Bayesian Random Phase- Amplitude Gaussian Process And Its Improvement

Enyu Li

May 8, 2025

Abstract

This report presents an in-depth exploration and implementation of the Random Phase-Amplitude Gaussian Process (RPAGP) model, a flexible Bayesian framework developed to enhance trial-level analysis of event-related potentials (ERPs) in electroencephalogram (EEG) studies. Traditional ERP analysis techniques frequently rely on averaging signals across trials and subjects to suppress noise. However, this approach may obscure critical trial-specific variations in amplitude and latency—features that are often linked to underlying cognitive and behavioral processes. The RPAGP model addresses this limitation by representing each trial as a phase- and amplitude-modified version of a shared structural signal, while modeling background brain activity with an autoregressive processn . By employing Gaussian process priors, the model flexibly captures the ERP waveform's structure, and full Bayesian inference enables uncertainty quantification of both shared and trial-specific parameters .

In addition to reviewing the theoretical and practical foundations of the RPAGP model, this report introduces two computational strategies—the nearest neighbor method and the projection method—to improve the scalability of Gaussian process inference. These methods are designed to reduce the computational burden typically associated with Gaussian processes, particularly in large-scale EEG datasets involving numerous trials and dense time series. Detailed mathematical explanations and implementation procedures for both acceleration techniques are provided. Empirical evaluations demonstrate that the projection method, in particular, achieves a significant reduction in computation time while preserving high estimation accuracy, as assessed by mean squared error (MSE) between the estimated and true signals.

Overall, the findings underscore the effectiveness of the RPAGP framework in generating cleaner and more interpretable ERP signals. With the incorporation of efficient computational methods, the model proves to be both statistically robust and computationally feasible for application to large-scale neuroscience research. These advancements enhance the practical utility of trial-level ERP analysis in cognitive and behavioral studies.

1 Introduction

Event-related potentials (ERPs) derived from electroencephalogram (EEG) recordings are widely used in cognitive neuroscience to investigate time-locked brain responses to external stimuli. Traditionally, ERP analyses rely on averaging EEG signals across multiple trials and subjects to suppress noise and emphasize neural activity patterns. Although this averaging method is effective in reducing random variability, it can obscure meaningful trial-specific differences in amplitude and latency—features that are often associated with cognitive, emotional, and behavioral processes. Accurately capturing these trial-level variations is essential for advancing the understanding of individual neural responses and improving the interpretability of EEG data.

To address these challenges, recent research has introduced the Random Phase-Amplitude Gaussian Process (RPAGP) model, a flexible Bayesian framework that infers single-trial ERPs by modeling each trial as a transformation of a shared structural signal (D.Pluta, 2024). The RPAGP model allows for variation in both amplitude and latency at the trial level while accounting for structured background brain activity through an autoregressive noise process. By incorporating Gaussian process priors, the model enables nonparametric estimation of the ERP waveform and supports full Bayesian inference to quantify uncertainty in both the shared signal and trial-specific characteristics. This approach has been shown to improve the signal-to-noise ratio, narrow confidence intervals, and enhance statistical power for detecting experimental effects when compared to traditional methods such as empirical bootstrapping, ANOVA, and ASEO.

One notable limitation of the RPAGP framework, however, is its computational complexity, especially when applied to large EEG datasets comprising numerous trials and high-resolution temporal measurements. To enhance scalability and feasibility for real-world applications, this report investigates two strategies for accelerating Gaussian process inference: the nearest neighbor method and the projection method. These techniques are integrated into the RPAGP framework to substantially reduce computational time while maintaining high estimation accuracy.

The report provides a comprehensive overview of the RPAGP model, including its mathematical foundations, modeling assumptions, and inference procedures. Theoretical and implementation details of the nearest neighbor and projection methods are also presented to clarify their role in improving computational efficiency. Experimental results demon-

strate that these enhancements lead to a significant reduction in runtime with minimal impact on accuracy, as measured by mean squared error (MSE). Taken together, these findings highlight the potential of the enhanced RPAGP framework as a robust and efficient tool for high-resolution ERP analysis in cognitive neuroscience research.

2 Gaussian Process

A Gaussian Process (GP) is a powerful non-parametric Bayesian model used for regression and classification. It is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution. This means that a GP defines a distribution over functions, allowing us to make predictions with associated uncertainty. GPs provide a principled, practical, and probabilistic approach to learning in kernel machines and are especially valued for their ability to quantify uncertainty in predictions.

Formally, a GP is defined as:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{1}$$

where $m(x)$ is the mean function, representing the expected value of the function at input x, and $k(x, x')$ is the covariance function (or kernel), which captures the similarity between function values at x and x'.

2.1 Mean and Covariance Functions

The behavior of functions drawn from a GP is determined by the choice of mean and covariance functions.

$$\begin{aligned} m(x) &= E[f(x)] \\ k(x, x') &= E[(f(x) - m(x))(f(x') - m(x'))] \end{aligned}$$

In practice, it is common to assume a zero mean function, $m(x) = 0$, without loss of generality, because any known mean can be subtracted from the data.

To clarify:

- x represents the observed input data points.
- x' refers to either the input grid points we select for making predictions or new data points where we want to estimate the output.

- $f(x)$ corresponds to the observed output (i.e., target or y values).
- $f(x')$ denotes the predicted output at the test points x'.

2.2 GP Regression

Suppose we are given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where each y_i is a noisy observation of an underlying latent function $f(x)$, i.e.,

$$y_i = f(x_i) + \epsilon_i, \quad \text{with } \epsilon_i \sim \mathcal{N}(0, \sigma_n^2) \tag{2}$$

Here, σ_n^2 denotes the variance of the observation noise, and ϵ_i represents independent identically distributed (i.i.d.) Gaussian noise. The goal is to infer the distribution of the function values $f(x)$ at some test inputs X_*, given the observed data.

Assuming a Gaussian Process prior over $f(x)$:

$$f(x) \sim \mathcal{GP}(0, k(x, x'))$$

we can compute the joint distribution of the observed outputs $\mathbf{y} = [y_1, \ldots, y_n]^\top$ and the function values $\mathbf{f}_*$ at the test points X_*. This joint distribution is multivariate normal:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K(X, X) + \sigma_n^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix}\right) \tag{3}$$

Here:

- $K(X, X)$ is the $n \times n$ covariance matrix computed using the kernel function over training inputs.
- $K(X, X_*)$ is the $n \times m$ covariance matrix between training and test points.
- $K(X_*, X_*)$ is the $m \times m$ covariance matrix over test points.
- I is the identity matrix accounting for Gaussian noise in the training outputs.

Using standard results for conditioning Gaussian distributions, the posterior distribution over the test function values $\mathbf{f}_*$ is also Gaussian:

$$\mathbf{f}_* \mid \mathbf{X}, \mathbf{y}, X_* \sim \mathcal{N}(\mu_*, \Sigma_*) \tag{4}$$

where the predictive mean and covariance are given by:

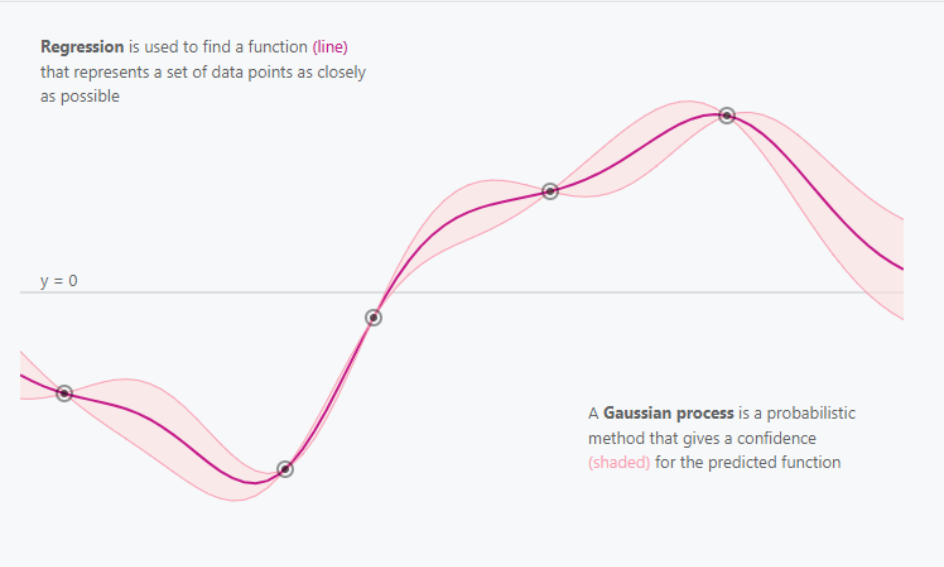

Figure 1: Gaussian Process regression example. The red curve represents the mean prediction, the shaded region shows the 95% confidence interval, black dots indicate training points.

$$\mu_* = K(X_*, X)\left[K(X, X) + \sigma_n^2 I\right]^{-1}\mathbf{y}$$
$$\Sigma_* = K(X_*, X_*) - K(X_*, X)\left[K(X, X) + \sigma_n^2 I\right]^{-1}K(X, X_*)$$

Interpretation:

- The predictive **mean** μ_* gives the best estimate of the function values at the test inputs.
- The predictive **covariance** Σ_* captures the uncertainty of the predictions — larger values indicate less confidence in the prediction at those inputs.
- The model automatically increases uncertainty in regions where fewer data points are available and reduces uncertainty where the data is dense.

Figure 1 visually illustrates the core idea of the projection method for handling large datasets in Gaussian Process modeling (Loper and Greydanus, 2019).

Advantages of GP Regression:

- It provides not just point predictions but also quantified uncertainty, which is critical in applications like active learning, Bayesian optimization, and robotics.
- GPs are non-parametric, meaning they can model very complex functions without explicitly defining a parametric form.
- The probabilistic nature of the model allows easy integration into larger Bayesian frameworks.

In summary, GP regression offers a flexible and elegant way to learn from data, particularly when we care about uncertainty in the predictions. However, its computational cost scales poorly with dataset size, which motivates the approximation strategies discussed in the next section.

2.3 Kernels and Their Roles

The kernel function $k(x, x')$ encapsulates assumptions about the underlying function we wish to learn. It controls the smoothness, periodicity, and general behavior of the function by defining the covariance between any two points x and x'.

The choice of kernel function in a Gaussian Process (GP) model significantly influences its ability to model the underlying function. Here are three widely used kernels, each suited for different modeling needs:

- **Squared Exponential (RBF) Kernel**:

$$k_{\text{RBF}}(x, x') = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2\ell^2}\right) \tag{5}$$

  This kernel assumes the function is very smooth and infinitely differentiable. It works well when the underlying function changes smoothly and gradually. However, it can be too restrictive for modeling data with abrupt changes or non-smooth behavior.

  Use when: You expect the function to be very smooth and differentiable, such as modeling physical processes or temperature changes over time.

- **Matérn Kernel**:

$$k_{\text{Matérn}}(x, x') = \sigma_f^2 \cdot \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}|x - x'|}{\ell} \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}|x - x'|}{\ell} \right) \quad (6)$$

  where $\nu > 0$ controls the smoothness, $\Gamma(\cdot)$ is the gamma function, and $K_\nu(\cdot)$ is the modified Bessel function of the second kind. The Matérn kernel allows for rougher functions depending on the value of ν:

  - $\nu = \frac{1}{2}$: Exponential kernel (not smooth)
  - $\nu = \frac{3}{2}$: Once differentiable
  - $\nu = \frac{5}{2}$: Twice differentiable

  Use when: You want to control the smoothness of the function. Ideal for modeling real-world phenomena that may not be perfectly smooth, such as geospatial data or financial time series.

- **Rational Quadratic Kernel**:

$$k_{\text{RQ}}(x, x') = \sigma_f^2 \left(1 + \frac{(x - x')^2}{2\alpha\ell^2} \right)^{-\alpha} \quad (7)$$

  This kernel is equivalent to a scale mixture of RBF kernels with varying length-scales. It can model data with both short-range and long-range variations more flexibly than a single RBF kernel.

  Use when: The smoothness of the function varies across the input space. Useful when the function exhibits both local and global patterns, such as heterogeneous sensor readings or variable user behaviors.

2.4 Computational Considerations

Despite their theoretical elegance, GPs face significant computational challenges when applied to large datasets. The core bottleneck lies in the inversion of the $n \times n$ covariance matrix, which is required for computing the posterior. This inversion has a time complexity of $\mathcal{O}(n^3)$ and a memory complexity of $\mathcal{O}(n^2)$, rendering standard GPs impractical for large n.

To address this, several approximation methods have been developed:

- **Sparse Gaussian Processes:** These methods introduce a smaller set of $m \ll n$ inducing points that summarize the training data. For instance, the Sparse Variational GP uses variational inference to optimize both the inducing inputs and their associated function values, reducing the complexity to $\mathcal{O}(nm^2)$.

- **Nyström Method:** This method constructs a low-rank approximation of the full kernel matrix using a selected subset of the data (landmark points). By performing eigen-decomposition on the smaller submatrix, it approximates the full matrix while preserving essential structure.

- **Nearest Neighbor and Projection-Based Methods:** These approaches further reduce computational cost by simplifying the modeling structure:

  - *Nearest Neighbor GP:* Only a fixed number of nearest training points are considered when predicting at a test point. This reduces the size of the kernel matrix and speeds up matrix operations.
  - *Projection Methods:* These reduce the dimensionality of the input space using techniques like Principal Component Analysis (PCA) or learned projections, thereby lowering the effective number of inputs the GP must handle.

In this report, I also apply both the nearest neighbor method and the projection method to reduce computational time when using Gaussian Processes. A detailed explanation of each method is provided. Moreover, empirical results show that these approximation methods significantly reduce runtime without sacrificing much accuracy — as measured by metrics such as Mean Squared Error (MSE).

3 Trial-Level ERP Analysis and the RPAGP Framework

Event-related potentials (ERPs) derived from electroencephalogram (EEG) recordings are crucial for understanding the brain's response to specific cognitive or sensory events. These signals are obtained by averaging time-locked EEG segments following a stimulus or cognitive event, which helps to reduce the impact of the brain's ongoing activity and other

noise. However, this averaging process assumes homogeneity across trials, neglecting potentially meaningful variability in trial-specific characteristics such as amplitude and latency. In this section, we provide an extended overview of the state-of-the-art in ERP analysis, the limitations of traditional methods, and the development and utility of the Random Phase-Amplitude Gaussian Process (RPAGP) model for trial-level ERP inference.

3.1 Limitations of Traditional ERP Analysis

Traditional ERP analysis typically relies on averaging EEG responses across trials and subjects. This process is justified by the need to enhance the signal-to-noise ratio (SNR) in the presence of substantial noise and weak ERP signals. Researchers often focus on specific components such as the P300 or the Late Positive Potential (LPP), defining time-windows of interest (e.g., 300–700 ms post-stimulus) and computing average voltages within these windows. These average values are then subjected to statistical tests, such as repeated-measures ANOVAs or bootstrap procedures, to identify group-level differences.

While these methods have yielded significant findings in cognitive neuroscience, they inherently overlook trial-level dynamics. In particular, the variability in individual trial responses can offer valuable insights into the neural correlates of behavior and cognition. Moreover, preprocessing steps, such as artifact rejection and baseline correction, may inadvertently suppress meaningful trial-specific variability. As a result, the derived average ERP may not accurately represent the diversity of neural responses across trials.

3.2 Introduction to Trial-Specific Modeling Frameworks

To capture the richness of ERP data at the single-trial level, researchers have proposed various modeling frameworks. One influential approach is the Variable Signal plus Ongoing Activity (VSPOA) model. In this framework, the observed EEG signal for each trial is modeled as a linear combination of a fixed number of components, each shifted by a trial-specific latency and amplitude, plus ongoing background activity.

Early implementations of VSPOA models focused on estimating trial-specific latencies (e.g., through cross-correlation methods), which were later extended to include amplitude estimation. A key advancement

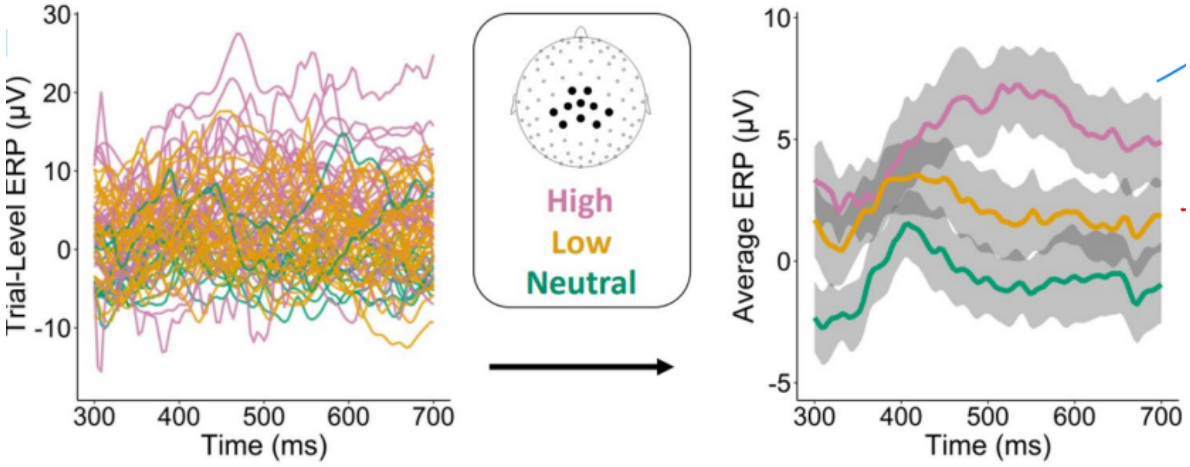

Figure 2: Single-subject analysis of trial-level ERP signals
Figure reproduced from Pluta et al., Scientific Reports 2024

was the development of the Differentially Variable Component Analysis (dVCA), a Bayesian approach that provides maximum a posteriori (MAP) estimates of component waveforms, latencies, and amplitudes. However, dVCA assumes that background activity is white noise and restricts latency estimates to integer multiples of the sampling interval, limiting its flexibility. The ERP plot or figure 2 shows that while averaging across trials produces smoother waveforms, it obscures substantial trial-to-trial variability in amplitude and latency. This highlights a key limitation of traditional averaging approaches, as meaningful neural dynamics are lost in the process (Pluta et al., 2024).

3.3 ASEO and its Improvements Over dVCA

To address the limitations of dVCA, the Analysis of Single-trial ERP and Ongoing activity (ASEO) model was proposed. ASEO builds on the VSPOA framework by assuming that background brain activity follows an autoregressive process rather than white noise. This assumption better reflects the temporal correlations in EEG data. Furthermore, ASEO allows latency estimates on a continuous scale, enhancing the temporal resolution of the inferred ERP components.

Despite these improvements, ASEO and dVCA share a common limitation: they require manual initialization of the number of components and their waveforms. In practice, this initialization is often based on segments of the average ERP, introducing potential bias. This data-driven

approach may reinforce patterns that are more reflective of the average than of genuine trial-specific dynamics, thus compromising the model's ability to uncover novel insights.

3.4 Alternative Approaches: Filtering and Machine Learning

Beyond VSPOA models, a variety of other techniques have been employed to analyze trial-level ERP data. These include:

- **Spatial and wavelet filtering:** These methods decompose EEG signals into spatial or frequency components, facilitating the extraction of ERP-related features.
- **Graph-based variability metrics:** These approaches characterize variability across trials using graph-theoretical measures.
- **Linear mixed models:** By modeling fixed and random effects, these models can account for trial-level and subject-level variability simultaneously.

In recent years, machine learning methods have gained popularity for single-trial EEG classification. Techniques such as support vector machines (SVMs), deep neural networks, and topological data analysis have been applied to raw EEG data to predict stimulus conditions or cognitive states. While effective in classification tasks, these methods are often black-box models that lack interpretability and are not suited for inference about ERP structure or trial-specific parameter estimation.

3.5 The RPAGP Model: Motivation and Structure

The Random Phase-Amplitude Gaussian Process (RPAGP) model addresses several of the shortcomings of earlier trial-specific modeling approaches. RPAGP is a flexible, fully Bayesian model that assumes each EEG trial is generated from a shared structural signal, modulated by trial-specific amplitude and latency shifts, and contaminated by autoregressive background activity.

Key features of RPAGP include:

- Use of Gaussian Process priors for the shared structural signal, allowing flexible non-parametric modeling.

- Continuous-time latency estimation, improving temporal accuracy.
- Autoregressive noise modeling, reflecting the real structure of background EEG activity.
- Posterior inference over all parameters, enabling full uncertainty quantification.

Unlike dVCA and ASEO, RPAGP does not require manual initialization of the number of components or their waveforms. Instead, it learns these quantities from the data, avoiding biases introduced by average ERP-driven initialization. This enables more objective and data-driven trial-level analysis.

3.6 Application to Emotion-Induced ERP Data

The RPAGP model was applied to a dataset in which subjects viewed images categorized as high, low, or neutral in emotional arousal. The ERP response of interest was the Late Positive Potential (LPP), measured from 300 to 700 ms post-stimulus. Traditional empirical analyses using bootstrap confidence intervals showed that while the high-arousal condition exhibited a stronger LPP response, the wide confidence bands made it difficult to determine statistically significant differences between the low and neutral conditions.

When RPAGP was applied, the posterior mean ERP curves by condition were similar in shape to the empirical means but featured much narrower credible intervals. The model effectively separated responses across conditions and filtered out ongoing background noise, enhancing the detection of meaningful differences.

Interestingly, in the neutral condition, the RPAGP estimate was nearly flat, while the empirical mean showed higher voltages. This discrepancy highlights the model's assumption of a shared ERP shape across conditions. In trials with low amplitude responses, such as those in the neutral condition, the posterior mean may be suppressed if the signal is not consistently present across all trials.

3.7 Comparison of Empirical vs. Model-Based Estimation

Empirical ERP estimation treats each condition independently and averages trials, leading to higher variability and potential contamination by

background activity. In contrast, RPAGP jointly models all conditions, assuming a shared structure. This allows the model to distinguish consistent signal patterns from random fluctuations, resulting in improved sensitivity and interpretability.

While this assumption may cause RPAGP to under-represent condition-specific deviations that are weak or inconsistent, it significantly improves inference in cases where signal differences are robust. The model-based approach is particularly valuable in detecting subtle differences that may be obscured by noise in traditional averaging methods.

3.8 Conclusion and Future Directions

The RPAGP model represents a substantial advancement in the analysis of trial-level ERP data. By combining the strengths of Bayesian inference, Gaussian Process modeling, and autoregressive noise assumptions, RPAGP enables flexible, interpretable, and precise estimation of neural response characteristics. It overcomes many limitations of earlier models and provides a practical framework for modern ERP studies.

Future work may extend RPAGP to multi-channel EEG data, incorporate subject-level random effects, or integrate it with classification frameworks for joint modeling and prediction. As ERP research continues to evolve, tools like RPAGP will be essential for bridging the gap between complex neural dynamics and interpretable cognitive and behavioral insights.

4 Bayesian Modeling of Trial-Level ERP Signals

In this section, we present a comprehensive methodological framework for the Bayesian modeling of trial-level Event-Related Potentials (ERP) using the Random Phase-Amplitude Gaussian Process (RPAGP) model. This includes descriptions of the experimental data, EEG signal preprocessing, probabilistic modeling formulation, prior specification, inference algorithm, and applications for signal estimation and hypothesis testing.

4.1 Participants and Experimental Design

A total of 161 participants were recruited from Harris County, TX. Inclusion criteria included being aged between 18–55, absence of psychiatric

diagnoses, negative drug and pregnancy tests, and stable housing. Participants completed a picture viewing task designed to elicit emotional responses while EEG data were recorded.

4.2 Stimuli and Procedure

Participants viewed 80 images sourced from the International Affective Picture System (IAPS) and supplemental sources categorized as high, low, or neutral arousal. Each image was shown for 4 seconds, followed by a fixation cross during a 3–5 second inter-trial interval. The task lasted around 20 minutes with scheduled breaks.

4.3 EEG Acquisition and Preprocessing

EEG signals were recorded with a 129-channel Geodesic Sensor Net, sampled at 250 Hz, and referenced to Cz. Data were filtered (0.1–100 Hz), re-referenced to the average, corrected for ocular/motion artifacts, segmented into 900-ms epochs (including a 100 ms pre-stimulus baseline), and cleaned via artifact rejection and channel interpolation. We averaged 10 centroparietal channels to produce a single ERP trace per trial.

4.4 Model Overview

We define the observed ERP for trial i at time t as:

$$y_i(t) = \beta_i f(t - \tau_i) + v_i(t), \tag{8}$$

where f is a latent shared structural signal (modeled as a Gaussian process), β_i is the amplitude, τ_i is the latency, and $v_i(t)$ is trial-specific noise (Pluta et al., 2024).

4.5 Autoregressive Noise Modeling

The trial-specific noise component $v_i(t)$ accounts for ongoing EEG activity and is modeled using an AR(p) process:

$$v_i(t) = \sum_{j=1}^{p} \phi_j v_i(t - j) + \epsilon_i(t), \quad \epsilon_i(t) \sim \mathcal{N}(0, \sigma^2). \tag{9}$$

This structure captures inherent autocorrelations in EEG noise and improves signal estimation by modeling background rhythmic activity. The

order p is selected based on the ACF/PACF plots, with $p = 2$ typically sufficient. AR parameters ϕ_j are given Gaussian priors, and noise variance σ^2 is assigned an inverse-gamma prior (Pluta et al., 2024). Including this noise structure is key to distinguishing between stimulus-locked and non-stimulus brain activity.

4.6 Posterior Inference via MCMC

We employ a Metropolis-within-Gibbs sampling strategy. The posterior over parameters $\theta = (\beta, \tau, f, \phi, \rho, \sigma^2)$ is explored via the following steps:

- **Sampling β_i and τ_i:** Trial-specific amplitudes and latencies are updated using Metropolis-Hastings steps with Gaussian proposal distributions tuned for each subject.
- **Sampling f:** The latent signal f is sampled from the GP posterior using conditioning formulas for multivariate Gaussians.
- **Sampling AR parameters:** ϕ is updated via regression on lagged residuals, and σ^2 is sampled from its conditional inverse-gamma posterior.

We ensure convergence using the Gelman-Rubin statistic ($\hat{R}$), trace plots, and autocorrelation checks. Posterior samples provide de-noised, trial-aligned reconstructions:

$$\tilde{y}_i(t) = \beta_i f(t - \tau_i). \tag{10}$$

These curves are used for further statistical analysis.

4.7 Statistical Inference and Summary Metrics

We summarize ERP activity using the Late Positive Potential (LPP) mean in the 300–700 ms window. The LPP for trial i is:

$$\mu_i^{\text{LPP}} = \frac{1}{T} \sum_{t=300}^{700} \tilde{y}_i(t), \tag{11}$$

where T is the number of time points in the window. Posterior distributions of group means are compared via pairwise differences. We declare a significant difference if the 90% credible interval does not contain zero. This Bayesian testing accounts for full uncertainty and does not rely on normality assumptions.

4.8 Performance Measures

We assess signal quality and model reliability via two core metrics:

Signal-to-Noise Ratio (SNR): Defined as:

$$\text{SNR} = \frac{E[\mu^{\text{LPP}}]}{\text{CI Width}}, \tag{12}$$

SNR reflects the certainty of condition means. Higher SNR indicates more informative posteriors.

Standardized Measurement Error (SME): SME is the standard deviation of bootstrap LPP estimates across trials, normalized by their mean. Low SME indicates robust estimates. It complements SNR by capturing reliability of trial-level reconstruction.

We also visualize posterior predictive checks and overlay recovered ERPs against original data to assess model fidelity.

4.9 Implementation Details

The model is implemented in Python with PyMC3 and SciPy. We run 8000 MCMC iterations (3000 burn-in) and fix $f(400ms) = 1$ for identifiability. Priors: $\rho \sim \text{Gamma}(12, 1)$, $\phi_j \sim \mathcal{N}(0, 1)$, $\sigma^2 \sim \text{IG}(2, 2)$. Chains are parallelized and evaluated with $\hat{R}$ and effective sample size.

Typical runtime per subject is 1–2 hours. Sensitivity tests confirm robustness to hyperparameter choices. These steps ensure accurate inference and scalability for future studies.

4.10 Results

The RPAGP model generates smooth, trial-specific ERP estimates that maintain important variability across trials while producing cleaner, less noisy averages. Unlike traditional averaging, the posterior mean ERP curves here are sharp and distinct across conditions, demonstrating RPAGP's ability to enhance signal detection without losing trial-level detail (Pluta et al., 2024).

The violin plots and posterior density plots show that RPAGP captures not only the condition-wise differences in LPP means but also the full uncertainty distribution at the trial level. This ability to quantify trial-specific variability and precisely estimate inter-condition differences

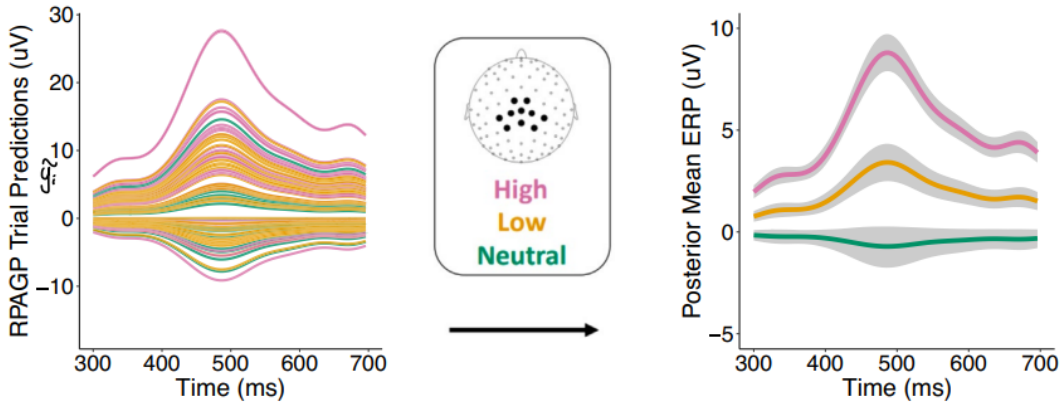

Figure 3: RPAGP Result
Figure reproduced from Pluta et al., Scientific Reports 2024

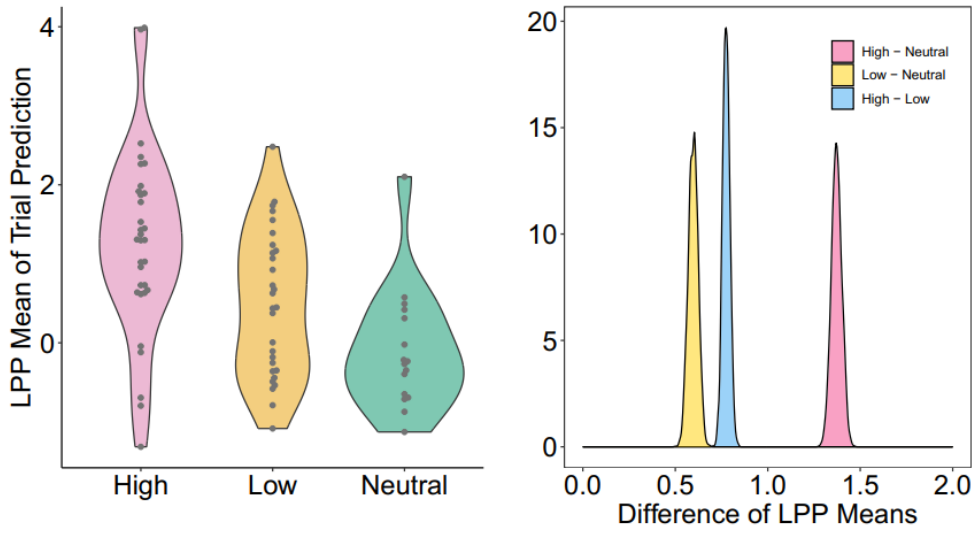

Figure 4. Inference on LPP means. (left) The posterior densities of the RPAGP estimates of trial LPP means by condition suggest that the selected subject exhibits the expected relations among the LPP means on average. There is also substantial variability apparent in the LPP means between trials within the same condition. (right) The posterior densities of the differences in LPP means are tightly concentrated and significantly greater than zero, in accordance with the expected relationships among conditions.

Figure 4: LPP Result
Figure reproduced from Pluta et al., Scientific Reports 2024

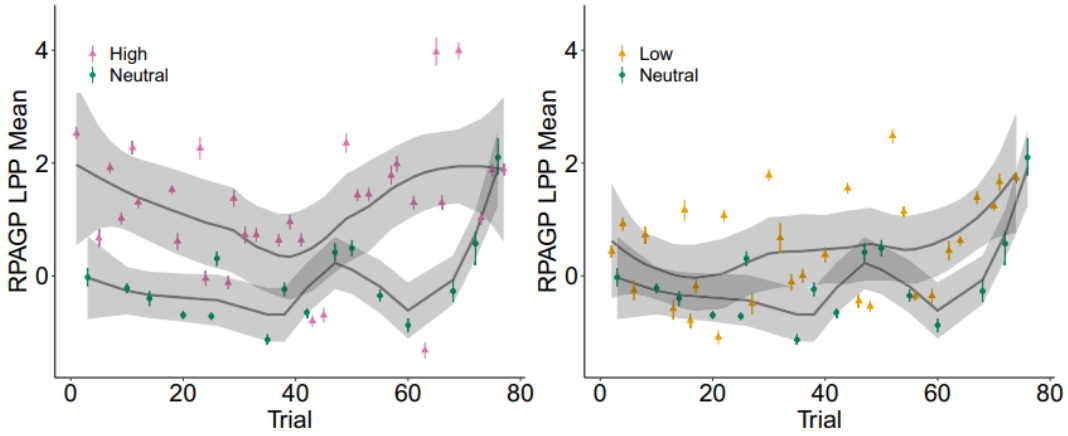

Figure 5: Trend Analysis Result
Figure reproduced from Pluta et al., Scientific Reports 2024

highlights RPAGP's strength in providing robust, interpretable statistical inference(Pluta et al., 2024).

RPAGP enables clear trend analysis across trials by modeling individual trial-level LPP means with credible intervals, revealing dynamic changes over time that traditional averaging would miss. The flexible Bayesian framework highlights subtle trends and uncertainties across trials and conditions, offering richer insights into temporal patterns in ERP data (Pluta et al., 2024).

5 Improvement on the Computational Time

Although the RPAGP model significantly reduces mean squared error (MSE) and improves accuracy, it is also computationally expensive. To address this issue, this section presents detailed explanations of the projection method and the nearest-neighbor method as strategies to reduce computation time.

5.1 Projection Method

Gaussian Processes (GPs) offer a flexible and probabilistically coherent framework for modeling functions whose exact forms are unknown. They are particularly useful because they provide not only point predictions but also associated measures of uncertainty. In GP models, we assume that any finite set of function values follows a multivariate Gaussian dis-

tribution. The mean and covariance structure of this distribution encapsulates our assumptions about smoothness, variability, and correlations across inputs.

However, a major drawback of GPs arises when handling large datasets. Specifically, if we have n observations, the GP requires inversion of an $n \times n$ covariance matrix, which typically incurs a computational complexity of $\mathcal{O}(n^3)$. This quickly becomes computationally impractical as n grows into the thousands or millions. Furthermore, even if computation were affordable, the storage requirements and potential numerical instabilities from poorly conditioned matrices can make the process unreliable. Thus, efficient approximation strategies are critical for enabling the use of GPs on modern large-scale datasets.

The **projection method** offers an elegant solution to this scalability issue. At its heart, the method proposes that instead of modeling the latent function at all n observed input points, we project it onto a lower-dimensional subspace defined by a much smaller set of representative points called *knots*. The goal is to summarize the major behavior of the function with these knots and reconstruct the function elsewhere using information about how each point relates to the knots.

Formally, let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ denote the full set of observed input locations, and let $\mathbf{z} = (z(x_1), z(x_2), \ldots, z(x_n))$ denote the latent GP values at these points. We also introduce a much smaller set of knot locations $\mathbf{x}^* = (x_1^*, x_2^*, \ldots, x_m^*)$, where $m \ll n$, and corresponding latent GP values $\mathbf{z}^* = (z(x_1^*), z(x_2^*), \ldots, z(x_m^*))$ (Savitsky et al., 2011) .

Instead of working directly with $\mathbf{z}$, we approximate it by the **conditional expectation** given the values at the knots. Using standard results from multivariate Gaussian theory, we know that the conditional distribution of $\mathbf{z}$ given $\mathbf{z}^*$ is Gaussian, and its mean is:

$$\mathbf{z}_{m \to n} = \mathbf{E}[\mathbf{z} \mid \mathbf{z}^*] = C(\mathbf{x}, \mathbf{x}^*) C(\mathbf{x}^*, \mathbf{x}^*)^{-1} \mathbf{z}^*,$$

where $C(\mathbf{x}, \mathbf{x}^*)$ is the $n \times m$ covariance matrix between the full data points and the knots, and $C(\mathbf{x}^*, \mathbf{x}^*)$ is the $m \times m$ covariance matrix among the knots themselves (Savitsky et al., 2011).

This formula can be interpreted as a *linear smoother* that reconstructs the full latent GP from a linear combination of the knot values $\mathbf{z}^*$. The weights in this combination are determined by how each data point correlates with the knots under the GP's covariance structure. Points that are more highly correlated with a particular knot will have a higher weight for that knot.

From an intuitive standpoint, this process is similar to how one might interpolate the value of a function based on a few strategically chosen support points. Imagine trying to reconstruct the shape of a long and winding road using a limited number of milestones. Although you lack detailed information about every bend and curve, by knowing the key landmarks and how the road behaves between them, you can form a fairly accurate mental picture of its entire path.

Having defined the projected latent GP $\mathbf{z}_{m\to n}$, the next step is to account for the real-world observations $\mathbf{y} = (y_1, \ldots, y_n)$. In practice, observations are rarely perfect reflections of the underlying function. They may be corrupted by measurement noise or affected by factors not captured in the model. To handle this, we assume that the observations are noisy versions of the projected GP values:

$$\mathbf{y} = \mathbf{z}_{m\to n} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \tau^2 I_n),$$

where $\boldsymbol{\epsilon}$ represents independent Gaussian noise with variance τ^2. The addition of $\boldsymbol{\epsilon}$ serves two purposes: it models intrinsic measurement error, and it compensates for the approximation error introduced by projecting the GP onto the lower-dimensional space.

The covariance structure of the observed data $\mathbf{y}$ now naturally follows from the properties of covariance under addition of independent random variables. Specifically, we obtain:

$$A_n = \mathrm{Cov}(\mathbf{y}) = \tau^2 I_n + C(\mathbf{x}, \mathbf{x}^*)C(\mathbf{x}^*, \mathbf{x}^*)^{-1}C(\mathbf{x}^*, \mathbf{x}).$$

The matrix A_n represents the total uncertainty in the observations, comprising both independent noise (the $\tau^2 I_n$ term) and structured spatial correlation inherited from the GP (the second term). It elegantly captures the idea that observations close together in input space are correlated due to the underlying smoothness of the function, while still allowing for random, independent variations at each point.

Despite this theoretical clarity, computational challenges still remain. To perform inference — such as predicting at new points or computing the marginal likelihood — we must invert A_n. A naive inversion would again cost $\mathcal{O}(n^3)$ operations, defeating the purpose of the approximation. To address this, we leverage the **Woodbury matrix identity**, a powerful algebraic tool that allows the inverse of a sum of matrices to be expressed in terms of smaller inverses:

$$(A + UCV^T)^{-1} = A^{-1} - A^{-1}U(C^{-1} + V^T A^{-1}U)^{-1}V^T A^{-1}.$$

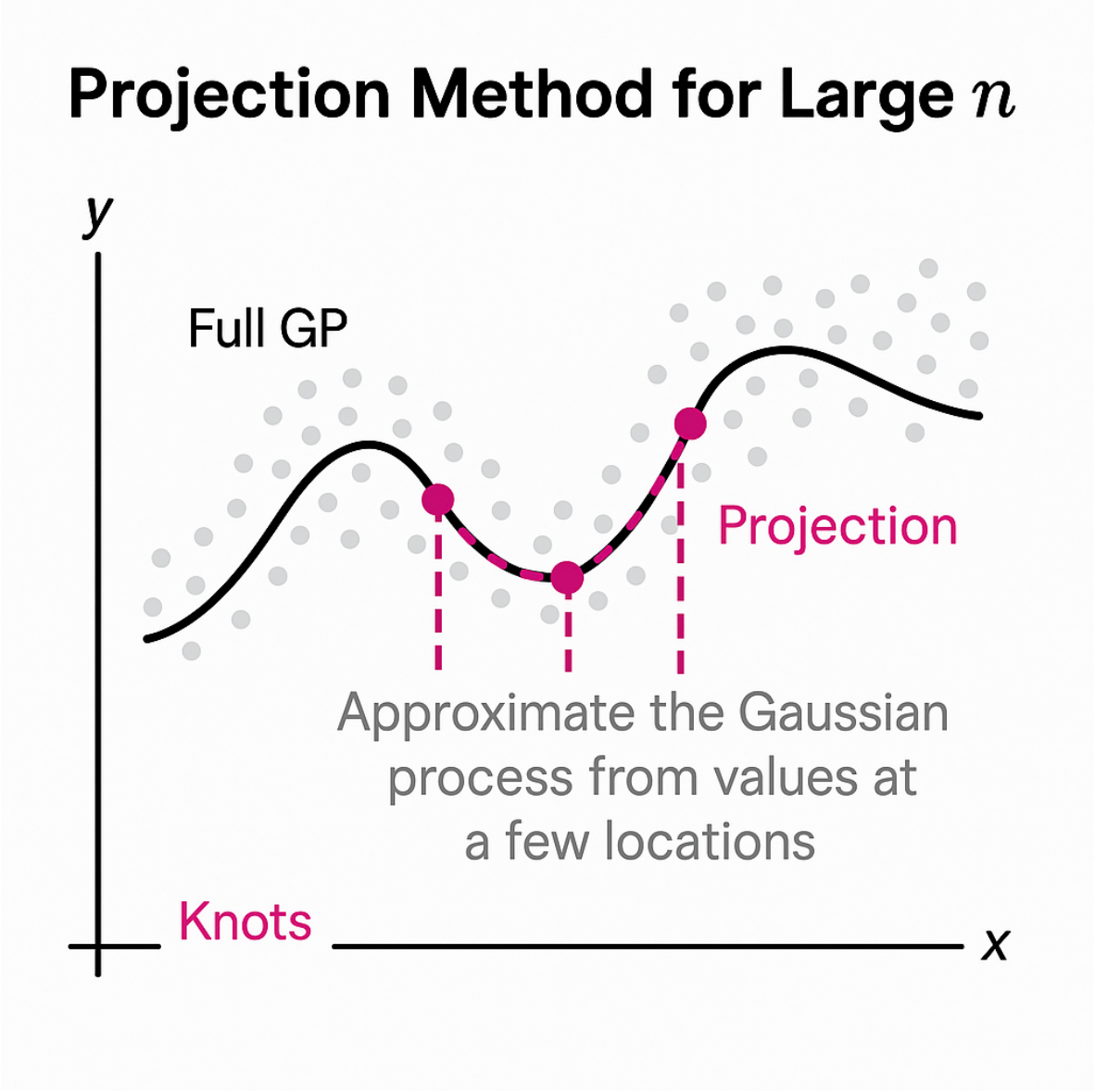

Figure 6: Single-subject analysis of trial-level ERP signals

Applying the Woodbury identity to A_n simplifies the problem dramatically:

$$A_n^{-1} = \tau^{-2} I_n - \tau^{-2} C(\mathbf{x}, \mathbf{x}^*) \left(C(\mathbf{x}^*, \mathbf{x}^*) + \tau^2 C(\mathbf{x}^*, \mathbf{x}) C(\mathbf{x}, \mathbf{x}^*) \right)^{-1} C(\mathbf{x}^*, \mathbf{x}) \tau^{-2}.$$

This transformation has an important consequence: instead of inverting an $n \times n$ matrix, we now invert an $m \times m$ matrix, where m is much smaller than n. This reduces the computational complexity to $\mathcal{O}(m^3)$, enabling scalable GP modeling for large datasets.

From an intuitive viewpoint, the Woodbury identity tells us that most of the complexity of the full covariance matrix arises from a simple

structure (the diagonal noise matrix) plus a low-rank adjustment (due to the GP structure projected onto the knots). Therefore, by carefully factoring the problem, we can adjust for the GP structure without recomputing everything from scratch.

Throughout this process, the choice of knots plays a crucial role. Ideally, the knots should be chosen to adequately represent the important variations in the input space. Techniques such as k-means clustering, random subset selection, or space-filling designs (like Latin Hypercube sampling) are often used to select knot locations. A poor choice of knots could result in a poor approximation and large errors, whereas a good choice leads to an efficient and accurate model.

In summary, the projection method embodies a fundamental strategy common to many areas of applied mathematics and statistics: approximate a complex, high-dimensional object using a simpler, lower-dimensional representation that captures most of the essential structure. By projecting the full GP onto the space spanned by a few carefully chosen knots, and by leveraging algebraic identities to facilitate computation, the projection method allows GPs to be used at scales that would otherwise be computationally prohibitive.

Ultimately, this approach retains the key advantages of Gaussian Processes — such as uncertainty quantification, smoothness assumptions, and probabilistic interpretation — while making them accessible for large modern datasets. It demonstrates how a deep understanding of mathematical structure can lead to practical, scalable algorithms that retain theoretical elegance.

Figure 6 visually illustrates the core idea of the projection method for handling large datasets in Gaussian Process modeling. The smooth black curve represents the full Gaussian Process (GP) function across the input space, while the many small gray dots represent the numerous observed data points. Instead of modeling the GP at all observed points, the method selects a smaller set of representative locations, called knots, shown as magenta points. These knots serve as anchors to reconstruct or approximate the function over the entire space. Dotted lines connecting the knots to the curve suggest the projection operation: information at the knots is used to predict the GP values at all other locations. By doing so, the complex full GP is efficiently approximated through a lower-dimensional structure, significantly reducing computational costs while preserving the main characteristics of the latent function. The method balances efficiency and fidelity by leveraging the natural smoothness and correlation properties inherent in the Gaussian Process framework.

6 Nearest-Neighbor Gaussian Processes: A Scalable Alternative to Full Gaussian Processes

6.1 Introduction to Full Gaussian Processes (GPs)

Gaussian Processes (GPs) are a fundamental modeling tool in statistics and machine learning, offering a flexible, probabilistic approach to function estimation. A GP defines a distribution over functions, such that for any finite set of input points $S = \{s_1, s_2, \ldots, s_n\}$, the corresponding function values $w_S = (w(s_1), w(s_2), \ldots, w(s_n))^\top$ are jointly Gaussian:

$$w_S \sim \mathcal{N}(0, K_\theta),$$

where K_θ is the covariance matrix determined by a covariance function $C_\theta(\cdot, \cdot)$ (Datta et al., 2016) . The covariance function captures assumptions about the function's smoothness, variability, and correlation over space. Common choices include the squared exponential and Matérn covariance functions.

Full GPs are particularly valued for their ability to provide not just point predictions but also uncertainty quantification. Given observed data, one can compute the posterior predictive distribution at new locations with closed-form expressions. This makes GPs a gold standard for non-parametric Bayesian regression and spatial modeling.

However, GPs suffer from significant computational challenges. Evaluating the likelihood, making predictions, and sampling from the posterior involve inverting the $n \times n$ covariance matrix K_θ, an operation that scales cubically with n ($\mathcal{O}(n^3)$). The memory requirement to store K_θ scales quadratically ($\mathcal{O}(n^2)$). Thus, traditional GP methods become impractical for datasets with more than a few thousand points (Datta et al., 2016).

6.2 Motivation for Approximation and Local Structure

To address these computational bottlenecks, various approximation strategies have been developed. A key insight motivating many of these approaches is the observation that in spatial data, the correlation between distant points often diminishes rapidly. That is, the strongest dependencies in the data are typically local. Consequently, when predicting the

function value at a location, it may suffice to condition only on nearby points rather than the entire dataset.

Sparse GP approximations leverage this local structure by introducing sparsity into the covariance or precision matrix. Methods such as inducing points, low-rank approximations, and local approximations have been proposed. Among these, nearest-neighbor-based methods offer a particularly intuitive and computationally efficient solution.

Nearest-Neighbor Gaussian Processes (NNGPs) capitalize on the locality principle by assuming that each function value depends primarily on a small subset of nearby observations. By restricting dependencies to local neighborhoods, NNGPs achieve scalability without sacrificing much predictive accuracy, especially in datasets where spatial locality dominates the correlation structure.

6.3 Transitioning from Full GP to Nearest-Neighbor GP

The transition from full GPs to NNGPs begins by reconsidering the factorization of the joint density. In a full GP, the joint distribution can be decomposed as a product of conditional densities:

$$p(w_S) = p(w(s_1)) \prod_{i=2}^{n} p(w(s_i) \mid w(s_1), \ldots, w(s_{i-1})).$$

Each conditional $p(w(s_i) \mid w(s_1), \ldots, w(s_{i-1}))$ depends on all previously observed points (Datta et al., 2016) . While exact, this formulation is computationally expensive.

NNGPs approximate each conditional by conditioning only on a small subset of the most relevant points, specifically the m nearest neighbors $N(s_i) \subset \{s_1, \ldots, s_{i-1}\}$, where $|N(s_i)| = m \ll n$:

$$p(w_S) \approx \prod_{i=1}^{n} p(w(s_i) \mid w(N(s_i))).$$

This approximation yields a sparse dependency structure, where each location is connected to only a few others (Datta et al., 2016). The resulting model remains a valid probabilistic model and offers substantial computational gains.

6.4 Detailed Explanation of Nearest-Neighbor Gaussian Processes (NNGP)

In the NNGP framework, each latent value $w(s_i)$ is modeled as a linear regression on its nearest neighbors plus a Gaussian residual term:

$$w(s_i) = \sum_{j \in N(s_i)} a_{ij} w(s_j) + \eta(s_i), \quad \eta(s_i) \sim \mathcal{N}(0, \Lambda_i).$$

The weights a_{ij} are obtained through kriging equations, ensuring optimal prediction based on neighbors:

$$a_i = C(s_i, N(s_i))C(N(s_i), N(s_i))^{-1},$$

where $C(s_i, N(s_i))$ is the covariance vector between s_i and its neighbors, and $C(N(s_i), N(s_i))$ is the covariance matrix among the neighbors. The residual variance Λ_i captures the uncertainty remaining after accounting for the neighbors (Datta et al., 2016):

$$\Lambda_i = C(s_i, s_i) - C(s_i, N(s_i))C(N(s_i), N(s_i))^{-1}C(N(s_i), s_i).$$

This local modeling structure introduces sparsity into the precision matrix (the inverse of the covariance matrix), enabling efficient computation of the likelihood and predictions. The sparsity also allows for parallel computation across locations, further enhancing scalability.

Importantly, although NNGPs introduce approximations, they define a valid stochastic process. The resulting process satisfies the Kolmogorov consistency conditions, ensuring that any finite-dimensional marginal distribution is multivariate Gaussian.

Figure 7 illustrates the fundamental concept behind Nearest-Neighbor Gaussian Processes (NNGP). In the plot, each blue dot represents a spatial data location where the latent process $w(s)$ is evaluated. Instead of modeling full pairwise dependencies between all locations — as is done in a full Gaussian Process — NNGPs restrict each point to depend only on a small number of its nearest neighbors, shown as green lines connecting a central point to a few surrounding ones. This local dependence structure forms the basis of a sparse approximation that significantly reduces computational complexity. The directed edges represent conditional dependencies, where each point is conditionally independent of all other points given its neighbors. By leveraging these sparse, localized interactions, NNGPs retain the key spatial relationships necessary for accurate prediction while enabling scalability to very large datasets.

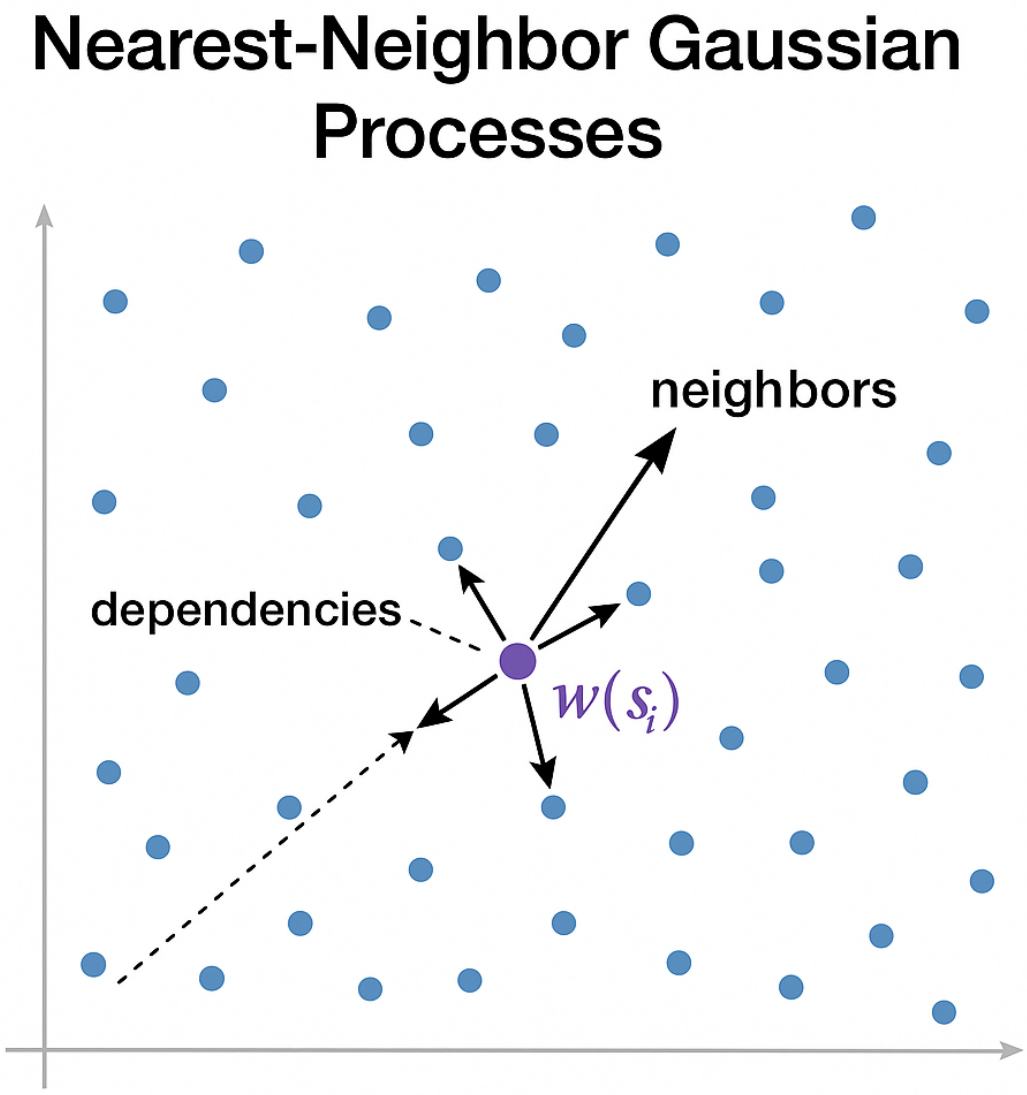

Figure 7: Single-subject analysis of trial-level ERP signals

6.5 Summary and Practical Implications

Nearest-Neighbor Gaussian Processes offer a scalable, principled alternative to full Gaussian Processes for modeling massive spatial datasets. By conditioning each location on a small number of neighbors, NNGPs preserve local spatial structure while significantly reducing computational costs. They enable GP modeling at scales previously inaccessible, with applications across environmental science, epidemiology, machine learning, and more.

While NNGPs introduce some approximation error, this is often negligible compared to the computational gains, particularly when neighbor sets are chosen carefully. The flexibility to trade off computational speed and statistical accuracy makes NNGPs a powerful tool in the modern statistician's and data scientist's toolbox.

7 Result of the Projection Method on Simulated Data

7.1 Graph Result

By applying the Projection Method to the simulated dataset, we demonstrated a significant improvement in computational efficiency. The simulated data consists of 78 rows and 100 columns, where each row corresponds to a discrete time point and each column represents a trial. We present several plots to illustrate the results: one showing the full Gaussian Process (GP) fit on the simulated data, and others showing the Projection Method results with different numbers of knots, specifically $m = 5$, $m = 15$, and $m = 25$ on the same simulated data.

For each different type (full GP) and others, we have two plots, one with all of the trials, the actual latent f function, empirical estimate of f(only using the data), and the GP estimate of f and the other with the actual latent f function, empirical estimate of f, GP estimate of f, and each estimate of f.

Notice that the full GP method can estimate the true latent function f very accurately with a computational time of around 6.338 seconds. The mean square error (MSE) is 0.0001637398.

Figure 8 contains all of the trials we observe, and figure 9 contains

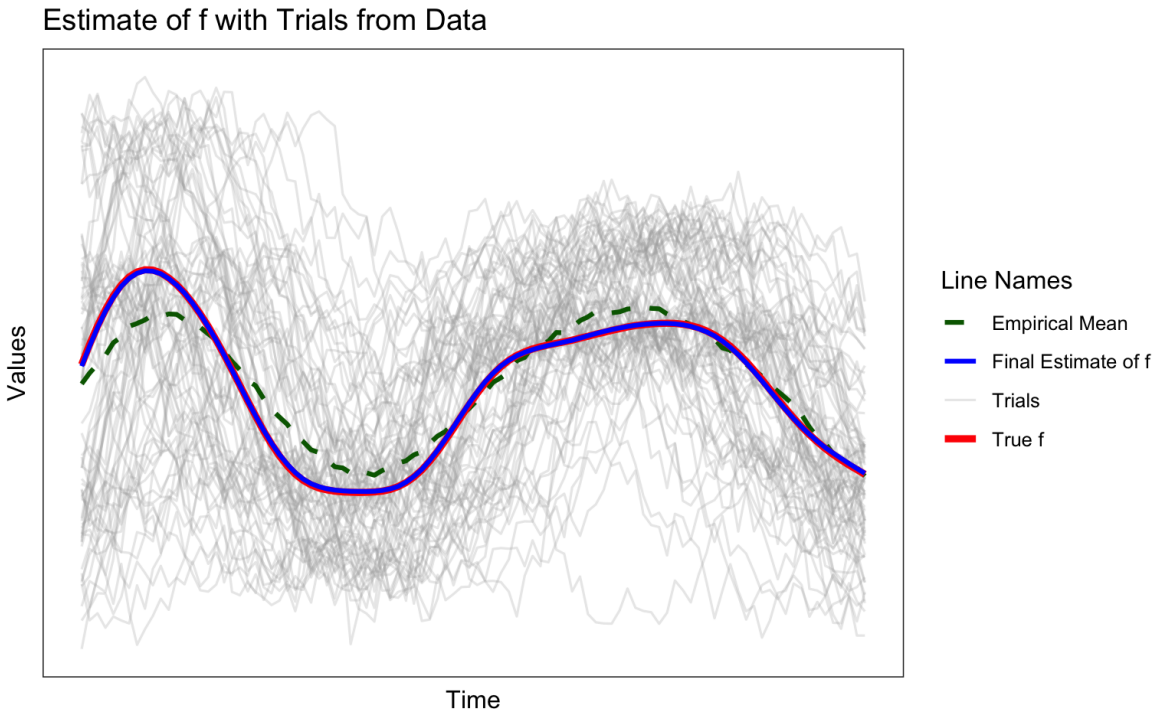

Figure 8: The Trial Plot of Using Full GP to Estimate the True f

all of the draws of the latent function f.

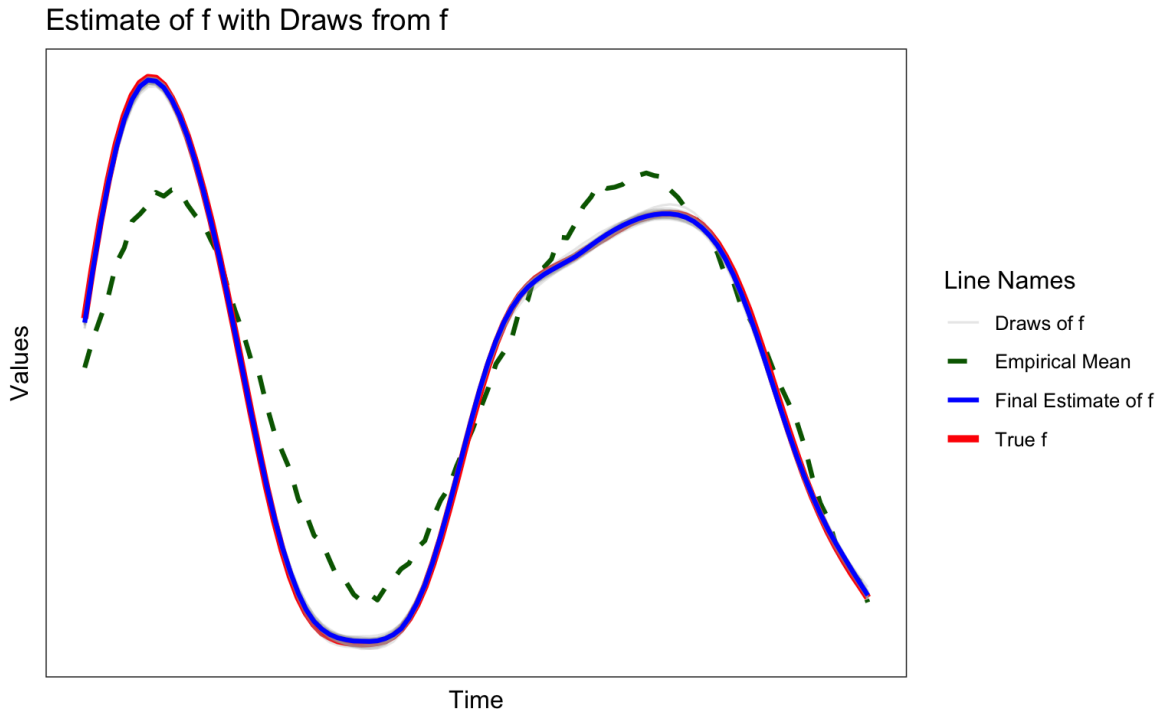

Figure 9: Th Plot of All Draws Using Full GP to Estimate the True f

Next part illustrates the performance of the projection methods in terms of the computational cost and the accuracy(in terms of MSE). Overall,the MSEs get exponentially closer to the full GP MSE with significantly lower computational cost. Now use the Projection Method by choosing $m = 5$ knots. Notice that the computational time is only 0.054 seconds, yet there exists a significant inaccuracy between the predicted f and the actual f, both demonstrated by the graph as well as the MSE, which is equal to 0.2600085.

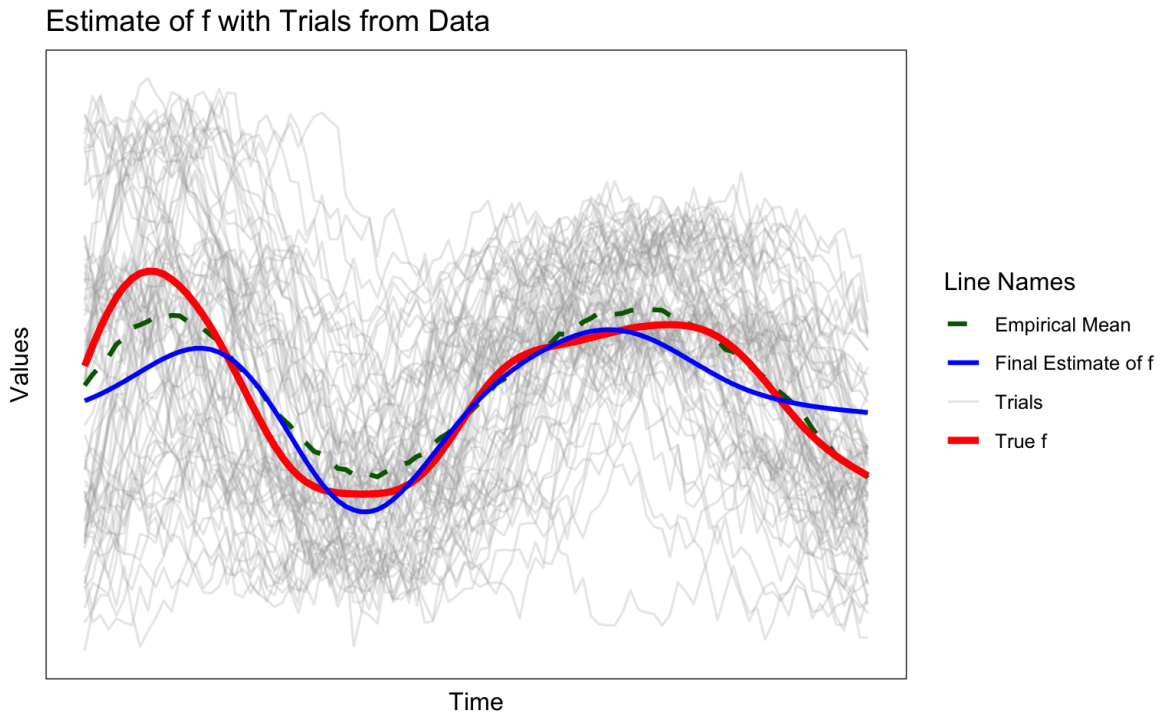

Figure 10: The Trial Plot of Using m = 5 Projection to Estimate the True f

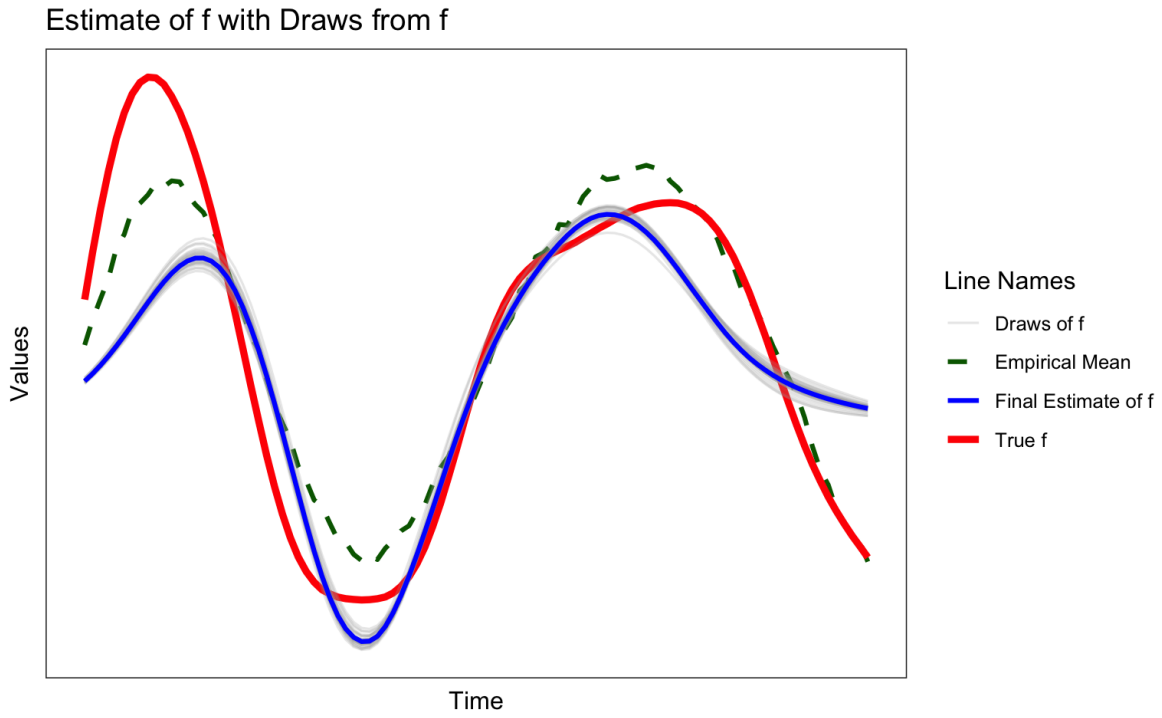

Figure 11: The Plot of All Draws Using m = 5 Projection to Estimate the True f

With m = 10, the prediction is more accurate. The MSE has decreased to 0.01420663 with a computational cost as low as 0.079 seconds.

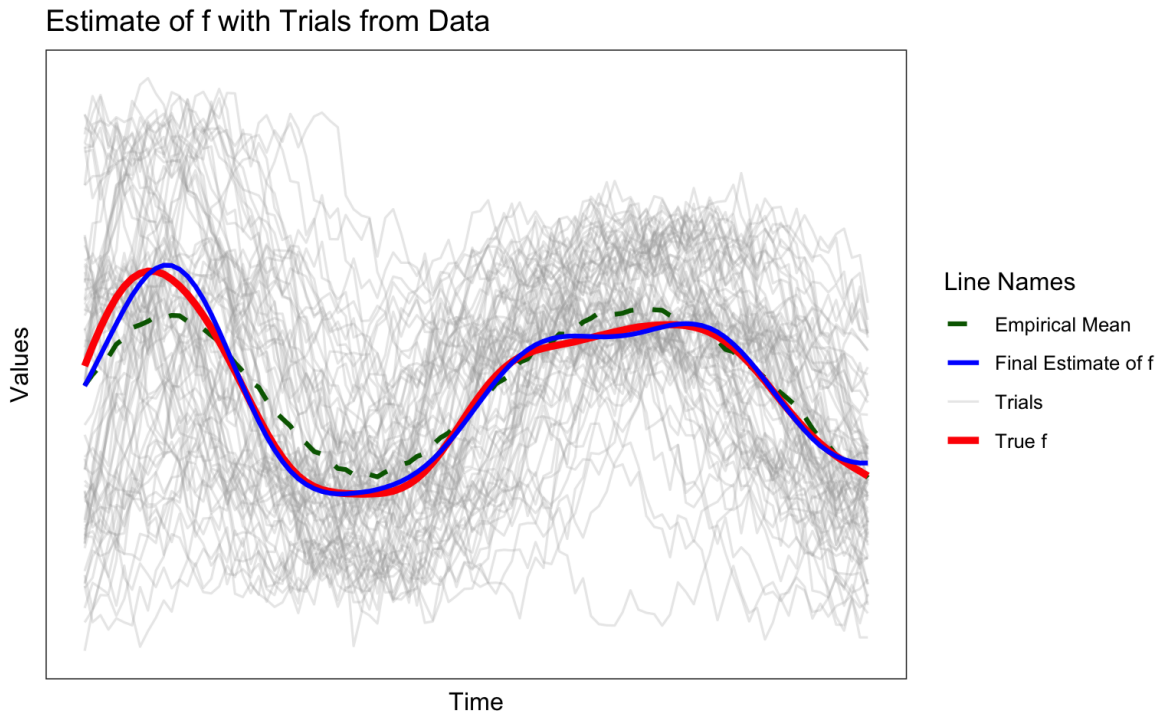

Figure 12: The Trial Plot of Using m = 10 Projection to Estimate the True f

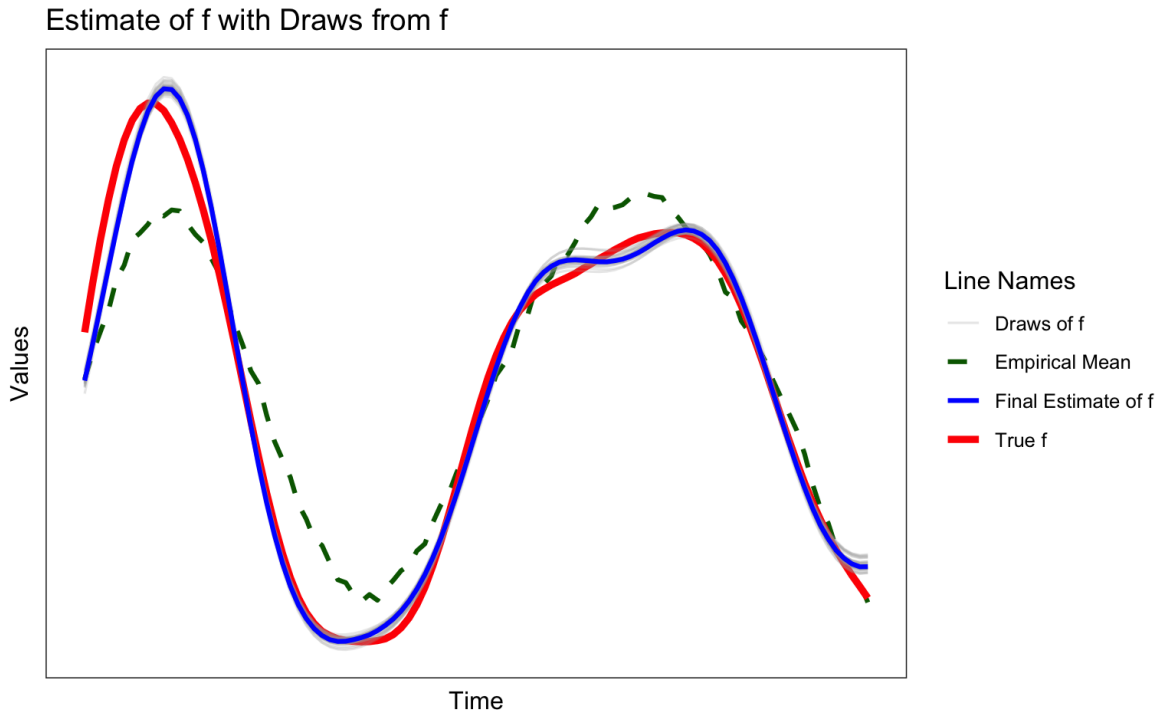

Figure 13: The Plot of All Draws Using m = 10 Projection to Estimate the True f

With m = 15, the prediction is much more accurate. The MSE has sharply decreased to 0.001779301 with a computational cost as low as 0.099 seconds.

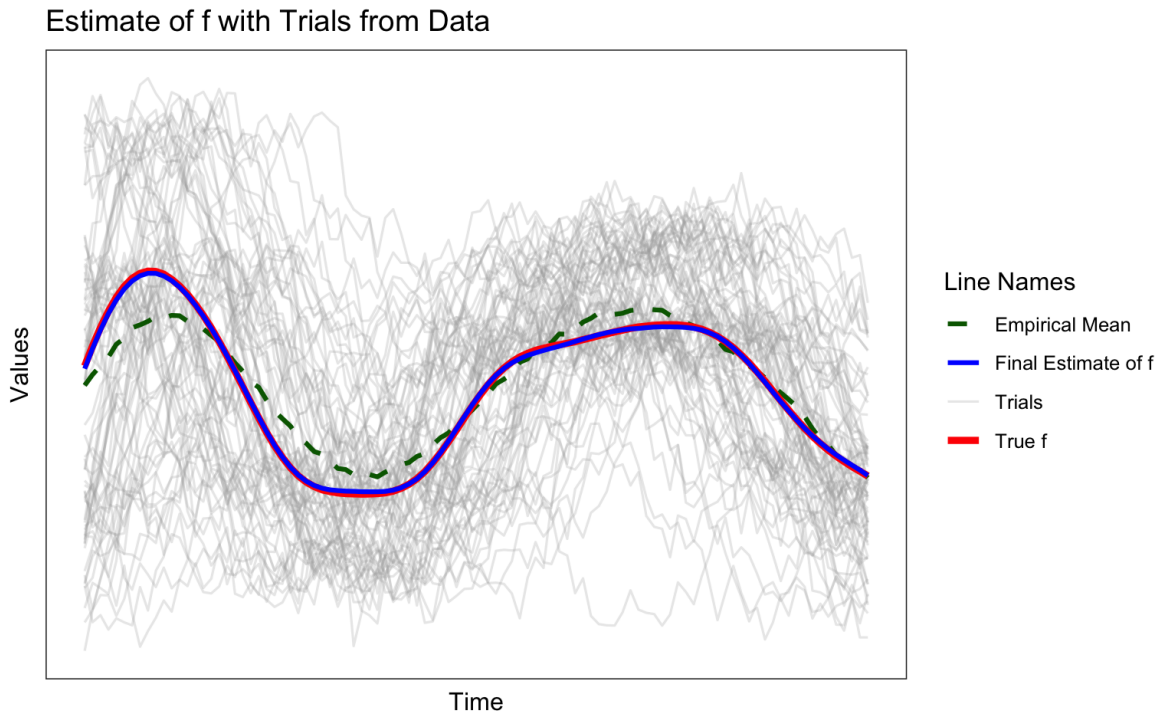

Figure 14: The Trial Plot of Using m = 15 Projection to Estimate the True f

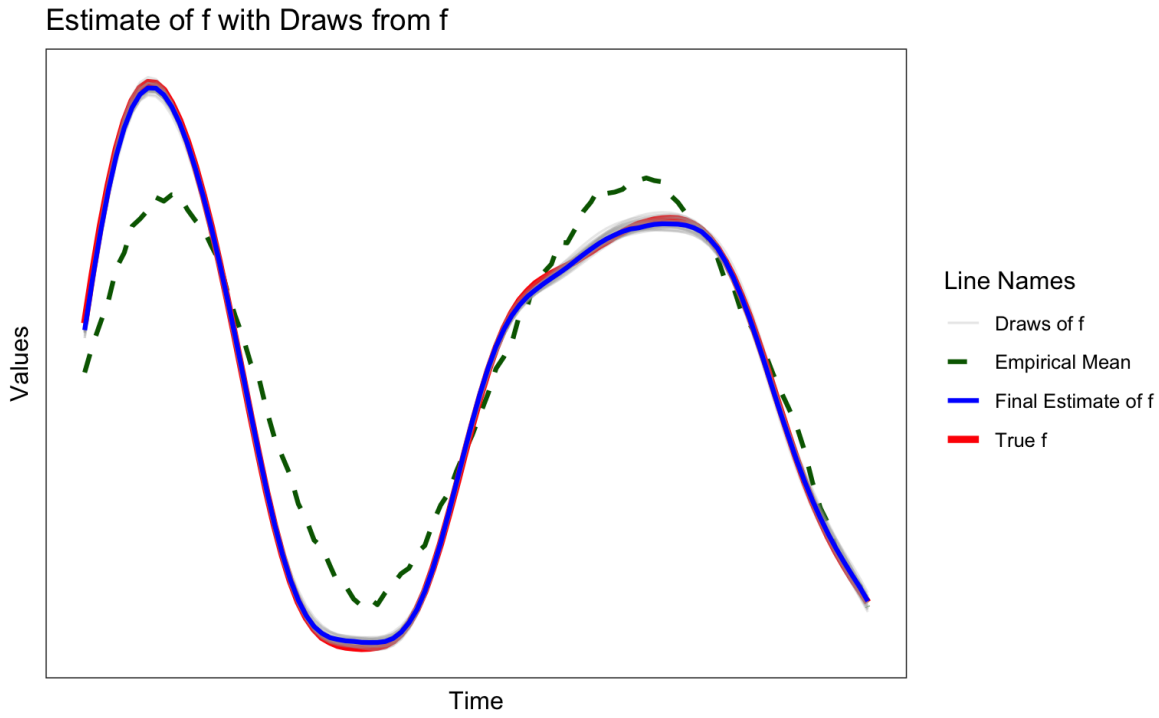

Figure 15: The Plot of All Draws Using m = 15 Projection to Estimate the True f

Last, with m = 25, the prediction is so accurate that it almost resembles the full GP predictions with a MSE of 0.0003586436 and a computational cost of 0.196 seconds.

So far we can see that the computational cost by using the projection method is much lower than that of the full GP. For example, the m = 25 projection method has a compuational cost of only 0.196 seconds, compared to the 6.338 seconds. This is 32 times more efficient. If we choose m = 15, then the cost is roughly 64 times more efficient.

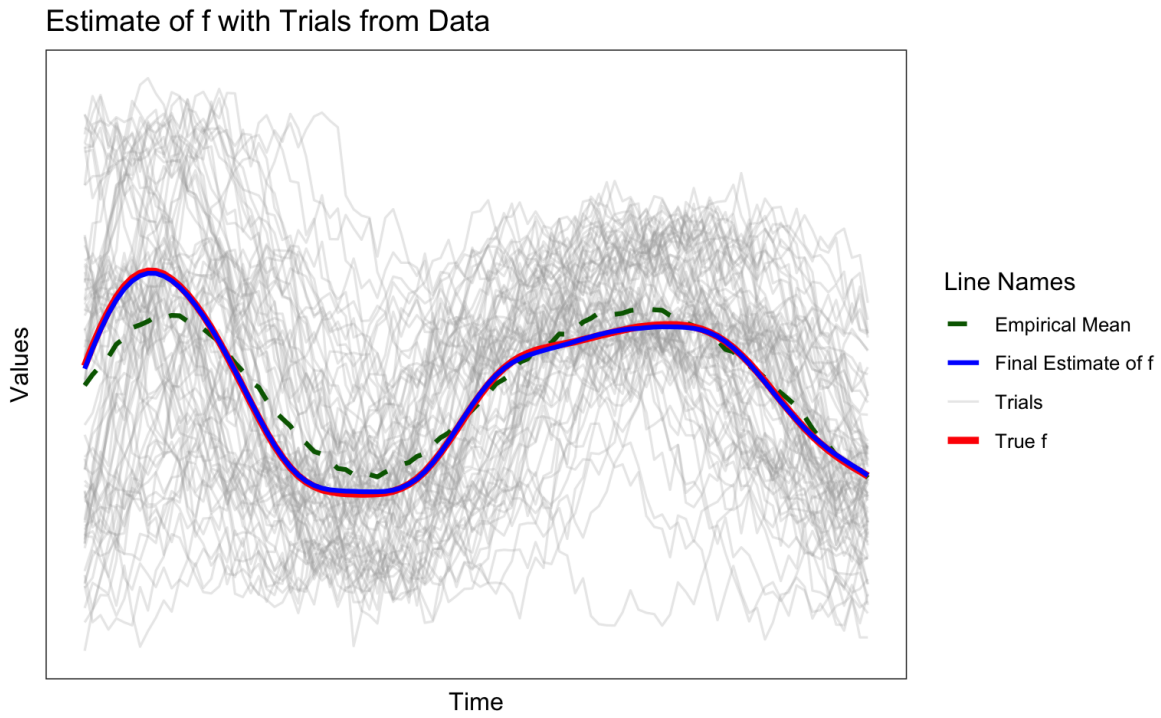

Figure 16: The Trial Plot of Using m = 25 Projection to Estimate the True f

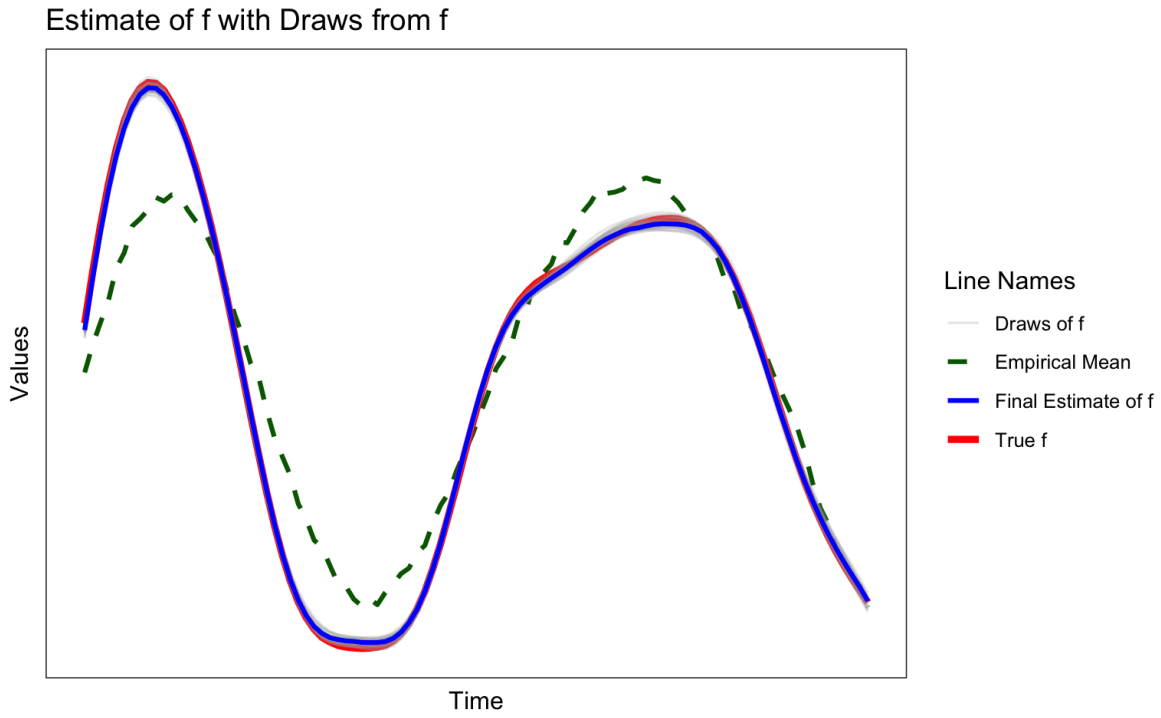

Figure 17: The Plot of All Draws Using m = 25 Projection to Estimate the True f

Below is a graph of the MSE with different m values to be chosen to see trend of the MSEs.

As we can see from here, the MSE with m ¿20 gets very close to the MSE created by the full GP method.

7.2 Codes Result

The complete source code is available at GPAGP repository.
The direct link is
https://github.com/el1997-TREY/RPAGP-Method.

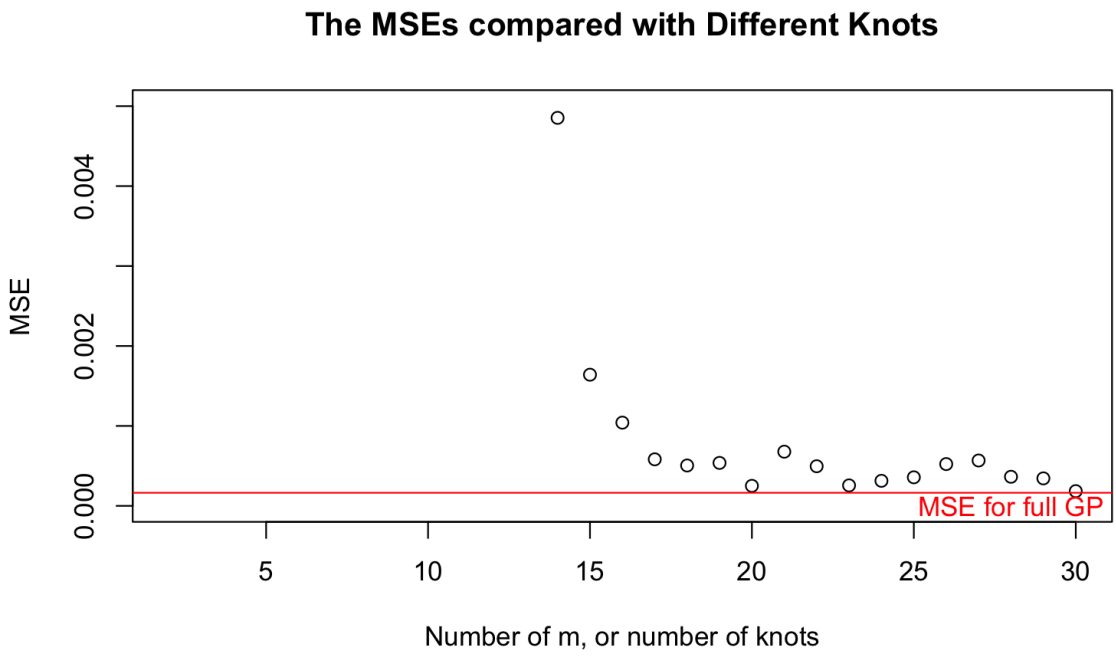

Figure 18: Plots of MSEs with Different m Values

References

[1] D. Pluta, B. Hadj-Amar, M. Li, Y. Zhao, F. Versace, and M. Vannucci. Improved data quality and statistical power of trial-level event-related potentials with Bayesian random-shift Gaussian processes. *Scientific Reports*, 14(8856), 2024. https://doi.org/10.1038/s41598-024-59579-2.

[2] M. Loper and S. Greydanus. Visual exploration of Gaussian processes. *Distill*, 2019. https://distill.pub/2019/visual-exploration-gaussian-processes/.

[3] T. Savitsky, M. Vannucci, and N. Sha. Variable selection for nonparametric Gaussian process priors: Models and computational strategies. *Statistical Science*, 26(1):130–149, 2011. https://doi.org/10.1214/11-STS354.

[4] A. Datta, S. Banerjee, A. O. Finley, and A. E. Gelfand. On nearest-neighbor Gaussian process models for massive spatial data. *WIREs Computational Statistics*, 8(5):162–171, 2016. https://doi.org/10.1002/wics.1383.