Master 2 in Mathematics and Applications, Data Science:
health formation initiale
Academic Year 2023-2024

Internship Report

Metamodeling for Variable Selection in High-Dimensional Complex Mixed Effects Models. (Application in Plant Breeding)

Student:	Viviana Gavilanes
Email:	viviana.gavilanes.g@gmail.com
Internship Period:	April-September 2024
Main Supervisor:	Maud Delattre (MaIAGE, INRAE Jouy-en-Josas).
Email:	maud.delattre@inrae.fr
Co-Supervisor:	Marion Naveau (MaIAGE, INRAE Jouy-en-Josas).
Email:	marion.naveau@inrae.fr

September 2024

Acknowledgment

Although the entire work is written in English, I feel it is important to write the Acknowledgments in French. This is my way of expressing gratitude to all the people who have supported me during my master's journey, especially those who have been directly involved in my academic and personal growth while in France.

Je souhaite exprimer ma sincère gratitude à mes professeurs de l'Université d'Evry Paris-Saclay. Pendant mon master, j'ai rencontré des professeurs formidables, toujours prêts à répondre à mes questions de manière aimable et claire. Je tiens à remercier particulièrement Marie Luce Taupin, qui a été la responsable du master M1 MINT et co-responsable du master M2 DS dans le cadre de mes études. Son soutien inestimable m'a permis de prendre les bonnes décisions pour mon parcours académique. Je la remercie profondément pour sa patience et ses conseils lors de moments difficiles. Un grand merci aussi à Christophe Ambroise, mon tuteur de stage en M1, pour son attention constante envers ses étudiants et son humanité.

Durant mon stage de M2, j'ai eu la chance de travailler à l'INRAE, un centre de recherche que j'ai découvert en M1 grâce à la gestion de Madame Taupin et Estelle Kuhn, une grande chercheuse qui se montre être une excellente leader dans son équipe DynEnvie. J'ai réalisé mes pratiques à l'Unité Mathématique et informatique appliquées du génome à l'environnement (MaIAGE - UR INRAE), sous la supervision de Maud Delattre et Marion Naveau. Je pense que je n'aurais pas pu choisir une meilleure expérience de stage. J'ai pu m'enrichir des caractéristiques propres aux chercheurs, que je suis en train d'acquérir. Je les remercie énormément pour la patience qu'elles ont eue avec moi et pour les révisions minutieuses de mon travail écrit. Je remercie tout particulièrement Maud d'avoir été toujours à l'écoute. Ses réponses rigoureuses m'ont aidée à améliorer le développement de mon stage, qui n'a pas été facile. Je la remercie pour sa gentillesse constante et ses remarques précieuses qui m'aideront à progresser dans ma vie académique et en tant que chercheuse. À Marion, merci d'avoir toujours été disponible pour moi, de m'avoir expliqué sans réserve tout ce dont j'avais besoin. Je ne pouvais pas rêver de meilleures tutrices! Je remercie également toute l'équipe de MaIAGE pour leur accueil chaleureux, même si j'ai parfois été réservée à cause de la barrière linguistique. Je m'efforce de m'améliorer et d'échanger plus librement avec vous bientôt. Merci pour tous les moments partagés.

Je suis heureuse de faire partie d'Amarun, une association qui vise à promouvoir les sciences exactes en Équateur. Un grand merci à son président, Diego Chamorro, pour la gestion de la bourse Sophie Germain de la Fondation Mathématique Jacques Hadamard (FMJH), sans laquelle ce master n'aurait pas été possible. *Ser miembro de Amarun es luchar contra marea.*

Je remercie de tout cœur ma famille, qui m'a toujours soutenue malgré la distance. Merci pour vos prières et vos bénédictions. Un remerciement spécial à ma mère Katy, mon père Gino, ma grand-mère Susana et mon frère Esteban. *Gracias familia, los amo mucho.*

Un salut à mes amis équatoriens qui ont pris la *loca* décision de partir étudier un postgrade à l'étranger, bon courage, *¡Vamos, sí se puede!*

Un remerciement spécial à Josué, *mi compañero de aventuras*, durant ces deux années, et j'espère pour encore bien d'autres. Merci pour tout.

Viviana Gavilanes Guerrero

Jouy-en-Josas, le 18 septembre 2024

Contents

Introduction

My internship was held from April 22, 2024, to September 20, 2024, at the "Institut National de Recherche pour l'Agriculture, l'Alimentation et l'Environnement (INRAE)". This internship follows the recent work of [Naveau et al., 2024], who developed the Stochastic Approximation Expectation Maximization for Variable Selection (SAEMVS) by coupling the SAEM algorithm with a Bayesian spike-and-slab prior for covariate selection in nonlinear mixed-effects models.

Given the profound challenges that climate change will impose on agricultural systems, informed decisions about plant breeding and crop management become crucial. Climate change significantly alters how plants interact with their environment, making the identification and selection of resilient plant varieties essential. The SAEMVS approach, developed in response to these challenges, utilizes statistical methods that accurately capture plant-environment interactions. This allows for the optimization of breeding strategies to select varieties that not only thrive in changing climates but also contribute to reducing agriculture's environmental footprint [Reynolds et al., 2020]. By employing the SAEM algorithm coupled with a Bayesian spike-and-slab prior, this method enhances our ability to make precise selections in the increasingly complex scenario of global climate change.

Given the profound challenges that climate change will impose on agricultural systems, it is crucial to make informed decisions about plant breeding and crop management. Recognizing how climate change affects agriculture is essential; we need to identify and select plant varieties that are resilient to these changes. This can be done effectively with statistical methods that accurately capture plant-environment interactions, allowing us to optimize breeding strategies.

The selection of crop varieties that perform well in agricultural settings depends on the use of complex nonlinear models, which are computationally intensive to evaluate. Estimating these models requires multiple iterations, where the nonlinear functions must be repeatedly assessed, leading to significant computational demands. My internship aimed to tackle this issue by applying metamodeling techniques to reduce computation time and improve the efficiency of the SAEMVS procedure, an iterative method [Naveau et al., 2024]. This approach is not based on new inventions, but rather on the application of established methods and insights gathered from various scientific articles and references. These metamodeling techniques have already proven effective in similar models, as demonstrated in the study by [Barbillon et al., 2021] on the parametric estimation of complex mixed models using a meta-model approach.

During this period, I had the privilege of being supervised by Maud Delattre, a dedicated research scientist at INRAE within the MaIAGE unit. Her expertise in statistical methodologies, particularly in Markovian and latent variable models, with applications in life sciences, provided invaluable guidance throughout my internship. I was also fortunate to be co-supervised by Marion Naveau, a highly skilled and motivated PhD candidate working under the supervision of Maud Delattre and Laure Sansonnet (MCF, AgroParisTech). Her research, which focuses on high-dimensional variable selection in nonlinear mixed-effects models applied to plant breeding, is a key part of the ANR project Stat4Plant [2021].

The first section of this report introduces the host institution, the biological and statistical context, and the objectives of the internship. The second section covers the EM algorithm and its extensions, such as SAEM and MCMC-SAEM, along with relevant stochastic processes, focusing on Gaussian processes. It also includes metamodeling techniques like Kriging and the

use of Latin Hypercube Sampling (LHS) for generating training data. The third section of the report focuses on the coupling of the SAEMVS method with Kriging. It begins with an overview of the SAEMVS method. The section then explains how Kriging is integrated into the mixed-effects model framework. By coupling SAEMVS with Kriging, the methodology aims to significantly reduce computational costs while preserving the accuracy of variable selection. The fourth section presents a simulation study validating this approach, exploring different Kriging configurations and comparing their performance. The report concludes with the results from the simulation study and perspectives.

1 Internship Framework

1.1 The Host Organization: INRAE Research Center at Jouy-en-Josas

INRAE, established in January 2020 through the merger of INRA (*Institut National de la Recherche Agronomique*) and IRSTEA (*Institut national de Recherche en Sciences et Technologies pour l'Environnement et l'Agriculture*), applies interdisciplinary analysis to advance scientific research and innovation across a wide range of fields, including climate change, agroecology, and biodiversity.

INRAE is a public scientific and technological institution, under the joint supervision of the Ministry of Higher Education, Research and Innovation and the Ministry of Agriculture and Food. The institute comprises 18 research centers. During my internship, I was based at the Jouy-en-Josas center, chaired by Nathalie Touze.

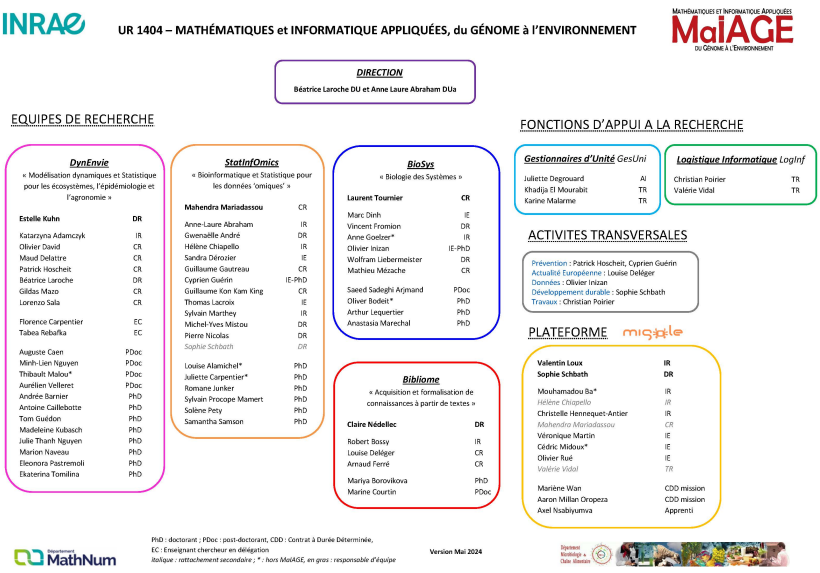

Figure 1: Organizational Chart of the MaIAGE Unit, INRAE Jouy-en-Josas

The MaIAGE research unit, directed by Béatrice Laroche, brings together experts in mathematics, computer science, bioinformatics, and biology to deal with questions related to biology and agroecology. The unit is divided into five teams, as shown in the organizational chart (Figure 1). I was part of the Dynenvie team, led by Estelle Kuhn, which focuses on dynamic and statistical modeling for ecosystems, epidemiology, and agronomy.

1.2 The ANR Stat4Plant Project

My internship was funded by the *Agence Nationale de la Recherche* (ANR) as part of the Stat4Plant [2021] project, led by Estelle Kuhn, a research director at the MaIAGE unit. This project aims to develop new statistical methods for characterizing plant-environment interactions in the context of climate change. Running from 2021 to 2025, the project brings together researchers in modeling and applied statistics, known for their interdisciplinary collaborations, and biologists specializing in genotype-phenotype relationships, from units such as MaIAGE, MIA Paris-Saclay, INRAE GQE (Quantitative Genetics and Evolution) Le Moulon, Centrale-Supelec MICS (Mathematics and Informatics for Complexity and Systems), UTC Heudiasyc (University of Technology of Compiègne, Heuristics and Diagnostics of Complex Systems), and IJPB (Institut Jean-Pierre Bourgin).

The Stat4Plant [2021] project is structured around four major research axes that bring together statisticians and biologists. My internship was specifically aligned with the third axis, which is dedicated to developing methods for high-dimensional variable selection in plant-related applications. This axis focuses on identifying the most influential genetic covariates from large datasets, particularly those that affect phenotypic traits either directly or in connection with critical agricultural events, such as flowering, fruiting, or harvest timing. To improve computational efficiency, metamodeling is introduced as an interesting and promising approach. My work specifically contributes to this effort, aiming to enhance the performance and practical applicability of these variable selection methods in agricultural scenarios.

1.3 Biological Context

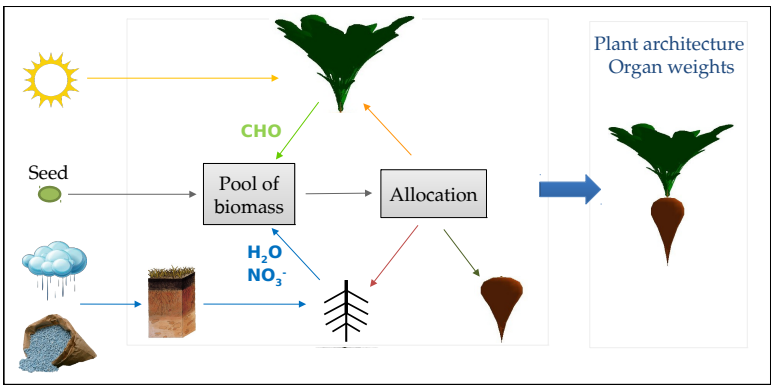

Figure 2: Stages from seed to mature plant and the allocation of resources necessary for growth (Image courtesy of Estelle Kuhn)

In plant breeding, one of the main goals is to predict plant development accurately based on genotype and environmental conditions. Figure 2 illustrates a conceptual model of plant development, showing how resources like water (H2O), nitrates (NO3-), and carbohydrates (CHO) are allocated throughout the plant's life cycle. Starting from the seed stage, these resources build biomass, shaping the plant's architecture and organ weights. The model highlights the complexity of plant development, where each stage of growth involves detailed biological processes (see Figure 2. This complexity underscores the challenges in accurately simulating and predicting plant growth, as each biological step contributes to the overall development process.

In this context, it is important to discuss crop models, which are recognized for their inherent complexity and the significant computational effort required to simulate detailed stages of plant growth. These models simulate various biological processes, such as phenological development, photosynthesis, and the allocation of dry matter, which depend on both genetic factors and environmental conditions. For instance, in the study conducted within the framework of the Stat4Plant project, a crop model that predicted the flowering times of 198 varieties of winter wheat across 10 different environments was employed. This model, which integrates genetic markers and environmental data, incorporates five sub-models that represent distinct physiological processes like stem and leaf production, root development, light interception, and photosynthesis [Weir et al., 1984]. However, due to its detailed nature, evaluating this crop model is much slower compared to simpler models, such as logistic growth models.

A single iteration of this crop model can take approximately five minutes, while a logistic model would complete the same iteration in just 1 second. When applying the SAEMVS algorithm, which requires hundreds of iterations and several model evaluations per iteration, the total computational time can extend beyond 24 hours for one set of simulations. In contrast, using a logistic model would complete the same task in a matter of minutes. This high computational demand becomes even more challenging when large-scale analyses, like parameter estimation or sensitivity analysis, require thousands of simulations [Brun et al., 2019].

Moreover, the high dimensionality of genetic data in plant breeding, which often includes thousands of covariates such as genetic markers or single nucleotide polymorphisms (SNPs), needs the use of variable selection techniques. These covariates are represented as a large number of variables in the model, creating a high-dimensional data scenario. Identifying the most influential genetic factors becomes essential to enhance the model's predictive accuracy. The challenge intensifies when the number of covariates exceeds the number of samples, making effective variable selection both critical and complex. This necessity for precise selection further motivates the development and optimization of computational methods in plant breeding research.

1.4 Internship Objectives

My internship work focuses on integrating metamodeling techniques into the process of variable selection in nonlinear mixed-effects models within a high-dimensional setting, specifically using the SAEMVS method. The challenge here lies in the significant computational cost, primarily due to the iterative nature of the SAEM algorithm, which necessitates multiple evaluations of a complex and computationally expensive crop model at each step. This is further complicated by the high dimensionality of genetic data.

The paper by [Barbillon et al., 2021] specifically addressed similar computational challenges, focusing on parametric inference within complex mixed models. They proposed metamodeling techniques to approximate the costly regression functions involved, significantly reducing the computational demands. However, their approach was developed with a focus on parametric estimation, not variable selection.

A recent study by [Naveau et al., 2024] has made a significant contribution in the area of variable selection by combining a spike-and-slab prior with the SAEM algorithm. Their approach offers a more computationally efficient alternative to traditional MCMC methods, showing strong

performance in both simulated and real datasets, particularly for identifying genetic markers in plant breeding. Building on this foundation, my work aims to extend these metamodeling techniques to the domain of high-dimensional variable selection within nonlinear mixed-effects models. This involves addressing the computational challenges posed by the SAEMVS method when applied to complex, high-dimensional datasets.

The objective of this internship, proposed by Maud Delattre and Marion Naveau, is to explore the use of metamodeling to improve the computational efficiency of the SAEMVS methodology. Building on the foundations of Naveau's research on Bayesian variable selection in high-dimensional nonlinear mixed-effects models, the focus is on developing a coupling between the Kriging method and the SAEMVS approach.

After studying the relevant literature, I implemented this new coupling in `R` and validated it through an intensive simulation study. In doing so, I utilized existing resources and methodologies from papers on Kriging, specifically incorporating tools such as `DiceKriging`, `DiceEval`, and `DiceOptim` to support the development and evaluation of the metamodeling approach.

2 Statistical Prerequisites

2.1 Nonlinear Mixed Effects Models (NLME)

Mixed-effects models are studied to combine fixed and random effects in the analysis of repeated observations on individuals under various conditions, such as temporal or environmental variations [Lavielle, 2014, Pinheiro and Bates, 2000]. These models are particularly relevant when it is essential to describe both intra- and inter-individual variation under different conditions.

For example, plant growth variations are monitored in response to various environmental factors, such as water, sunlight, and temperature. These data allow us to model both the variability specific to each individual and the variability between individuals. The goal is to model plant growth under these conditions using a two-level model:

- **Global level**: A common global growth trend is modeled by a function g, which is consistent across all individuals.
- **Individual level**: The parameters that vary between individuals are denoted by φ_i and are influenced by both fixed effects (common across all individuals) and random effects specific to each individual.

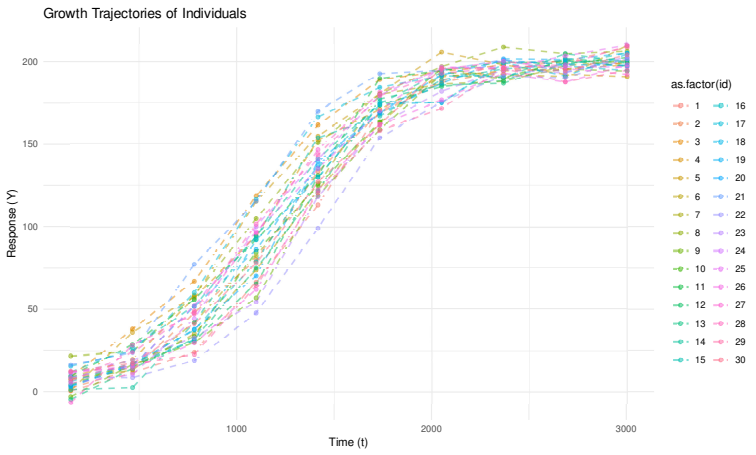

Figure 3: Growth trajectories of individuals over time

Figure 3 illustrates the growth trajectories of multiple individuals over time, revealing a consistent overall trend where most individuals experience rapid initial growth that slows and plateaus as time progresses. However, there is notable inter-individual variability, with some individuals growing significantly faster than others, indicating differences in growth rates likely due to genetic, environmental, or other factors. This variability underscores the complexity of growth dynamics within the population, even as a common growth pattern is observed.

Statistical Model A non-linear mixed-effects model is presented as follows, composed of two levels defined by Equations (1) and (2). This model accounts for intra-individual variability through the function g, which depends on the specific profile of the crop model under study [Lavielle, 2014].

Let n denote the number of individuals, and J the number of observations per individual. For simplicity, it is assumed that J is constant across all individuals $i \in \{1, \ldots, n\}$. For each $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, J\}$, the response of individual i under condition x_{ij}, denoted by y_{ij}, is given by:

$$y_{ij} = g(\varphi_i, \psi, x_{ij}) + \epsilon_{ij}, \quad \epsilon_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2), \tag{1}$$

$$\varphi_i = \mu + \beta^\top V_i + \xi_i, \qquad \xi_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}_q(0, \Gamma). \tag{2}$$

In this model:

- $\psi \in \mathbb{R}^r$ denotes the unknown fixed effects parameters common across the population (in the following simulation case $r = 2$).
- $\varphi_i \in \mathbb{R}^q$ represents the unobserved individual-specific parameters.
- g is a known function governing intra-individual behavior based on the conditions observed, depending nonlinearly on the unobserved individual parameter φ_i and the fixed effects ψ.
- $y_{ij} \in \mathbb{R}$ represents the response of individual i under condition x_{ij}.
- $\sigma^2 > 0$ is the unknown residual variance in the observations.
- μ is the overall intercept.
- β is a matrix of fixed effects related to covariates V_i.

- ξ_i is a random variable following a multivariate normal distribution, $\mathcal{N}_q(0, \Gamma)$, accounting for differences between individuals that are not explained by the observed covariates V_i.
- $V_i = (V_{i1}, \ldots, V_{ip})^T$ represents the p covariates associated with individual i.
- Γ is the covariance matrix characterizing the variability of φ_i across the population.

Two Levels of Variability

Intra-Individual Variability: This is captured by the error term ϵ_{ij} in equation (1), representing random noise around each individual's response predictions.

Inter-Individual Variability: Equation (2) captures inter-individual variability. This variability is represented as the sum of two terms: a linear combination of covariates $\beta^\top V_i$, which describes differences between individuals, and a random effect ξ_i, which accounts for unmeasured characteristics not explained by the covariates [Lavielle, 2014, Pinheiro and Bates, 2000].

These two levels together provide a comprehensive description of both intra- and inter-individual variability.

The population parameters are denoted as $\theta = (\mu, \beta, \psi, \sigma^2, \Gamma)$, while $\varphi = (\varphi_i)_{1 \leq i \leq n}$ represents latent random variables, which are not directly observed. In this model, the individual-specific parameters φ_i serve as such latent variables.

In this internship, we are particularly interested in scenarios where the number of covariates p is significantly larger than the number of observations n. This high-dimensional context ($p \gg n$) makes the problem computationally challenging.

2.2 EM Algorithm and Its Extensions

In this context, where φ_i are latent variables for all $i \in \{1, \ldots, n\}$, as is common in most latent variable models, the inference process is typically indirect and not explicit. Consequently, specialized algorithms are required to perform this inference effectively. This necessity leads us to the discussion of the EM algorithm and its extensions.

2.2.1 EM Algorithm

The Expectation-Maximization (EM) algorithm was originally introduced by [Dempster et al., 1977] to overcome difficulties in maximizing likelihoods in the presence of latent variables. The intuition behind the EM algorithm can be illustrated with the "chicken-and-egg" analogy:

1. If we knew the latent variables φ, we could easily compute the complete-data log-likelihood $\log p_\theta(y, \varphi)$.
2. If we knew the parameters θ, we could estimate the latent variables φ through their posterior distribution $p_\theta(\varphi \mid y)$.

However, since the latent variables φ are not observed, we aim to maximize the observed-data likelihood $p_\theta(y) = \int p_\theta(y, \varphi) d\varphi$. The goal is to find the maximum likelihood estimate:

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \log p_\theta(y).$$

The EM algorithm constructs a sequence $\{\theta^{(k)}\}$ converging to a local maximum of the likelihood function:

1. **Initialization**: Choose an initial value $\theta^{(0)} \in \Theta$.
2. **Iteration (for $k \geq 0$)**:
   (a) **E-step:** Compute the expected value of the complete-data log-likelihood with respect to the current estimate of the parameters:

$$Q(\theta \mid \theta^{(k)}) = \mathbb{E}_{\varphi|y,\theta^{(k)}}[\log p_\theta(y, \varphi)].$$

   (b) **M-step:** Update the parameter estimates by maximizing $Q(\theta \mid \theta^{(k)})$:

$$\theta^{(k+1)} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, Q(\theta \mid \theta^{(k)}).$$

Each iteration of the EM algorithm increases the observed-data log-likelihood $\log p_\theta(y)$, and the algorithm converges to a local maximum under mild regularity conditions (Section A.1) [Wu, 1983]. It is important to note that the convergence point may depend on the choice of the initial value $\theta^{(0)}$, and different initializations may lead to different local maxima. Therefore, careful consideration of the initialization strategy is crucial in practice [McLachlan and Krishnan, 2007]. We can also rewrite the likelihood to work with the ELBO dimension as can be seen in Section A.3.

Convergence theorems of the EM algorithm The convergence of the EM algorithm is supported by several theoretical results, which are detailed in Annex A.2.

2.2.2 SAEM Algorithm (Stochastic Approximation EM)

The Stochastic Approximation EM (SAEM) algorithm was introduced by [Delyon et al., 1999] as an extension of the EM algorithm to handle situations where the E-step cannot be computed explicitly. This is often the case in nonlinear mixed-effects models, as described in Section 2.1. The SAEM algorithm achieves this by incorporating simulation and stochastic approximation methods.

Objective: To compute $Q(\theta \mid \theta^{(k)})$ by approximating the conditional expectation of the complete-data log-likelihood.

The SAEM algorithm replaces the deterministic E-step of the standard EM algorithm with a stochastic approximation:

1. **S-step (Simulation):** Generate a realization $\varphi^{(k)}$ of the latent variables from the conditional distribution $p_{\theta^{(k)}}(\varphi \mid y)$.
2. **SA-step (Stochastic Approximation):** Update the approximation of $Q(\theta \mid \theta^{(k)})$ using the stochastic update rule:

$$Q_{k+1}(\theta) = Q_k(\theta) + \gamma_k \left(\log p_\theta(y, \varphi^{(k)}) - Q_k(\theta) \right),$$

   where γ_k is a step size sequence satisfying $\sum_k \gamma_k = \infty$ and $\sum_k \gamma_k^2 < \infty$. The sequence $(\gamma_k)_k$ is typically decreasing, converging to 0, and starts at 1.

3. **M-step:** Maximize the updated $Q_{k+1}(\theta)$ with respect to θ:

$$\theta^{(k+1)} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, Q_{k+1}(\theta).$$

The choice of this step size sequence $\{\gamma_k\}$ is crucial for the convergence of the SAEM algorithm. The conditions $\sum_k \gamma_k = \infty$ and $\sum_k \gamma_k^2 < \infty$ ensure that the algorithm explores the parameter space sufficiently while the diminishing step size reduces the impact of the stochastic noise over time.

A common practical choice is to set:

$$\gamma_k = \begin{cases} 1 & \text{if } k \leq k_0, \\ (k - k_0)^{-\alpha} & \text{if } k > k_0, \end{cases}$$

where k_0 is the number of iterations in the initial exploration phase (burn-in), and $\alpha \in (0.5, 1]$ controls the rate at which the step size decreases [Lavielle, 2014].

The SAEM algorithm iteratively updates the parameter estimates by combining stochastic simulations of the latent variables with stochastic approximations of the expected complete-data log-likelihood. This approach allows for efficient parameter estimation even when the exact E-step is computationally infeasible.

Convergence of SAEM The convergence properties of the SAEM algorithm have been studied extensively. Under suitable regularity conditions, the sequence $\{\theta^{(k)}\}$ generated by the SAEM algorithm converges almost surely to a critical point of the observed-data log-likelihood function. Specifically, [Delyon et al., 1999] proved the theorem cited in Annex A.4.

2.2.3 Simplified Version with Exponential Family

To improve computational efficiency, we consider models within the exponential family, which enables stochastic approximation to be performed on these statistics rather than directly on the likelihood [Delyon et al., 1999]. In subsequent sections, we will discuss the selection of priors for the SAEMVS algorithm, specifically Gaussian spike-and-slab priors. These priors allow us to remain within the exponential family framework, benefiting from its computational advantages. By choosing priors that are conjugate to the likelihood within the exponential family, we can simplify the updates in the SAEM algorithm and ensure that the sufficient statistics remain tractable [Lavielle, 2014].

For models in the exponential family, the joint distribution of the observed data y and the latent variables φ can be written as:

$$p_\theta(y, \varphi) = \exp\left(-\psi(\theta) + \langle S(y, \varphi), \phi(\theta) \rangle\right),$$

where $S(y, \varphi)$ are the sufficient statistics, $\phi(\theta)$ are the natural parameters, and $\psi(\theta)$ is the log-partition function.

The complete-data log-likelihood is then:

$$\log p_\theta(y, \varphi) = -\psi(\theta) + \langle S(y, \varphi), \phi(\theta) \rangle .$$

Consequently, the Q function in the EM algorithm becomes:

$$Q\left(\theta \mid \theta^{(k)}\right) = -\psi(\theta) + \left\langle \mathbb{E}_{\theta^{(k)}}\left[S(y, \varphi) \mid y\right], \phi(\theta) \right\rangle .$$

In this context, it suffices to approximate the expected sufficient statistics $\mathbb{E}_{\theta^{(k)}}\left[S(y,\varphi) \mid y\right]$ in order to approximate $Q(\theta \mid \theta^{(k)})$. This simplifies the SAEM algorithm, as the stochastic approximation can focus on the sufficient statistics rather than the entire log-likelihood function.

The convergence of SAEM under the given conditions provides theoretical assurance for its application in complex models where traditional EM algorithms fail due to intractable E-steps. By carefully selecting the step size sequence and ensuring the model satisfies the regularity conditions, we can confidently apply SAEM to obtain reliable parameter estimates.

2.2.4 MCMC-SAEM Algorithm: Coupling with MCMC Procedures

In nonlinear mixed-effects models, where the conditional distribution of the latent variables φ given the observed data y, i.e., $p_\theta(\varphi \mid y)$, is not available in closed form, the S-step of the SAEM algorithm becomes infeasible. To address this issue [Kuhn and Lavielle, 2004] proposed coupling the SAEM algorithm with Markov Chain Monte Carlo (MCMC) methods to simulate the latent variables.

MCMC Method and Metropolis-Hastings Algorithm

Markov Chain Monte Carlo (MCMC) methods construct a Markov chain $(\varphi^{(k)})_{k\geq 0}$ whose stationary distribution is the target distribution $p_\theta(\varphi \mid y)$. One of the most commonly used MCMC algorithms is the Metropolis-Hastings algorithm [Metropolis et al., 1953, Hastings, 1970], which operates as follows:

1. **Proposal Step:** Propose a candidate sample φ^c from a proposal distribution $q(\varphi^c \mid \varphi^{(k-1)})$.
2. **Acceptance Probability:** Compute the acceptance probability:
$$\alpha(\varphi^{(k-1)}, \varphi^c) = \min\left(1, \frac{p_\theta(\varphi^c \mid y)}{p_\theta(\varphi^{(k-1)} \mid y)} \cdot \frac{q(\varphi^{(k-1)} \mid \varphi^c)}{q(\varphi^c \mid \varphi^{(k-1)})}\right).$$
3. **Acceptance Step:** Generate $u \sim \mathcal{U}(0,1)$. If $u \leq \alpha(\varphi^{(k-1)}, \varphi^c)$, accept the candidate and set $\varphi^{(k)} = \varphi^c$; otherwise, reject the candidate and set $\varphi^{(k)} = \varphi^{(k-1)}$.

This method ensures that the Markov chain has $p_\theta(\varphi \mid y)$ as its stationary distribution. Candidates that increase the target density $p_\theta(\varphi \mid y)$ are always accepted, while others are accepted with probability α, maintaining detailed balance and ergodicity Robert and Casella [2004].

By integrating the Metropolis-Hastings algorithm into the SAEM framework, we can approximate the S-step by generating samples from the conditional distribution $p_{\theta^{(k)}}(\varphi \mid y)$ using MCMC methods.

The use of MCMC methods in the S-step allows us to handle complex models where the conditional distribution of φ is intractable, enabling the SAEM algorithm to be applied more broadly.

[Kuhn and Lavielle, 2004] established the convergence of the MCMC-SAEM algorithm under certain regularity conditions. The main idea is that, despite the stochastic nature of the MCMC procedure, the overall algorithm behaves like a stochastic approximation algorithm, the theorem is also cited in Annex A.5.

Practical Considerations In practice, the proposal distribution $q(\varphi^c \mid \varphi^{(k-1)})$ is often chosen to be easy to sample from and to ensure a reasonable acceptance rate. Common choices include:

- **Random Walk Proposal:** $q(\varphi^c \mid \varphi^{(k-1)}) = \mathcal{N}(\varphi^{(k-1)}, \sigma^2 I)$.
- **Independent Proposal:** $q(\varphi^c)$ independent of $\varphi^{(k-1)}$.
- **Another Proposal:** $q(\varphi^c \mid \varphi^{(k-1)}) = \mathcal{N}(\mu^{(k)} + \beta^{(k)} V_i, \Gamma^{(k)})$.

The number of MCMC iterations M at each SAEM iteration can also be adjusted to balance computational cost and convergence speed. A single MCMC iteration per SAEM iteration is often sufficient due to the stochastic approximation nature of the algorithm.

2.3 SAEMVS: Variable Selection in High-Dimensional Nonlinear Mixed-Effects Models

Variable selection in high-dimensional settings, especially within nonlinear mixed-effects models, poses significant challenges due to the presence of latent variables and the complex structure of the data. The SAEMVS method (Stochastic Approximation EM Variable Selection), introduced by [Naveau et al., 2024], addresses these challenges by combining the Stochastic Approximation EM (SAEM) algorithm with a Bayesian spike-and-slab prior for variable selection. This approach extends the EMVS framework proposed by Rockova and George [Ročková and George, 2014] for linear models to the nonlinear mixed-effects context.

Considering the NLME model defined by (1) and (2) for all $1 \leq i \leq n$ and $1 \leq j \leq J$, we assume that the variabilities are independent and identically distributed (i.i.d.). Under this assumption, the model can be expressed as follows:

$$\begin{aligned} y_{ij} &\overset{\text{i.i.d.}}{\sim} \mathcal{N}(g(\varphi_i, \psi, x_{ij}), \sigma^2), \\ \varphi_i &\overset{\text{i.i.d.}}{\sim} \mathcal{N}_q(\mu + \beta^\top V_i, \Gamma). \end{aligned}$$

In such models, identifying the influential covariates that explain the variability between individuals is crucial. This task translates to determining the non-zero elements of the fixed-effects matrix $\boldsymbol{\beta}$. For each $(\ell, m) \in \{1, \ldots, p\} \times \{1, \ldots, q\}$, the coefficient $\beta_{\ell m}$ represents the effect of covariate ℓ on the individual-specific parameter φ_{im} of the nonlinear model g. A zero coefficient $\beta_{\ell m} = 0$ indicates that covariate ℓ has no influence on the m-th individual parameter across all individuals i, while $\beta_{\ell m} \neq 0$ suggests a significant effect.

The objective is to identify the support of $\boldsymbol{\beta}$, defined as

$$S^* = \{(\ell, m) \in \{1, \ldots, p\} \times \{1, \ldots, q\} \mid \beta^*_{\ell m} \neq 0\},$$

where $\boldsymbol{\beta}^*$ denotes the true fixed-effects matrix. In high-dimensional scenarios where $p \gg n$, it is reasonable to assume sparsity in $\boldsymbol{\beta}^*$, meaning that many of the $\beta^*_{\ell m}$ are zero.

2.3.1 Details of the SAEMVS Algorithm

To facilitate variable selection, SAEMVS uses a Bayesian approach. [Naveau et al., 2024] introduces binary latent variables $\boldsymbol{\delta} = (\delta_{\ell,m})_{1 \leq \ell \leq p; 1 \leq m \leq q}$, where $\delta_{\ell m} \in \{0, 1\}$. Each $\delta_{\ell m}$ indicates

whether covariate ℓ indicates that the covariate m provides information on the individual parameter m. This is formalized by setting $\delta = (\delta_{\ell m})_{1\leq \ell \leq p; 1 \leq m \leq q}$ are introduced, such as:

$$\forall 1 \leq \ell \leq p, \forall 1 \leq m \leq q, \delta_{\ell m} = \begin{cases} 1 \text{ if } (\ell, m) \text{ is to be included in model } S^* \\ 0 \text{ otherwise.} \end{cases}$$

The SAEMVS method proceeds as follows:

1. **Introduction of latent variables and priors**: Incorporate the latent variables δ_ℓ and assign spike-and-slab priors on the coefficients β_ℓ [Mitchell and Beauchamp, 1988],which serve as a regularization mechanism through the hyperparameters ν_0 and ν_1.
   Specifically, the spike-and-slab prior for each $\beta_{\ell m}$ given $\delta_{\ell m}$, for all $1 \leq l \leq p$, $1 \leq m \leq q$ is defined as:

   $$\pi(\beta_{\ell m} \mid \delta_{\ell m}) = \mathcal{N}(0, a_{\ell m}), \quad \text{where} \quad a_{\ell m} = (1 - \delta_{\ell m})\nu_0 + \delta_{\ell m}\nu_1,$$

   with $0 < \nu_0 < \nu_1$. Here, ν_0 is small, inducing strong shrinkage towards zero (the "spike"), and ν_1 is larger, allowing for non-zero coefficients (the "slab").
2. **Bayesian estimation via MCMC-SAEM**: Use a Bayesian approach to estimate all parameters θ, including $\boldsymbol{\beta}$ and the latent variables δ and other model parameters, by employing the MCMC-SAEM algorithm to obtain the Maximum A Posteriori (MAP) estimator $\hat{\theta}$.
3. **Variable selection**: Perform variable selection by thresholding the estimated coefficients $\widehat{\beta}^{\text{MAP}}$. Covariates corresponding to non-negligible estimated coefficients are selected, i.e., covariates ℓ for which $|\widehat{\beta}^{\text{MAP}}_{\ell m}| \geq s_\beta$ are included in the final model, where s_β is a threshold function.
4. **Model selection over hyperparameters**: Repeat steps 2 and 3 over a grid of values for the shrinkage hyperparameter ν_0. The optimal value $\hat{\nu}_0$ is chosen based on a model selection criterion such as the extended Bayesian Information Criterion (eBIC).
   Specifically, the optimal ν_0 is selected by minimizing the eBIC:

   $$\hat{\nu}_0 = \underset{\nu_0 \in \Delta}{\text{argmin}} \ \text{eBIC}\left(\widehat{S}_{\nu_0}\right),$$

   where Δ denotes the grid of ν_0 values, $\widehat{S}_{\nu_0}$ is the selected model at ν_0, and

   $$\text{eBIC}\left(\widehat{S}_{\nu_0}\right) = -2 \log p\left(y; \hat{\theta}^{\text{MLE}}_{\nu_0}\right) + \text{pen}_{\text{eBIC}}\left(\nu_0\right),$$

   with the penalty term

   $$\text{pen}_{\text{eBIC}}\left(\nu_0\right) = B_{\nu_0} \log n + 2 \log \binom{p}{B_{\nu_0}},$$

   where B_{ν_0} is the number of non-zero coefficients in the model $\widehat{S}_{\nu_0}$.

The spike-and-slab prior effectively distinguishes between relevant and irrelevant covariates by shrinking negligible coefficients towards zero while allowing significant ones to remain. The figure below illustrates the spike-and-slab distribution:

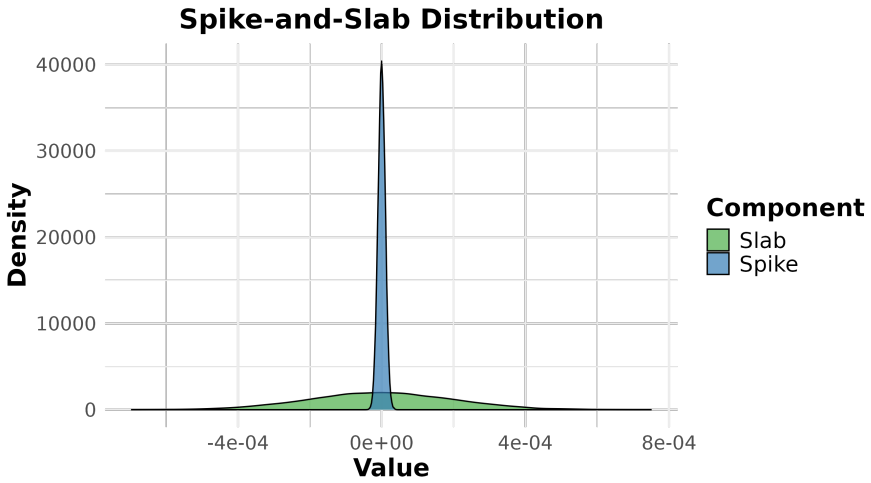

Figure 4: Spike-and-Slab Distribution

In this plot, the "spike" at zero represents the prior belief that many coefficients are exactly zero, while the "slab" allows for coefficients to take on a range of values, capturing the significant effects.

Employing Gaussian priors within the SAEM algorithm allows the model to remain within the exponential family, which offers computational benefits. The exponential family structure facilitates the calculation of sufficient statistics and simplifies the update steps in the SAEM algorithm, enhancing numerical stability a crucial factor in high-dimensional settings [Lavielle, 2014].

2.3.2 Final Variable Selection

After obtaining the MAP estimates $\widehat{\beta}$ for each $\nu_0 \in \Delta$, the final set of selected variables $\widehat{S}_{\nu_0}$ is determined using a thresholding rule based on the estimated coefficients:

$$\widehat{S}_{\nu_0} = \left\{(\ell, m) \in \{1, \ldots, p\} \times \{1, ..., q\} \mid |\widehat{\beta}_{\ell m}^{\text{MAP}}| \geq s_\beta \left(\nu_0, \nu_1, \widehat{\alpha}_m^{\text{MAP}}\right)\right\}, \tag{3}$$

where s_β is a threshold function depending on ν_0, ν_1, and the estimated hyperparameters $\widehat{\alpha}_m^{\text{MAP}}$. This step ensures that only covariates with sufficiently large estimated effects are selected.

The following graphical plot was designed by [Naveau et al., 2024] and it is used to illustrate the regularization process in model selection using the SAEMVS method. It shows how the coefficients of covariates change as the regularization parameter varies, helping to identify the most relevant covariates. Additionally, it illustrates how the eBIC criterion selects the optimal model by balancing model complexity and fit quality.

In Figure (A), the regularization plot visualizes how the estimates of model parameters change with the regularization parameter ν_0. he blue lines represent the values of the components of $\widehat{\beta}^{\text{MAP}}$ associated with relevant covariates, while the black lines correspond to the covariates with null effects. The red lines indicate the selection thresholds for the covariates. For each ν_0, the selected covariates $\widehat{S}_{\nu_0}$ are those for which $(\widehat{\beta}_{\nu_0}^{\text{MAP}})_\ell$ lies outside the two red threshold lines.

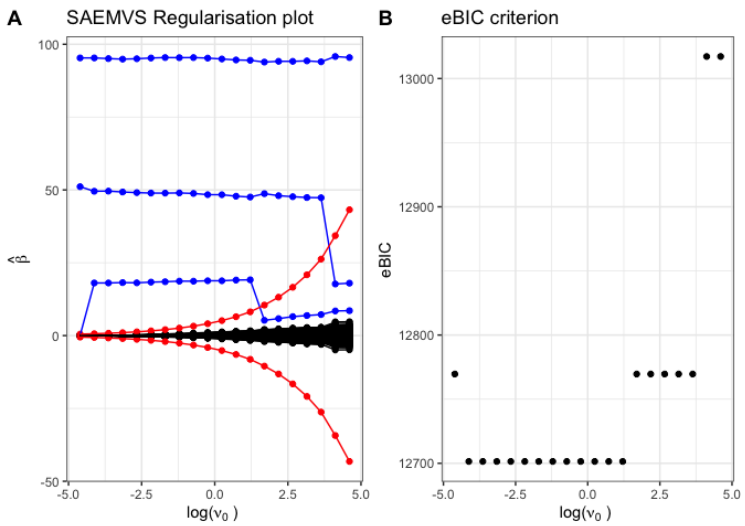

Figure 5: (A) SAEMVS Regularisation plot and (B) eBIC criterion

Figure (B) illustrates the eBIC criterion across all values of ν_0 in Δ. As desired, the eBIC reaches its minimum when the correct model is selected. The procedure selects the second value of $\nu_0 \in \Delta$, which corresponds to $\hat{\nu}_0 \approx 0.016$, and the associated covariates $\widehat{S}_{\hat{\nu}_0} = \{1, 2, 3\}$. In this simulated scenario, SAEMVS successfully identifies the correct model, which includes only the first three covariates.

2.3.3 Computational Considerations

Implementing the SAEMVS algorithm involves substantial computational effort, particularly due to the need to run the SAEM algorithm multiple times over the grid of ν_0 values. For each grid point, the algorithm iteratively evaluates the model g, described below, across all individuals, which can be computationally expensive, especially when g is complex or when the sample size n is large.

Moreover, the choice of the grid Δ for ν_0 is critical. An inadequately chosen grid may fail to capture the optimal ν_0, leading to suboptimal variable selection—either by missing important covariates (underselection) or including irrelevant ones (overselection). Balancing the granularity of the grid with computational feasibility is thus essential.

The total number of SAEM iterations is determined by two loops. The first loop runs over $\nu_0 \in \Delta$, where $\mathcal{M}$ is the number of values in the grid Δ. For each ν_0, the MCMC-SAEM algorithm performs K iterations to compute the MAP estimate $\hat{\Theta}_{\nu_0}^{MAP}$, resulting in a total of $\mathcal{M} \times K$ iterations for the first loop. The second loop processes unique sub-models $\hat{S}_{\nu_0}$, where U represents the number of unique sub-models obtained after thresholding. For each sub-model, the MCMC-SAEM algorithm runs for K' iterations to compute the MLE estimate $\hat{\Theta}_{\nu_0}^{MLE}$, contributing $U \times K'$ iterations for the second loop. Therefore, the total number of SAEM iterations is the sum of both loops, given by the formula: $\mathcal{M} \times K + U \times K'$.

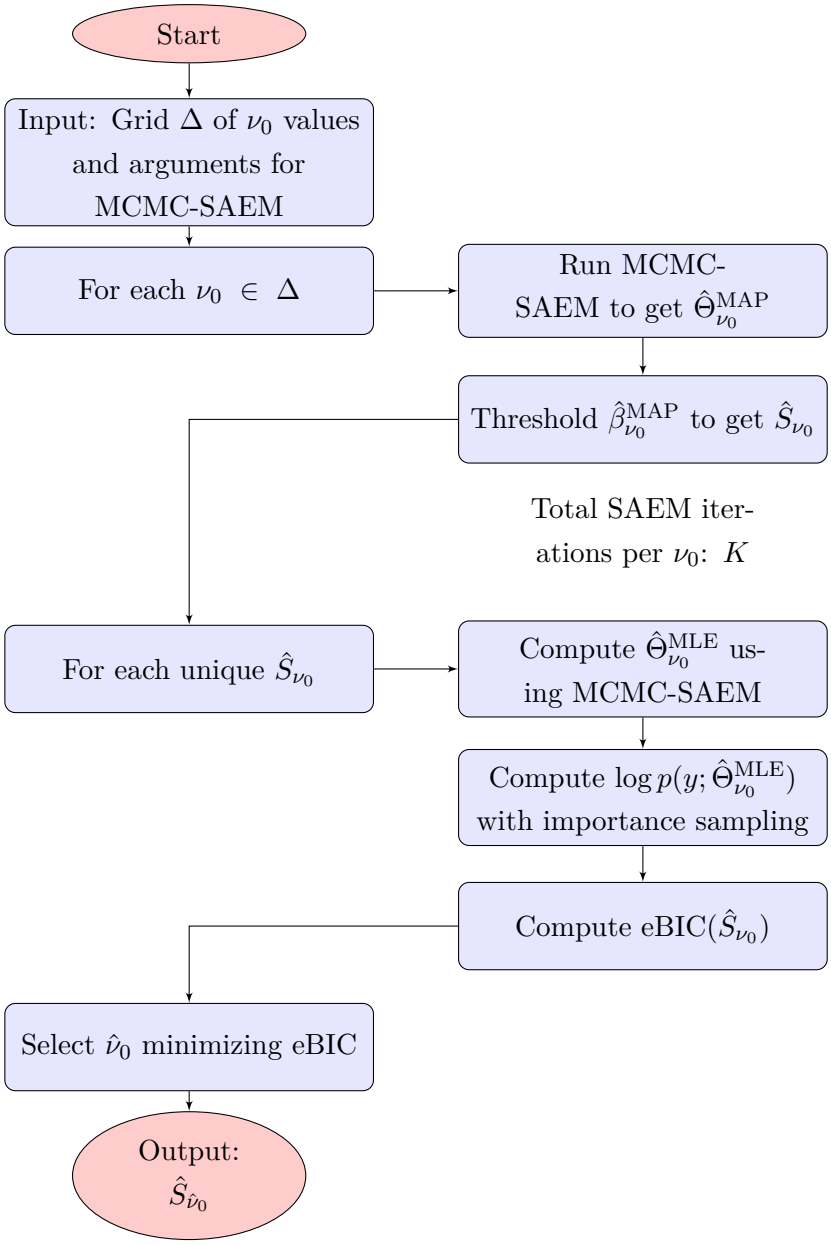

Figure 6: Illustration of the computational Workflow of the SAEMVS Algorithm

2.4 Introduction to Metamodeling

Metamodeling, also known as surrogate modeling, is a technique that aims to create simplified models that approximate the behavior of more complex systems. This approach is especially valuable when direct simulation or evaluation of the original model is computationally expensive or time-consuming. The concept of metamodeling can be traced back to the work of George E. P. Box, whose pioneering efforts in statistics laid the groundwork for many modern metamodeling methods [Box, 1976].

A metamodel, also known as a *surrogate model*, *black-box model*, or *emulator*, serves as a "model of a model". These simplified approximations aim to capture the essence of the relationship between the inputs and outputs of a complex model, offering a precise approximation with lower computational cost [Wang et al., 2014].

In their comparative study, [Wang et al., 2014] concluded that Gaussian Processes (GPs) are

the most effective method for constructing metamodels when compared to Polynomial Regression, Regression Trees, Random Forests, Multivariate Adaptive Regression Splines (MARS), Support Vector Machines (SVM), and Artificial Neural Networks (ANN). The methods were applied to benchmark problems like the Six-Dimensional Hartman function and a real-world hydrological modeling problem. Based on these findings, we chose Gaussian Processes for our metamodeling.

Our work primarily focuses on coupling the SAEMVS method with metamodeling. Kriging, originally developed in geostatistics, is a method for spatial interpolation and prediction based on Gaussian Processes. It was introduced by Daniel G. Krige in 1951 [Krige, 1951] and further formalized by Georges Matheron [Matheron, 1963]. Kriging provides the Best Linear Unbiased Prediction (BLUP) at unobserved locations based on observations at nearby locations.

Gaussian Process Regression (GPR) is a generalization of Kriging to arbitrary input spaces and is not limited to spatial data. In fact, GPR and Kriging are mathematically equivalent under certain assumptions [Rasmussen and Williams, 2006, Cressie, 1990]. Both methods use Gaussian Processes to model the underlying function and make predictions based on the observed data.

[Sacks et al., 1989] proposed the joint use of Kriging models as metamodels together with Latin Hypercube (LH) designs as an efficient method for approximating complex functions. In our work, we employ Gaussian Process Regression, commonly known as Kriging in the field of computer experiments, to construct metamodels that can approximate such functions. Moreover, we can have outputs in many cases that are time-dependent. This observation inspires our approach, where we develop a Kriging model based with both a time-dependent experimental design and a global design. To achieve this, we employ Latin Hypercube Sampling (LHS) to construct the design of experiments.

In the context of Stochastic Approximation Expectation-Maximization for Bayesian Variable Selection (SAEMVS), Kriging can be used as an effective metamodeling technique to approximate complex objective functions or likelihoods that are otherwise expensive to evaluate directly.

Introduction to Gaussian Processes To fully define a Gaussian Process (GP), it is necessary to specify a mean function μ and a covariance function K.

Definition 1 (Kernel Function)**.** *Let $\Omega \subset \mathbb{R}^d$. A function $K : \Omega \times \Omega \to \mathbb{R}$ is called a kernel function if it is symmetric and positive semi-definite (p.s.d.), meaning that for any set of points $x_1, \ldots, x_n \in \Omega$, the matrix $K(X, X)$ is p.s.d.*

For the matrix $K(X, X)$ to be p.s.d., it must satisfy the condition:

$$\text{For all } \boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)^\top \in \mathbb{R}^n : \quad \boldsymbol{\lambda}^\top K(X, X)\boldsymbol{\lambda} = \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j K(x_i, x_j) \geq 0.$$

We denote the Gaussian Process as $G \sim \mathcal{GP}(\mu, K)$, meaning that

$$\begin{aligned} &\mu : \Omega \to \mathbb{R}, \\ &\quad x \mapsto \mu(x) = \mathbb{E}[G(x)], \end{aligned} \quad \text{and} \quad \begin{aligned} &K : \Omega \times \Omega \to \mathbb{R}, \\ &\quad (x, y) \mapsto K(x, y) = \mathrm{Cov}(G(x), G(y)). \end{aligned}$$

That is, for any set of indices $x_1, \ldots, x_n \in \Omega$, we have

$$\begin{pmatrix} G(x_1) \\ \vdots \\ G(x_n) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu(x_1) \\ \vdots \\ \mu(x_n) \end{pmatrix}, \mathbf{\Sigma} \right), \quad \text{with} \quad \mathbf{\Sigma} = (K(x_i, x_j))_{1 \leq i,j \leq n}.$$

In this case, the probability density function of the Gaussian vector is given by

$$f_{G(x_1),\ldots,G(x_n)}(\boldsymbol{z}) = \frac{1}{(2\pi)^{n/2} \det(K(X,X))^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{z} - \mu(X))^\top K(X,X)^{-1} (\boldsymbol{z} - \mu(X)) \right),$$

where $\boldsymbol{z} = (z_1, \ldots, z_n)^\top$ and $\mu(X) = (\mu(x_1), \ldots, \mu(x_n))^\top$.

To ensure that the Gaussian Process is well-defined, the mean function μ does not require additional properties. However, the covariance function K must be symmetric and positive semi-definite, as detailed in the previous definition.

The key advantage of Gaussian Processes is their flexibility, allowing them to model complex functions without requiring a predefined parametric form [Rasmussen and Williams, 2006]. Examples of Kernels are shown in Annex B.

2.5 Gaussian Process Regression

Consider a set of observations $y_1, \ldots, y_n$ at associated locations $x_1, \ldots, x_n$ in a space $\mathbb{X}$, originating from an unknown function f [Roustant, 2023]. The goal is to build a function (metamodel) $\hat{f}$ that interpolates the data, i.e., $\hat{f}(x_i) = y_i$ for $i = 1, \ldots, n$, or at least approximates the data when the observations are noisy. Among the various techniques available, we focus on Gaussian Process Regression (GPR), which offers two primary advantages in the context of metamodeling. First, GPR is a probabilistic method, allowing for the quantification of uncertainty in unexplored regions. Second, it is parameterized by two functions (mean and kernel), providing flexibility and the ability to incorporate expert knowledge or domain-specific insights.

The origins of GP regression trace back to geostatistics, with foundational work by [Krige, 1951] and later extensions by [Matheron, 1963] in two or three dimensions. Its extension to higher dimensions emerged in the late 1980s, motivated by the need to analyze large computer simulations [Sacks et al., 1989]. Understanding the various facets of GP regression is beneficial. For example, the Universal Kriging variance formula from geostatistics may serve as a good trade-off for uncertainty quantification, partially accounting for parameter uncertainty without resorting to time-consuming Bayesian inference.

2.5.1 The Gaussian Process Approach

The concept of GP regression is based on the assumption that the unknown function f is a realization of a Gaussian Process $Y \sim \mathcal{GP}(\mu, K)$. The desired approximation is then obtained by conditioning on the observations $Y(x_i) = y_i$ for $i = 1, \ldots, n$. Using the properties of Gaussian vectors, this conditional process remains a GP, with explicit expressions for its mean and covariance functions. We distinguish two cases:

Noise-Free Observations. The conditional process $Y(x)$ given $Y(x_i) = y_i$ for $i = 1, \ldots, n$ is a GP with mean $m_{\mathrm{c}}(x)$ and covariance $K_{\mathrm{c}}(x, x')$, given by:

$$m_c(x) = \mu(x) + K(x, X)K(X, X)^{-1}(\boldsymbol{y} - \mu(X)),$$
$$K_c(x, x') = K(x, x') - K(x, X)K(X, X)^{-1}K(X, x'),$$

where $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$, $\mu(X) = (\mu(x_1), \ldots, \mu(x_n))^\top$, $K(x, X) = (K(x, x_1), \ldots, K(x, x_n))$, and $K(X, X) = (K(x_i, x_j))_{1 \le i,j \le n}$.

$m_c(x)$ interpolates the data: $m_c(x_i) = y_i$, and the uncertainty is zero at the design points: $K_c(x_i, x_i) = 0$. Additionally, $K_c(x_i, x) = 0$ for all $x \in \mathbb{X}$. Notably, $m_c(x)$ is affine with respect to the observations $\boldsymbol{y}$, and $K_c(x, x')$ does not depend on them.

Noisy Observations. When observations are noisy, we express them as

$$y_i = Y(x_i) + \epsilon_i,$$

where ϵ_i are mutually independent $\mathcal{N}(0, \sigma^2)$ noise terms, independent of Y. The prediction problem then becomes one of filtering, where we aim to predict the underlying value $Y(x)$ conditional on the noisy observations $y_1, \ldots, y_n$. The resulting conditional process $Y(x)$, given $Y(x_i) + \epsilon_i = y_i$ for $i = 1, \ldots, n$, is a GP with mean $m_c(x)$ and covariance $K_c(x, x')$, given by:

$$m_c(x) = \mu(x) + K(x, X)[K(X, X) + \sigma^2 I]^{-1}(\boldsymbol{y} - \mu(X)),$$
$$K_c(x, x') = K(x, x') - K(x, X)[K(X, X) + \sigma^2 I]^{-1}K(X, x'),$$

where I is the identity matrix of appropriate dimension. In this case, $m_c(x)$ no longer interpolates the data, and the uncertainty at observed locations is not zero, reflecting the presence of noise.

2.5.2 The Geostatistical Approach: Kriging

Simple Kriging. Simple Kriging assumes that the mean function $\mu(x)$ is known and constant across the domain, i.e., $\mu(x) = \mu$. The predictor relies solely on the spatial correlation structure provided by the covariance function $K(x, x')$. While this assumption simplifies calculations, it is often unrealistic in practice.

Universal Kriging. To address the limitations of Simple Kriging, Universal Kriging incorporates a trend model into the mean function. We assume that $Y(x)$ has a mean of the form $\mu(x) = f(x)^\top \boldsymbol{\lambda}$, where $f(x)$ is a vector of known basis functions, and $\boldsymbol{\lambda}$ is a vector of unknown coefficients [Cressie, 1990].

Theorem 1 (Universal Kriging Predictor [Cressie, 1990])**.** *Let $Y(x) = f(x)^\top \boldsymbol{\lambda} + Z(x)$, where $Z(x)$ is a zero-mean Gaussian Process with covariance function $K(x, x')$. The Universal Kriging predictor $\hat{Y}(x)$ minimizes the mean squared error $\mathbb{E}[(Y(x) - \hat{Y}(x))^2]$ subject to the unbiasedness condition $\mathbb{E}[\hat{Y}(x)] = \mathbb{E}[Y(x)]$. The predictor and its associated variance are given by:*

$$\hat{Y}(x) = f(x)^\top \hat{\boldsymbol{\lambda}} + K(x, X)K^{-1}(\boldsymbol{y} - F\hat{\boldsymbol{\lambda}}),$$
$$\sigma^2_{UK}(x) = K(x, x) - K(x, X)K^{-1}K(X, x) + [f(x) - K(x, X)K^{-1}F]^\top (F^\top K^{-1}F)^{-1}[f(x) - K(x, X)K^{-1}F],$$

where F is the design matrix with rows $f(x_i)^\top$, $K = K(X, X)$ is the covariance matrix, and $\hat{\boldsymbol{\lambda}}$ is the generalized least squares estimator:

$$\hat{\boldsymbol{\lambda}} = (F^\top K^{-1}F)^{-1}F^\top K^{-1}\boldsymbol{y}.$$

Proof. See Cressie [1990, Chapter 3] for a detailed derivation. □

These expressions account for the uncertainty in estimating the trend coefficients $\boldsymbol{\lambda}$ and provide a more accurate quantification of the prediction variance [Cressie, 1990].

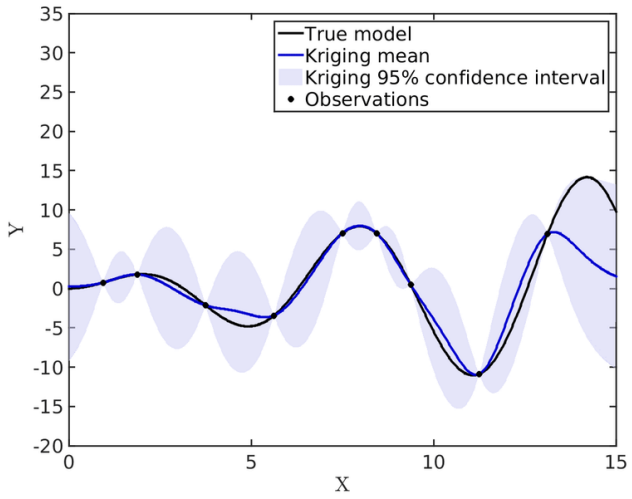

Figure 7: An example of a Kriging metamodel [Lataniotis et al., 2015].

This image from [Lataniotis et al., 2015] represents a Kriging metamodel. The black line shows the true model, the blue line represents the Kriging mean prediction, and the shaded area indicates the 95% confidence interval of the Kriging prediction. The Kriging mean closely follows the true model around the observation points (black dots), and the confidence intervals are narrow near these points, indicating higher prediction accuracy. However, the intervals widen in regions with fewer observations, reflecting greater uncertainty in the predictions. Overall, Kriging effectively interpolates between the known observations while providing a measure of uncertainty in the unobserved areas.

2.5.3 Hyperparameter Inference

In practical applications, we often consider Gaussian Processes with a linear trend, which can be written as

$$Y(x) = \mu(x) + Z(x),$$

where:

- $\mu(x) = f(x)^\top \boldsymbol{\lambda}$ is a linear trend with known basis functions $f(x)$ and unknown coefficients $\boldsymbol{\lambda}$,
- $Z(x)$ is a zero-mean GP with covariance function $K(x, x'; \phi)$, where ϕ represents hyperparameters of the covariance function.

Here, $\boldsymbol{\lambda}$ and ϕ are vectors of unknown parameters, commonly referred to as hyperparameters. We now describe a standard approach for estimating these hyperparameters.

Maximum Likelihood Estimation. Maximum Likelihood Estimation (MLE) involves finding the parameters that maximize the likelihood of the observed data. The joint distribution of the observations $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$ is multivariate normal:

$$\boldsymbol{y} \sim \mathcal{N}(F\boldsymbol{\lambda}, K(\phi)),$$

where F is the design matrix with rows $f(x_i)^\top$, and $K(\phi) = K(X, X; \phi)$ is the covariance matrix computed using the covariance function $K(x, x'; \phi)$. The log-likelihood function is given by:

$$\ell(\boldsymbol{\lambda}, \phi) = -\frac{1}{2}\left[n \log(2\pi) + \log\det K(\phi) + (\boldsymbol{y} - F\boldsymbol{\lambda})^\top K(\phi)^{-1}(\boldsymbol{y} - F\boldsymbol{\lambda})\right].$$

The MLEs of $\boldsymbol{\lambda}$ and ϕ are obtained by maximizing $\ell(\boldsymbol{\lambda}, \phi)$. This optimization problem is typically solved numerically, as there is no closed-form solution due to the dependence of $K(\phi)$ on ϕ [Rasmussen and Williams, 2006].

Remark. While the MLE approach provides point estimates of the hyperparameters, it does not account for their uncertainty in the predictions. Universal Kriging partially addresses this issue by incorporating the uncertainty of the trend coefficients $\boldsymbol{\lambda}$ into the prediction variance. This method serves as a good trade-off for uncertainty quantification without resorting to computationally intensive Bayesian inference [Stein, 1999, Cressie, 1990].

2.6 Design of Computer Experiments

Designing computer experiments is a critical aspect of studying complex systems where the function of interest, denoted as f, represents a simulator, a machine learning algorithm, or a regression function. Unlike traditional physical experiments where data is collected from real-world observations, computer experiments involve data generated through simulations. This allows researchers to select the input points where the function f is evaluated, a process commonly referred to as Design of Experiments (DoE).

The primary goal of designing computer experiments is to create a set of input points that effectively explore the input space $\mathbb{X}$, leading to accurate models or predictions of f while minimizing the number of costly evaluations. This approach is particularly important when dealing with expensive-to-evaluate functions, where each simulation run may require significant computational resources.

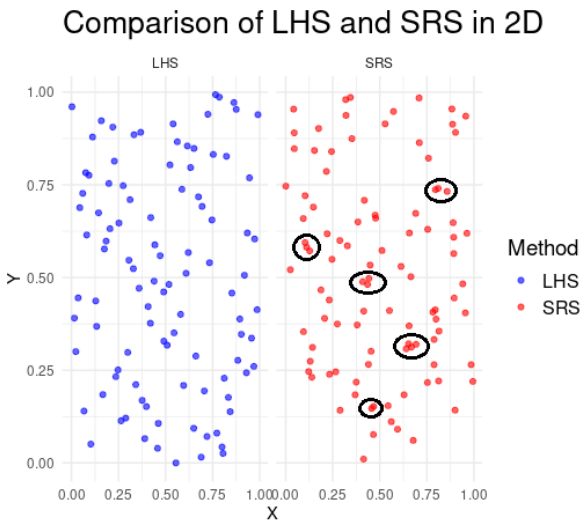

Figure 8: Simple Random Sampling (SRS) and Latin Hypercube Sampling (LHS) in 2D with 100 samples

Simple Random Sampling (SRS) is a basic sampling method in which each point is selected randomly and independently from the entire space, with each having an equal probability of being chosen [Fang et al., 2005]. While this method can be straightforward, it often results in clusters or gaps, as shown on the right plot (red points). In contrast, Latin Hypercube Sampling (LHS) (left plot, blue points) ensures more uniform coverage by dividing the space into non-overlapping intervals or strata for each variable, selecting one point from each interval. This leads to a more even distribution of points across the entire design space, reducing the risk of leaving areas unsampled, as evident from the comparison of both methods in the figure.

According to Fang et al. [2005], the design of computer experiments can be broadly categorized into static and adaptive strategies. A static strategy involves creating a fixed set of design points prior to any evaluations, often aiming for a space-filling design that evenly covers the input space without specific model information. In contrast, an adaptive strategy incrementally adds design points based on the outcomes of previous evaluations, typically guided by criteria that seek to optimize specific objectives such as model accuracy, optimization, or uncertainty reduction.

2.6.1 Space-Filling Designs

Space-filling designs are foundational in computer experiments where the function f is assumed to be complex and nonlinear, with negligible experimental noise [Fang et al., 2005]. The primary goal of a space-filling design is to distribute the design points as uniformly as possible across the input space $\mathbb{X}$, ensuring that no region is left unexplored [Roustant, 2023].

Key properties desirable in space-filling designs include:

- **Uniform Coverage**: The design should cover the entire domain evenly, enabling comprehensive exploration of f.
- **Avoidance of Replication**: Since the noise in computer experiments is negligible, it is inefficient to evaluate f multiple times at the same point. Thus, replication should be avoided.

- **Projection Properties**: The design should maintain its space-filling properties even when projected onto lower-dimensional subspaces, which is important when f depends on fewer variables than the dimension of $\mathbb{X}$.

While uniform grids might seem a natural choice for space-filling, they often lead to clustering and poor coverage of the input space, especially with a limited number of points. As an alternative, more sophisticated designs, such as LHS, have been developed to address these issues.

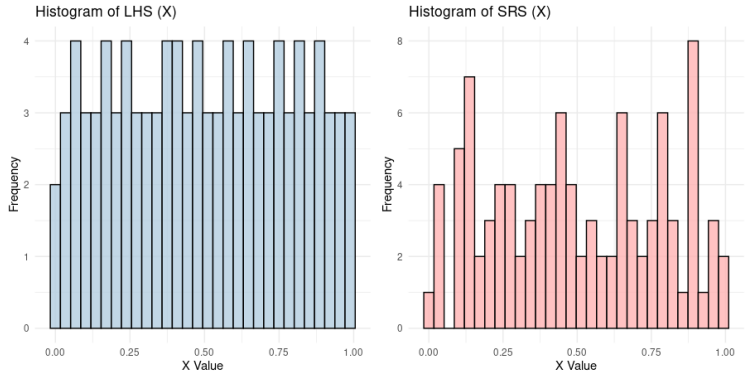

Figure 9: Histograms comparing Latin Hypercube Sampling (LHS) and Simple Random Sampling (SRS).

The histogram on the left represents the distribution of sample values from the LHS. It shows a relatively uniform distribution, as expected from LHS, where the sample points are spread evenly across the range. The frequency for each bin remains fairly constant, with minimal variation between different intervals.

The histogram on the right shows the distribution of sample values from SRS. In contrast to LHS, the distribution is less uniform, with noticeable clustering in some bins and gaps in others. This uneven distribution is characteristic of Simple Random Sampling, where randomness can result in both under-sampling and over-sampling in different regions of the range.

2.6.2 Latin Hypercube Sampling (LHS)

Given the importance of space-filling designs in computer experiments, we now focus on Latin Hypercube Sampling (LHS), one of the most widely used techniques in this context. LHS was introduced to address the limitations of simple random sampling by ensuring a more uniform and comprehensive coverage of the input space [McKay et al., 1979, Fang et al., 2005].

LHS is a stratified sampling technique that divides each input variable's range into m equal intervals, ensuring that each interval is sampled exactly once. Consider a d-dimensional input space $\mathbb{X} = [0,1]^d$. In fact, the LHS can be defined in terms of the Latin hypercube design [Fang et al., 2005].

Definition 2 (Latin hypercube design (LHD) [McKay et al., 1979])**.** *A LHD with m runs and d input variables, denoted by $LHD(m,d)$, is an $m \times d$ matrix, in which each column is a random permutation of $\{1,2,...,m\}$.*

An LHS can be generated by the following algorithm (LHSA). An m-point LHS on $\mathbb{X}$ is constructed as follows:

1. Independently take d permutations $\pi_j(1), \ldots, \pi_j(m)$ of the integers $1, \ldots, m$ for $j = 1, \ldots, d$, i.e., generate an $LHD(m, d)$.

2. Take md uniform variates $U_k^j \sim U(0,1)$, for $k = 1, \ldots, m$, $j = 1, \ldots, d$, which are mutually independent. Let $x_k = (x_k^1, \cdots, x_k^d)$, where

$$x_k^j = \frac{\pi_j(k) - U_k^j}{m}, \quad k = 1, \ldots, m, \quad j = 1, \ldots, d.$$

Then, $D_m = \{x_1, \ldots, x_m\}$ is a Latin Hypercube Sample (LHS) and is denoted by $LHS(m, d)$.

Example 1 (From Fang et al. [2005]). *For generating an LHS for $m = 8$, $d = 2$, in the first step we generate two permutations of $\{1, 2, \ldots, 8\}$ as $(2, 5, 1, 7, 4, 8, 3, 6)$ and $(5, 8, 3, 6, 1, 4, 7, 2)$. Then we generate $16 = 8 \times 2$ random numbers to form an 8×2 matrix on the right below:*

$$\begin{pmatrix} 2 & 5 \\ 5 & 8 \\ 1 & 3 \\ 7 & 6 \\ 4 & 1 \\ 8 & 4 \\ 3 & 7 \\ 6 & 2 \end{pmatrix}, \quad \begin{pmatrix} 0.9501 & 0.8214 \\ 0.2311 & 0.4447 \\ 0.6068 & 0.6154 \\ 0.4860 & 0.7919 \\ 0.8913 & 0.9218 \\ 0.7621 & 0.7382 \\ 0.4565 & 0.1763 \\ 0.0185 & 0.4057 \end{pmatrix}$$

Now an LHS is given by:

$$\frac{1}{8}\left[\begin{pmatrix} 2 & 5 \\ 5 & 8 \\ 1 & 3 \\ 7 & 6 \\ 4 & 1 \\ 8 & 4 \\ 3 & 7 \\ 6 & 2 \end{pmatrix} + \begin{pmatrix} 0.9501 & 0.8214 \\ 0.2311 & 0.4447 \\ 0.6068 & 0.6154 \\ 0.4860 & 0.7919 \\ 0.8913 & 0.9218 \\ 0.7621 & 0.7382 \\ 0.4565 & 0.1763 \\ 0.0185 & 0.4057 \end{pmatrix}\right] = \begin{pmatrix} 0.1312 & 0.5223 \\ 0.5961 & 0.9444 \\ 0.0491 & 0.2981 \\ 0.8143 & 0.6510 \\ 0.3886 & 0.0098 \\ 0.9047 & 0.4077 \\ 0.3179 & 0.8530 \\ 0.7477 & 0.1993 \end{pmatrix}.$$

LHS is particularly advantageous in the context of computer experiments due to its ability to ensure better coverage of the input space compared to random sampling. This systematic spread of sample points reduces the likelihood of clustering and ensures that the entire input space is explored.

Mathematically, LHS reduces the variance in the estimation of integrals, which is critical when the objective is to approximate an expected value or the average behavior of f. Stein [1987] showed that for certain classes of functions, the variance of the sample mean using LHS is lower than that obtained using simple random sampling. Specifically, if $\hat{f} : \mathbb{X} \to \mathbb{R}$ is a function with finite second moment, and $x_1, \ldots, x_n$ are points from an LHS, then:

$$\mathrm{Var}\left(\frac{1}{m}\sum_{k=1}^{m} \hat{f}(x_i)\right) \leq \mathrm{Var}\left(\frac{1}{m}\sum_{k=1}^{m} \hat{f}(u_i)\right),$$

where $u_1, \ldots, u_n$ are independent and identically distributed random variables over $\mathbb{X}$. This inequality highlights the efficiency of LHS in reducing uncertainty, especially as the number of dimensions or sample points increases.

In practice, LHS is widely used in the simulation of complex systems, serving as the basis for creating surrogate models or emulators [Fang et al., 2005]. By providing a well-distributed set of input points, LHS enables more accurate predictions of the function's behavior across the entire input space, which is essential when the function is expensive to evaluate.

2.6.3 Connection to Universal Kriging

The design of experiments has a direct impact on the quality of surrogate models like Universal Kriging. Universal Kriging not only models the spatial correlation through a covariance function but also includes a trend component to capture global variations in the data [Cressie, 1990].

In Universal Kriging, the choice of design points affects the estimation of both the trend coefficients and the covariance parameters. A well-designed experiment, such as one based on LHS, ensures that the input space is adequately sampled, which leads to more accurate estimates of these parameters and, consequently, better predictions [Fang et al., 2005].

Space-filling designs, especially Latin Hypercube Sampling, provide a systematic way to cover the input space, ensuring that the surrogate models built upon them can accurately predict the behavior of the underlying function f. This is particularly beneficial when using advanced metamodeling techniques like Universal Kriging, where the quality of predictions is highly dependent on the design of experiments.

3 Coupling of SAEMVS Method and Kriging

To couple the Stochastic Approximation Expectation-Maximization Variable Selection (SAEMVS) method with Kriging, we replace the computationally expensive evaluations of the nonlinear function $g(\varphi_i, x_{ij})$ with predictions from a metamodel constructed using Kriging. This approach aims to improve computational efficiency without significantly compromising accuracy.

Consider the statistical model defined in Section 2.1, where $n \in \mathbb{N}$ represents the number of individuals, $J \in \mathbb{N}$ the number of observations per individual, and $p \in \mathbb{N}$ the number of covariates. We assume a one-dimensional individual parameter φ_i and exclude fixed effects ψ. For each individual $i = 1, \ldots, n$ and each observation $j = 1, \ldots, J$, the model is defined as:

$$\begin{cases} y_{ij} = g\left(\varphi_i, x_{ij}\right) + \varepsilon_{ij}, & \varepsilon_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \sigma^2\right), \\ \varphi_i = \mu + \boldsymbol{\beta}^\top \mathbf{V}_i + \xi_i = \tilde{\boldsymbol{\beta}}^\top \tilde{\mathbf{V}}_i + \xi_i, & \xi_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}\left(0, \Gamma^2\right), \end{cases} \tag{4}$$

where:

- $\tilde{\boldsymbol{\beta}} = (\mu, \boldsymbol{\beta})^\top \in \mathbb{R}^{p+1}$ is the augmented coefficient vector.
- $\tilde{\mathbf{V}}_i = (1, V_{i1}, \ldots, V_{ip})^\top \in \mathbb{R}^{p+1}$ is the augmented covariate vector.

For our following simulation studies we will use the logistic growth model g defined in equation 5.

$$g(\varphi_i, \psi, t_{ij}) = \frac{\psi_1}{1 + \exp\left(-\frac{\varphi_i - x_{ij}}{\psi_2}\right)} \tag{5}$$

3.1 Metamodel Construction via Kriging

The metamodel $\hat{g}$ approximates the function g using a Gaussian Process (GP) regression model, also known as Kriging. To build this metamodel, we create an experimental design using Latin Hypercube Sampling (LHS) and employ the `DiceDesign` and `DiceKriging` packages developed by Roustant et al. [2012].

The steps for constructing the Kriging metamodel are as follows:

1. **Generate LHS design:** Create an LHS design of m points in the input space of (φ, x), ensuring a good coverage of the domain.
2. **Evaluate the true function:** Compute $g(\varphi_i, x_{ij})$ at the design points to obtain the responses.
3. **Train the kriging model:** Use the design points and corresponding responses to train a Kriging model $\hat{g}$, specifying appropriate covariance functions and trend models.
4. **Validate the metamodel:** Assess the performance of $\hat{g}$ using metrics such as RMSE, MAE, and R^2, possibly with a validation dataset.

Remarks on the nugget effect: In the construction of the Kriging metamodel, we introduce a small nugget effect to account for potential observation noise and to ensure numerical stability. The nugget, initially introduced by [Matheron, 1963], adds a small variance to the diagonal of the covariance matrix:

$$K(\mathbf{x}_i, \mathbf{x}_j) = K_0(\mathbf{x}_i, \mathbf{x}_j) + \delta_{ij}\tau^2,$$

where δ_{ij} is the Kronecker delta and τ^2 is the nugget variance. This adjustment prevents the covariance matrix from becoming bad-conditioned, facilitating its inversion during the Kriging model training [Stein, 1999, Cressie, 1993].

3.2 Integration into the MCMC-SAEM Algorithm

The integration of the Kriging metamodel into the MCMC-SAEM algorithm is achieved by replacing the direct evaluations of $g(\varphi_i, x_{ij})$ with predictions from $\hat{g}(\varphi_i, x_{ij})$. Specifically, the metamodel is used in two critical steps:

1. **E-Step (Stochastic Approximation):** During the simulation of the latent variables φ_i, the likelihood computations involving $g(\varphi_i, x_{ij})$ are replaced with $\hat{g}(\varphi_i, x_{ij})$.
2. **M-Step (Maximization):** In updating the parameters μ, $\boldsymbol{\beta}$, γ^2, and σ^2, the sufficient statistics are computed using $\hat{g}(\varphi_i, x_{ij})$.

By utilizing $\hat{g}$ within the SAEM algorithm, we reduce the computational burden associated with evaluating g at each iteration, as Kriging predictions are computationally less expensive.

3.3 Implementation of the SAEMVS procedure with Kriging

The SAEMVS procedure, as developed by [Naveau et al., 2024], is adapted to incorporate the Kriging metamodel. The key idea is to use $\hat{g}$ in place of g throughout the procedure, particularly in the computation of the maximum a posteriori (MAP) estimates and the evaluation of the extended Bayesian information criterion (eBIC). The implementation of the code in R for the Kriging by time can be found in Annex C.1 and for the global Kriging in Annex C.1.7.

3.3.1 Model Reduction Step

For each value of the regularization parameter ν_0 in a predefined grid Δ, we perform the following steps:

1. **Compute MAP estimate:** Use the MCMC-SAEM algorithm with $\hat{g}$ to compute the MAP estimate $\hat{\Theta}_{\nu_0}^{\text{MAP}}$ of the parameters.
2. **Thresholding:** Apply a threshold to the estimated coefficients $\hat{\beta}_{\nu_0}^{\text{MAP}}$ to define the sub-model $\hat{S}_{\nu_0}$, retaining only the significant covariates.

3.3.2 eBIC Criterion Computation

For each unique sub-model $\hat{S}_{\nu_0}$, we compute the eBIC as follows:

1. **Compute MLE Estimate:** Use the MCMC-SAEM algorithm with $\hat{g}$ to compute the maximum likelihood estimate (MLE) $\hat{\Theta}_{\nu_0}^{\text{MLE}}$.
2. **Log-Likelihood estimation:** Employ importance sampling techniques to estimate the log-likelihood $\log p(y; \hat{\Theta}_{\nu_0}^{\text{MLE}})$ based on $\hat{g}$.
3. **Compute eBIC:** Calculate the eBIC for each sub-model $\hat{S}_{\nu_0}$ using the estimated log-likelihood.

The optimal level of sparsity is determined by selecting the value of ν_0 that minimizes the eBIC. This results in the selection of the sub-model $\hat{S}_{\hat{\nu}_0}$, which balances model fit and complexity.

3.3.3 Summary of the Coupling Procedure

The coupling of the SAEMVS method and Kriging can be summarized as follows:

- **Metamodel Construction:** Build a Kriging metamodel $\hat{g}$ to approximate g.
- **SAEMVS Algorithm with Metamodel:** Use $\hat{g}$ within the SAEMVS algorithm to compute MAP and MLE estimates.
- **Model Selection:** Evaluate models using eBIC computed with $\hat{g}$ and select the optimal model.

This approach is claimed to reduce computational costs by avoiding direct evaluations of g, but only for complex models. Moreover, this has not been numerically verified during my internship, as there was not enough time to explore the complex crop model or real data.

4 Simulation Study

4.1 Important elements to consider when designing simulations

Kriging has been successfully applied in various fields, such as engineering design and geostatistics, as noted in [Krige, 1951, Sacks et al., 1989]. However, our primary goal was to verify whether it's possible to use a metamodel for variable selection without significantly degrading performance. The key question is: under what conditions can this be achieved?

It is not particularly meaningful to compare computational time in this context because we are currently working with a very simple case logistic growth regression, where the exact evaluation of the model is nearly instantaneous. When using the metamodel in this simple example, we actually end up with larger computational times due to the overhead of handling large linear combinations. In this simple case, running SAEMVS with the metamodel takes

more time than using the exact model. Nevertheless, this is not a major problem, as the simulated model serves to help us understand the behavior of the SAEMVS method coupled with metamodeling. This is an important preliminary step before moving on to more complex models, where the real value of the [Naveau et al., 2024] methodology development lies.

4.1.1 Time-Dependent vs. Global Kriging in application to real problems

It is important to clarify why both Time Kriging and Global Kriging approaches were considered in this study. In practical experimental settings, particularly in biological experiments, time points at which measurements are collected are often fixed. For instance, we may know in advance that results will be measured at specific times, such as 5 and 15. In such cases, it would be unnecessary to try to model or measure the function at a time like 2, because there is no data available for that point. This is the underlying assumption of Time Kriging: we fix the time variable and focus the modeling on other variables, such as φ, to see if it improves the performance of Kriging. This concept is crucial as it directly relates to real-world applications where time might not be the key variable.

A relevant biological case could illustrate how Time Kriging can be applied to future data analysis, especially when the "time" variable in the study is qualitative rather than temporal. For example, in the [Stat4Plant, 2021] project, repetitions occur not over time, but across different environments. These environments are qualitative variables, such as soil fertilizer types or environmental conditions. In such cases, it does not make sense to sample beyond the support defined by the observed environments in the experiment. This highlights the value of using Time Kriging in scenarios where the variable fixed in the model, while labeled as time, could represent other qualitative factors.

In contrast, Global Kriging models the function across a continuous time space, rather than being limited to specific time points as in the Time Kriging case. The idea for comparing Time Kriging and Global Kriging is to determine whether treating time as a continuous variable, rather than fixing specific time points, has a significant impact on the performance of Kriging. This comparison is fundamental to the construction of the metamodel and my analysis of its effectiveness, as each approach has distinct strengths depending on the experimental design.

4.1.2 Design of Experiments using Latin Hypercube Sampling (LHS)

The implementation of the metamodel construction relies on generating training data through *Latin Hypercube Sampling* (LHS). This technique ensures that the input space is explored uniformly, providing better coverage than simple random sampling [McKay et al., 1979]. The `DiceDesign` package is employed for the LHS design, and the metamodel is built using Kriging with the `DiceKriging` package [Roustant et al., 2012]. There are two main approaches to how LHS is applied, depending on whether the Kriging model is global or time-based.

The main distinction between the global and time-based Kriging approaches lies in how the LHS design is applied to the variables. In the **global Kriging approach**, both the covariate φ and the time variable t are sampled simultaneously usign LHS. This ensures that the design provides a comprehensive representation across both variables. The φ values are then scaled to the range $[0, 1100]$, and the time values are scaled to $[150, 3000]$, ensuring that the Kriging model captures the global relationship between φ and t. This global sampling strategy treats time as

an integral part of the input space, assuming that the relationship between the covariates and the response variable remains consistent across all time points.

In contrast, the **time-based Kriging approach** applies LHS independently for each time point. Instead of sampling φ and t together, as we kwon the time points are fixed, and LHS is used only for the covariate φ at each specific time (which is also scaled to $[150, 3000]$. Additionally, stratified sampling is applied to ensure that the data is split into training and test sets in a way that respects the temporal structure, which is not done in the global approach (because it use a random sampling).

4.1.3 Kriging Model and metamodeling in SAEMVS

Once the design points and responses are obtained, the Kriging model is trained with several configurations. For example, the choice of the Mater5/2 kernel is supported in literature by [Stein, 1999], where it has been demonstrated to offer a good balance between smoothness and flexibility, especially in cases involving non-differentiable functions or when smoothness assumptions may not hold. But we also try to run it with Gauss and Exponential kernels. The model's predictions at new points are obtained using Universal Kriging (UK) with bias correction. UK is known for its robustness in situations where the underlying trend is unknown or not constant [Cressie, 1993].

Kriging serves as a surrogate model during the SAEMVS iterations, replacing the direct evaluation of g (see equation (5)). This substitution allows for substantial computational savings, as the model approximates g based on a limited number of evaluations, while still maintaining high prediction accuracy. The Kriging predictions are used to compute the log-likelihood and update the sufficient statistics during the S-step of the SAEMVS algorithm (see Section 3.3). The efficiency gains are further enhanced by utilizing parallelization techniques, which allow for multiple Kriging models to be evaluated simultaneously, as shown in the code structure using `foreach` and `doParallel` packages [Folashade, 2022, Weston, 2022].

4.1.4 Covariance Structure and Trend:

The choice of covariance function, or kernel, is essential for the performance of the Kriging model, as it defines the spatial relationships between data points. Using the `km` function from the `DiceKriging` package, we experimented with several kernels, including Gaussian, Exponential, and Matern 5/2. The Gaussian kernel is known for its ability to model smooth and continuous processes effectively, while the Matern 5/2 kernel offers flexibility in capturing both smoothness and irregularities, as highlighted by Stein [1999].

The exponential kernel, unlike the Gaussian, is suited for modeling more abrupt variations and lacks the smoothness assumptions inherent in the Gaussian kernel. This kernel is characterized by a shorter-range correlation structure, meaning that it places higher emphasis on nearby points and decays faster with distance, making it appropriate for modeling processes with rapid changes [Roustant et al., 2012]. However, in our experiments, it performed less effectively in capturing the more gradual variations present in the data, leading to lower accuracy in predictions.

We also examined the role of the trend function in the Kriging model. Although we tested more complex polynomial trends, these did not lead to substantial improvements in model

accuracy. In our case, the relationship between input variables and the response was primarily captured by the covariance structure. Consequently, a simple constant trend was sufficient, which reduced computational overhead without compromising the model's effectiveness. This result aligns with previous observations where Kriging models often capture the data's underlying structure without needing complex trend functions [Roustant et al., 2012]. Thus, while the choice of covariance function significantly impacted the model's performance, particularly favoring the Matern 5/2 kernel, the trend component had minimal effect in our specific case as we will see in Section 4.3.

4.2 Simulation Design

In this simulation study, we generated 100 datasets to evaluate the performance of the SAEMVS algorithm coupled with metamodeling via Kriging. Each dataset was created under the following specifications:

- **Number of individuals (n)**: Each dataset consisted of $n \in \{100, 200\}$ observations.
- **Number of covariates (p)**: We considered $p \in \{500, 2000\}$ covariates, of which only the first three $(\beta_1, \beta_2, \beta_3)$ are truly influential.
- **Model specification**: The underlying model for each dataset was defined as in Equation 5.
- **True parameter values**:
  - **Intercept (μ)**: 1200.
  - **Covariate effects (β)**:
    $$\beta = (\beta_1, \beta_2, \beta_3, \beta_4, \ldots, \beta_p),$$
    with $\beta_1 = 100$, $\beta_2 = 50$, $\beta_3 = 20$, and $\beta_\ell = 0$ for $\ell = 4, \ldots, p$.
  - **Random effect variance (Γ^2)**: 200 (inter-individual variance of φ_i).
  - **Residual variance (σ^2)**: 30.
- **Function $g(\varphi_i, \psi, t_{ij})$**: Defined as
  $$g(\varphi_i, \psi, t_{ij}) = \frac{\psi_1}{1 + \exp\left(-\dfrac{t_{ij} - \varphi_i}{\psi_2}\right)},$$
  where $\psi = (\psi_1, \psi_2)$ with $\psi_1 = 200$ and $\psi_2 = 300$.
- **Time points (t_{ij})**: Equally spaced between 150 and 3000, with $J = 10$ time points for each individual.

The SAEMVS algorithm was initialized with the following parameter values:

- **Initial estimates of β**:
  $$\beta^{(0)} = (\beta_0^{(0)}, \beta_1^{(0)}, \ldots, \beta_p^{(0)}),$$
  where
  $$\beta_0^{(0)} = 1400, \quad \beta_\ell^{(0)} = 100 \text{ for } \ell = 1, \ldots, 10, \quad \beta_\ell^{(0)} = 1 \text{ for } \ell = 11, \ldots, p.$$
- **Initial residual variance ($\sigma^{2(0)}$)**: 100.
- **Initial random effect variance ($\Gamma^{2(0)}$)**: 5000.
- **Initial inclusion probability ($\alpha^{(0)}$)**: 0.1.
- **Initial value for τ**: 0.98 (associated with the initialization of Γ^2).

The hyperparameters were set as follows:

- **Spike parameter (ν_0)**: Explored through a grid during model selection. The value 0.04 is only used for testing purposes in SAEM.
- **Slab parameter (ν_1)**: 12000.
- **Prior parameters for Γ^2**: Shape $\nu_\Gamma = 1$, Scale $\lambda_\Gamma = 1$.
- **Prior parameters for σ^2**: Shape $\nu_\sigma = 1$, Scale $\lambda_\sigma = 1$.
- **Prior parameters for inclusion probability α**: $a = 1$, $b = p$.

- **Prior variance for intercept** μ: $\sigma_\mu^2 = 3000$.

Δ is the grid of values for ν_0, explored as part of a hyperparameter grid search. The values of Δ are defined on a logarithmic scale as:

$$\log_{10}(\Delta) = \left\{-2 + k \times \frac{4}{19},\ k \in \{0, \ldots, 19\}\right\}$$

This grid spans a range from 10^{-2} to 10^2, providing 20 different values for Δ to ensure robustness in model selection.

These settings were consistent across all simulations to ensure comparability of results and initial values were selected based on the [Naveau et al., 2024] article.

All simulations were performed using a fixed random seed to ensure reproducibility. The consistent use of these parameter settings across all datasets allows for a fair comparison of the SAEMVS algorithm's performance with and without metamodeling.

Application of SAEMVS-Metamodel

For each of the 100 simulated datasets, the SAEMVS-metamodel procedure was applied across all configurations of the number of training points ($m = 50, 100, 150, 200$) and covariance structures (Matern 5/2, Gaussian or Exponential kernels). The steps followed were:

1. **Design of experiments:** for each configuration, training points were generated using LHS with the specified m.
2. **Metrics calculation:** performance metrics (RMSE, MAE, R^2) were computed in order to evaluate the quality of the metamodel.
3. **Kriging model fitting:** a Kriging model was fitted using the `DiceKriging` package with the chosen covariance function.
4. **SAEMVS execution:** the SAEMVS algorithm was run using the Kriging metamodel to approximate g during the iterative estimation process.

4.3 Interpretation of the Results

4.3.1 Analysis of Global Kriging Results

The following graphs illustrate the behavior of variable selection and parameter estimation based on various simulations conducted during the internship. It is essential to be rigorous in the conclusions drawn from these simulations and to explain the observed patterns thoroughly, as this numerical analysis is a central component of the work.

Bar plot of correctly selected models: In Figure 10, we observe the evolution of the coupling's ability to correctly select the three non-zero effect variables as the number of points m increases. The plot shows a clear improvement in selection accuracy as m grows, eventually approaching the reference result obtained from SAEMVS applied to the true model without Kriging. This behavior is expected: with more data points, the Kriging metamodel can more effectively approximate the underlying function g, leading to better variable selection.

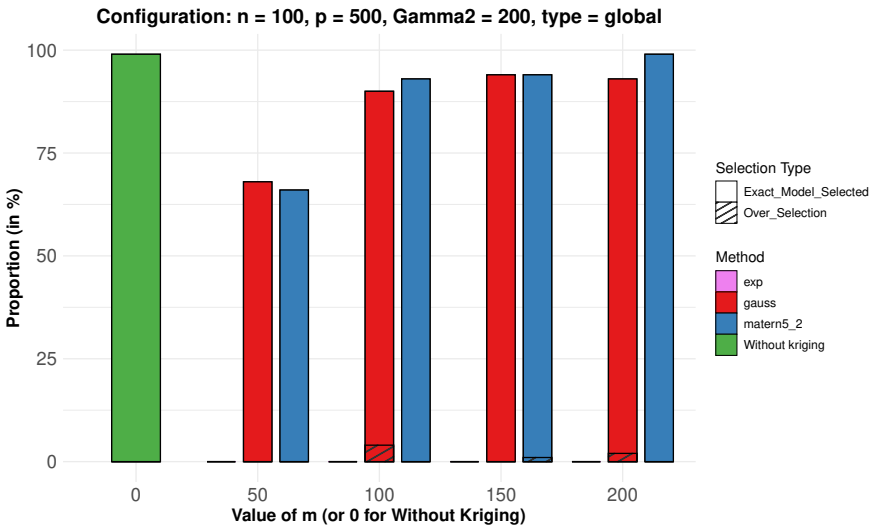

Figure 10: Bar plot of correct selected models for $n = 100$, $p = 500$, $\Gamma^2 = 200$, by global Kriging.

However, what is particularly interesting here is the significant influence of the covariance kernel used in Kriging on the performance of the selection process. When using the exponential kernel, the performance is notably poor, as the model struggles to select the correct variables. This can be attributed to the smoothness properties of the exponential kernel, it lacks the flexibility needed to adapt to more complex underlying relationships between the variables, which leads to a loss of accuracy in the selection process.

In contrast, the Gaussian and Matern $\nu = 5/2$ kernels show marked improvements. The Gaussian kernel, being more flexible than the exponential kernel, allows for better handling of the spatial correlations in the data, leading to more accurate variable selection. The best results, however, are achieved using the Matern $\nu = 5/2$ kernel. This is likely because the Matern kernel, particularly with $\nu = 5/2$, provides a good balance between smoothness and flexibility, making it more suited to capturing the relationships between the covariates and the response.

This strong dependence of the results on the chosen kernel underscores the importance of selecting an appropriate covariance function for Kriging. In scenarios where the underlying function g is complex, choosing a kernel with insufficient flexibility (such as the exponential kernel) can severely degrade performance. Therefore, kernel selection is a critical step that directly impacts the effectiveness of the metamodel in aiding variable selection.

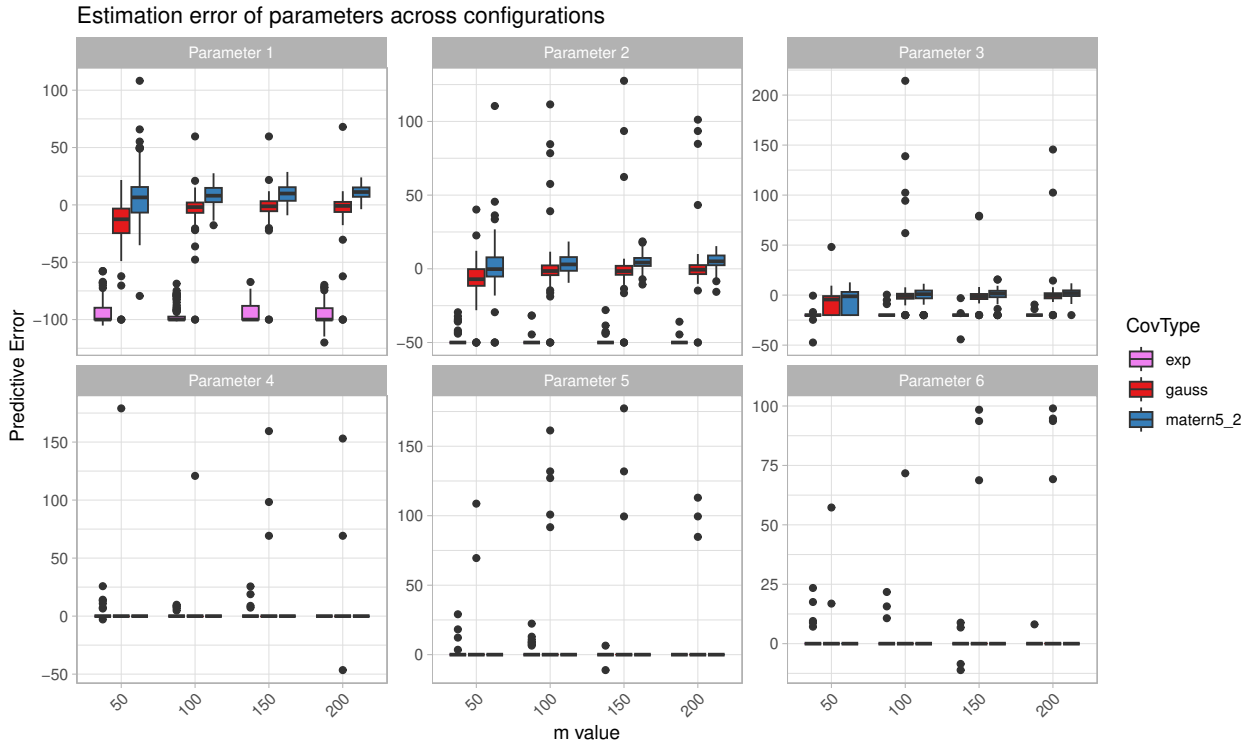

Figure 11:
footnotesize Boxplot of estimation error of parameters β_j for $j \in \{1, ..., 6\}$ with $n = 100$, $p = 500$, $\Gamma^2 = 200$ across coniguration by global Kriging.

Figure 11 displays the estimation error of parameters β_j for $j \in \{1, ..., 6\}$ across different covariance types (exponential, Gaussian, and Matern $\nu = 5/2$) and varying values of m (the number of data points). This plot complements the earlier analysis of the barplot of correct selection, as it offers a deeper insight into how parameter estimation accuracy impacts the overall selection process.

Firstly, it is evident that the choice of covariance kernel (or covariance function) has a direct impact on both the correct selection of variables and the accuracy of parameter estimates. The Matern $\nu = 5/2$ kernel (represented in blue) consistently outperforms the other kernels, particularly the exponential kernel (in purple), which displays large and highly variable estimation errors across most parameters, particularly for Parameters 1, 2, and 3. This aligns with our previous observation that the exponential kernel struggles to select the correct variables effectively, which can be attributed to its poor parameter estimation performance. The Gaussian kernel (in red) performs better than the exponential, but it still exhibits higher error variability compared to Matern $\nu = 5/2$, especially for small values of m.

As m increases, we see a clear reduction in estimation error for the Gaussian and Matern kernels, which leads to improved variable selection accuracy as shown in the barplot of correct selection. In particular, for Parameters 1 and 2, which show greater variability in the estimation error, the correct selection of these variables improves significantly with the Matern kernel as more points are used. This reinforces the idea that accurate parameter estimation is crucial for the correct selection of influential variables, particularly when their effect sizes are relatively small.

On the other hand, the exponential kernel shows limited improvement as m increases, with

consistently large estimation errors and wider interquartile ranges. This suggests that the exponential kernel is not flexible enough to capture the underlying structure of the data, leading to both poor parameter estimation and suboptimal variable selection. This is consistent with the barplot analysis, where we noted that the exponential kernel fails to correctly select the influential variables, especially when they have weaker effects.

The lower estimation errors for parameters 4, 5, and 6, particularly for the Matern $\nu = 5/2$ kernel, suggest that these parameters are easier to estimate, which is reflected in the better variable selection for these covariates as seen in the barplot. The stability of the Matern kernel across different parameter settings reinforces its suitability for Kriging, particularly in cases where precision in both variable selection and parameter estimation is required.

This boxplot provides strong evidence that the choice of covariance kernel is a critical factor in both parameter estimation and variable selection accuracy. The Matern $\nu = 5/2$ kernel shows the best performance, with smaller estimation errors and less variability, leading to better variable selection outcomes as m increases. The poor performance of the exponential kernel highlights the need for careful kernel selection, as inadequate choices can result in both large estimation errors and incorrect variable selection, especially in complex models where some variables have weaker effects.

It's important to clarify that these parameter estimates are obtained using the maximum a posteriori (MAP) method, not maximum likelihood estimation (MLE). SAEMVS relies on MAP estimates, which take into account prior information, potentially leading to more robust parameter estimates in the context of sparse data. The MAP method can provide a more stable solution in variable selection, especially in cases where traditional MLE might struggle due to limited data or high-dimensional spaces.

Three-variable bar plot: Figure 12 offers insights into the effect of covariates with varying coefficients. In this experiment, the covariates are not equally prominent, as their coefficients differ significantly in magnitude. Covariates with larger coefficients carry more information, making them easier to select. As expected, these high-information covariates are reliably selected even with a smaller number of points.

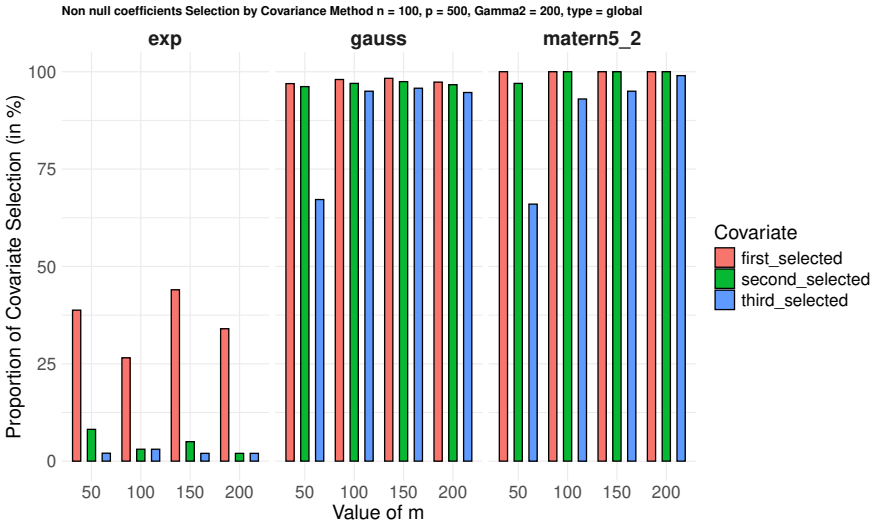

Figure 12: Bar plot of selected non-zero coefficients for $n = 100$, $p = 500$, $\Gamma^2 = 200$, global Kriging.

Conversely, covariates with smaller coefficients, despite being influential, are harder to detect with fewer points. This difficulty arises because their contribution to the response is relatively subtle, making them less distinguishable when the sample size is small. However, as the number of points increases, the selection process improves significantly, and the model is able to select these weaker covariates more frequently. We see this in the increasing height of the selection bars, which approach nearly 100% as m increases. This illustrates that with more data, the metamodel becomes increasingly reliable in identifying even subtle effects in the covariates.

4.3.2 Analysis of time kriging results

In the Time Kriging approach, the experimental design differs significantly from Global Kriging, primarily due to how the data is partitioned across time points. For smaller values of m, the model has fewer training points, which impacts its ability to generalize and make accurate predictions. For instance, with $m = 50$, where 80% of the data is used for training and 20% for testing, each time point $j \in \{1, ..., J\}$ only has around 4 points for training. Naturally, this limitation severely affects the model's performance.

Impact of limited training data: As a direct consequence, Time Kriging struggles with limited training data, particularly for smaller m values. This leads to a worst performance in the selection of non-null effect variables and higher prediction errors. The model simply doesn't have enough data to accurately capture the relationships between the covariates, especially when these relationships are subtle.

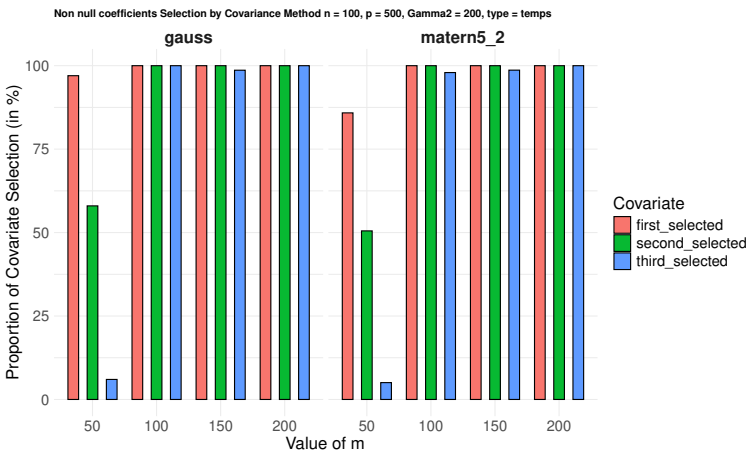

Figure 13: Bar plot of selected non-zero coefficients for $n = 100$, $p = 500$, $\Gamma^2 = 200$, Time Kriging.

Looking at Figure 13, the model's ability to correctly select non-null coefficients significantly declines when the available training data is limited. This effect is most pronounced at lower values of m, where the model has very few points to learn from, leading to a drop in performance. However, we see that as m increases, the selection improves significantly, even for $m = 100$.

4.3.3 Comparison between Time Kriging and Global Kriging

When we compare this with Global Kriging, several differences emerge. Global Kriging benefits from having access to the entire dataset for training, which allows it to perform better, even

with smaller m values. Time Kriging, on the other hand, suffers from fewer points per time slice. However, as m increases, Time Kriging begins to show distinct advantages over Global Kriging. This improvement occurs because Time Kriging fixes the time dimension, allowing the model to focus entirely on the design over φ, which enhances the accuracy of the variable selection process.

In Figure 14, the results show that for smaller m, the correct model selection accuracy in Time Kriging is notably lower than in Global Kriging. However, as m increases, the performance of Time Kriging not only catches up to Global Kriging but starts to outperform it.

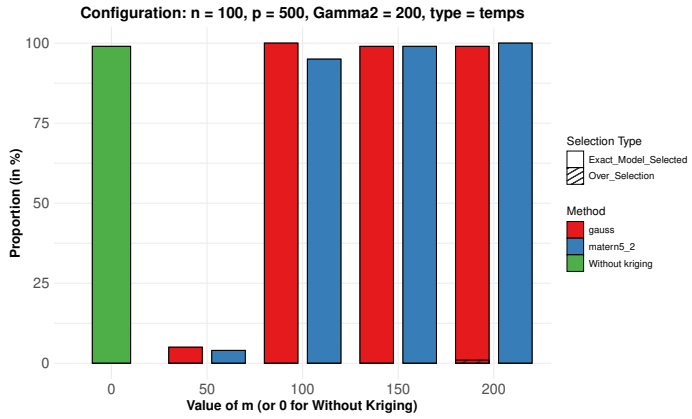

Figure 14: Bar plot of correct selected models for $n = 100$, $p = 500$, $\Gamma^2 = 200$, Time Kriging.

In Figure 15, for smaller m, the predictive error is relatively high compared to the results obtained using Global Kriging. But as m increases, the predictive error for Time Kriging reduces dramatically, reflecting the model's ability to focus better on the covariates and improve its parameter estimation capabilities.

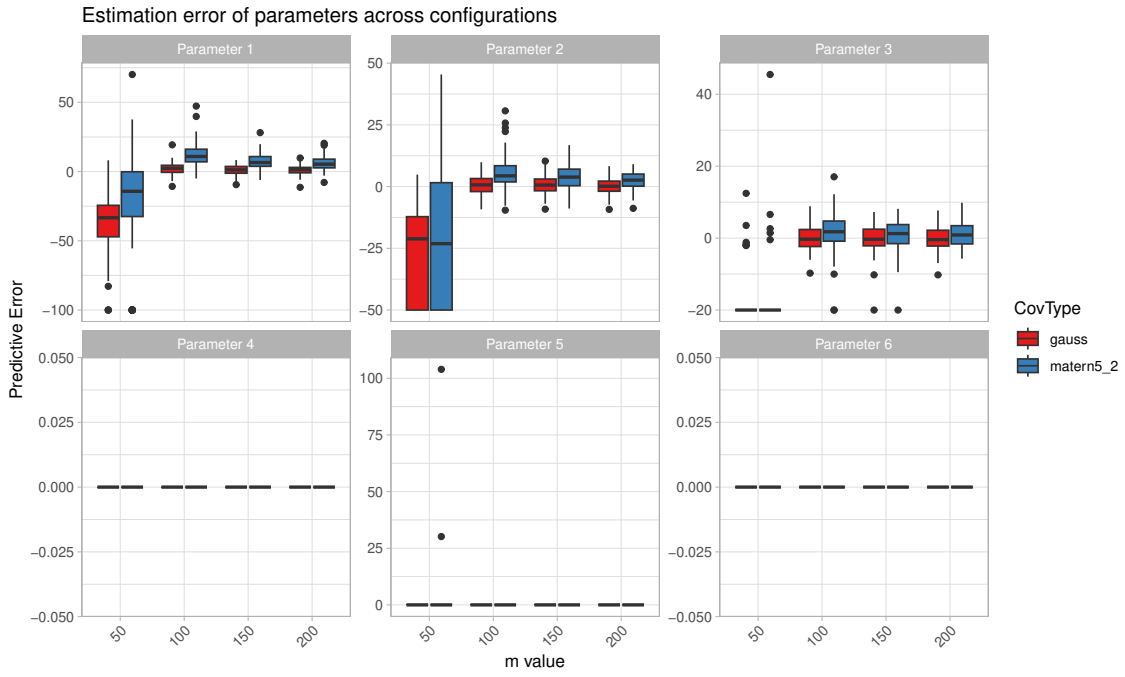

Figure 15: Boxplot of estimation error of parameters β_j for $\{j \in \{1, ..., -\}$ with $n = 100$, $p = 500$, $\Gamma^2 = 200$ by Time Kriging.

4.3.4 Impact of RMSE, MAE, and R^2 on Variable Selection in Kriging Model

In this part, we analyze the performance of our results using the Gauss kernel with parameters $n = 100$, $p = 500$, $\Gamma_2 = 200$, and $m = 200$. We chose the Gauss kernel because it did not perform as well as the `Mater5_2` method to look up the impact of Kriging in SAEMVS.

The results are summarized in Table 1, which reports the RMSE, MAE, R^2, and variable selection metrics for various simulated datasets. These metrics provide insights into the predictive performance of the model and its effectiveness in selecting the correct variables.

Table 1: RMSE, MAE, R^2, and Selection Variables in Model Performance for $n = 100$, $p = 500$, $\Gamma_2 = 200$, $m = 50$, and Gauss kernel

s	**RMSE**	**MAE**	R^2	**Sens.**	**Spec.**	**Exact Sel.**	**Corr. Inc.**	**Over Sel.**	**Model Sel. Kriging**	**SAEMVS Non-Kriging**
31	49.51	39.80	-0.001	1.000	0.994	0	1	1	1, 2, 3, 7, 8, 9	1, 2, 3
41	55.86	44.18	-0.003	0.000	0.994	0	0	0	6, 7, 10	1, 2, 3
57	50.33	40.16	-0.005	0.000	1	0	0	0	9	1, 2, 3
84	49.45	38.96	-0.014	1.000	0.994	0	1	1	1, 2, 3, 5, 7, 10	1, 2, 3
6	55.49	42.62	-0.025	1.000	0.996	0	1	1	1, 2, 3, 5, 10	1, 2, 3
14	53.67	43.29	-0.038	0.667	0.996	0	0	0	1, 2, 5, 8, 9, 10	1, 2, 3
19	62.79	47.58	-0.080	0.000	0.992	0	0	0	8, 10	1, 2, 3
94	45.99	38.87	-0.089	0.333	0.992	0	0	0	1, 5, 6, 7, 10	1, 2, 3
90	59.57	44.57	-0.119	0.333	0.996	0	0	0	1, 4	1, 2, 3
73	0.42	0.08	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
80	0.23	0.05	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
50	0.17	0.03	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
55	0.20	0.04	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
74	0.10	0.03	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
2	0.08	0.02	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3
72	0.08	0.01	0.999	0.667	1.000	0	0	0	1, 2	1, 2
65	0.07	0.02	0.999	1.000	1.000	1	1	0	1, 2, 3	1, 2, 3

The metrics presented in the columns: **RMSE**, **MAE**, R^2, **Sensitivity**, **Specificity**, **Exact Selection**, **Correctly Incorrect**, **Over Selection**, and **Model Selection Kriging** are the metrics derived from applying Kriging to SAEMVS. In contrast, the column **SAEMVS Non-Kriging** reflects the variables selected by the original SAEMVS method without the use of a metamodel.

The table columns are interpreted as follows:

- **s**: Index of each simulated dataset.
- **RMSE** (Root Mean Squared Error): Measures the square root of the average squared difference between predicted and actual values. Lower values indicate better prediction accuracy.
- **MAE** (Mean Absolute Error): Represents the average magnitude of prediction errors. As with RMSE, lower values correspond to more accurate predictions.
- R^2 (Coefficient of Determination): Indicates how well the model explains the variability of the data. Values close to 1 imply a good fit, whereas negative values suggest the model performs worse than a mean-based predictor.
- **Sensitivity** (Sens.): Reflects the model's ability to correctly identify true positives.
- **Specificity** (Spec.): Captures the model's capacity to correctly identify true negatives, avoiding false positives.
- **Exact Selection** (Exact Sel.): A binary indicator of whether the model successfully selected the exact correct variables.
- **Correctly Incorrect** (Corr. Inc.): Denotes whether the correct variables are included in the model, even if the selection is not exact.

- **Over Selection** (Over Sel.): Indicates whether the model has over-selected variables, meaning more variables than necessary were chosen.
- **Model Selection** (Model Sel.): Lists the variables selected by the Kriging model.
- **SAEMVS Non-Kriging**: Variables selected by the SAEMVS method without using Kriging.

Experiments such as $s = 31, 41, 57, 84, 6, 14, 19, 94$ and $s = 90$ show high RMSE and MAE values, alongside negative R^2 scores. The negative R^2 indicates that these models perform poorly. Such models consistently fail to select the correct or exact variables, as indicated by zeros in the "Exact Sel." or "Corr. Inc." columns. In examining this models with poor performance, we observe that in seeds $s = 31, 57, 84$, the model **overselects** variables, this means it includes irrelevant variables along with some of the correct ones. While these models show high sensitivity due to selecting true positives, their specificity is reduced because of the inclusion of false positives. In seeds $s = 41, 57, 19$, the model **incorrectly selects** the covariates and does not select any true values. The failure to select the correct variables evidence the limited utility of these models with bad metrics in practice.

We also observe that when the Kriging model has good metrics, it behaves the same as the SAEMVS without Kriging (SAEMVS Real). In experiments with $s = 73, 80, 50, 55, 74, 2, 65$, the models exhibit the lowest RMSE and MAE values, along with R^2 values close to 1, indicating a high level of predictive precision. They also successfully select the exact correct variables, as reflected by ones in the "Exact Sel." and "Corr. Inc." columns as the real SAEMVS.

It is important to note that the model indexed by $s = 72$ is under-selecting variables, particularly the third variable, which is known to have the lowest effect. This under-selection could be attributed to the SAEMVS algorithm nature, which may prune weaker variables.

All this analysis suggest that the incorporation of the Kriging model does not negatively affect the performance of SAEMVS when the metrics are favorable. Therefore, under optimal conditions, the model with Kriging can match the performance of the SAEMVS without Kriging in terms of predictive accuracy and effectiveness in variable selection.

5 Conclusion and Discussion

In this study, we looking to address a fundamental question: Can we use a metamodel to perform variable selection without degrading the results? Through extensive simulations and analyses, we explored the integration of Kriging metamodeling into the SAEMVS algorithm for variable selection within nonlinear mixed-effects models. Our findings indicate that metamodeling can indeed be effectively utilized for variable selection without significant loss of performance, provided certain conditions are met.

Our simulation results demonstrated that the SAEMVS algorithm, when coupled with Kriging metamodeling, maintains a high level of accuracy in variable selection. Specifically, we observed that as the number of design points (m) used in the Kriging metamodel increases, the performance of variable selection approaches that of the SAEMVS algorithm applied directly to the exact model without metamodeling. This suggests that metamodeling does not inherently degrade the results of variable selection and can be a viable tool in this context.

The effectiveness of using a metamodel for variable selection is contingent upon several key conditions:

Number of Design Points (m)

A sufficient number of design points is critical for the Kriging metamodel to accurately approximate the underlying function g. Our simulations showed that with a small m, the metamodel struggled to capture the complex relationships between covariates and the response variable, leading to decreased performance in variable selection. As m increased, the metamodel's accuracy improved, enhancing the variable selection process. Therefore, practitioners should ensure that an adequate number of design points are included in the experimental design to achieve reliable results.

An important aspect to consider is the scale of the parameter space over which the Latin Hypercube Sampling (LHS) was performed in our experiments. We designed our LHS over a wide interval to ensure comprehensive coverage of the input space. However, in cases where the parameters of interest lie within smaller intervals, it may be feasible to reduce the number of design points (m) required for effective metamodeling. A smaller parameter space would inherently require fewer points to achieve adequate coverage, potentially enhancing computational efficiency without compromising the accuracy of the metamodel. Future work could explore the impact of parameter space scaling on the performance of the LHS design and determine optimal strategies for different model complexities and parameter ranges.

Choice of covariance kernel

The selection of the covariance kernel in the Kriging model plays a pivotal role in the metamodel's performance. Our findings highlighted that kernels with greater flexibility and smoothness, such as the Matern $\nu = 5/2$ kernel, outperformed less flexible kernels like the exponential kernel. The Matern $\nu = 5/2$ kernel provided a better balance between smoothness and adaptability, allowing the metamodel to capture the intricate patterns in the data more effectively. This underscores the importance of choosing an appropriate covariance kernel tailored to the specific characteristics of the data and the underlying function g.

Experimental design considerations

The way in which the Latin Hypercube Sampling (LHS) is applied significantly affects the metamodel's performance. We investigated two approaches: Time Kriging and Global Kriging. In Time Kriging, separate Kriging models are constructed for each fixed time point, which can be advantageous in experiments where time points are predetermined and fixed. This method allows for a more focused modeling of covariate effects at specific times. Conversely, Global Kriging involves building a single Kriging model over both covariates and time, treating time as a continuous variable. While Global Kriging reduces computational overhead by training only one model, it may sacrifice some local temporal accuracy, especially in scenarios where temporal dynamics are complex.

RMSE, MAE, and R^2 considerations

The analysis presented underscores the critical relationship between RMSE, MAE, and R^2 in model evaluation. Models with lower RMSE and MAE values consistently outperform those with higher values, not only in terms of prediction accuracy but also in their ability to select

the correct variables. The models with the highest R^2 values—those closest to 1 are the most reliable, both in terms of their predictive performance and their capacity to include and identify the correct variables.

5.1 Key considerations for practitioners

For practitioners, such as biologists seeking to apply this methodology, several considerations are important. Determining the number of design points (m) is crucial, as the optimal number depends on factors like model complexity, dimensionality of the covariate space, and computational resources. A balance should be achieved between sufficient design points to capture the function g and the feasibility of the metamodeling process. Additionally, selecting an appropriate covariance kernel is key to the Kriging model's success, though experimenting with other kernels, such as the Gaussian, may yield better results depending on the data. While metamodeling introduces computational overhead, particularly in simpler cases where direct model evaluation is faster, its potential benefits in more complex, high-dimensional scenarios remain to be explored, especially in terms of computational savings for variable selection. Ultimately, selecting models with low RMSE and MAE values, combined with high R^2, leads to optimal performance in the context of Kriging. These models should be prioritized for further analysis and application in practical scenarios.

5.2 Limitations of the study and future work

It is important to acknowledge the limitations of our study. The simulations were conducted using a simple logistic growth regression model, where the exact evaluation of the logistic growth model was computationally trivial. Consequently, the computational benefits of metamodeling were not realized in this context, and the use of metamodeling actually increased computation time due to the overhead of constructing the Kriging model. However, this limitation is acceptable in the scope of our study because the primary objective was to understand the behavior of the SAEMVS method when coupled with metamodeling in a controlled setting.

Moreover, we did not extend our analysis to real data or more complex models where the evaluation of g is computationally expensive. In such scenarios, the benefits of metamodeling are expected to be more pronounced, potentially reducing computation time significantly and enabling the handling of more complex variable selection problems.

Future research should focus on applying the SAEMVS algorithm with Kriging metamodeling to more complex models and real datasets. This would provide a better understanding of the practical benefits and challenges of metamodeling in variable selection. Additionally, exploring adaptive methods for selecting the number of design points and optimizing the covariance kernel selection based on the data characteristics could further enhance the applicability of this methodology.

To facilitate reproducibility and allow others to replicate our results, the complete and well-documented code used in this study is publicly available on the Stat4Plant GitHub repository. The repository includes all scripts, data generation processes, and instructions necessary to reproduce the simulations and figures presented in this document. Researchers and practitioners are encouraged to access the code to explore the implementation details, run the simulations, and adapt the code to their own data and models.

A EM algorithm and tts extensions

A.1 Mild regularity conditions

The "mild regularity conditions" referred to in the convergence theorems are specific assumptions that ensure the EM algorithm's convergence to a stationary point of the observed-data log-likelihood function. These conditions are typically satisfied in practical applications and are outlined as follows:

1. **Closed Parameter Space**:
   - The parameter space Θ is a closed subset of $\mathbb{R}^p$. This ensures that limit points of sequences within Θ also belong to Θ.
2. **Continuity and Differentiability**:
   - The observed-data log-likelihood function $\log p_\theta(y)$ is continuous and differentiable with respect to θ over Θ.
   - The complete-data log-likelihood function $\log p_\theta(y, \varphi)$ is continuous with respect to θ for each (y, φ).
3. **Support Condition**:
   - The support of the conditional distribution $p_\theta(\varphi \mid y)$ does not depend on θ. That is, the set of possible values for the latent variables φ given the observed data y is the same for all $\theta \in \Theta$.
4. **Well-Defined Expectations**:
   - The expected value $Q(\theta \mid \theta^{(k)}) = \mathbb{E}_{\varphi|y,\theta^{(k)}}[\log p_\theta(y, \varphi)]$ is well-defined and finite for all $\theta \in \Theta$ and all iterations k.
5. **Monotonicity and Convergence**:
   - The EM algorithm increases the observed-data log-likelihood at each iteration unless it has already reached a stationary point.
   - Any limit point θ^* of the sequence $\{\theta^{(k)}\}$ is a stationary point of the observed-data log-likelihood function $\log p_\theta(y)$.
6. **Differentiability of the Q Function**:
   - The function $Q(\theta \mid \theta^{(k)})$ is continuous and differentiable with respect to θ for all $\theta \in \Theta$ and $k \geq 0$.

These regularity conditions are essential for the proofs of the convergence theorems presented earlier. Specifically, they allow us to apply results from optimization theory to establish that:

- The sequence $\{\theta^{(k)}\}$ generated by the EM algorithm remains within the parameter space Θ.
- The observed-data log-likelihood function $\log p_\theta(y)$ behaves smoothly enough for the algorithm to make consistent progress toward a stationary point.
- The support condition ensures that the E-step computes expectations over a consistent domain, which is crucial for the monotonic increase of the likelihood.

Under these conditions, the EM algorithm is guaranteed to converge to a stationary point of the observed-data log-likelihood function, which may be a local maximum or a saddle point. It is important to note that global convergence to the absolute maximum of the likelihood function is not guaranteed without additional assumptions [Wu, 1983].

In practical applications, verifying these regularity conditions involves checking that:

- The parameter estimates remain within a reasonable and predefined range.
- The likelihood functions are well-behaved (continuous and differentiable) with respect to the parameters.
- The latent variables' support does not change with different parameter values.

A.2 Convergence theorems of the EM algorithm

Theorem 2 (Monotonicity of the likelihood [Dempster et al., 1977])**.** *At each iteration of the EM algorithm, the observed-data log-likelihood does not decrease:*

$$\log p_{\theta^{(k+1)}}(y) \geq \log p_{\theta^{(k)}}(y).$$

Proof. This property follows from the construction of the Q function and the use of Jensen's inequality in the E-step. Specifically, the EM algorithm ensures that

$$\log p_{\theta^{(k+1)}}(y) - \log p_{\theta^{(k)}}(y) \geq Q(\theta^{(k+1)} \mid \theta^{(k)}) - Q(\theta^{(k)} \mid \theta^{(k)}) \geq 0.$$

□

Theorem 3 (Convergence to stationary points [Wu, 1983])**.** *Under mild regularity conditions, any limit point θ^* of the sequence $\{\theta^{(k)}\}$ generated by the EM algorithm is a stationary point of the likelihood function $\log p_\theta(y)$, meaning:*

$$\nabla_\theta \log p_\theta(y)\big|_{\theta=\theta^*} = 0.$$

These theorems provide a theoretical foundation for the convergence behavior of the EM algorithm:

- **Monotonicity**: Ensures that each iteration does not decrease the likelihood, promoting consistent progress toward maximizing $\log p_\theta(y)$.
- **Convergence to stationary points**: Guarantees that the algorithm converges to a point where the gradient of the likelihood function is zero.

A.3 About the Evidence Lower Bound (ELBO)

An alternative perspective on the EM algorithm involves the Evidence Lower Bound (ELBO), which provides further insight into the algorithm's convergence properties and its connection to variational inference.

The observed-data log-likelihood can be decomposed using any distribution $q(\varphi)$ over the latent variables φ:

$$\begin{aligned}
\log p_\theta(y) &= \int q(\varphi) \log p_\theta(y)\, d\varphi \\
&= \int q(\varphi) \log \left(\frac{p_\theta(y, \varphi)}{p_\theta(\varphi \mid y)} \right) d\varphi \\
&= \int q(\varphi) \left[\log p_\theta(y, \varphi) - \log p_\theta(\varphi \mid y) \right] d\varphi \\
&= \underbrace{\int q(\varphi) \log p_\theta(y, \varphi)\, d\varphi - \int q(\varphi) \log q(\varphi)\, d\varphi}_{\mathcal{L}(q,\theta)} + \underbrace{\int q(\varphi) \log \frac{q(\varphi)}{p_\theta(\varphi \mid y)}\, d\varphi}_{\mathrm{KL}(q(\varphi) \| p_\theta(\varphi|y))}.
\end{aligned}$$

Rewriting, we have:

$$\log p_\theta(y) = \mathcal{L}(q, \theta) + \mathrm{KL}(q(\varphi) \parallel p_\theta(\varphi \mid y)),$$

where:

- $\mathcal{L}(q, \theta) = \int q(\varphi) \log \frac{p_\theta(y,\varphi)}{q(\varphi)}\, d\varphi$ is the **Evidence Lower Bound (ELBO)**.
- $\mathrm{KL}(q(\varphi) \parallel p_\theta(\varphi \mid y))$ is the Kullback-Leibler divergence between $q(\varphi)$ and the true posterior $p_\theta(\varphi \mid y)$.

Since the KL divergence is always non-negative, the ELBO serves as a lower bound on the observed-data log-likelihood:

$$\mathcal{L}(q, \theta) \leq \log p_\theta(y).$$

The EM algorithm can be interpreted as iteratively maximizing the ELBO with respect to $q(\varphi)$ and θ:

1. **E-step (Maximize w.r.t. $q(\varphi)$)**:

   $$q^{(k+1)}(\varphi) = \operatorname*{argmax}_{q(\varphi)} \mathcal{L}(q, \theta^{(k)}) = p_{\theta^{(k)}}(\varphi \mid y).$$

   In this step, setting $q(\varphi)$ to the true posterior distribution maximizes the ELBO because it minimizes the KL divergence (which becomes zero).
2. **M-step (Maximize w.r.t. θ)**:

   $$\theta^{(k+1)} = \operatorname*{argmax}_{\theta \in \Theta} \mathcal{L}(q^{(k+1)}, \theta).$$

   Here, we maximize the ELBO with respect to θ while keeping $q^{(k+1)}(\varphi)$ fixed.

By alternately maximizing the ELBO with respect to $q(\varphi)$ and θ, the EM algorithm ensures that the observed-data log-likelihood $\log p_\theta(y)$ does not decrease at each iteration.

The increase in the observed-data log-likelihood can be understood through the ELBO:

$$\begin{aligned}
\log p_{\theta^{(k+1)}}(y) &= \mathcal{L}(q^{(k+1)}, \theta^{(k+1)}) + \mathrm{KL}(q^{(k+1)}(\varphi) \parallel p_{\theta^{(k+1)}}(\varphi \mid y)) \\
&\geq \mathcal{L}(q^{(k+1)}, \theta^{(k+1)}) \\
&\geq \mathcal{L}(q^{(k+1)}, \theta^{(k)}) \\
&= \int p_{\theta^{(k)}}(\varphi \mid y) \log \frac{p_{\theta^{(k)}}(y, \varphi)}{p_{\theta^{(k)}}(\varphi \mid y)} \, d\varphi \\
&= \log p_{\theta^{(k)}}(y).
\end{aligned}$$

This chain of inequalities shows that:

$$\log p_{\theta^{(k+1)}}(y) \geq \log p_{\theta^{(k)}}(y),$$

confirming the monotonic increase of the observed-data log-likelihood at each iteration [Neal and Hinton, 1998].

A.4 Convergence of SAEM

Theorem 4 (Convergence of SAEM Delyon et al. [1999])**.** *Assume that:*

(a) The complete-data model $p_\theta(y, \varphi)$ belongs to the curved exponential family, i.e.,

$$p_\theta(y, \varphi) = h(y, \varphi) \exp\left(\langle S(y, \varphi), \phi(\theta) \rangle - \psi(\theta)\right),$$

where $S(y, \varphi)$ are sufficient statistics, $\phi(\theta)$ are natural parameters, and $\psi(\theta)$ is the log-partition function.

(b) The parameter space Θ is an open subset of $\mathbb{R}^d$.

(c) The function $\psi(\theta)$ is twice continuously differentiable on Θ.

(d) The sequence of step sizes $\{\gamma_k\}$ satisfies:

$$\sum_{k=1}^{\infty} \gamma_k = \infty, \quad \sum_{k=1}^{\infty} \gamma_k^2 < \infty, \quad 0 < \gamma_k \leq 1.$$

(e) The observations y are such that the log-likelihood function $\ell(\theta) = \log p_\theta(y)$ is continuous and has isolated critical points.

Then, the sequence $\{\theta^{(k)}\}$ generated by the SAEM algorithm converges almost surely to a (local) maximum of the observed-data log-likelihood $\ell(\theta)$.

Sketch of Proof. The proof relies on viewing the SAEM algorithm as a stochastic approximation algorithm for estimating the maximum likelihood estimator (MLE). The main steps are:

1. **Representation as a stochastic approximation:** The SAEM update can be written as a stochastic approximation of the form:
$$\theta^{(k+1)} = \theta^{(k)} + \gamma_k H(\theta^{(k)}, S^{(k)}),$$
where $S^{(k)}$ is a realization of the sufficient statistics at iteration k, and $H(\theta, S)$ is a function guiding the updates.
2. **Asymptotic behavior:** Under the given conditions, it can be shown that the noise in the stochastic approximation diminishes appropriately, and the algorithm behaves like a deterministic ordinary differential equation (ODE) in the limit.
3. **Stability and convergence:** By applying the Robbins-Monro stochastic approximation theory and the ODE method [Kushner and Yin, 2003], it can be established that $\{\theta^{(k)}\}$ converges almost surely to a stationary point of $\ell(\theta)$.
4. **Convergence to Local Maximum:** Given that the critical points of $\ell(\theta)$ are isolated and that the algorithm increases the likelihood, the limit point must be a local maximum.

For a detailed proof, see [Delyon et al., 1999]. □

A.5 Convergence of MCMC-SAEM

Theorem 5 (Convergence of MCMC-SAEM [Kuhn and Lavielle, 2004])**.** *Assume that:*

(a) The conditions of Theorem 4 hold for the SAEM algorithm.

(b) The Markov chain used in the S-step is uniformly ergodic, i.e., there exists a constant $\rho \in (0,1)$ *such that for any initial state* $\varphi^{(0)}$*,*

$$\left\| P^n(\varphi^{(0)}, \cdot) - p_{\theta^{(k)}}(\varphi \mid y) \right\|_{TV} \leq C\rho^n,$$

where P^n *is the n-step transition kernel,* C *is a constant, and* $\|\cdot\|_{TV}$ *denotes the total variation norm.*

Then, the sequence $\{\theta^{(k)}\}$ *generated by the MCMC-SAEM algorithm converges almost surely to a (local) maximum of the observed-data log-likelihood function* $\ell(\theta)$*.*

A detailed proof can be found in [Kuhn and Lavielle, 2004].

B Kernel Functions in the the `DiceKriging` Package

Table 2: Examples of Kernel Functions in the `DiceKriging` Package

Kernel	**Expression**	**Parameters**
Gaussian	$K(x,y) = \sigma^2 \exp\left(-\frac{\|x-y\|^2}{2l^2}\right)$	σ^2, l
Exponential	$K(x,y) = \sigma^2 \exp\left(-\frac{\|x-y\|}{l}\right)$	σ^2, l
Matern 3/2	$K(x,y) = \sigma^2 \left(1+\sqrt{3}\frac{\|x-y\|}{l}\right) \exp\left(-\sqrt{3}\frac{\|x-y\|}{l}\right)$	σ^2, l
Matern 5/2	$K(x,y) = \sigma^2 \left(1+\sqrt{5}\frac{\|x-y\|}{l}+\frac{5\|x-y\|^2}{3l^2}\right) \exp\left(-\sqrt{5}\frac{\|x-y\|}{l}\right)$	σ^2, l

The parameters σ^2 and l play crucial roles in defining the behavior of each kernel function. Here, σ^2 represents the variance, controlling the overall variability or amplitude of the Gaussian Process. The length-scale parameter l determines how quickly the correlation between points decays as the distance $\|x-y\|$ increases. A smaller l implies that the function can vary more rapidly, capturing finer details, whereas a larger l leads to smoother and more gradual changes [Roustant et al., 2012].

Regarding the types of covariance functions:

- **Gaussian Kernel**: Also known as the Radial Basis Function (RBF), it is infinitely differentiable, making it suitable for modeling very smooth functions.
- **Exponential Kernel**: Less smooth than the Gaussian kernel, it models functions that are continuous but not differentiable.
- **Matern 3/2 Kernel**: Provides a balance between smoothness and flexibility, assuming the underlying function is once differentiable.
- **Matern 5/2 Kernel**: Allows for twice differentiable functions, offering more flexibility in capturing the function's behavior compared to Matern 3/2.

These covariance functions enable Gaussian Processes to model a wide range of function behaviors by adjusting the smoothness and correlation structure through their parameters.

C SAEMVS with Kriging implementation code in R

C.1 Code for Kriging by time

The following R code sets up and runs simulations for the SAEMVS algorithm with Kriging metamodeling. Below, we explain each part of the code, describe what it does, and outline the steps involved.

C.1.1 Initialization and Parameter Setting

Sets the seed based on a command-line argument:

```
s_setparam = as.numeric(commandArgs(trailingOnly=TRUE)[1]) # Seed
s = s_setparam
```

Defining Parameters for Data Generation

```
n <- 200 # Number of individuals
J <- 10 # Number of repetitions
p <- 2000 # Number of covariates
Gamma2 <- 200 # Inter-individual variance of phi_i
sigma2 <- 30 # Residual variance of y
mu <- 1200 # Intercept
beta <- c(100, 50, 20, rep(0, p - 3)) # Covariate fixed effects vector
beta_tilde <- c(mu, beta) # Useful reformulation
psi1 <- 200 # Parameter of the function g
psi2 <- 300 # Parameter of the function g
psi <- c(psi1, psi2)
time_points <- seq(150, 3000, length.out = J)
```

The function g is implemented as:

```
g <- function(phi_i, psi, t_ij) {
  return(psi[1] / (1 + exp(-(t_ij - phi_i) / psi[2])))
}
```

Parameters for the SAEMVS algorithm are set:

```
# SAEMVS algorithm parameters
niter <- 500          # Number of iterations
nburnin <- 350        # Number of burn-in iterations
niterMH_phi <- 1      # Number of Metropolis-Hastings iterations for phi
M <- 20               # Number of grid points for nu0
Delta <- 10^(seq(-2, 2, length.out = M))  # Grid of nu0 values
```

Kriging Parameters Parameters for the Kriging metamodel are defined:

```
# Kriging parameters
m_values <- c(40, 50, 100, 150, 200)  # Number of design points for Kriging
cov_types <- c("gauss","matern5_2")  # Covariance functions to be tested
# Trend formulas for Kriging models
trend_formulas <- list(
  list(formule = ~1, trend = 1)  # Constant trend
)
trend_formule <- trend_formulas[[1]]$formule
trend <- trend_formulas[[1]]$trend
```

C.1.2 Data Simulation

The code simulates the dataset by generating random effects φ_i, covariate matrix V, and observations y_{ij}. Random effects $\varphi_i \sim \mathcal{N}(0, \Gamma^2)$, the covariate matrix V is generated with independent standard normal entries. Finally, observations y_{ij} are generated using the model:

$$y_{ij} = V_i^\top \beta + g(\varphi_i, \psi, t_{ij}) + \epsilon_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$$

C.1.3 Design of Experiments with Latin Hypercube Sampling (LHS)

The code creates a design of experiments using LHS to generate the design points for the Kriging metamodel.

```r
# Function to create an LHS design for all times
create_lhs_design_for_all_times <- function(m, time_points, seed = NULL) {
  if (!is.null(seed)) {
    set.seed(seed)
  }
  num_points_per_time = m / length(time_points)
  # Ensure the total number of points matches m
  total_points <- length(time_points) * num_points_per_time
  if (m != total_points) {
    stop("Total number of points does not match m.")
  }
  # Create LHS design for varphi
  lhs_design <- lhsDesign(n = m, dimension = 1, seed = seed)$design
  lhs_design <- data.frame(varphi = lhs_design[,1] * 1100)
  # Repeat time points
  time_values <- rep(time_points, each = num_points_per_time)
  # Associate time values with the LHS design
  design <- data.frame(varphi = lhs_design$varphi, t = time_values)
  return(design)
}
```

For each m in m_values, the design is created, and responses are computed using the function g.

C.1.4 Training and Validation Sets

The design and responses are split into training and validation sets using stratified sampling based on time points.

```r
lhs_design$unique_id <- seq_along(lhs_design$t)
set.seed(s)
split_data <- stratified(lhs_design, group = "t", size = 0.8, bothSets = TRUE)
train_design <- split_data$SAMP1
test_design <- split_data$SAMP2
# Generate responses for training and test sets
train_response <- sapply(1:nrow(train_design), function(i) {
  g(train_design$varphi[i], psi, train_design$t[i])
})
test_response <- sapply(1:nrow(test_design), function(i) {
  g(test_design$varphi[i], psi, test_design$t[i])
})
```

An additional validation set is generated with a new seed.

C.1.5 Kriging Model Training and Prediction

For each covariance type and trend formula, Kriging models are trained for each unique time point.

```r
for (covtype in cov_types) {
  for (k in seq_along(trend_formulas)) {
    # Initialize lists for storing predictions and models
    predictions_test <- c()
    predictions_val <- c()
    trend_formula <- trend_formulas[[k]]
    models_by_model_J <- list()

    # Loop over unique time points
    for (j in seq_along(unique_times)) {
      time <- unique_times[j]
      # Select training data for current time point
      train_design_time <- subset(train_design, t == time, select = varphi)
      indices_train_design_time <- as.numeric(unlist(subset(train_design, t == time,
          select = unique_id)))
      train_response_time <- lhs_response[indices_train_design_time]
```

```
16
17        # Train Kriging model
18        set.seed(m + s)
19        model <- km(
20          formula = trend_formula$formule,
21          design = train_design_time,
22          covtype = covtype,
23          response = train_response_time,
24          nugget = 1e-8
25        )
26
27        models_by_model_J[[j]] <- model
28
29        # Predictions on test and validation sets
30        test_design_temp <- subset(test_design, t == time, select = varphi)
31        predictions_test <- c(predictions_test,
32                              predict(model, newdata = test_design_temp,
33                                      type = "UK", bias.correct = TRUE)$mean)
34
35        val_design_temp <- subset(val_design, t == time, select = varphi)
36        predictions_val <- c(predictions_val,
37                             predict(model, newdata = val_design_temp,
38                                     type = "UK", bias.correct = TRUE)$mean)
39      }
40    }
41  }
```

C.1.6 SAEMVS with Kriging

The SAEMVS algorithm is run using the Kriging metamodel instead of directly evaluating g.

```
1  # Load the Kriging model
2  kriging_model <- get_model_by_triplet(charging_models$models, m_values[1],
3                                        cov_types[1], trend_formule)
4  # Initialize parameters for SAEMVS
5  betatilde_init <- c(1400, rep(100, 10), rep(1, p - 10)) # Initial beta_tilde
6  sigma2_init <- 100
7  Gamma2_init <- 5000
8  alpha_init <- 0.1
9  tau <- 0.98
10 param_init <- list(
11   beta_tilde = betatilde_init, alpha = alpha_init, Gamma2 = Gamma2_init,
12   sigma2 = sigma2_init
13 )
14 # Hyperparameters
15 nu0 <- 0.04
16 nu1 <- 12000
17 nu_Gamma <- 1
18 lb_Gamma <- 1
19 nu_sigma <- 1
20 lb_sigma <- 1
21 a <- 1
22 b <- p
23 sigma2_mu <- 3000^2
24 hyperparam <- list(
25   nu0 = nu0, nu1 = nu1, nu_Gamma = nu_Gamma, lb_Gamma = lb_Gamma,
26   nu_sigma = nu_sigma,
27   lb_sigma = lb_sigma,
28   a = a, b = b, sigma2_mu = sigma2_mu, psi = psi, tau = tau
29 )
30 # Run the SAEMVS algorithm with Kriging
```

```
results_Mod_select <- Model_selection_Kriging(Delta, niter, nburnin, niterMH_phi, data$Y,
    data$t, data$id, V_tilde, param_init, hyperparam, s, kriging_model)
```

C.1.7 Global Kriging Metamodel Construction

In the global Kriging approach, we model the function g as a function of both φ_i and t_{ij} simultaneously, rather than building separate models for each time point.

Design of Experiments with Latin Hypercube Sampling We use Latin Hypercube Sampling (LHS) to create the design points for the Kriging metamodel:

```
for (m in m_values) {
  set.seed(s)
  # Use LHS for training data
  lhs_design <- lhsDesign(n = m, dimension = 2, seed = s)$design
  lhs_design <- data.frame(varphi = lhs_design[,1]*1100,
                           t = lhs_design[,2] * (3000 - 150) + 150)
  lhs_response <- numeric(m)
  for (i in 1:m) {
    lhs_response[i] <- g(lhs_design$varphi[i], psi, lhs_design$t[i])
  }
```

Here, we sample φ_i and t_{ij} jointly over their ranges.

Splitting Data into Training and Test Sets The data is divided into training and test sets:

```
  # Split data into training and test sets
  train_indices <- sample(1:m, size = 0.8 * m)
  test_indices <- setdiff(1:m, train_indices)
  train_design <- lhs_design[train_indices, ]
  train_response <- lhs_response[train_indices]
  test_design <- lhs_design[test_indices, ]
  test_response <- lhs_response[test_indices]
```

Generating Validation Data An additional LHS design is created for validation:

```
  # Generate validation data
  m_val <- m
  lhs_val_design <- lhsDesign(n = m_val, dimension = 2, seed = s + 2)$design
  val_design <- data.frame(varphi = lhs_val_design[, 1]*1100,
                           t = lhs_val_design[, 2] * (3000 - 150) + 150)

  val_response <- numeric(m_val)
  for (i in 1:m_val) {
    val_response[i] <- g(val_design$varphi[i], psi, val_design$t[i])
  }
```

Training Kriging Models For each covariance type and trend formula, a Kriging model is trained using the training data:

```
  for (covtype in cov_types) {
    for (ii in seq_along(trend_formulas)) {
      trend_ii <- trend_formulas[[ii]]
      trend_formula <- trend_ii$formule
      trend_trend <- trend_ii$trend
      set.seed(m + s)
      # Fit Kriging model
      model <- km(formula = trend_formula, design = train_design,
                  covtype = covtype, response = train_response, nugget = 1e-6)
```

Running the SAEMVS Algorithm The SAEMVS algorithm is executed:

```
# Run the SAEMVS algorithm with Kriging
results_Mod_select <- Model_selection_Kriging(Delta, niter, nburnin, niterMH_phi, data$Y,
    data$t, data$id, V_tilde, param_init, hyperparam, s, m_values[1], cov_types[1],
    trend, kriging_model)
```

D Comparison of kriging predictions instead of using the real model

In this section, we focus on the specific differences in how the Kriging predictions replace the function $g(\varphi_i, \psi, t_{ij})$ in the SAEMVS procedure across the three scripts. We provide detailed explanations and highlight the changes in the code that implement this replacement.

D.1 Original Implementation in `Functions_SAEMVS.R`

In the original script, the function g is defined and used directly in the SAEMVS algorithm.

```
g <- function(phi_i, psi, t_ij) {
  return(psi[1] / (1 + exp(-(t_ij - phi_i) / psi[2])))
}
```

Within the `SAEM_MAP` function, g is called directly to compute the model predictions during the Metropolis-Hastings step and the calculation of sufficient statistics.

Metropolis-Hastings Step

```
# S-step: Metropolis Hastings within Gibbs
for (i in 1:n) {
  for (r in 1:niterMH_phi){
    phi_ci = rnorm(1, mean = Vbetak[i], sd = sqrt(Gamma2[k]))
    logratio = sum(dnorm(yi[,i], mean = g(phi_ci, psi, ti[,i]), sd = sqrt(sigma2[k]), log
        = TRUE)) -
          sum(dnorm(yi[,i], mean = g(phi_tilde[i,r], psi, ti[,i]), sd = sqrt(sigma2[
              k]), log = TRUE))
  }
}
```

Updating Sufficient Statistics

```
# SA-step: updating of sufficient statistics
for (i in 1:n) {
  mco[i, ] = (yi[, i] - g(phi[i, k + 1], psi, ti[, i]))^2
}
```

In both cases, g is evaluated directly for each individual and time point.

D.2 Modification in the original script for the global kriging

In this case, g is approximated using a Kriging metamodel that depends on both φ and t. The Kriging predictions replace direct evaluations of g.

Definition of Kriging Prediction Function A new function `predict_f_kriging` is defined to compute the Kriging predictions:

```
predict_f_kriging <- function(varphi, t, kriging_model) {
  newdata <- cbind(varphi, t)  # Combine varphi and t into a matrix
  predict(kriging_model, newdata = newdata, type = 'UK')$mean
}
```

This function takes φ and t as inputs and returns the predicted value from the Kriging model.

Replacement in Metropolis-Hastings Step In the `SAEM_MAP_Kriging` function, the direct evaluations of g are replaced with `predict_f_kriging`:

```
# S-step: Metropolis Hastings within Gibbs
for (i in 1:n) {
  for (r in 1:niterMH_phi) {
    phi_ci = rnorm(1, mean = Vbetak[i], sd = sqrt(Gamma2[k]))
    logratio = sum(dnorm(yi[, i], mean = predict_f_kriging(phi_ci, ti[, i], kriging_model
        ),
                         sd = sqrt(sigma2[k]), log = TRUE)) -
               sum(dnorm(yi[, i], mean = predict_f_kriging(phi_tilde[i, r], ti[, i],
                   kriging_model),
                         sd = sqrt(sigma2[k]), log = TRUE))
  }
}
```

Similarly, when updating the sufficient statistics:

```
# SA-step: updating of sufficient statistics
for (i in 1:n) {
  mco[i, ] = (yi[, i] - predict_f_kriging(phi[i, k + 1], ti[, i], kriging_model))^2
}
```

The residuals are computed using the Kriging predictions instead of direct evaluations of g.

D.3 Modification in the original script for the time kriging

In this script, time-dependent Kriging models are used. A separate Kriging model is trained for each time point, and the Kriging predictions depend only on φ.

A new function `predict_f_kriging_temps` is defined:

```
predict_f_kriging_temps <- function(varphi, kriging_model) {
  newdata <- data.frame(varphi = varphi)
  prediction <- predict(kriging_model, newdata = newdata, type = 'UK', bias.correct =
      TRUE)
  return(prediction$mean)
}
```

This function takes φ and a Kriging model corresponding to a specific time point and returns the predicted value.

Replacement in Metropolis-Hastings Step In the `SAEM_MAP` function, the Kriging predictions replace g during the Metropolis-Hastings step. Since there are separate Kriging models for each time point, we loop over the observations:

```
# S-step: Metropolis Hastings within Gibbs
for (i in 1:n) {
  for (r in 1:niterMH_phi) {
    phi_ci = rnorm(1, mean = Vbetak[i], sd = sqrt(Gamma2[k]))
    logratio = sum(sapply(1:J, function(j) {
      dnorm(yi[j, i],
            mean = predict_f_kriging_temps(phi_ci, kriging_model[[j]]),
            sd = sqrt(sigma2[k]), log = TRUE)
    })) - sum(sapply(1:J, function(j) {
      dnorm(yi[j, i],
            mean = predict_f_kriging_temps(phi_tilde[i, r], kriging_model[[j]]),
            sd = sqrt(sigma2[k]), log = TRUE)
    }))
  }
}
```

For each observation j, we use the corresponding Kriging model `kriging_model[[j]]` to predict the mean response given φ_i.

Replacement in Updating Sufficient Statistics When updating the sufficient statistics:

```
# SA-step: updating of sufficient statistics
for (i in 1:n) {
  mco[i, ] = sapply(1:J, function(j) {
    (yi[j, i] - predict_f_kriging_temps(phi[i, k + 1], kriging_model[[j]]))^2
  })
}
```

The residuals for each observation are computed using the Kriging predictions from the corresponding time-specific model.

References

Pierre Barbillon, Célia Barthélémy, and Adeline Samson. Parametric estimation of complex mixed models based on meta-model approach. arXiv preprint arXiv:2111.09516, November 2021. URL https://arxiv.org/abs/2111.09516.

George EP Box. Science and statistics. Journal of the American Statistical Association, 71(356):791–799, 1976.

Francois Brun, Daniel Wallach, David Makowski, and James W. Jones. Working with Dynamic Crop Models. Academic Press, third edition edition, 2019. ISBN 9780128117569. URL http://libgen.li/file.php?md5=65c1b2a685e0d0bfa6de76deab4ed833.

Noel Cressie. The origins of kriging. Mathematical Geology, 22(3):239–252, 1990.

Noel A. C. Cressie. Statistics for Spatial Data. John Wiley & Sons, New York, revised edition, 1993. ISBN 978-0-471-00255-0.

Bernard Delyon, Marc Lavielle, and Eric Moulines. Convergence of a stochastic approximation version of the em algorithm. Annals of Statistics, 27(1):94–128, 1999.

Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society: Series B (Methodological), 39(1):1–22, 1977.

K.-T. Fang, R. Li, and A. Sudjianto. Design and Modeling for Computer Experiments. Chapman & Hall/CRC, 2005.

Daniel Folashade. foreach: Provides Foreach Looping Construct, 2022. URL https://CRAN.R-project.org/package=foreach. R package version 1.5.2.

W. Keith Hastings. Monte carlo sampling methods using markov chains and their applications. Biometrika, 57 (1):97–109, 1970.

D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. Journal of the Chemical, Metallurgical and Mining Society of South Africa, 52(6):119–139, 1951.

E. Kuhn and M. Lavielle. Coupling a stochastic approximation version of em with an mcmc procedure. ESAIM: Probability and Statistics, 8:115–131, 2004.

Harold J. Kushner and G. George Yin. Stochastic Approximation and Recursive Algorithms and Applications. Springer, New York, 2nd edition, 2003.

Christos Lataniotis, Stefano Marelli, and Bruno Sudret. UQLAB User Manual – Kriging (Gaussian process modelling), 07 2015.

Marc Lavielle. Mixed Effects Models for the Population Approach: Models, Tasks, Methods and Tools. Taylor and Francis Group, 07 2014. ISBN 9781482226508. doi: 10.1201/b17203.

G. Matheron. Principles of geostatistics. Economic Geology, 58:1246–1266, 1963.

Matthew D McKay, Richard J Beckman, and William J Conover. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics, 21(2):239–245, 1979.

G.J. McLachlan and T. Krishnan. The EM Algorithm and Extensions. Wiley, Hoboken, NJ, 2007.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. The Journal of Chemical Physics, 21(6):1087–1092, 1953.

T.J. Mitchell and J.J. Beauchamp. Bayesian variable selection in linear regression. Journal of the American Statistical Association, 83(404):1023–1032, 1988.

M. Naveau, G. Kon Kam King, R. Rincent, et al. Bayesian high-dimensional covariate selection in non-linear mixed-effects models using the saem algorithm. Statistical Computing, 34:53, 2024. doi: 10.1007/s11222-023-10367-4. URL https://doi.org/10.1007/s11222-023-10367-4.

Radford M. Neal and Geoffrey E. Hinton. A View of the Em Algorithm that Justifies Incremental, Sparse, and other Variants, pages 355–368. Springer Netherlands, Dordrecht, 1998. ISBN 978-94-011-5014-9. doi: 10.1007/978-94-011-5014-9_12. URL https://doi.org/10.1007/978-94-011-5014-9_12.

José C Pinheiro and Douglas M Bates. Mixed-effects Models in S and S-PLUS. Springer, New York, 2000. ISBN 978-1-4419-0318-1.

Carl Edward Rasmussen and Christopher KI Williams. Gaussian processes for machine learning. MIT Press, 2006.

M. Reynolds, S. Chapman, L. Crespo-Herrera, G. Molero, S. Mondal, D. N. L. Pequeno, F. Pinto, F. J. Pinera-Chavez, J. Poland, C. Rivera-Amado, C. Saint Pierre, and S. Sukumaran. Breeder friendly phenotyping. Plant Science, 295:110396, June 2020. doi: 10.1016/j.plantsci.2019.110396. URL https://doi.org/10.1016/j.plantsci.2019.110396. Epub 2020 Jan 18.

Christian P. Robert and George Casella. Monte Carlo Statistical Methods. Springer, New York, 2nd edition, 2004.

Olivier Roustant. Metamodeling with gaussian processes. Online, 2023. URL https://olivier-roustant.fr/wp-content/uploads/2020/05/gp_talk.pdf. Lecture notes, September 2023.

Olivier Roustant, David Ginsbourger, and Yves Deville. Dicekriging, diceoptim: Two r packages for the analysis of computer experiments by kriging-based metamodeling and optimization. Journal of Statistical Software, 51 (1):1–55, 2012. doi: 10.18637/jss.v051.i01. URL https://www.jstatsoft.org/index.php/jss/article/view/v051i01.

Veronika Ročková and Edward I. George. Emvs: The em approach to bayesian variable selection. Journal of the American Statistical Association, 109(506):828–846, 2014.

J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. Statistical Science, 4(4):409–435, 1989.

Stat4Plant. Méthodes statistiques pour caractériser les interactions entre la plante et son environnement. https://stat4plant.mathnum.inrae.fr/, 2021.

M. Stein. Large sample properties of simulations using latin hypercube sampling. Technometrics, 1987.

Michael L. Stein. Interpolation of Spatial Data: Some Theory for Kriging. Springer Series in Statistics. Springer-Verlag New York, New York, 1999. doi: 10.1007/978-1-4612-1494-6.

Chen Wang, Qingyun Duan, Wei Gong, Aizhong Ye, Zhenhua Di, and Chiyuan Miao. An evaluation of adaptive surrogate modeling based optimization with two benchmark problems. Environmental Modelling and Software, 60:167–179, 2014. ISSN 1364-8152. doi: https://doi.org/10.1016/j.envsoft.2014.05.026. URL https://www.sciencedirect.com/science/article/pii/S1364815214001698.

A. H. Weir, P. L. Bragg, J. R. Porter, and J. H. Rayner. A winter wheat crop simulation model without water or nutrient limitations. The Journal of Agricultural Science, 102(2):371–382, 1984. doi: 10.1017/S0021859600042702.

Steve Weston. doParallel: Foreach Parallel Adaptor for the 'parallel' Package, 2022. URL https://CRAN.R-project.org/package=doParallel. R package version 1.0.17.

Chien-Jen Wu. On the convergence properties of the em algorithm. The Annals of Statistics, pages 95–103, 1983.