Gaussian Process Regression for the Prediction of the Solid-State Transformation Rate

Anna Grebennikova
Student, M1 Applied Mathematics
University of Grenoble Alpes

Supervised by:

Clémentine Prieur
Professor, University of Grenoble Alpes
AIRSEA Team, LJK Laboratory

Abstract

In recent years, Gaussian Processes have become a powerful tool for modeling and prediction tasks across a wide range of fields — from industry and material science to thermal physics. Their main advantage lies in the ability to build flexible nonlinear models that incorporate prior uncertainty, even when only a limited amount of experimental data is available, while requiring relatively low computational resources. GP models are particularly relevant in physics, where system behavior is often governed by complex and partially observable functions, and it is important not only to predict a value but also to quantify the uncertainty of that prediction.

In this study, Gaussian Processes were applied to construct a model capable of predicting the rate of change in the solid phase fraction of a material during phase transitions. The input data was obtained from calorimetry experiments using the Phase Change Material RT58, a material with well-characterized properties. During the experiments, the sample was subjected to repeated heating and cooling cycles at various controlled rates to ensure reliable data collection.

The purpose of the model is to estimate the rate at which the material changes—how quickly it transitions between solid and liquid phases—based on temperature, its time derivative, and other relevant factors. This model provides a basis for predicting the solid-state transformation rate in more complex and untested materials, which can then be used to investigate their physical properties.

Contents

1 Gaussian Processes and Gaussian Vectors

Gaussian Processes generalize Gaussian vectors. Therefore, this section begins with the definition of Gaussian vectors and an outline of their main properties, which serve as a basis for understanding Gaussian Processes, as presented in the lecture notes by François Bachoc [1].

1.1 Definition of a Gaussian Vector

A Gaussian Vector is defined as a random vector with values in $\mathbb{R}^n$. Consider n random variables $X_1, X_2, \ldots, X_n$ organized into a vector:

$$\mathbf{V} = (X_1, X_2, \ldots, X_n)^T \sim \mathcal{N}(\mu, \Sigma).$$

We tell that $\mathbf{V}$ is normally distributed with:

- $\mu \in \mathbb{R}^n$: the mean vector, where $\mu_i = \mathbb{E}[X_i]$ for $i = 1, \ldots, n$,
- $\Sigma \in \mathbb{R}^{n \times n}$: the covariance matrix, which is symmetric and positive semi-definite. Its elements are defined as:

$$\Sigma_{ij} = \mathrm{Cov}(X_i, X_j) = \mathbb{E}[(X_i - \mu_i)(X_j - \mu_j)].$$

The joint probability density function (PDF) of $\mathbf{V}$ is given by:

$$f_{\mathbf{V}}(\mathbf{X}) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma)}} \exp\left(-\frac{1}{2}(\mathbf{X} - \mu)^T \Sigma^{-1} (\mathbf{X} - \mu)\right),$$

where $\mathbf{X} \in \mathbb{R}^n$.

1.2 Characterizations of a Gaussian Vector

A random vector $\mathbf{V} \in \mathbb{R}^n$ is called a Gaussian vector if it satisfies any of the following equivalent conditions:

1. **Linear Combination Property:** Any linear combination of the components of $\mathbf{V}$ follows a Gaussian distribution. Specifically, for any fixed $n \times 1$ vector $\mathbf{a}$, there exist $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}^+$ such that:
$$\mathbf{a}^\top \mathbf{V} = \sum_{i=1}^{n} a_i V_i \sim \mathcal{N}(\mu, \sigma^2).$$

2. **Characteristic Function Property:** There exist $\mathbf{m} \in \mathbb{R}^n$ and an $n \times n$ symmetric non-negative definite matrix Σ such that, for all $n \times 1$ vectors $\mathbf{u}$, the characteristic function of $\mathbf{V}$ is given by:
$$\phi_{\mathbf{V}}(\mathbf{u}) = \mathbb{E}\left(e^{i\mathbf{u}^\top \mathbf{V}}\right) = \exp\left(i\mathbf{u}^\top \mathbf{m} - \frac{1}{2}\mathbf{u}^\top \Sigma \mathbf{u}\right)$$

3. **Linear Transformation Property:** There exists a $n \times 1$ vector $\mathbf{m}'$, a $n \times r$ matrix K (with $r \leq n$), and an $r \times 1$ random vector $\mathbf{W}$ with independent components following $\mathcal{N}(0, 1)$, such that:
$$\mathbf{V} = \mathbf{m}' + K\mathbf{W}.$$

   Moreover, with $\mathbf{m}$ and Σ from 2 and $\mathbf{m}'$ and K from 3, we have the following relationships:

   - Mean vector: $\mathbf{m} = \mathbf{m}' = \mathbb{E}(\mathbf{V})$,
   - Covariance matrix: $\Sigma = KK^\top = \mathrm{Cov}(\mathbf{V})$.

1.3 Gaussian Conditioning Theorem

We consider a $(n_1 + n_2) \times 1$ Gaussian vector

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix}$$

where Y_1 is of size $n_1 \times 1$ and Y_2 is of size $n_2 \times 1$. We write its mean vector as

$$\begin{pmatrix} m_1 \\ m_2 \end{pmatrix}$$

where m_1 is of size $n_1 \times 1$ and m_2 is of size $n_2 \times 1$. Finally, we write its covariance matrix as

$$\begin{pmatrix} \Sigma_1 & \Sigma_{1,2} \\ \Sigma_{1,2}^\top & \Sigma_2 \end{pmatrix}$$

where Σ_1 is of size $n_1 \times n_1$, $\Sigma_{1,2}$ is of size $n_1 \times n_2$, and Σ_2 is of size $n_2 \times n_2$. We assume that Σ_1 is invertible. Then, conditionally on $Y_1 = y_1$, the random vector Y_2 is a Gaussian vector with mean vector

$$\mathbb{E}(Y_2 \mid Y_1 = y_1) = m_2 + \Sigma_{1,2}^\top \Sigma_1^{-1}(y_1 - m_1)$$

and covariance matrix

$$\mathrm{Cov}(Y_2 \mid Y_1 = y_1) = \Sigma_2 - \Sigma_{1,2}^\top \Sigma_1^{-1} \Sigma_{1,2}.$$

Connection to Gaussian Processes

A Gaussian process can be thought of as an extension of the concept of a Gaussian vector to an infinite collection of random variables indexed on a continuous domain, such as space or time. While a Gaussian vector describes a finite set of random variables that follow a joint Gaussian distribution, a Gaussian process generalizes this idea by defining a random function.

For every realization of a Gaussian process, we obtain a specific function, known as a sample path or trajectory. These sample paths represent potential outcomes of the random process. Importantly, for any finite collection of points within the domain, the values of the Gaussian process at these points form a Gaussian vector.

1.4 Definition of a Gaussian Process

Let $(\Omega, \mathcal{A}, P)$ be a probability space, where $(\Omega, \mathcal{A})$ is a measurable space of elementary events, and P is a probability measure, i.e. any non-negative measure on $(\Omega, \mathcal{A})$ such that $P(\Omega) = 1$. Consider a function Z:

$$\begin{aligned} Z : (\Omega, \mathcal{A}, P) \times [0,1]^d &\to \mathbb{R}, \\ (\omega, x) &\mapsto Z(\omega, x). \end{aligned}$$

We say that Z is a Gaussian process on $[0,1]^d$ when, for all $n \in \mathbb{N}$ and for all $x_1, \ldots, x_n \in [0,1]^d$, the function

$$\omega \mapsto (Z(\omega, x_1), \ldots, Z(\omega, x_n))$$

is a Gaussian vector. In the sequel, we will often write $Z(x)$ instead of $Z(\omega, x)$.

- The function μ:
  $$\mu : [0,1]^d \to \mathbb{R}, \quad x \mapsto \mathbb{E}(Z(x)),$$
  is called the mean function of Z.
- The function K:
  $$K : [0,1]^d \times [0,1]^d \to \mathbb{R}, \quad (x, y) \mapsto \mathrm{Cov}(Z(x), Z(y)),$$
  is called the covariance function of Z.

If the function K depends only on $x - y$, that is, $x - y = x' - y' \Rightarrow K(x, y) = K(x', y')$, then we say that K is stationary.

1.5 Prediction Using the Gaussian Conditioning Theorem

Let Y be a Gaussian process on $[0,1]^d$ with mean function μ and covariance function K. Given n observations $Y(x_i) = y_i$, we aim to predict a function $f(x)$ such that $Y(x) = f(x)$ for any $x \in [0,1]^d$.

Les us define:

$$Y^{(n)} = \begin{pmatrix} Y(x_1) \\ \vdots \\ Y(x_n) \end{pmatrix}, \quad y^{(n)} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}, \quad \mu_y = \begin{pmatrix} \mu(x_1) \\ \vdots \\ \mu(x_n) \end{pmatrix},$$

and let R be the $n \times n$ covariance matrix of the observations defined as:

$$R = \big(K(x_i, x_j)\big)_{i,j=1,\dots,n}.$$

For a new point $x \in [0,1]^d$, define the $n \times 1$ covariance vector $r(x)$ as:

$$r(x) = \begin{pmatrix} K(x, x_1) \\ \vdots \\ K(x, x_n) \end{pmatrix}.$$

Then,

$$\begin{pmatrix} Y^{(n)} \\ Y(x) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_y \\ \mu(x) \end{pmatrix}, \begin{pmatrix} R & r(x) \\ r(x)^\top & K(x,x) \end{pmatrix} \right).$$

Thus, using the Gaussian conditioning theorem, the conditional expectation of $Y(x)$ given the observed data is:

$$\mathbb{E}[Y(x) \mid Y^{(n)}] = \mu(x) + r(x)^\top R^{-1}(y^{(n)} - \mu_y).$$

The function $\hat{Y}(x) = \mathbb{E}(Y(x) \mid Y^{(n)} = y^{(n)})$ is a metamodel of $f(x)$, and the computation of the metamodel has a negligible cost compared to performing an additional computer experiment to compute $f(x)$.

2 Learning with Gaussian Processes

The choice of the covariance kernel in Gaussian Processes is crucial because it directly determines the predictor's regularity and the data's dependency structure. The kernel is defined by a set of parameters that influence its behavior, and these parameters must be estimated using the model's available observations. Once estimated, the kernel is utilized in conjunction with the Gaussian Conditioning Theorem to compute predictions for the black-box function at new points.

The parameters of the covariance kernel are typically estimated using two main approaches: maximum likelihood estimation or cross-validation. Maximum likelihood estimation involves optimizing the likelihood of the observed data under the Gaussian Process model to determine the parameter values. Alternatively, cross-validation selects parameters by minimizing the prediction error on held-out data. In this work, we focus specifically on demonstrating the estimation of these parameters using the likelihood approach.

The selection of the initial kernel is guided by expert knowledge about the expected regularity of the black-box function. This insight helps ensure that the chosen kernel is well-aligned with the characteristics of the underlying function, enabling more effective modeling and accurate predictions.

2.1 Kernels and their properties

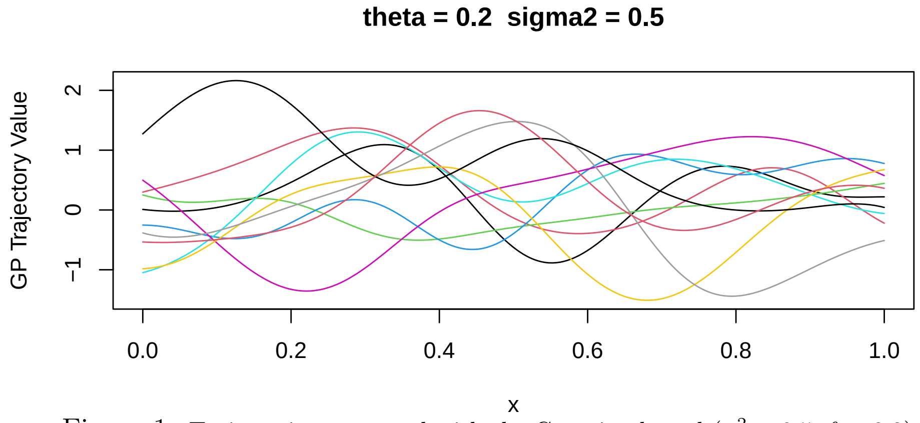
The most basic type of kernel is the Gaussian (or squared exponential) kernel [3], which is defined as:

$$k(x, x') := \sigma^2 \exp\left(-\frac{h^2(x, x')}{2\theta^2}\right)$$

where $h(x, x')$ represents the Euclidean distance between x and x', σ^2 is the variance, and θ is the length scale parameter. For inputs $x \in \mathbb{R}^d$, the Gaussian kernel becomes:

$$k(x, x') = \sigma^2 \exp\left(-\frac{\sum_{i=1}^{d}(x_i - x_i')^2}{2\theta^2}\right)$$

This kernel is infinitely differentiable, enabling it to model highly smooth functions. Figure 1 illustrates trajectories generated using the Gaussian kernel with $\sigma^2 = 0.5$ and $\theta = 0.2$. However, this level of smoothness may not always reflect real-world data, which often exhibit irregularities. Stein, in his work [4], argues that imposing such strict regularity constraints is often unrealistic for modeling physical processes.



theta = 0.2 sigma2 = 0.5

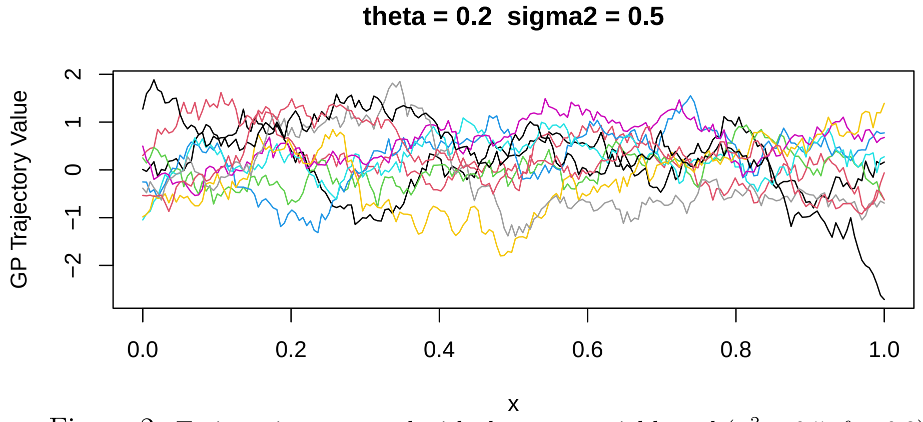
Figure 1: Trajectories generated with the Gaussian kernel ($\sigma^2 = 0.5$, $\theta = 0.2$).

Another commonly used kernel is the exponential kernel, defined as:

$$k(x, x') := \sigma^2 \exp\left(-\frac{|h(x, x')|}{\theta}\right)$$

For inputs $x \in \mathbb{R}^d$, this becomes:

$$k(x, x') = \sigma^2 \exp\left(-\frac{\sum_{i=1}^{d}|x_i - x_i'|}{\theta}\right)$$



theta = 0.2 sigma2 = 0.5

Figure 2: Trajectories generated with the exponential kernel ($\sigma^2 = 0.5$, $\theta = 0.2$).

The exponential kernel is not differentiable, making it more suitable for black-box functions where the underlying process is likely to be non-smooth. Simulated trajectories based on the exponential kernel are shown in Figure 2, highlighting its ability to model less smooth behavior.

In addition to these basic kernels, more flexible options like the Matérn kernels are widely used in practice [4, 5, 6]. These kernels introduce a smoothness parameter that allows for control over the differentiability of the resulting functions.

The Matérn 3/2 kernel is given by:

$$k(x,x') = \sigma^2 \left(1 + \sqrt{3}\|x - x'\|_\theta\right) \exp\left(-\sqrt{3}\|x - x'\|_\theta\right)$$

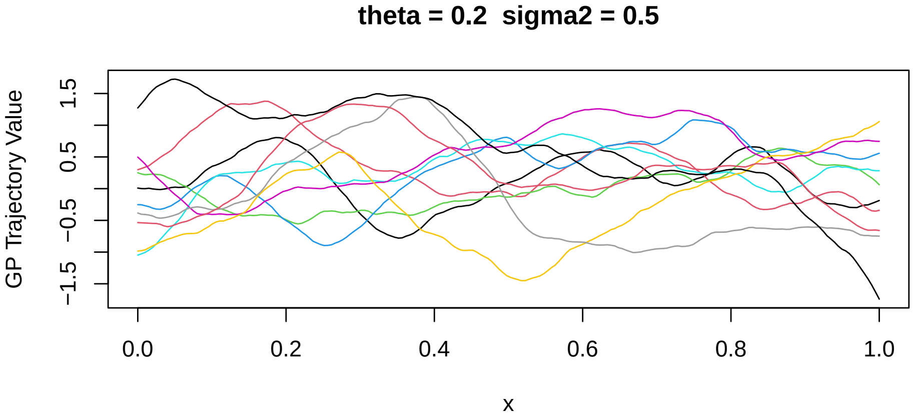


Figure 3: Trajectories generated with the Matérn 3/2 kernel ($\sigma^2 = 0.5$, $\theta = 0.2$).

This kernel is typically used to model C^1-differentiable functions, which are continuous but have less smooth derivatives. Trajectories generated using this kernel are shown in Figure 3.

The Matérn 5/2 kernel, on the other hand, provides smoother functions and is commonly used for modeling C^2-differentiable functions. Its expression is:

$$k(x,x') = \sigma^2 \left(1 + \sqrt{5}\|x - x'\|_\theta + \frac{5}{3}\|x - x'\|_\theta^2\right) \exp\left(-\sqrt{5}\|x - x'\|_\theta\right)$$

Simulations with this kernel are depicted in Figure 4.

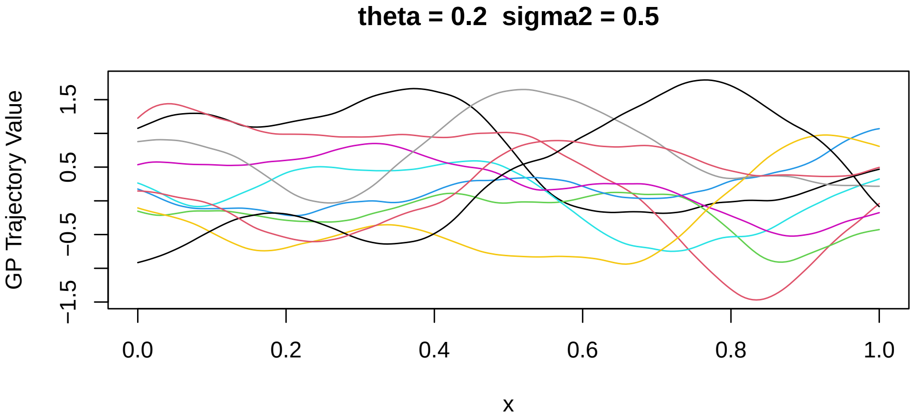


Figure 4: Trajectories generated with the Matérn 5/2 kernel ($\sigma^2 = 0.5$, $\theta = 0.2$).

The choice of kernel should align with the underlying characteristics of the data, balancing smoothness and flexibility to capture the system's behavior effectively.

2.2 Parameter estimation

To estimate the parameters of a Gaussian process, we rely on the principle of *maximum likelihood estimation (MLE)* [4]. The main idea is to find the parameter values that maximize the likelihood of the observed data.

The likelihood function is defined as:

$$L(\theta) = \prod_{i=1}^{m} f(X_i; \theta),$$

where:

- m is the number of observations,
- $f(X_i; \theta)$ is the probability density function of the n-dimensional Gaussian vector X_i,
- θ represents the parameters of the distribution (e.g., variance σ and length scale ℓ).

For a multivariate Gaussian distribution, the density function is:

$$f(\mathbf{X}; \mu; \Sigma(\sigma, \theta)) = \frac{1}{\sqrt{(2\pi)^n \det(\Sigma(\sigma, \theta))}} \exp\left(-\frac{1}{2}(\mathbf{X} - \mu)^T \Sigma^{-1}(\sigma, \theta)(\mathbf{X} - \mu)\right),$$

where:

- μ is the mean vector (assumed constant),
- $\Sigma(\sigma, \theta)$ is the covariance matrix parameterized by variance σ and other parameters θ.

To simplify the calculations, we work with the *log-likelihood function*, defined as:

$$\ell(\theta) = \ln L(\theta) = \sum_{i=1}^{m} \ln f(X_i; \theta).$$

For a Gaussian vector with m observations, the log-likelihood function becomes:

$$\ell(\mu, \Sigma(\sigma, \theta)) = \sum_{i=1}^{m} \ln\left(f(\mathbf{X}_i; \mu; \Sigma(\sigma, \theta))\right).$$

Substituting the Gaussian density function, we have:

$$\ell(\mu, \Sigma(\sigma, \theta)) = \sum_{i=1}^{m} \left(-\frac{n}{2}\ln(2\pi) - \frac{1}{2}\ln\det(\Sigma(\sigma, \theta)) - \frac{1}{2}(\mathbf{X}_i - \mu)^T \Sigma^{-1}(\sigma, \theta)(\mathbf{X}_i - \mu)\right).$$

Simplifying further:

$$\ell(\mu, \Sigma(\sigma, \theta)) = -\frac{mn}{2}\ln(2\pi) - \frac{1}{2}\sum_{i=1}^{m}\ln\det(\Sigma(\sigma, \theta)) - \frac{1}{2}\sum_{i=1}^{m}(\mathbf{X}_i - \mu)^T \Sigma^{-1}(\sigma, \theta)(\mathbf{X}_i - \mu).$$

The parameters μ, σ, and θ are estimated by maximizing the log-likelihood function:

$$\hat{\mu}, \hat{\sigma}, \hat{\theta} = \arg\max_{\mu, \sigma, \theta} \ell(\mu, \Sigma(\sigma, \theta)).$$

2.3 Model validation

The validation of a model's predictive performance is crucial to assess its generalization ability and goodness of fit. One common validation criterion used in regression tasks is the Q^2-metric. This metric measures the proportion of variance in the observed data that is explained by the model. The Q^2-score is calculated as:

$$Q^2 = 1 - \frac{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

where y represents the true values, $\hat{y}$ are the predicted values from the model, and $\bar{y}$ is the mean of the true values. The numerator represents the mean squared error (MSE), quantifying the average squared difference between the true and predicted values. The denominator normalizes this by the variance of the observed data. A Q^2-value close to 1 indicates that the model explains most of the variance, while a value near 0 suggests poor performance. Therefore, the Q^2-metric is particularly valuable for evaluating models designed to generalize well to new, unseen data.

To assess the model's ability to generalize, leave-one-out (LOO) cross-validation was employed in this study. In LOO, the model is trained on all data points except for one, and the prediction for the excluded data point is computed. This process is repeated for each data point, ensuring that every data point serves as a validation set. This method is computationally intensive, requiring the training of N distinct models (where N is the total number of data points). However, it provides an estimation of the predictive error without new model evaluations. By using LOO, we ensure that the model is tested on unseen data, thereby mimicking real-world scenarios where predictions are made for new, unknown data. The Q^2-metric, when applied to the LOO predictions, offers a robust measure of the model's generalization ability, as it evaluates how well the model performs on each data point when trained on all others.

2.4 Twinning Sampling

Let $\mathcal{D} = \{\mathbf{Z}_i = [\mathbf{X}_i, Y_i]\}_{i=1}^N \subset \mathbb{R}^{N\times d}$ represent a dataset, where $\mathbf{X}_i$ is a $(d-1)$-dimensional vector corresponding to the $d-1$ features of the i^{th} data point, and Y_i is the associated response variable.

The concept of `data twinning`, as introduced by Akhil Vakayil and V. Roshan Joseph [7] aims to partition a dataset into two disjoint subsets, referred to as twins. These subsets, while not necessarily of equal size, should share similar statistical properties. Denoting the twins by $\mathcal{D}^1 = \{\mathbf{U}_i\}_{i=1}^n$ and $\mathcal{D}^2 = \{\mathbf{V}_j\}_{j=1}^{N-n}$, they are constructed such that:

$$\mathcal{D}^1 \cap \mathcal{D}^2 = \emptyset \quad \text{and} \quad \mathcal{D}^1 \cup \mathcal{D}^2 = \mathcal{D}.$$

To quantify the similarity between the statistical distributions of the two subsets, the energy distance metric was used as:

$$\overline{\mathbb{ED}}_{n,N-n} = \frac{2}{n(N-n)}\sum_{i=1}^{n}\sum_{j=1}^{N-n}\|\mathbf{U}_i - \mathbf{V}_j\|_2 - \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\|\mathbf{U}_i - \mathbf{U}_j\|_2 - \frac{1}{(N-n)^2}\sum_{i=1}^{N-n}\sum_{j=1}^{N-n}\|\mathbf{V}_i - \mathbf{V}_j\|_2.$$

The goal of twinning is to minimize $\overline{\mathbb{ED}}_{n,N-n}$ while ensuring the disjointness and completeness conditions for $\mathcal{D}^1$ and $\mathcal{D}^2$:

$$\{\mathbf{U}_i^*\}_{i=1}^n, \{\mathbf{V}_j^*\}_{j=1}^{N-n} \in \underset{\{\mathbf{U}_i\}_{i=1}^n, \{\mathbf{V}_j\}_{j=1}^{N-n}}{\arg\min} \overline{\mathbb{ED}}_{n,N-n}$$

$$\text{subject to:} \quad \{\mathbf{U}_i\}_{i=1}^n \cap \{\mathbf{V}_j\}_{j=1}^{N-n} = \emptyset, \quad \{\mathbf{U}_i\}_{i=1}^n \cup \{\mathbf{V}_j\}_{j=1}^{N-n} = \mathcal{D}.$$

To achieve this, the authors proposed a sequential algorithm. Assuming a splitting ratio of $\gamma = 1/r$, where r is an integer such that $N = rn$, the dataset $\mathcal{D}$ is divided into n disjoint subsets, each containing r points. From each subset, one point is assigned to $\mathcal{D}^1$ and the remaining $r-1$ points are assigned to $\mathcal{D}^2$, preserving the desired splitting ratio γ.

Given a starting point $u_1 \in \mathcal{D}$, the $r-1$ points closest to u_1 in the Euclidean sense are identified to form the first subset:

$$\mathcal{S}_1 = \{u_1, v_1^1, v_1^2, \ldots, v_1^{r-1}\}.$$

The points in $\mathcal{S}_1$ are ranked according to their distances from u_1, satisfying:

$$\|u_1 - v_1^k\|_2 \leq \|u_1 - v_1^{k+1}\|_2, \quad \forall k = 1, \ldots, r-2.$$

The point u_1 is added to $\mathcal{D}^1$, while the remaining points $\{v_1^1, v_1^2, \ldots, v_1^{r-1}\}$ are assigned to $\mathcal{D}^2$. The next starting point u_2 is chosen as the point in $\mathcal{D} \setminus \mathcal{S}_1$ that is closest to v_1^{r-1}. A new subset is then formed:

$$\mathcal{S}_2 = \{u_2, v_2^1, v_2^2, \ldots, v_2^{r-1}\},$$

where u_2 is added to $\mathcal{D}^1$ and the remaining points $\{v_2^1, v_2^2, \ldots, v_2^{r-1}\}$ are assigned to $\mathcal{D}^2$.

This process continues iteratively until all points in $\mathcal{D}$ are distributed into the two subsets $\mathcal{D}^1$ and $\mathcal{D}^2$. This sequential approach ensures that the energy distance $\overline{\mathbb{ED}}_{n,N-n}$ is minimized, balancing the closeness of the twins with their internal diversity, and providing a statistically consistent partitioning of the dataset.

For the purposes of demonstration, consider a simple two-dimensional dataset consisting of $N = 50$ observations, generated according to the following distribution:

$$X_i \sim N(0,1) \quad \text{and} \quad Y_i \mid X_i \sim N(X_i^2, 1) \quad \text{for} \quad i = 1, \ldots, N.$$

The dataset is divided with a division rate $r = 5$, which results in the subset $\mathcal{D}_1$ containing 10 points and $\mathcal{D}_2$ containing 40 points. A graphical representation demonstrating the algorithm's operation is provided below.

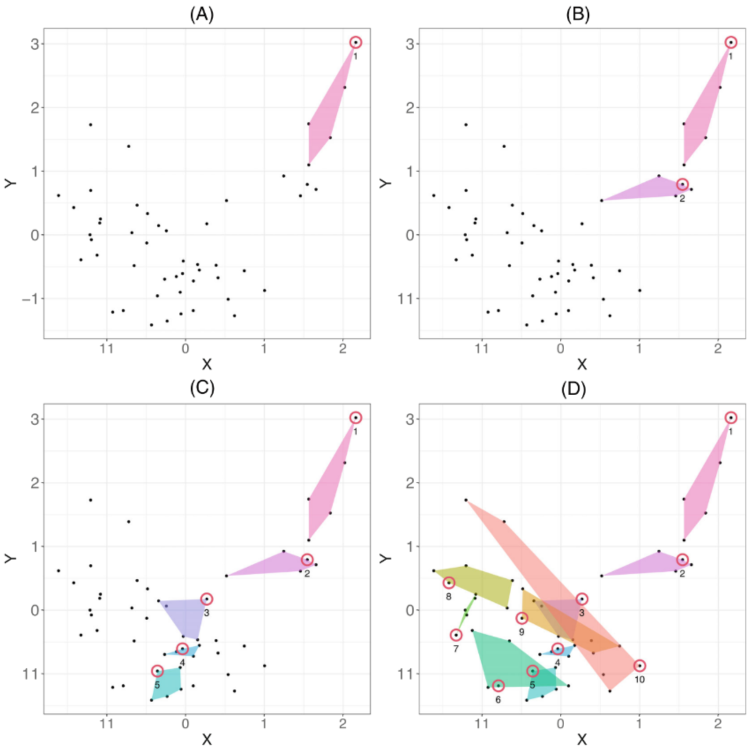


Figure 5: The convex hull of subsets identified by Twinning at the end of iterations 1, 2, 5, and 10 for the sample dataset described in Section 3. Points in $\mathcal{D}_1$ are shown as encircled points, and they are numbered in the order they were selected. (A) Iteration 1. (B) Iteration 2. (C) Iteration 5. (D) Iteration 10.

2.5 Adaptive Learning

The performance of a model can vary across the input domain, resulting in different Mean Squared Error (MSE) values at prediction points. High MSE values highlight regions of greater uncertainty in the model's predictions. To minimize the overall error and enhance model reliability, it is essential to strategically add new data points. This is achieved using an adaptive learning approach [10], where the model is updated iteratively by including new points selected based on specific criteria, such as regions with the highest uncertainty or variance.

The process begins with an initial training sample size of $n = 5d$, where d is the number of input dimensions. Additional points are then added step by step. First, the model evaluates the test set and calculates the MSE for each point. The next point is chosen near the region with the highest MSE and added to the training data. The model is then retrained with $n+1$ points. This procedure is repeated, with new points continuously selected in areas of high MSE, until the model's performance becomes satisfactory.

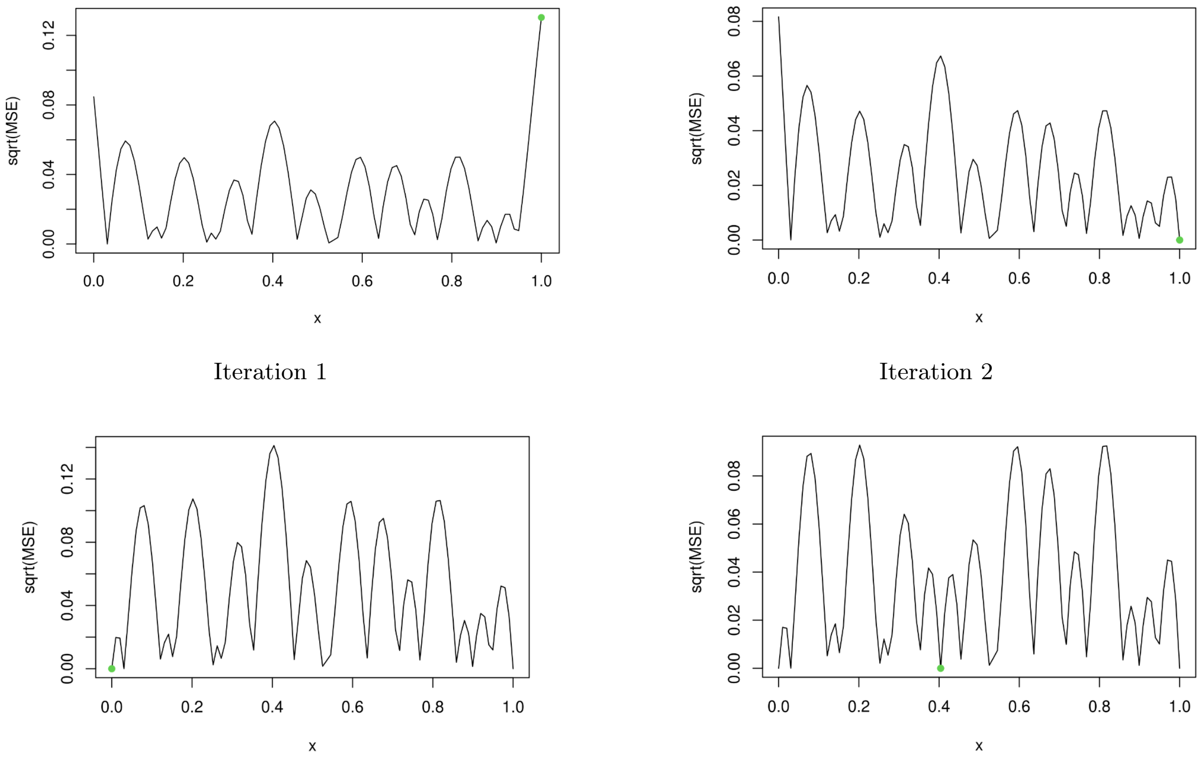
The adaptive learning process continues until the model achieves satisfactory performance, as determined by a chosen evaluation criterion. This iterative approach ensures that the model focuses on exploring regions of the input space where it is least certain, thereby improving its predictive capabilities.

To demonstrate the method in practice, we consider the analytical one-dimensional function defined on the interval $[0, 1]$:

$$f(x) = \sin\big(30(x-0.9)^4\big)\cos\big(2(x-0.9)\big) + (x-0.9).$$

A training set of 15 points, optimally distributed within the interval, was selected, and a metamodel was constructed using the Matérn 5/2 kernel.

Figure 6 presents the root mean squared error evaluated at each point during the first four iterations of the adaptive learning process.



Iteration 1

Iteration 2

Iteration 3

Iteration 4

Figure 6: Optimization process showing the point with maximum root MSE (green) during the first four iterations.

As we can see, after adding a point to the training set, the MSE becomes equal to zero for the point that had the maximum error in the previous iteration.

2.6 Latent Variable Gaussian Process

In Gaussian Process (GP) modeling, qualitative inputs are typically handled by assuming an independent response surface for each combination of levels of the qualitative factors. While this approach is standard, it treats qualitative variables as fixed categories rather than dynamic contributors to the model, potentially oversimplifying their influence.

The Latent Variable Gaussian Process (LVGP) method, introduced by Zhang, Tao, Chen, and Apley [8], addresses this limitation by mapping qualitative factor levels to latent, unobservable quantitative variables. After obtaining this mapping, the GP covariance model over $(\mathbf{x}, t)$ can be any standard GP covariance model for quantitative variables over $(\mathbf{x}, \mathbf{z}(t))$, where $\mathbf{z}(t)$ is the numerical vector of mapped latent variables. This transformation allows GP models to treat qualitative and numerical inputs equivalently, reflecting their effects through similar physical mechanisms.

This approach incorporates a single qualitative factor t with m levels, represented as $t = 1, 2, \ldots, m$, by introducing a two-dimensional latent variable $\mathbf{z}(t) \in \mathbb{R}^2$. Each of the m levels is mapped to a corresponding latent numerical value $\{\mathbf{z}(1) = (z_1(1), z_2(1)), \ldots, \mathbf{z}(m) = (z_1(m), z_2(m))\}$ in a 2D latent space. Consequently, the input $\mathbf{w} = (\mathbf{x}, t)$ is transformed into $(\mathbf{x}, \mathbf{z}(t))$, allowing the Gaussian correlation function to be defined as:

$$R\big(y(\mathbf{x}, t), y(\mathbf{x}', t')\big) = \exp\left(-\sum_{i=1}^{p} \phi_i (x_i - x_i')^2 - \|\mathbf{z}(t) - \mathbf{z}(t')\|_2^2\right),$$

where ϕ_i are the correlation parameters for the quantitative inputs $\mathbf{x}$. The latent variables $\{\mathbf{z}(1), \mathbf{z}(2), \ldots, \mathbf{z}(m)\}$ are estimated via maximum likelihood estimation (MLE), with $\mathbf{z}(1)$ fixed at $(0, 0)$. This scaling ensures that no additional correlation parameter is required for $\mathbf{z}$.

By parameterizing the levels of the qualitative factor in the latent space, this approach allows the Gaussian Process (GP) model to treat categorical variables in the same manner as numerical inputs. This enables seamless integration of categorical variables into the GP framework, facilitating their modeling alongside numerical variables with minimal adjustments.

The authors emphasize that a 2D latent space is often the most suitable choice for handling categorical variables. It provides sufficient flexibility in terms of correlation structure and significantly reduces the likelihood of singularity, as points can be more freely positioned in the 2D space. However, in some cases, a 2D latent space may not be sufficient, and higher-dimensional latent spaces may be required for better representation.

3 Experiment Description

In this study, we analyzed data provided by Gilles Fraisse from the Laboratory of Energy Processes and Building Materials (LOCIE) at Université Savoie Mont Blanc. His research focuses on the thermal behavior of materials, with particular emphasis on the Solid-State Transformation Rate—a critical parameter for determining heat capacity, which is a fundamental property of materials [14].

The data were obtained through an experiment conducted using a scanning calorimeter. The material studied was a typical Phase Change Material (PCM) named RT58 [14], with a mass of 200 grams. The calorimeter was programmed to follow a controlled temperature routine in which the sample was gradually heated from 20°C to 70°C, and then cooled back to 20°C. This heating and cooling cycle was repeated several times to enhance the reliability and consistency of the results.

The dataset collected from the experiment included the following columns: t (time), T (temperature), and α (the solid-state fraction of the material). From these, the rate of temperature change, β, was computed as $\beta = dT/dt$, and the second derivative, $d\beta/dt$, was also calculated. These processed data served as inputs for developing a model to predict $d\alpha/dt$, the rate of solid-state transformation.

For the material under consideration, our colleague employed an analytical formula:

$$\frac{d\alpha}{dt} = -\beta^{n_4} \cdot \mathrm{sign}(\beta) \cdot \left\{ k_1 \left(\frac{\alpha^{m_1} (\alpha_{\max} - \alpha)^{n_1}}{\alpha_{\max}^{m_1+n_1}} \right) + k_2 \left(\frac{(\alpha - \alpha_{\min})^{m_2} (1 - \alpha)^{n_2}}{(1 - \alpha_{\min})^{m_2+n_2}} \right) \right\} \cdot \left(1 - k_3 \frac{d\beta}{dt} \right)^{n_3}$$

During the calculations of this formula, it was assumed that $\frac{d\beta}{dt} = 0$. However, in our metamodel, we treated $d\beta/dt$ as a parameter to be included in the model training process, allowing for a more flexible representation of the transformation rate The coefficients used in the formula were determined using the Excel Solver with the "GRG Nonlinear" method, resulting in the following values: $n_1 = 1.26$, $n_2 = 0.68$, $n_4 = 1.04$, $k_1 = 0.42$, $k_2 = 0.08$, $\alpha_{\text{max}} = 0.75$, $\alpha_{\text{min}} = 0.38$, $m_1 = 0.54$, and $m_2 = 1.04$.

This formula and the associated coefficients will serve as a comparison with our results.

4 Application of GP regression to fit $d\alpha/dt$

We wish to model:

$$\left(\alpha, \beta, \frac{d\beta}{dt}, \text{sign}(\beta)\right) \rightarrow \frac{d\alpha}{dt}$$

where β can be either positive or negative.

Since Gaussian Processes typically work with quantitative variables, although β is a numerical variable, it can essentially be considered binary because it takes values of ±1. Therefore, it should be treated as a categorical variable. Consequently, our initial approach is to split the dataset into two groups based on the sign of β: one group where $\text{sign}(\beta) = 1$ and another where $\text{sign}(\beta) = -1$.

4.1 Two Groups Using Twinning Sampling

The initial dataset contained approximately 11,000 observations, with some entries having $\text{sign}(\beta) = 0$, $\text{sign}(\beta) = 1$, or $\text{sign}(\beta) = -1$. First, the dataset was divided into two groups based on the sign of β: one group with $\text{sign}(\beta) = 1$ and another with $\text{sign}(\beta) = -1$.

The models to be built are as follows:

$$\left(\alpha, \beta, \frac{d\beta}{dt}, \text{sign}(\beta) = 1\right) \rightarrow \frac{d\alpha}{dt} \qquad \text{and} \qquad \left(\alpha, \beta, \frac{d\beta}{dt}, \text{sign}(\beta) = -1\right) \rightarrow \frac{d\alpha}{dt}$$

Each group contains approximately 4,000 observations. However, building models using the entire set of observations in each group is computationally expensive. To address this, we will use the `twinning` library [9] to select a subset of input and output with a distribution similar to that of the full dataset.

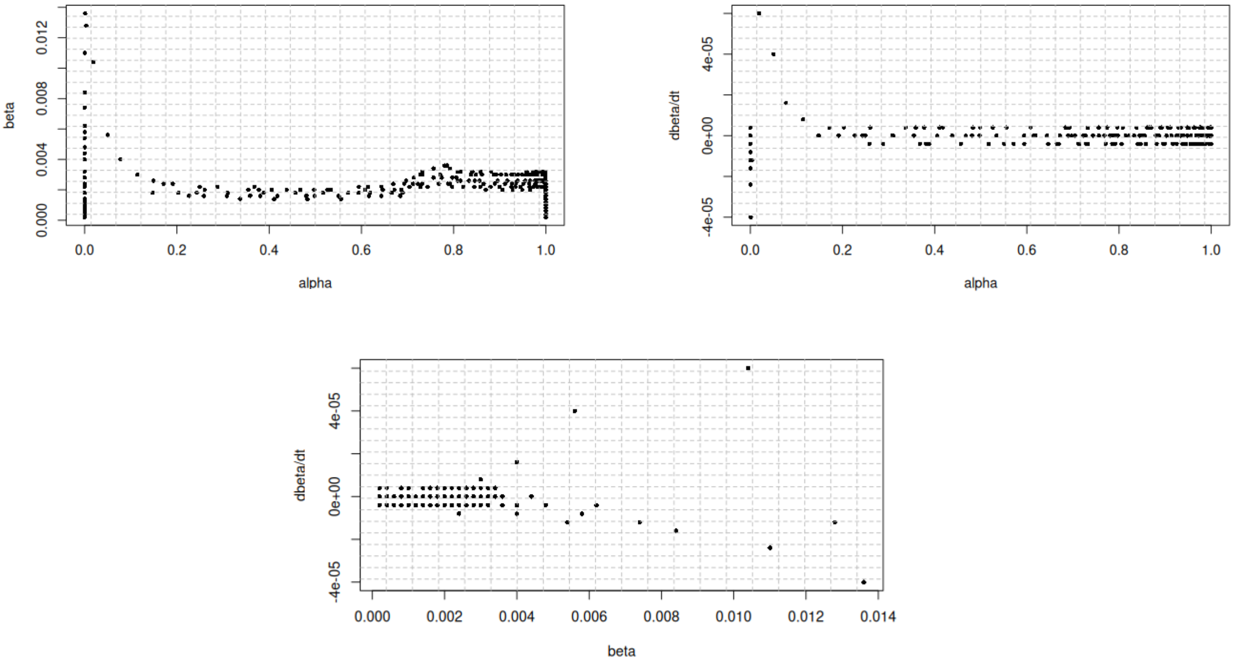
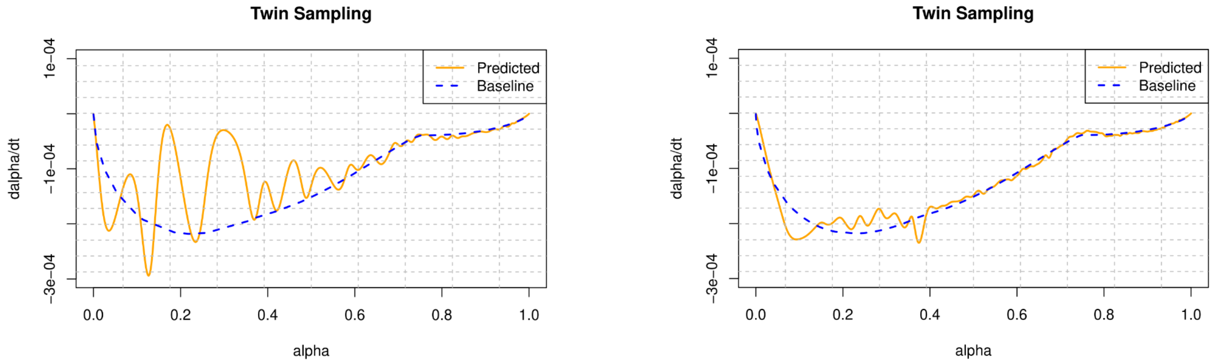


Figure 7: Projections of the input data obtained by twin sampling onto different planes $\beta > 0$.

After twin sampling, we obtained the distributions for α, β, and $d\beta/dt$. To visualize how the points are distributed in the space, we plot the projections onto three parameter planes in Figure 7.

After training the model using the Matérn 5/2 kernel and the `DiceKriging` library [11, 12], the model's performance was evaluated using the Q^2-metric. This metric was computed with both a test set and Leave-One-Out estimation.

For the graphical representation of the predicted function, the values of β and $\frac{d\beta}{dt}$ were fixed, although these parameters were not held constant during the training process. Specifically, the median values of β and $d\beta/dt$ were chosen as fixed parameters within the model: $\beta = 0.002$ and $d\beta/dt = 0$.



138 observations; $Q^2 = 0.8813$, $Q^2_{\text{LOO}} = 0.8381$

412 observations; $Q^2 = 0.9901$, $Q^2_{\text{LOO}} = 0.9886$

Figure 8: Examples of less accurate simulations with fixed seed `set.seed(123)` for $\beta > 0$.

It was observed that variations in the results occurred when different values were used for `set.seed()`, particularly when the number of training points was insufficient. Therefore, the selection of points by twin sampling produced dramatically different outcomes depending on the seed value, resulting in good or bad results intermittently. An example of poor results due to insufficient points is presented in Figure 8 (left). These inaccuracies are attributed to convergence issues encountered during the optimization of the kernel parameters.

To mitigate the variability associated with the choice of seed parameter, the number of data points was increased, thereby ensuring more consistent results across different seeds. However, too big number of points does not necessarily guarantee better results, as illustrated in Figure 8 (right). This phenomenon arises due to numerical instabilities caused by closely spaced points, which lead to large errors when inverting the covariance matrix.

The optimal number of observations was 275, with the best convergence to the baseline solution achieved as shown in Figure 9:

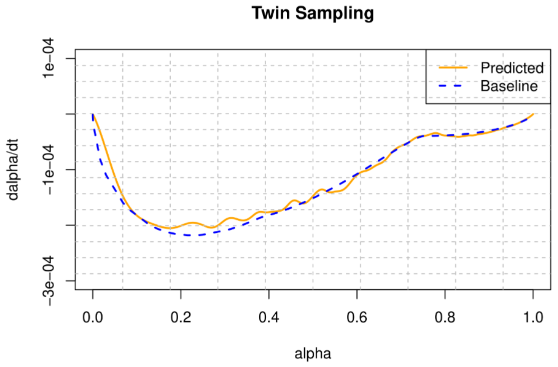


Figure 9: 275 observations; $Q^2 = 0.9980$, $Q^2_{\text{LOO}} = 0.9424$ for $\beta > 0$

The same analysis was performed for a group of data with $\beta < 0$. The best results were obtained with 209 observations, using fixed parameters within the model: $\beta = 0.002$ and $\frac{d\beta}{dt} = 0$.

These results achieved $Q^2 = 0.9986$ and are presented in Figure 10:

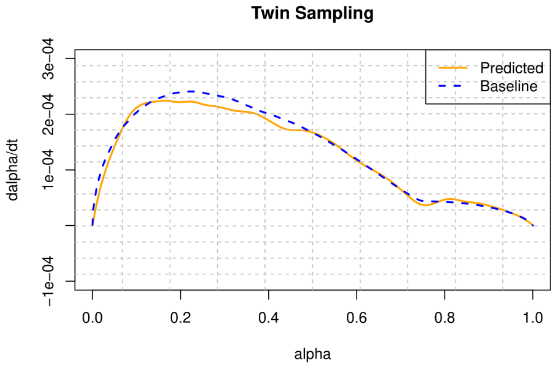
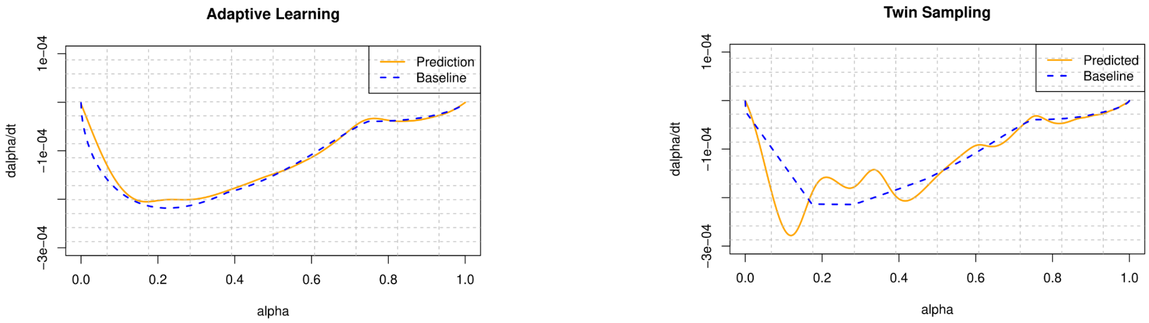


Figure 10: 417 observations; $Q^2 = 0.9986$, $Q^2_{\text{LOO}} = 0.9965$ for $\beta < 0$

4.2 Two Groups Using Adaptive Learning

As discussed in the previous section, the challenge of selecting initial points can result in unpredictable model behavior. To address this, an alternative method known as adaptive learning was employed.

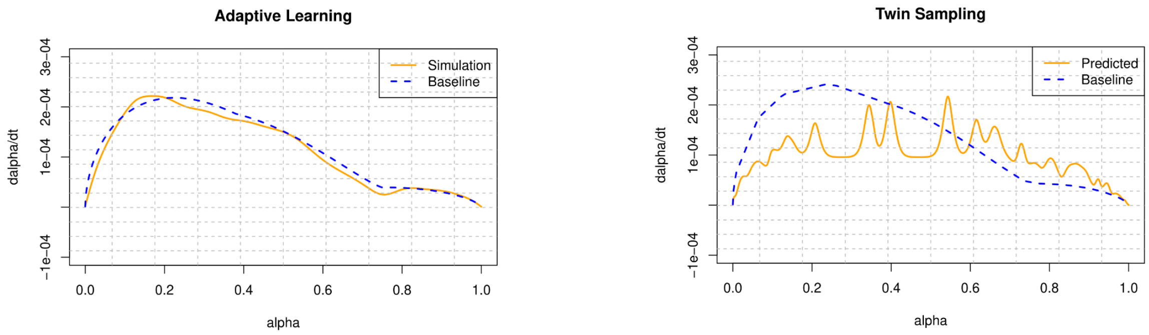
In this approach, an initial set of 15 points was randomly selected from the dataset. Subsequently, 35 additional points were iteratively added to refine the model. These additional points were drawn from a pool of approximately 200 observations generated using twinning sampling. The results obtained from this analysis are presented in Figure 11



Adaptive Learning. 50 observations; $Q^2 = 0.9955$, $Q^2_{\text{LOO}} = 0.9529$ for $\beta > 0$.

Twinning Sampling. 50 observations; $Q^2 = 0.9377$, $Q^2_{\text{LOO}} = 0.8471$ for $\beta > 0$.

Figure 11: Comparison of adaptive learning and twinning sampling; $\beta > 0$; `set.seed(123)`; 50 observations.



Adaptive Learning. 80 observations; $Q^2 = 0.9980$, $Q^2_{\text{LOO}} = 0.9973$ for $\beta < 0$.

Twinning Sampling. 80 observations; $Q^2 = 0.8315$, $Q^2_{\text{LOO}} = 0.8498$ for $\beta < 0$.

Figure 12: Comparison of adaptive learning and twinning sampling; $\beta < 0$; `set.seed(123)`; 80 observations.

As observed, the adaptive learning approach allows for a more strategic selection of points to include in the training sample. For the case of positive β, the results demonstrate a significant improvement in accuracy when compared to twinning sampling with the same number of points (50 observations). Same was observed with negative β in Figure 12.

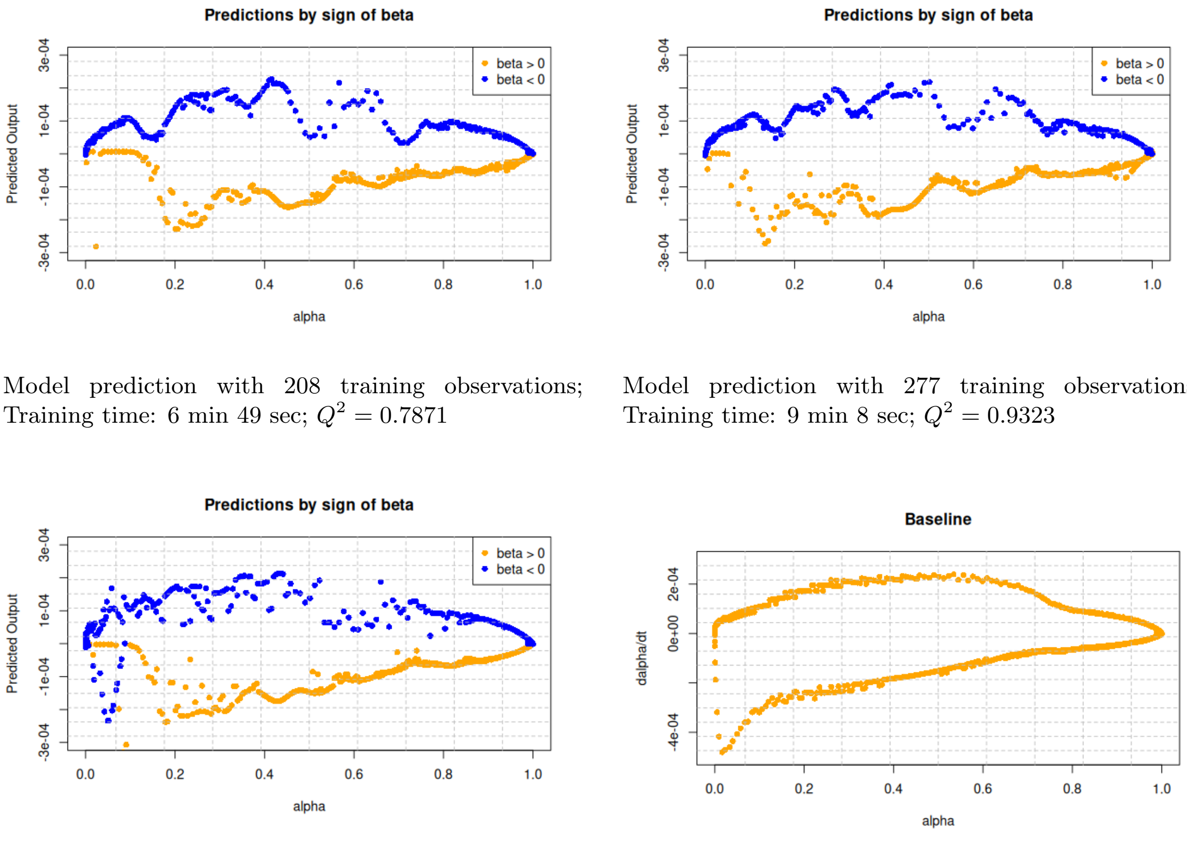
These results demonstrate that adaptive learning is a more efficient method for selecting points to train the model. It offers greater stability across different values of `set.seed()`, making it a more robust approach.

4.3 LVGP approach

As mentioned earlier, although β is a numerical variable, it can essentially be considered binary because it takes values of ±1. Therefore, it should be treated as a categorical variable. Building on this, our goal is to link the categorical variable sign(β) to latent variables in a 2D space to construct a GP model using it as an input.

Three experiments were conducted using different sizes of training samples: 208, 277, and 415 observations. Observations were selected from the entire dataset using twinning sampling. The model was then trained on these samples to establish a connection between the levels of the categorical variable and the latent variables by minimizing the likelihood function [13]. The training time was recorded to demonstrate the increase in computational time with larger training set sizes.

Subsequently, the trained model was applied to data consisting of 1,500 observations to compare the baseline results with those predicted by the model.



Model prediction with 208 training observations; Training time: 6 min 49 sec; $Q^2 = 0.7871$

Model prediction with 277 training observations; Training time: 9 min 8 sec; $Q^2 = 0.9323$

Model prediction with 415 training observations; Training time: 20 min 34 sec; $Q^2 = 0.8668$

Baseline prediction

Figure 13: Model predictions on a test sample of size 1500 compared to baseline results.

As observed, the statistical model shows many oscillations in the middle of the α-interval across all training sample sizes when compared to the theoretical model. Additionally, the training time increases significantly with the size of the training sample. If the training size is too high it seems that we encounter numerical issues as Q^2 decreases.

5 Conclusion

In this work, Gaussian processes and experimental modeling were analyzed and applied. The study focused on key aspects of Gaussian process metamodel, such as selecting kernels, parameter estimation, and model validation. Additionally, techniques for selecting well-distributed training points were investigated, including twin sampling to identify subsets with matching distributions, and adaptive learning for iterative point selection. Furthermore, a latent variable approach was explored to address categorical variables within Gaussian Process Regression.

The methodology was applied to data provided by colleagues studying solid-state transformation to analyze and validate relevant formulas. Since the material and its characteristics are well-known, the results were compared with established formulas. Promising outcomes were achieved using adaptive learning and twin sampling. However, the robustness of twin sampling was found to depend on the choice of the seed parameter. This approach typically requires selecting approximately 300–400 training points to achieve reliable model performance, while adaptive learning achieves stable and good results with only about 50 training points, independent of the seed parameter.

The results obtained using Latent Variable Gaussian Processes were less satisfactory. The model struggled to make accurate predictions in the middle of the interval for α, and the situation did not improve even with the addition of more data points. Furthermore, fitting the levels of categorical variables to latent variables is time consuming, making Leave-One-Out validation impractical due to the significant increase in training time.

In addition, incorporating adaptive learning into the LVGP framework could theoretically enhance its performance, making this a potential area for future research. However, it is important to note that training this model would require significantly more time.

The methodology we developed is inherently general, both in its design and the tools utilized. Using this approach, we achieved strong results for a well-studied material. This inspires confidence in the potential of our method to be applied to more complex and previously unstudied materials aiding in the optimization of material studies.

References

[1] François Bachoc. Lecture notes on Gaussian processes and sensitivity analysis for computer experiments. Available at: `https://www.math.univ-toulouse.fr/~fbachoc/Lecture_notes_gaussian_processes_and_sensitivity_analysis.pdf`.

[2] A. O'Hagan. *On curve fitting and optimal design for regression.* Journal of the Royal Statistical Society. Series B (Methodological), 40(1):1–42, 1978.

[3] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* Springer, 2006.

[4] M. L. Stein. *Interpolation of Spatial Data.* Springer Series in Statistics. Springer, 1999.

[5] P. Guttorp and T. Gneiting. "Studies in the history of probability and statistics XLIX: On the Matérn correlation family." *Biometrika*, 93(4):989–995, 2006.

[6] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments.* Springer Science & Business Media, 2013.

[7] A. Vakayil and V. R. Joseph. "Data Twinning." *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 15(5):598–610, 2022.

[8] Y. Zhang, S. Tao, W. Chen, and D. W. Apley. "A Latent Variable Approach to Gaussian Process Modeling with Qualitative and Quantitative Factors." *Journal of Mechanical Design*, 144(11):111703, 2022.

[9] Twinning Package. *Data Twinning.* Version 1.0, October 14, 2022. Available at: `https://cran.r-project.org/web/packages/twinning/index.html`.

[10] P. Semler and M. Weiser. "Adaptive Gaussian Process Regression for Efficient Building of Surrogate Models in Inverse Problems." *Inverse Problems*, 39(12):125003, 2023.

[11] O. Roustant, D. Ginsbourger, and Y. 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.

[12] DiceKriging Package. *Kriging Methods for Computer Experiments.* Version 1.5.6, 2023. Available at: `https://cran.r-project.org/web/packages/DiceKriging/index.html`.

[13] LVGP Package. *Latent Variable Gaussian Process Modeling.* Version 1.0, 2023. Available at: `https://cran.r-project.org/web/packages/LVGP/index.html`.

[14] M. Thonon, G. Fraisse, L. Zalewski, and M. Pailha. *Towards a better analytical modelling of the thermodynamic behaviour of phase change materials.* HAL open archive, 2022.