Umlauf, N., Klein, N., & Zeileis, A. (2018). BAMLSS: Bayesian Additive Models for Location, Scale, and Shape (and Beyond). *Journal of Computational and Graphical Statistics*, *27*(3), 612–627. https://doi.org/10.1080/10618600.2017.1407325

Ushey, K., Allaire, J., & Tang, Y. (2024). *Reticulate: Interface to 'python'*. https://CRAN.R-project.org/package=reticulate

Vaughan, D., & Dancho, M. (2022). *furrr: Apply mapping functions in parallel using futures*. https://CRAN.R-project.org/package=furrr

Wickham, H. (2019). *assertthat: Easy pre and post assertions*. https://CRAN.R-project.org/package=assertthat

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

Wickham, H., Pedersen, T. L., & Seidel, D. (2023). *scales: Scale functions for visualization*. https://CRAN.R-project.org/package=scales

Wieling, M. (2018). Analyzing dynamic phonetic data using generalized additive mixed modeling: A tutorial focusing on articulatory differences between L1 and L2 speakers of English. *Journal of Phonetics*, *70*, 86–116. https://doi.org/10.1016/j.wocn.2018.03.002

Winter, B., & Wieling, M. (2016). How to analyze linguistic change using mixed models, Growth Curve Analysis and Generalized Additive Modeling. *Journal of Language Evolution*, *1*(1), 7–18. https://doi.org/10.1093/jole/lzv003

Wood, S. N. (2003). Thin Plate Regression Splines. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *65*(1), 95–114. https://doi.org/10.1111/1467-9868.00374

Wood, S. N. (2017a). *Generalized Additive Models*. Chapman; Hall/CRC. https://doi.org/10.1201/9781315370279

Wood, S. N. (2017b). *Generalized additive models: An introduction with r* (2nd ed.). Chapman; Hall/CRC.

Wüllhorst, V., Wüllhorst, R., Overmeyer, R., & Endrass, T. (2025). Comprehensive Analysis of Event-Related Potentials of Response Inhibition: The Role of Negative Urgency and Compulsivity. *Psychophysiology*, *62*(2). https://doi.org/10.1111/psyp.70000

Xie, Y. (2014). knitr: A comprehensive tool for reproducible research in R. In V. Stodden, F. Leisch, & R. D. Peng (Eds.), *Implementing reproducible computational research*. Chapman; Hall/CRC.

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

Xie, Y. (2025). *knitr: A general-purpose package for dynamic report generation in R*. 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

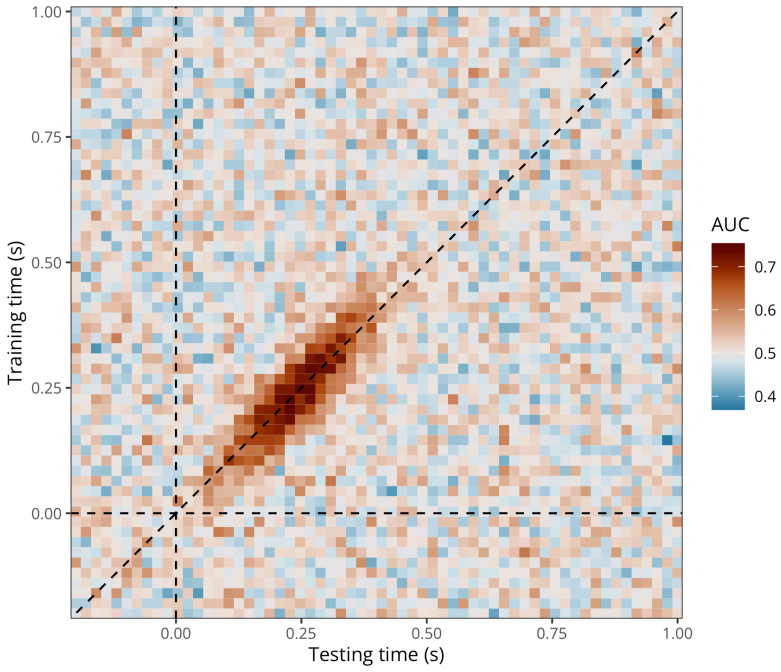
Yeung, N., Bogacz, R., Holroyd, C. B., & Cohen, J. D. (2004). Detection of synchronized oscillations in the electroencephalogram: An evaluation of methods. *Psychophysiology*, *41*(6), 822–832. https://doi.org/10.1111/j.1469-8986.2004.00239.x

## Appendix A
## Application to 2D time-resolved decoding results (cross-temporal generalisation)

Assume we have M/EEG data and we have conducted cross-temporal generalisation analyses (King & Dehaene, 2014). As a result, we have a 2D matrix where each element contains the decoding accuracy (e.g., ROC AUC) of a classifier trained at timestep training$_i$ and tested at timestep testing$_j$ (Figure A1).

**Figure A1**

*Exemplary (simulated) group-level average cross-temporal generalisation matrix of decoding performance (ROC AUC).*




To model cross-temporal generalisation matrices of decoding performance (ROC AUC), we extended the initial (decoding) GAM to take into account the bivariate temporal distribution of AUC values, thus producing naturally smoothed estimates (timecourses) of AUC values and posterior probabilities. This model can be written as follows:

$$\begin{aligned} \text{AUC}_i &\sim \text{Beta}(\mu_i, \phi) \\ g(\mu_i) &= f\left(\text{train}_i, \text{test}_i\right) \end{aligned}$$

where we assume that AUC values come from a Beta distribution with two parameters $\mu$ and $\phi$. We can think of $f\left(\text{train}_i, \text{test}_i\right)$ as a surface (a smooth function of two variables) that we can model using a 2-dimensional splines. Let $\mathbf{s}_i = (\text{train}_i, \text{test}_i)$ be some pair of training and testing samples, and let $\mathbf{k}_m = (\text{train}_m, \text{test}_m)$ denote the $m^{\text{th}}$ knot in the domain of $\text{train}_i$ and $\text{test}_i$. We can then express the smooth function as:

$$f\left(\text{train}_i, \text{test}_i\right) = \alpha + \sum_{m=1}^{M} \beta_m b_m \left(\tilde{s}_i, \tilde{k}_m\right)$$

Note that $b_m(,)$ is a basis function that maps $R \times R \to R$. A popular bivariate basis function uses *thin-plate splines* (Wood, 2003), which extend to $\mathbf{s}_i \in \mathbb{R}^d$ and $\partial l_g$ penalties. These splines are designed to interpolate and approximate smooth surfaces over two dimensions (hence the "bivariate" term). For $d = 2$ dimensions and $l = 2$ (smoothness penalty involving second order derivative):

$$f\left(\tilde{s}_i\right) = \alpha + \beta_1 x_i + \beta_2 z_i + \sum_{m=1}^{M} \beta_{2+m} b_m\left(\tilde{s}_i, \tilde{k}_m\right)$$

using the the radial basis function given by:

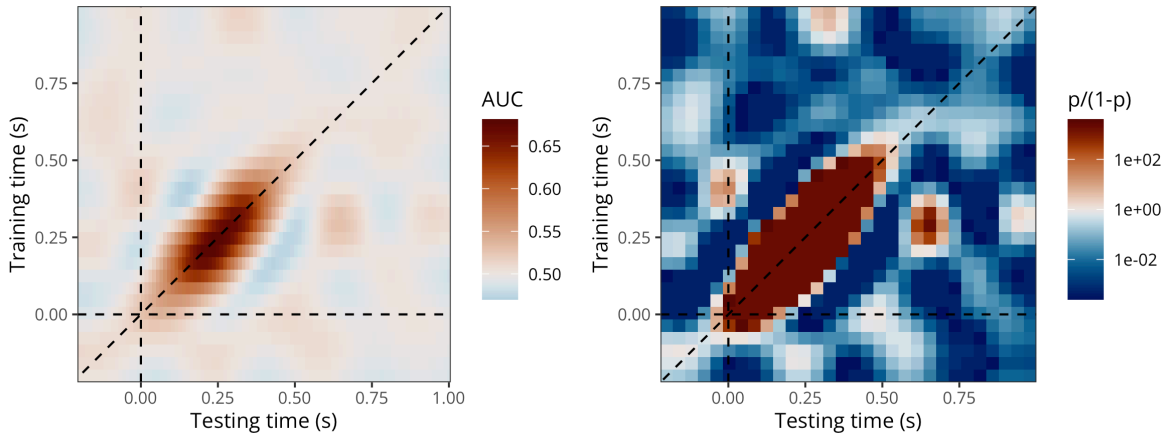
$$b_m\left(\tilde{s}_i, \tilde{k}_m\right) = \left\|\tilde{s}_i - \tilde{k}_m\right\|^2 \log\left\|\tilde{s}_i - \tilde{k}_m\right\|$$

where $\|\mathbf{s}_i - \mathbf{k}_m\|$ is the Euclidean distance between the covariate $\mathbf{s}_i$ and the knot location $\mathbf{k}_m$. We fitted this model using `brms`...

```
# fitting a GAM with two temporal dimensions
timegen_gam <- brm(
    # 2D thin-plate spline (tp)
    auc ~ t2(train_time, test_time, bs = "tp", k = 20),
    # auc ~ t2(train_time, test_time, bs = "tp", k = 30),
    data = timegen_data,
    family = Beta(),
    warmup = 1000,
    iter = 2000,
    chains = 8,
    cores = 8,
    file = "models/timegen_gam_t2.rds"
    # file = "models/timegen_gam_t2_k30.rds"
    )

# fitting a GP with two temporal dimensions
# timegen_gp <- brm(
#     auc ~ gp(train_time, test_time, k = 20),
#     data = timegen_data,
#     family = Beta(),
#     control = list(adapt_delta = 0.95),
#     iter = 2000,
#     chains = 4,
#     cores = 4,
#     file = "models/timegen_gp.rds"
#     )
```

**Figure A2**

*Predicted AUC values (left) and posterior probabilies of decoding accuracy being above chance level (right) according to the bivariate GAM.*




Could be extended to spatial and temporal dimensions with formulas such as `te(x, y, Time, d = c(2, 1) )`...

## Appendix B
## Alternative to GAMs: Approximate Gaussian Process regression

A Gaussian process (GP) is a stochastic process that defines the distribution over a collection of random variables indexed by a continuous variable, that is $\{f(t) : t \in \mathcal{T}\}$ for some index set $\mathcal{T}$ (Rasmussen & Williams, 2005; Riutort-Mayol et al., 2023). Whereas Bayesian linear regression outputs a distribution over the parameters of some predefined parametric model, the GP approach, in contrast, is a non-parametric approach, in that it finds a distribution over the possible functions that are consistent with the observed data. However, note that nonparametric does not mean there aren't parameters, it means that there are infinitely many parameters.

From brms documentation: A GP is a stochastic process, which describes the relation between one or more predictors $x = (x_1, \ldots, x_d)$ and a response $f(x)$, where $d$ is the number of predictors. A GP is the generalization of the multivariate normal distribution to an infinite number of dimensions. Thus, it can be interpreted as a prior over functions. The values of $f()$ at any finite set of locations are jointly multivariate normal, with a covariance matrix defined by the covariance kernel $k_p(x_i, x_j)$, where $p$ is the vector of parameters of the GP:

$$\left(f(x_1), \ldots f(x_n) \sim \text{MVN}\left(0, \left(k_p(x_i, x_j)\right)_{i,j=1}^{n}\right)\right.$$

The smoothness and general behaviour of the function $f$ depends only on the choice of covariance kernel, which ensures that values that are close together in the input space will be mapped to similar output values...

From this perspective, $f$ is a realisation of an infinite dimensional normal distribution:

$$f \sim \text{Normal}(0, C(\lambda))$$

where $C$ is a covariance kernel with hyperparameters $\lambda$ that defines the covariance between two function values $f(t_1)$ and $f(t_2)$ for two time points $t_1$ and $t_2$ (Rasmussen & Williams, 2005). Similar to the different choices of the basis function for splines, different choices of the covariance kernel lead to different GPs. In this article, we consider the squared-exponential (a.k.a. radial basis function) kernel, which computes the squared distance between points and converts it into a measure of similarity. It is defined as:

$$C(\lambda) := C(t_1, t_2, \sigma, \gamma) := \sigma^2 \exp\left(-\frac{||t_1 - t_2||^2}{2\gamma^2}\right)$$

with hyperparameters $\lambda = (\sigma, \gamma)$, expressing the overall scale of GP and the length-scale, respectively (Rasmussen & Williams, 2005). The advantages of this kernel are that it is computationally efficient and (infinitely) smooth making it a reasonable choice for the purposes of the present article. Here again, $\lambda$ hyperparameters are estimated from the data, along with all other model parameters.

Taken from https://michael-franke.github.io/Bayesian-Regression/practice-sheets/10c-Gaussian-processes.html: For a given vector $\mathbf{x}$, we can use the kernel to construct finite multi-variate normal distribution associated with it like so:

$$\mathbf{x} \mapsto_{GP} \text{MVNormal}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}))$$

where $m$ is a function that specifies the mean for the distribution associated with $\mathbf{x}$. This mapping is essentially the Gaussian process: a systematic association of vectors of arbitrary length with a suitable multi-variate normal distribution.

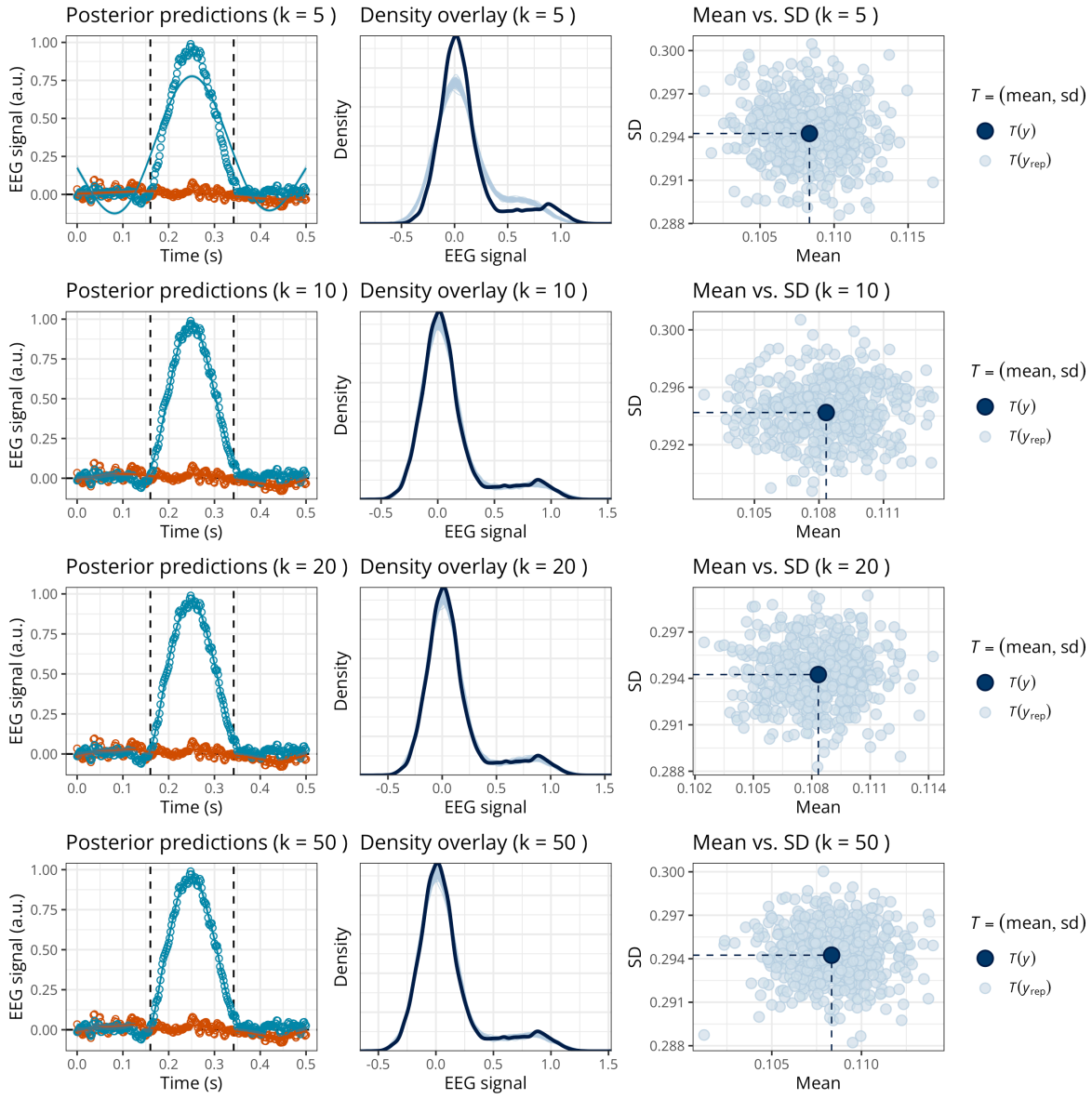
Low-rank approximate Gaussian processes are of main interest in machine learning and statistics due to the high computational demands of exact Gaussian process models (Riutort-Mayol et al., 2023)...

## Appendix C
## How to choose the GAM basis dimension?

Here provide recommendation about how to define `k`. An option is to vary `k` and examine the predictions and posterior predictive checks (PPCs) of each model... In this example (Figure C1)... However, it is not possible to provide general recommendations, as the optimal `k` depends on the sampling rate, the preprocessing steps (e.g., signal-to-noise ratio, low-pass filtering, etc), and the neural dynamics of the phenomenon under study.

### Figure C1

*Posterior predictions and posterior predictive checks for the GAM with varying k (in rows).*



## Appendix D
## `R` package and integration with `MNE-Python`

For users who are already familiar with `brms`, the recommended pipeline is to import ERPs or decoding results in `R` and analyse these data using the code provided in the main paper. However, it is also possible to call functions from the `neurogam R` package (available at https://github.com/lnalborczyk/neurogam), which come with sensible defaults.

```r
# installing (if needed) and loading the neurogam R package
# remotes::install_github("https://github.com/lnalborczyk/neurogam")
library(neurogam)

# using the testing_through_time() function from the neurogam package
# this may take a few minutes (or hours depending the machine's performance
# and data size)...
gam_onset_offset <- testing_through_time(
    # dataframe with M/EEG data in long format
    data = raw_df,
    # threshold for defining clusters (20 by default)
    threshold = 20,
    # the *_id arguments are use to specify the relevant columns in data
    participant_id = "participant", meeg_id = "eeg",
    time_id = "time", predictor_id = "condition",
    # number of warmup MCMC iterations
    warmup = 1000,
    # total number of MCMC iterations
    iter = 5000,
    # number of MCMCs
    chains = 4,
    # number of parallel cores to use for running the MCMCs
    cores = 4
    )

# displaying the results
gam_onset_offset$clusters
```

The `neurogam` package can also be called from `Python` using the `rpy2` module, and can easily be integrated into `MNE-Python` pipelines. For example, we use it below to estimate the onset and offset of effects for one EEG channel from a MNE evoked object. Note that the code used to reshape the `sample MNE` dataset is provided later, and we refer to the MNE documentation about converting `MNE` epochs to `Pandas` dataframes in long format.

```python
# loading the Python modules
import rpy2.robjects as robjects
from rpy2.robjects.packages import importr
from rpy2.robjects import pandas2ri
from rpy2.robjects.conversion import localconverter

# importing the "neurogam" R package
neurogam = importr("neurogam")

# activating automatic pandas-R conversion
pandas2ri.activate()
```

```
# assuming reshaped_df is some M/EEG data reshaped in long format
with localconverter(robjects.default_converter + pandas2ri.converter):

    reshaped_df_r = robjects.conversion.py2rpy(reshaped_df)


# using the testing_through_time() function from the neurogam R package
gam_onset_offset = neurogam.testing_through_time(
    data=reshaped_df_r,
    threshold=10,
    multilevel=False
    )

# displaying the results
print(list(gam_onset_offset) )
```

```
# loading the Python modules
import mne
from mne.datasets import sample
import numpy as np
import pandas as pd

# defining the path to the "sample" dataset
path = sample.data_path()

# loading the evoked data
evokeds = mne.read_evokeds(path / "MEG" / "sample" / "sample_audvis-ave.fif")

# defining a function to reshape the data
def reshape_eeg_channels_as_participants(evokeds, condition_labels):

    all_dfs = []

    for evoked, condition in zip(evokeds, condition_labels):

        # selecting only EEG channels
        picks = mne.pick_types(evoked.info, meg=False, eeg=True)
        data = evoked.data[picks, :]
        channel_names = np.array(evoked.ch_names)[picks]

        n_channels, n_times = data.shape
        # repeating time for each channel
        times = np.tile(evoked.times, n_channels)
        meeg_values = data.flatten()
        pseudo_participant_ids = np.repeat(channel_names, n_times)

        # converting to dataframe
        df = pd.DataFrame({
            "participant": pseudo_participant_ids,
            "time": times,
```

```
            "eeg": meeg_values,
            "condition": condition
            })

        all_dfs.append(df)

    return pd.concat(all_dfs, ignore_index=True)

# picking two evoked conditions
faces = evokeds[0]
scrambled = evokeds[1]

# reshaping data by pretending each EEG channel is a participant
reshaped_df = reshape_eeg_channels_as_participants(
    evokeds=[faces, scrambled],
    condition_labels=["faces", "scrambled"]
    )
```