Systems of Gaussian process models for directed chains of solvers

Francois Sanson, Olivier Le Maitre, Pietro Marco Congedo

Uncertainty Propagation Framework for Systems of Solvers

Francois Sanson

Inria Bordeaux Sud-Ouest, 200 Avenue de la Vieille Tour, 33405 Talence, France

Olivier Le Maitre

LIMSI, CNRS, Université Paris Saclay, Orsay, France

Pietro Marco Congedo

Inria Saclay Île-de-France, 1 Rue d'Estienne d'Orves, 91120, Palaiseau, France

Abstract

The simulation of complex multi-physics phenomena often relies on System of Solvers (SoS), which we define here as a set of interdependent solvers where the output of an upstream solver is the input of downstream solvers. Performing Uncertainty Quantification (UQ) analyses in SoS is challenging as they generally feature a large number of uncertain input parameters so that classical UQ methods, such as spectral expansions or Gaussian process models, are affected by the curse of dimensionality. In this work, we develop an original mathematical framework, based on Gaussian Process (GP) models, to construct a global surrogate model of the uncertain directed SoS, (i.e. merely featuring one-way dependences between solvers). The key ideas of the proposed approach are i) to determine a local GP model for each solver constituting the SoS and, ii) to define the prediction as the composition of the individual GP models constituting a system of GP models (SoGP). We further propose different adaptive sampling strategies for the construction of the SoGP. These strategies are based on the decomposition of the SoGP prediction variance into individual contributions of the constitutive GP models and on extensions of the Maximum Mean Square Predictive Error criterion to system of GP models. The performance of the SoGP framework is then assessed on several SoS involving different numbers of solvers and structures of input dependencies. The results show that the SoGP framework is very flexible and can handle different types of SoS, with a significantly reduced construction cost (measured by the number of training samples) compared to the direct GP model approximation of the SoS.

Keywords: Surrogate Model, Gaussian Process, System of solvers, Variance Decomposition, Adaptive sampling

1. Introduction

Many engineering problems are solved in a multiphysics environment requiring the resolution of multiple physical phenomena. The global solution to these problems is generally obtained by coupling different solvers, each one devoted to a specific aspect of the problem. Example of multiphysics problems can be found in nuclear safety application, weather forecast and space object reentry. In the latest domain, for instance, the following phenomena have to be modeled: fluid flow, structural

mechanics, ablation, trajectory propagation, material behavior, thermodynamics, etc. The prediction and the modeling of all the phenomena involved in the reentry of a space object, from the de-orbiting manoeuver to the ground impact, present a great challenge due to the complexity and the diversity of the physics involved. In practice, dedicated solvers are individually developed by teams of experts for each aspect of the physics (aero-thermics, trajectory propagation, tank thermodynamics, ...); these solvers are subsequently coupled to form what we call in this work a system of solvers (SoS). Formally, we define a SoS as a set of interdependent solvers. The composing solvers are linked by their inputs and outputs such as the output of a solver may be the input of another one. A SoS is said to be directed if the information can only be transferred forward in the system : with respect to a particular solver, the outputs of its downstream solvers cannot be inputs of any of its upstream solvers. On the contrary, in a strongly coupled SoS the inputs of a solver can be the outputs of a downstream solver.

In this work, we restrict ourselves to the case of directed SoS. A schematic representation of such a SoS is provided in Fig. 1. This modular aspect of SoS makes them flexible for industrial applications. Nevertheless, the application of Uncertainty Quantification (UQ) methods to SoS is particularly challenging. First, systems of solvers commonly involve a large number of uncertain inputs. This property generally challenges the efficiency of UQ methods based on functional approximations (see *e.g.*, [1]). Second, the SoS are usually computationally expensive to run as they require the sequential run of several solvers. These two challenges can rule out the application of standard UQ methods designed for single solvers. In this work, we propose an approach that exploits the structure of the SoS to efficiently propagate uncertainties in the SoS. The general procedure proposed in [2] is used: first a surrogate model of the SoS is constructed, and then a Monte Carlo approach is used to estimate the statistics of the quantities of interest using the surrogate model. Two approaches are classically used in the industry to build surrogate models and quantify uncertainties in SoS: the black box approach and the fragmented approach. In the first approach, the SoS is seen as a whole and the structure of the SoS along with its internal solvers is not taken into account in the UQ analysis. A surrogate model is built in order to create a direct mapping between the global uncertain inputs and the quantities of interest (global outputs). Within this approach, Polynomial Chaos expansions [1, 3, 4], GP models [5, 6], low rank approximations [7] are possible alternatives to construct a global surrogate. One major drawback of these alternatives is their computational cost that can dramatically increase with the number of (global) uncertain inputs, commonly referred to as Curse of Dimensionality [8]. Further, these methods can be challenged by the highly non-linear dependences between the global inputs and outputs induced by the structure of the SoS.

An alternative approach called the *fragmented approach* in the following, consists in building a surrogate model for each solver. In the fragmented approach, each surrogate relates the inputs of a solver to its outputs, and a prediction of the global outputs can be made by substituting each solver with its surrogate model. This approach is common and particularly suited in situations where the individual solvers are developed, maintained and run by distinct teams. The fragmented approach has a clear advantage with respect to the global black box approach when the inputs dimensionality of each solver is lower than the dimensionality of the global inputs. In this situation, it can be more effective to construct several low dimensional surrogate models, rather than constructing a single high-dimensional global one, therefore mitigating the curse of dimensionality. In addition, the individual solvers may exhibit simple mappings from inputs to outputs, where their composition may yield complex dependencies. On the other hand, the definition of the inputs probability measures to be used when constructing the individual surrogates represent a major drawback of the *fragmented appraoch*. The probability measure of an input that is the output of an upstream solver is unknown

a priori. One solution to this issue consists in assuming an a priori distribution for these inputs. However, proposing a distribution a priori is a difficult task: being too conservative (*e.g.* considering large input ranges) can be detrimental to the overall efficiency, when disregarding possible input values can result in large prediction errors. A possibility to overcome the difficulties in defining the input distributions is to rely on training sets resulting from a global run of entire SoS corresponding to a sample set of the global inputs. This approach ensures the consistency between the sample sets of inputs for the individual solvers. However, this approach, that relies on the sequential nature of the directed SoS, prevents the possibility of performing parallel runs of a solver and to focus the computational resources on particular solvers demanding larger training sets to construct their surrogate models.

Recently, systems of solvers have received interest from the UQ community trying to develop efficient UQ methods for SoS. In [9], the authors proposed a method based on importance sampling to decouple the uncertainty propagation process of individual solvers in order to gain flexibility. Other recent works focused on adapting Global Polynomial Chaos (gPC) based methods to SoS, with the challenge of deriving efficient quadrature rules on intermediate inputs with a priori unknown distributions. Using the structure of SoS, the authors of [10] proposed a method for propagating uncertainty in a composite function by adapting the quadrature rule of intermediate inputs in the SoS, thus limiting the number of quadrature points with respect to a global black box approach. This work used the recursive formula for orthogonal polynomials and Lanczos algorithms. The same authors generalized this idea in [11] to a full SoS. Their approach is based on Galerkin projection methods at intermediate layers of the SoS. By solving an optimization problem, they proposed a quadrature rule for latent variables, regularized in order to promote sparsity in the weights, thus reducing the number of quadrature points. In [12], the authors tackled the problem of strongly coupled systems. Their main idea is that the dimension of the coupling variables and the amount of information that is transmitted from one solver to another is not as high as the actual inputs dimensionality. Consequently, they use Karhunen–Loève expansions to reduce the inputs space of each solver at each iteration, therefore lowering the computational cost, while propagating the uncertainty through the coupled system. In [13], the authors proposed a framework for uncertainty propagation for directed SoS. Their framework applies to intrusive and non-intrusive methods such as Monte Carlo, non-intrusive polynomial chaos and Galerkin method. In [14, 15] the framework presented in [13] is generalized to non-directed systems. Their approach relies on restriction and expansion operators adjusted to the dimension of the intermediate inputs. The UQ methods rely on a global polynomial approximation but with quadrature rules adapted to the local problems. Hence, the local quadrature rules are improved compared to the global black-box approach. From a Bayesian perspective, the authors of [16] proposed a UQ propagation framework in SoS for data assimilation in multiphysics problems.

Our approach tackles the problem from a different perspective. We introduce a new predictive model called System of Gaussian Processes (SoGP) suitable for directed systems of solvers. In our approach, a GP model is constructed for each solver of the SoS, and the global prediction is built by propagating the GPs predictions. Our framework carries similarities with Deep Gaussian Processes [17] and Multi-step ahead predictions [18], although the objectives and construction differ in our framework. A similar framework for a two-solvers problem is presented in [19]. An important contribution of our work is the formulation of suitable criteria to design efficient adaptive training strategies. The key ideas are to simultaneously exploit the advantages of a fragmented approach (low dimensionality, flexibility) when building the surrogate model of a solver, with the use of global criteria to weight the importance of each solver on the global outputs prediction error. Specifically, a

decomposition of the SoGP prediction variance is presented. It provides a ranking according to GP model contirubtion to the global error that can be used in order to enhance the overall predictive capabilities of the SoGP. This decomposition leads to efficient training algorithms that identify the GP model that should be refined in order to improve the prediction of the global outputs.

In Section 2, basic ingredients of Gaussian Processes (GP) modeling are recalled and the System of Gaussian Processes (SoGP) approximating the SoS is introduced. In Section 3, the decomposition of the prediction variance of a SoGP is presented, and some suggestions for its estimation are provided. Section 4 illustrates several strategies for the training of SoGPs, based mainly on the Maximum Mean Square Predictive Error (MMSPE) criterion, extended to the case of SoGP. Performance results on several test cases are presented in Section 5 for evaluating the performances of the proposed framework. Different structures of the SoS and solvers dependencies between inputs and outputs are contrasted. Conclusions and some perspectives are finally drawn in Section 6.

2. System of Gaussian Processes

In Section 2.1, we discuss the structure of the SoS considered in this work; this structure is subsequently exploited to construct a system of Gaussian processes (SoGP) approximating the original SoS. The essential elements of Gaussian Processes (GP) modeling are recalled in Section 2.2. The System of Gaussian Processes (SoGP) approximating the SoS is discussed in section 2.3 where we focus on the definition of the SoGP predictions to elucidate the predictive distribution. This results will serve as a basis to propose several estimates and decompositions of the predictive variance, in Section 3, and derive adaptive sampling strategies in Section 4.

2.1. Directed Systems of Solver

Generically, a SoS is an organized ensemble of connected solvers used to compute output values referred to as the quantities of interest (QoI) or global output in the following. Without loss of generality, we shall restrict ourselves to the case of scalar global output in the following. The solvers constituting the systems can be very different in terms of computational cost, inputs dimension and influence on the global output. They are organized such that outputs of a solver can be inputs of other solvers. We shall call global inputs, the inputs of the solvers that are not an output of any solver. In this work, we restrict ourselves to the particular case of *directed systems of solvers* where the solvers can be ordered along the upstream to downstream direction. Specifically, an output of a solver can only be an input of a downstream solver, such that the information (simulation results) flows in one direction only. Hence, all upstream solvers must have been run prior to running the downstream ones. This restriction rules out the case of strongly coupled solvers, which must be considered as a whole.

Fig 1 shows an example of such a directed SoS. In the plot, the boxes labeled with letters represent the constituting solvers; the arrows are used to represent the inputs (arrows coming in) and outputs (arrows coming out) connecting the solvers. In this example, the solver E is an upstream solver with respect to the solvers F, G, I and J, and a downstream solver with respect to A, B, C, and D. We remark that the directed SoS could be also partitioned into non-overlapping blocks with boundaries corresponding to logical computational barriers reflecting the structure of the system. Such a block can be composed of a single or a set of solvers, with outputs of upstream blocks (or global inputs) as inputs, and outputs being inputs of downstream blocks. A key point is that solvers constituting a block can be run independently in parallel. The example depicted in Fig 1 illustrates the non-uniqueness of the partition of the SoS into blocks, which are represented by the dashed lines

rectangles in the figure. Indeed, the solver H, belonging with the solver E to the third block, could have been also attached to the first or second blocks. In our framework, we construct a surrogate model for each solver, such that the non-uniqueness of the block decomposition is not a concern.

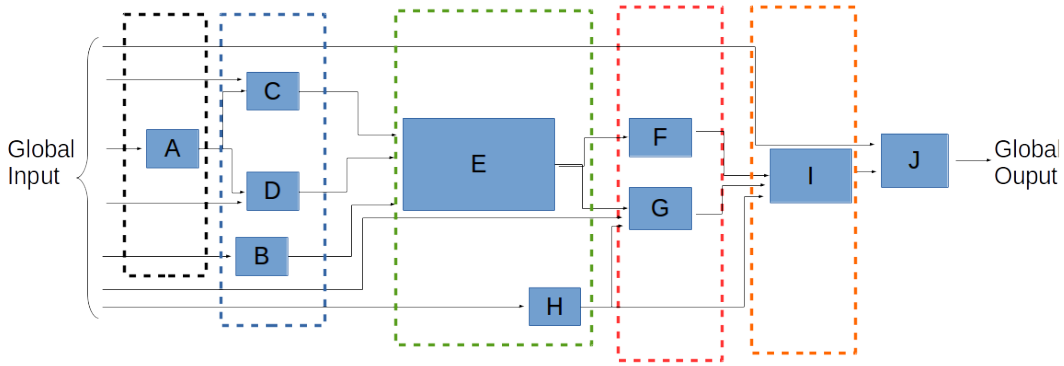


Figure 1: Example of directed SoS.

2.2. Gaussian Process models

The basic constitutive element of the SoGP are Gaussian process models [20]. GP models are probabilistic approximations of generic functions $f : \Omega \subseteq \mathbf{R}^n \mapsto \mathbf{R}^m$. For simplicity, we restrict the presentation to the case of scalar functions f, that is, $m = 1$. GP models have been widely used in uncertainty propagation [21], sensitivity analysis [6] and inverse problem [21]. We only provide a brief overview of the construction of GP models; a complete and detailed introduction to GP models can be found for instance in the reference book [22].

The GP model of f is obtained by updating a prior distribution over the space of Gaussian processes. The update uses observations of f, possibly noisy ones, and the resulting model is a Gaussian process in the sense that any set of model evaluations follows a Gaussian distribution. Consequently, the GP model is completely defined by its second order properties, namely its mean and a covariance functions. We shall consider the simplest situation of GP models having zero mean and covariance function $k(x, y)$ for prior. Then, from a set $\mathcal{A}$ of (noise free) observations of f, consisting in P pairs (x_p, y_p), with $x_p \in \Omega$ and $y_p \equiv f(x_p)$, the GP model of f is characterized by the posterior mean μ and variance σ^2 given respectively by

$$\mu(x) = \mathbf{k}_{\mathcal{A}}(x) K_{\mathcal{A}}^{-1} \mathbf{y}_{\mathcal{A}}, \tag{1}$$

$$\sigma^2(x) = k(x, x) - \mathbf{k}_{\mathcal{A}}(x) K_{\mathcal{A}}^{-1} \mathbf{k}_{\mathcal{A}}(x)^T. \tag{2}$$

In the previous expressions, we have denoted $\mathbf{k}_{\mathcal{A}}(x) = (k(x, x_1) \cdots k(x, x_p))^T$ and $K_{\mathcal{A}i,j} = k(x_i, x_j)$ the vector and matrix of the prior covariance between the point x, where f is to be predicted, and the points x_p in the observation set $\mathcal{A}$. Similarly, $\mathbf{y}_{\mathcal{A}} = (y_1 \cdots y_P)^T$ is the vector of observed values.

The posterior mean in Eq.(1) is the best prediction (in the mean squared sense) of $f(x)$; in fact, for the noise free construction above, we have $\mu(x = x_p) = y_p$ for all $x_p \in \mathcal{A}$. The prediction variance in Eq.(2)) measures the confidence in the prediction, a characterization constituting the starting points of many Bayesian optimizations and adaptive design of computer experiments. One objective of the present work is to extend these concepts to the case of SoGP.

The choice of the covariance function k is critical to obtain high-quality predictions (see for instance [23]), and it should be made dependent on the observations in $\mathcal{A}$ and prior knowledge about the function to approximate. The covariance, which in the literature is often referred to as the kernel, is classically selected by fixing some hyper-parameters that span a whole family of functions k. The selection of the hyper-parameters rests generally on cross-validation procedures or on the optimization of some quantities measuring the approximation quality, *e.g.*, the log-marginal likelihood of $\mathcal{A}$ [22]. In the present work, we considered the smooth isotropic squared exponential kernel,

$$k(x, y) = \sigma_k^2 \exp\left(-\frac{\|x - y\|^2}{2L^2}\right), \tag{3}$$

with hyper-parameters σ_k^2, the prior variance, and L, the prior correlation length. The kernel in (3) yields GP models having infinite smoothness, a characteristic that raises a lot of criticisms (see for instance [24]); it is, however, appropriate for the examples proposed below. It is also noted that all the developments proposed in the following are independent of the particular kernel family used in the GP models construction.

2.3. SoGP predictions

As discussed above, the SoGP considered in this work are obtained by substituting the solvers of a directed SoS with GP models. The focus of the present section is to define the SoGP prediction, as resulting from the composition of the GP models. The composition of GPs has been studied in machine learning as the GP-based equivalent of Neural Networks (called Deep Gaussian Processes -DGP- [17]). However, our SoGP case is different from the one usually considered in machine learning:

- there are no latent variables since all intermediate variables are observed (inputs and outputs of the constitutive solvers),
- the GPs (layers) are derived from the SoS structure, and their definition is not left to the user choice (each solver has a corresponding GP).

To illustrate the prediction of SoGP output, we consider the simple system consisting of just 2 solvers as illustrated in Fig. 2. The two solvers (f_1 and f_2 respectively) are substituted with two GPs (G_1 and G_2 respectively). Even in this trivial situation, the prediction using the SoGP is not unique; two possibilities can be readily proposed:

- Given the global input x_0, the best prediction of G_1 (the mean $\mu_1(x_0)$) can be used as the input of G_2 to retrieve its best prediction of the QoI. That is, for the SoGP of Fig. 2,

  $$f_2 \circ f_1(x_0) \approx \mu_2(\mu_1(x_0)) = \mu_2 \circ \mu_1(x_0).$$

  This approach is easily generalized to more complex SoGP and will be called the **composition of the averages** in the following. It is remarked that this composition of averages provides a deterministic prediction of the QoI, with no characterization of its uncertainty. However, it is clear that the manipulation and propagation of deterministic values in the SoGP is computationally convenient.

- Alternatively, one can keep the whole Gaussian distribution of $X_1 = \mathrm{G}_1(x_0)$ as the input of G_2, and defines the prediction as the resulting average, namely,

$$f_2 \circ f_1(x_0) \approx \mathbb{E}\left[\mathrm{G}_2 \circ \mathrm{G}_1(x_0)\right],$$

where $\mathbb{E}[\cdot]$ denotes the expectation operator. In the following, we call this approach the **averaged composition of GPs**. In general, the full distribution of the prediction is transmitted from a GP to the next, all along the SoGP. Note that in the case of the system shown in Fig. 2 it comes

$$f_2 \circ f_1(x_0) \approx E\left[\mathrm{G}_2 \circ \mathrm{G}_1(x_0)\right] = \frac{1}{\sqrt{2\pi\sigma_1^2(x_0)}} \int \mu_2(x_1) \exp\left(-\frac{(x_1 - \mu_1(x_0))^2}{2\sigma_1^2(x_0)}\right) dx_1. \quad (4)$$

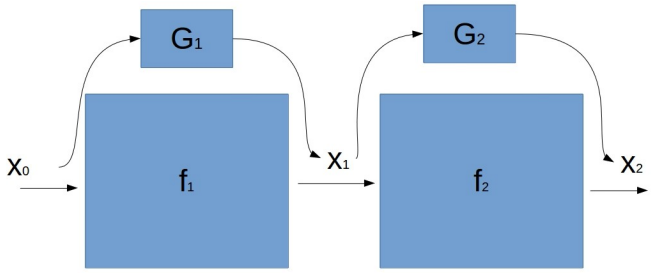


Figure 2: Example of 2 solvers directly chain and the corresponding SoGPs.

It is clear that, in general,

$$\mathbb{E}\left[\mathrm{G}_2 \circ \mathrm{G}_1(x_0)\right] \neq \mu_2 \circ \mu_1(x_0),$$

so that the composition of averages and the averaged composition of GPs are not equivalent. While the composition of the averages is computationally the fastest and the easiest to implement, the predictive distribution is lost because of the intermediate averaging of the GP outputs. On the contrary, the averaged composition of GPs propagates the full predictive distribution through the SoGP, therefore allowing to estimate the confidence in the predicted QoI. The main issue with this second approach is that even for just 2 chained solvers, the distribution of $\mathrm{G}_2 \circ \mathrm{G}_1(x_0)$ is in general not Gaussian [17, 18], preventing the derivation of explicit formulas (such as in (4)) for the corresponding prediction. The loss of Gaussianity when composing the GPs is due to the nonlinear character of the mapping between the inputs and outputs of a GP model. It is thus tempting to recover a Gaussian prediction using local linearizations of the GPs, around the inputs mean value, as proposed in [18]. For instance, the example would lead to the Gaussian approximation of the composition

$$\mathrm{G}_2 \circ \mathrm{G}_1(x_0) \approx N(\mu_2(\mu_1(x_0)), \sigma_2^2 + |\mu_2'(\mu_1(x_0))|^2 \sigma_1^2(x_0)),$$

where μ_2' is the derivative of $\mu_2(x_1)$, and $N(\mu, \sigma^2)$ denotes the normal variable with mean μ and variance σ^2. It is remarked that the mean prediction for the linearized approach coincides with the prediction using the composition of the averages. In fact, the linearization can be seen as an approximated approach to propagate variances of the outputs along the SoGP and come-up with a Gaussian prediction of the QoI. This idea is further exploited in section 3.

The distribution of the averaged composition of GPs prediction can also be recast in a (high dimensional) integral of conditional probabilities. For instance, the case of the system with four solvers shown in Fig. 3 leads to

$$p(x_4|x_0) = \int_{x_1}\int_{x_2}\int_{x_3} p(x_4, x_3, x_2, x_1|x_0)dx_1dx_2dx_3 \tag{5}$$

$$= \int_{x_1}\int_{x_2}\int_{x_3} p(x_4, x_3, x_2|x_1)p(x_1|x_0)dx_1dx_2dx_3 \tag{6}$$

$$= \int_{x_1}\int_{x_2}\int_{x_3} p(x_4|x_3)p(x_3|x_2)p(x_2|x_1)p(x_1|x_0)dx_1dx_2dx_3. \tag{7}$$

In the previous expressions, the elementary conditional densities $p(x_i|x_{i-1})$ are all Gaussian; specifically

$$p(x_i|x_{i-1}) = \frac{1}{\sqrt{2\pi\sigma_i^2(x_{i-1})}} \exp\left[-\frac{(x_i - \mu_i(x_{i-1}))^2)}{2\sigma_i^2(x_{i-1})}\right]. \tag{8}$$

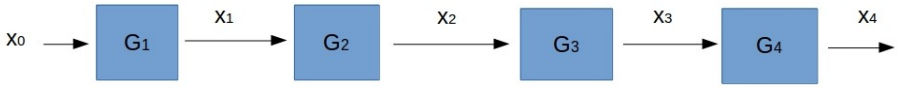


Figure 3: Example of SoGP (block view).

This expression of $p(x_4|x_0)$ and the Gaussian nature of the conditional densities (8) show that, in principle, one can accurately evaluate the averaged composition of GPs by means of tensorized Gaussian quadrature rules. However, the computational cost of tensorized quadrature rules would increase exponentially with the number of chained GPs and inputs, limiting its applicability to simple trivial systems. Sparse quadrature rules could be employed to estimate at a reduced cost the high dimensional integrals, but we found more effective to proceed by Monte Carlo sampling in the present work. Indeed Monte Carlo methods are insensitive to dimensionality and can be easily applied to SoGPs with complex structures. Specifically, for the example of Fig. 3, one generates randomly joint samples of (X_1, X_2, X_3, X_4), using the elementary Gaussian conditional probabilities in (8), in order to estimate the averaged composition of GPs prediction $\mathbb{E}[X_4]$. Note that these Monte Carlo samples can also be used to estimate the variance and (non-Gaussian) density of any of the intermediate output X_i of the SoGP, and assess their respective uncertainty as further discussed in the following.

3. Decomposition of the predictive variance

In this section, we present the decomposition of the prediction variance of a SoGP. For simplicity, we restrict ourselves to the case of n Gaussian models G_i, directly chained one after the other, with the output of G_i being the (only) input of G_{i+1}. We shall denote $x_0 \in \Omega_0$ the global input of G_1. Further, to alleviate notational burden we shall consider that $\mathrm{G}_i : \mathbb{R} \mapsto \mathbb{R}$, although the derivations below can be easily extended to more complex situations with higher dimensional inputs and outputs (see also discussion in Section 3.3).

Our objective is to decompose the SoGP prediction variance into individual contributions V_i related to the model G_i and rank the importance of the GP models in view of improving the overall predictive capabilities of the SoGP. The decomposition of the SoGP prediction variance is exploited in the next section to propose dedicated training strategies. In the following, we first define the elements V_i of the decomposition (Section 3.1) and discuss their practical estimation (Section 3.2). Two approximations of the estimator of V_i, differing in their computational cost and accuracy, are subsequently proposed in Sections 3.2.2 and 3.2.3. Finally, we discuss the decomposition of the variance in the case of directed SoGPs having more general structures in Section 3.3.

3.1. Variance Decomposition

For convenience, we set

$$\mathrm{G}_{j\mapsto i} := \mathrm{G}_i \circ \cdots \circ \mathrm{G}_j, \quad 1 \leq j < i \leq n. \tag{9}$$

With this notation, the predictive variance associated to $x_0 \in \Omega_0$ is $\mathbb{V}\left[\mathrm{G}_{1\mapsto n}(x_0)\right]$. To access the contributions of different solvers onto the predictive variance, we define $V_{1\mapsto i}(x_0)$ as variance of the expected prediction conditioned on $\mathrm{G}_{1\mapsto i}(x_0)$, that is

$$V_{1\mapsto i}(x_0) := \mathbb{V}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]\right], \quad i = 1, \ldots, n. \tag{10}$$

The variance $V_{1\mapsto i}$ is interpreted as the variance in the (final) prediction of the global output caused by the predictive variability of $\mathrm{G}_{1\mapsto i}$, that is the SoGP up to the i-th solver. Setting $V_{1\mapsto 0}(x_0) := 0$, we note that $\{V_{1\mapsto i}\}_{i=0}^{i=n}$ forms an increasing sequence from $V_{1\mapsto 0}(x_0) := 0$ to $V_{1\mapsto n}(x_0) = \mathbb{V}\left[\mathrm{G}_{1\mapsto n}(x_0)\right]$, such that $V_{1\mapsto j}(x_0) \leq V_{1\mapsto i}(x_0)$ for $0 \leq j \leq i \leq n$ and $\forall x_0 \in \Omega_0$. Therefore, we define the predictive variance incurring to the G_i as

$$V_i(x_0) := V_{1\mapsto i}(x_0) - V_{1\mapsto i-1}(x_0) \geq 0, \quad i = 1, \ldots, n. \tag{11}$$

Observing that $\mathbb{E}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]\right] = \mathbb{E}\left[\mathrm{G}_{1\mapsto n}(x_0)\right]$, Eq. (10) becomes

$$V_{1\mapsto i}(x_0) = \mathbb{E}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]^2\right] - \mathbb{E}\left[\mathrm{G}_{1\mapsto n}(x_0)\right]^2,$$

and the expression of V_i can be recast to

$$\begin{aligned} V_i(x_0) &= V_{1\mapsto i}(x_0) - V_{1\mapsto i-1}(x_0) \\ &= \mathbb{E}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]^2\right] - \mathbb{E}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i-1}(x_0)\right]^2\right], \end{aligned} \tag{12}$$

setting $\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto 0}(x_0)\right] := x_0$.

3.2. Practical estimation

3.2.1. Monte Carlo estimation

To estimate the partial variances V_i at a given $x_0 \in \Omega_0$, one could consider computing first the variances $V_{1\mapsto i}$ using (10), that is through the estimation of the variance of conditional expectations. This approach would lead to a stratified Monte Carlo (MC) method with nested loops on samples. Although the computations would involve low-cost SoGP evaluations, and would not rely on the original solvers, the stratified MC is known to be inefficient in this situation [25]. For computational efficiency, we propose to use an MC sampling strategy inspired by [26]. In view of (12), the

computation of the $V_i(x_0)$ amounts to the computation of expected value of the squared conditional expectations $\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]^2$,called $E_{1\mapsto i}(x_0)$ hereafter. The expectation $E_{1\mapsto i}(x_0)$ can be rewritten as:

$$E_{1\mapsto i}(x_0) = \mathbb{E}\left[\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]\right], \tag{13}$$

leading to the (unbiased) MC estimate

$$E_{1\mapsto i}(x_0) \approx \frac{1}{M}\sum_{j=1}^{M} Y_j Y_j', \tag{14}$$

where Y_j, Y_j' are two independent random samples of $\mathrm{G}_{i+1\mapsto n} \circ X_j$ where X_j is a random sample of $\mathrm{G}_{1\mapsto i}(x_0)$. The estimation of the $E_{1\mapsto i}$ for given $x_0 \in \Omega_0$ can be performed in parallel for different GP model i, and can eventually reuse samples X_j from one level i to another. Using the same number M of samples for all the $E_{1\mapsto i}$, and recycling samples, the estimation of the n partial variance V_i has a computational cost of the order of $\mathcal{O}(Mn)$.

As we shall see later, the training strategy may require the evaluations of the partial variances $V_i(x_0)$ at multiple input points $x_0 \in \Omega_0$. In this case, the estimator in (14) may be too expensive, in particular if the variance of $\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right]$ is large and high accuracy on $E_{1\mapsto i}(x_0)$ is demanded. We then propose in the following two approximations of $E_{1\mapsto i}$ aiming at reducing the computational cost of computing the V_i.

3.2.2. Composition of averages

Following the discussion of Section 2.3, the expected value of composed GP models can be substituted with the composition of the averaged GP predictions. Specifically, we propose to use the following approximation of the conditional average,

$$\mathbb{E}\left[\,\mathrm{G}_{1\mapsto n} \mid \mathrm{G}_{1\mapsto i}(x_0)\right] \approx \mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0), \tag{15}$$

where we have consistently denoted $\mu_{j\mapsto i} := \mu_i \circ \cdots \circ \mu_j$. Using this approximation in (12), the contribution of G_i to the total variance is approximated through

$$V_i(x_0) \approx \widehat{V}_i(x_0) = \mathbb{E}\left[\left(\mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0)\right)^2 - \left(\mu_{i\mapsto n} \circ \mathrm{G}_{1\mapsto i-1}(x_0)\right)^2\right]. \tag{16}$$

Finally, letting $\widehat{E}_{1\mapsto i}(x_0) = \mathbb{E}\left[\left(\mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0)\right)^2\right]$, we use the MC estimate

$$\widehat{E}_{1\mapsto i}(x_0) \approx \frac{1}{M}\sum_{j=1}^{M} (Y_j)^2, \tag{17}$$

where Y_j are independent random samples of $\mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0)$. Compared to the previous estimator, in (14), the composition of averages still calls for a full sampling of the whole SoGP chain to get all the V_i at given x_0. Relying on the composition of averages ($\mu_{i\mapsto n}$) instead of the composition of GP processes ($\mathrm{G}_{i\mapsto n}$) reduces the computational cost by reducing the number of random numbers to be generated and also by reducing, to some extent, the variance of the estimator with possibly a lower sampling error in the MC estimate for fixed M.

3.2.3. Linearized approximation

The MC estimation of $\widehat{V}_i(x_0)$ is still random and the sampling noise can cause problems when solving for x_0 the optimization problems associated to the training strategies introduced in Section 4. These optimization problems are non-convex and their resolution requires a large number of accurate evaluations of the $\widehat{V}_i$ at multiple x_0. This fact has motivated the second approximation of $V_i(x_0)$ that is both fast to estimate and free of sampling noise.

Starting from the expression of $V_{1\mapsto i}$ in (10), we first use (15) to obtain

$$V_{1\mapsto i}(x_0) \approx \mathbb{V}\left[\mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0)\right]. \tag{18}$$

Relying on a local linearization, we have

$$\mathbb{V}\left[\mu_{i+1\mapsto n} \circ \mathrm{G}_{1\mapsto i}(x_0)\right] \approx \left(\mu'_{i+1\mapsto n}(\mu_{1\mapsto i}(x_0))\right)^2 \mathbb{V}\left[\mathrm{G}_{1\mapsto i}(x_0)\right]. \tag{19}$$

The first order derivative of the composition of averages, $\mu'_{i+1\mapsto n}$, can be computed by chain rule differentiation or more generally by finite difference formula. In addition, it is noted that this derivative is considered at the composition of averages $\mu_{1\mapsto i}(x_0)$ rather than at $\mathbb{E}\left[\mathrm{G}_{1\mapsto i}(x_0)\right]$ in order to avoid having to estimate the average of the composition. At this point, Eq. (19) provides an approximation of $V_{1\mapsto i}(x_0)$, which, we recall, characterizes the variance induced by the GP models up to G_i. To single-out the effect of G_i and approximate $V_i(x_0)$, we finally consider

$$V_i(x_0) \approx \widetilde{V}_i(x_0) = \left(\mu'_{i+1\mapsto n}(\mu_{1\mapsto i}(x_0))\right)^2 \sigma_i^2\left(\mu_{1\mapsto i-1}(x_0)\right), \tag{20}$$

where it is recalled that σ_i^2 is the predictive variance of G_i. By definition, this definition is deterministic and does not call for any MC computations. Moreover, the approximation $\widetilde{V}_i(x_0)$ will be accurate provided that the prediction variances σ_i^2 are small. However, we stress that the estimate will be used to select new training points and Gaussian models to be improved and from this perspective, it needs not be necessarily very accurate.

3.3. Generalization

To close this section, we discuss the generalization of the proposed predictive variance decomposition and its approximations above, in the case of more complex SoGP. First, the MC estimates of $V_i(x_0)$ and $\widehat{V}_i(x_0)$ can be readily extended to the case of chained vector-valued SoGP, provided that the final prediction remains scalar, that is $\mathrm{G}_{1\mapsto n}(x_0) \in \mathbb{R}$. The linearized approximation $\widetilde{V}_i(x_0)$ can also be extended to more general chained SoGP, albeit the introduction of the gradient of $\mu_{i+1\mapsto n}$ and the covariance matrix Σ_i^2 of the predictions of G_i.

In addition, one can extend the previous concepts of predictive variance decomposition to more generic SoGP, that is not simply chained one, provided that it remains directed. Specifically, $V_{1\mapsto i}$ becomes the variance of the expected final prediction conditioned all GP predictions upstream of and including G_i, instead of $\mathrm{G}_{1\mapsto i}$. Identically, the composition of averages and the linearized approximations can be derived for more general directed SoGP by substituting the GP predictions G_j downstream of G_i with their averaged prediction μ_j. Note that introducing a tree representation of the SoGP (and SoS) may help to automate the set of GP models appearing in the conditioning of the variance contribution of a specific GP model. An example of a not simply chained SoS is provided in the result section below.

4. Training strategies

In this section, we discuss several strategies for the training of SoGPs. These adaptive training strategies are based on the classical Maximum Mean Square Predictive Error (MMSPE) reduction, which is extended to the SoGP case.

For simplicity, we consider as previously the case of n simply chained solvers, with real scalar inputs and outputs, and global input x_0 uniformly distributed in a bounded domain $\Omega_0 \subset \mathbb{R}$. We denote $\mathcal{X}_0 = \{x_0^{(l)} \in \Omega_0, l = 1, \ldots, m\}$ a uniform sample set of m global input points; for $i = 1, \ldots, n$ let $\mathcal{X}_i \doteq f_i(\mathcal{X}_{i-1})$ be the images of $\mathcal{X}_{i-1}$ by the solver f_i, such that reusing the notation of the previous section

$$x_i^{(l)} = f_{1 \mapsto i}(x_0^{(l)}), \quad i = 1, \ldots, n.$$

The Gaussian Process G_i, approximating $f_i : \Omega_{i-1} \mapsto \Omega_i$, can be constructed using the sample sets $\mathcal{X}_{i-1}$ and its image $\mathcal{X}_i$ by f_i. Space-filling techniques, such as Latin Hypercube Sampling (LHS) [27] and Sobol sequence [28], can be used to generate the driving sample set $\mathcal{X}_0$, with very satisfying results [29]. The direct application of LHS on Ω_0 will serve as a reference to be contrasted with our sampling strategies proposed below. One advantage of considering sample sets $\mathcal{X}_i$ that are the successive images $\mathcal{X}_0$, is that they implicitly follow the input distribution induced by $f_{1 \mapsto i}$, without having to estimate the distribution of $x_i \in \Omega_i$. As a result, different regions of Ω_i are sampled with a density of training points that reflects their importance. This is a desirable property as it will enforce higher accuracy for the G_i in the regions where they are likely to be queried. However, this sampling method is completely *a priori* and may not be optimal, in particular for limited size sample set, with a dominant error in less likely regions that have not been sampled. In other words, an adaptive sampling of Ω_0 can yield an error lower than for an a priori LHS and for the same computational complexity measured by the size of the samples set. In addition, adapting the input training set $\mathcal{X}_i$ to each Gaussian Processes appears as a possible way to reduce the error while minimizing the computational complexity, possibly by adapting the size of the samples sets associated to the construction of the different GP models.

4.1. Maximum Mean Square Prediction Error

Given the samples sets $\mathcal{X}_i$, possibly not image of each others and having different sizes, of points $x_i^{(l)} \in \Omega_i$, we denote

$$Q(\mathcal{X}_0, \ldots, \mathcal{X}_{n-1}) = \max_{x_0 \in \Omega_0} \mathbb{V}\left[\mathrm{G}_{1 \mapsto n}(x_0)\right], \tag{21}$$

where the GP model G_i is constructed using the training sets $\mathcal{X}_{i-1}$ *and its image* $f_i(\mathcal{X}_{i-1})$. In words, Q measures the maximum of the global prediction variance for $x_0 \in \Omega$, given the input samples sets of each GP model. The prediction variance is classically assumed to be representative of the model error $f_i - \mathrm{G}_i$. Further, the selection of the samples sets $\mathcal{X}_i$ in order to minimize Q is known in the literature as the Maximum Mean Square Prediction Error (MMSPE) criterion [2] or the minimization of the Mean Square Error (MSE) of the Best Linear Predictor [30]. Clearly, computing the $\mathcal{X}_i$ such that $Q(\mathcal{X}_0, \ldots, \mathcal{X}_{n-1})$ is minimal is a very difficult task even for fixed samples sets size, search over finite sets of candidates $x_i^{(l)}$ [31], or even reducing the search space in Ω_0 and imposing the samples sets to be the images of one to another.

4.2. Adaptive training strategies

Adaptive training strategies (or active learning methods) intend to approach the solution of the optimal sampling problem in a greedy fashion, by progressively enriching the samples sets $\mathcal{X}_i$. The generation of effective training sets is an active research area and advanced methods are available, see for instance [2, 32, 33, 34]. In the present work, we do not aim to develop an original adaptive method, but rather to propose adaptations of the MMSPE criterion-based method of [2] to systems of GP models. In the following, we propose three different strategies. They are implemented and compared in the section 5. As a side note, we remark that in the prediction of a SoGP being non-Gaussian, in general, the minimization of the MMSPE criterion is not equivalent to entropy minimization.

4.2.1. Global Composition Criterion (GCC)

Following a greedy approach, we propose to select the input point $\tilde{x}_0 \in \Omega_0$ presenting the highest global predictive variance selected to seed the enrichment of the training sets. Specifically, the seeding point is defined as

$$\tilde{x} := \arg\max_{x \in \Omega_0} \mathbb{V}\left[\mathrm{G}_{1\mapsto n}(x)\right]. \tag{22}$$

We set $\mathcal{X}_0 \leftarrow \mathcal{X}_0 \cup \tilde{x}$ and update the other input samples sets $\mathcal{X}_i \leftarrow \mathcal{X}_i \cup \{\mu_{1\mapsto i}(\tilde{x})\}$, that is using the successive composed averaged predictions applied to $\tilde{x}$. The update of the training sets for the GCC strategy is outlined in Algorithm 1. In view of (22), we call Global Composition Criterion (GCC) this training strategy.

Algorithm 1 GCC: selection of new training points using the Global Composition Criterion.

1: **procedure** SELECTGCC($\{\mathrm{G}_{i=1,\dots,n}\}$)
2: Find $\tilde{x}_0 \in \Omega_0$ solution of (22)
3: **for** $i = 0, \dots, n-1$ **do**
4: $\tilde{x}_{i+1} = \mu_i(\tilde{x}_i)$
5: **return** $\{\tilde{x}_{i=0,\dots,n-1}\}$

Once the training sets $\mathcal{X}_i$ have been enriched, one can proceed with the update of the GP models G_i. The update of the G_i using $\mathcal{X}_{i-1}$ and its image by solver f_i, the computational complexity of GCC amounts to one resolution of *all* the solvers in the system, for every new training point seed $\tilde{x}$. Note that defining the new training points by successive compositions of averages allows parallelizing the computation of their images by f_i. Defining instead $\tilde{x}_{i+1} = f_i(\tilde{x}_i)$ at line 4 of Algorithm 1 would result in a sequential update procedure as f_i must be solved before proceeding with its composition with the next solver f_{i+1}. Furthermore, numerical tests have shown that using the composition of exact images by f_i has no significant impact on the efficiency of GCC.

4.2.2. Local Contributions Criteria (LCC)

The strategy GCC is a direct adaption of the classical MMSPE criterion used to train single GP model. It misses the chain structure of the SoGP, which can incorporate solvers with very different complexity and influence on the on the global output. As a result, significant error at input points that are not necessarily the composed images of a single seed $\tilde{x} \in \Omega_0$.Therefore, we propose here to select for each GP model G_i the input point $\tilde{x}_{i-1} \in \Omega_0$ yielding the highest contribution

$V_i(\tilde{x}_{i-1})$. We recall that V_i is the contribution to the global predictive variance of GP model G_i (see Section 3.1 and Eq. (11)). Specifically, we consider

$$\tilde{x}_{i-1} = \arg\max_{x \in \Omega_0} V_i(x), \quad i = 1, \ldots, n. \tag{23}$$

Note that the search space for the $\tilde{x}_i$ is *always* Ω_0. In the following, we call LCC the training strategy based on Local Contribution Criteria in (23). Once the input points $\tilde{x}_i \in \Omega$ have been determined, we enrich the respective training sets through $\mathcal{X}_i \cup \{\mu_{1 \mapsto i}(\tilde{x}_i)\}$ as underlined by the procedure reported in Algorithm 2. Note that the training point added to $\mathcal{X}_i$ is determined by a composition of averages, $\mu_{1 \mapsto i}(\tilde{x}_i)$, and not using the composition of models, $f_{1 \mapsto i}(\tilde{x}_i)$, for computational complexity reduction purposes.

Algorithm 2 LCC: selection of new training points using Local Contribution Criteria.

```
procedure SelectLCC({G_{i=1,...,n}})
    for i = 1, ..., n do
        Find x̃_* ∈ Ω_0 solution of (23)
        x̃_{i-1} = μ_{1↦i-1}(x̃_*)
    return {x̃_{i=0,...,n-1}}
```

Because the initial seed $\tilde{x}_i$ is changing from a GP model to another, the LCC strategy does not generate enrichment points that are composed images of one another (neither by f_i or μ_i), with potentially a better reduction of the MMSPE criterion compared to GCC. Comparing further GCC and LCC, updating the GP models for the two strategies has the same computational cost that is one evaluation of every solver in the SoS to compute the image by f_i of the new point added to $\mathcal{X}_{i-1}$. Finally, note that each seed $\tilde{x}_i$ in LCC calls for the resolution of a distinct optimization problem (23), whose complexity is comparable to the *unique* optimization problem (22) of GCC. However, these optimization problems in (23) can be carried out in parallel.

4.2.3. Single Model Selection (SMS)

The GCC and LCC strategies require the evaluation of *all* the solvers and update all the GP models. In practice, especially for limited size sample sets (*e.g.* at the start of the adaptive procedure), the MMSPE $\mathbb{V}\left[\mathrm{G}_{1 \mapsto n}\right]$ can be dominated by the contributions V_i of few GP processes. In this situation, enriching a single training set $\mathcal{X}_i$, or less aggressively just a few of them, may constitute a more efficient strategy to focus the computational resources on the improvement of selected GP models, disregarding the update of relatively more accurate ones. To this end, we propose a strategy called Single Model Selection (SMS), which selects a pair of one seed point and index of the GP the featuring the largest contribution to global predictive variance. The pair solves the following optimization problem

$$(\tilde{x}, \tilde{l}) = \arg \max_{\substack{x \in \Omega_0 \\ i \in \{1, \ldots, n\}}} V_i(x). \tag{24}$$

Note again that the optimal point is sought in the global input domain Ω_0, so we have to propagate it to define the new training point of the selected model $\mathrm{G}_{\tilde{l}}$ to be improved. As for the other strategy, we enrich the input training set (and its image by $f_{\tilde{l}}$) through $\mathcal{X}_{\tilde{l}-1} \cup \{\tilde{x}_{\tilde{l}-1}\}$, where the new training point is obtained by the composition of averages $\tilde{x}_{\tilde{l}-1} := \mu_{1 \mapsto \tilde{l}-1}(\tilde{x})$. As a result of the selection

of a single model to be updated, the SMS strategy has training sets $\mathcal{X}_i$ with variable sizes and, in contrast to the other strategies, only the selected GP model $\mathrm{G}_{\tilde{l}}$ needs be updated in SMS. In addition, because of possible large heterogeneities between solvers, it may be interesting to account for the computational cost of solving f_i when selecting the new training point. This can be achieved by extending the optimization problem (24) to

$$(\tilde{x}, \tilde{l}) = \arg \max_{\substack{x \in \Omega_0 \\ i \in \{1,\dots,n\}}} -\alpha C_i, \tag{25}$$

where C_i is an estimate of the computational cost of solver i and $\alpha > 0$ a user defined constant. The procedure for selecting points in SMS is outlined in Algorithm 3.

Algorithm 3 SMS: selection of a unique new training point by the Single Model Selection.

1: **procedure** SELECTSMS($\{\mathrm{G}_{i=1,\dots,n}\}$,$\alpha$)
2: Find couple $(\tilde{x}, \tilde{l})$ solution of (25)
3: $x_{\tilde{l}-1} = \mu_{1 \mapsto \tilde{l}-1}(\tilde{x})$
4: **return** $(x_{\tilde{l}-1}, \tilde{l})$

4.3. Training algorithm

The three strategies presented above are greedy and add a single training point per samples set $\mathcal{X}_i$ (in all sets for GCC and LCC, in a single set for SMS) before updating the GP model(s). For parallelization purposes, one may be interested in adding a batch of training points instead of a single one, in order to run in parallel multiple evaluations of the model f_i. In the following, we denote N_{add} the number of training points added at a time to reduce the prediction error of the SoGP. The main difficulty in adding N_{add} points at once is that the three strategies above rely on optimization problems that will produce the same new training points unless the SoGP is updated. In other words, these strategies are sequential and it is necessary to update the SoGP for each new training point to obtain the next one.

One possibility to circumvent the sequential nature of the strategies consists in substituting the evaluations of the models f_i, at the new training points, with the current best predictions of the corresponding G_i, that is using μ_i in place of f_i. Doing so, one does not change the prediction but locally reduces (in the neighborhood of the new training point) the predictive variance σ_i. This, in turn, affects the global predictive variance and its decomposition, such that the next training points will be found at different locations. The corresponding procedure for adding a batch of N_{add} new training points is outlined in Algorithm 4. The procedure uses one of the three selection procedures (see line 5) to construct the enrichment $\tilde{\mathcal{X}}_i$ of the initial training sets, while updating the GP models G_i (see line 7) using the initial training points with their images by f_i and the enrichment points and their prediction with μ_i. Note that in the case of the SMS strategy, only $\mathrm{G}_{\tilde{l}}$ needs be updated and that one can keep the hyper-parameters of the GP models constant during these updates to further reduce the computational load. The procedure in Algorithm 4 returns the enriched sets to train each model. To this end, the images by the solvers f_i of the N_{add} new training points must be computed first, possibly in parallel, as sought by the approach. Then, the GP models can be recomputed with the exact images and a validation of the hyper-parameters.

As a final comment, we observe that the optimization problems in (22)-(25) are nonconvex, in general, and present many local optima. Furthermore, depending on the approximation of global

Algorithm 4 Training algorithm.

procedure SELECTBATCH($\{\mathrm{G}_{i=1,\dots,n}\}$, $\{\mathcal{X}_{i=0,\dots,n-1}\}$, N_{add}, [,α])
  for $i = 0, \dots, n-1$ **do**
    $\tilde{\mathcal{X}}_i = \emptyset$
  for $p = 0, \dots, N_{\mathrm{add}}$ **do**
    $\{\tilde{\mathcal{X}}_{i=0,\dots,n}\} \leftarrow \{\tilde{\mathcal{X}}_{i=0,\dots,n}\} \cup$ SelectStrategy($\{\mathrm{G}_{i=1,\dots,n}\}[, \alpha]$))
    for $i = 1, \dots, n$ **do**
      Update G_i using $(\mathcal{X}_{i-1}, f_i(\mathcal{X}_{i-1}))$ and $(\tilde{\mathcal{X}}_{i-1}, \mu_i(\tilde{\mathcal{X}}_{i-1}))$
  return $\{(\mathcal{X}_i \cup \tilde{\mathcal{X}}_i)_{i=0,\dots,n}\}$

predictive variance or contributions V_i considered, only noisy evaluations of the objective functions may be available, such that appropriate optimization procedures must be considered. Fortunately, the precise computation of the optimal points is not critical to the efficiency of the training procedure, in particular when using a batch of points. In the present work, we relied on genetic algorithms [35] to approximate the solutions of the optimization problems (22)-(25).

5. Test problems

The proposed SoGP methodology is now applied on several test-cases corresponding to different types of SoS, in terms of dependencies and structure. It is then applied to a realistic engineering system of solvers in the context of the space object reentry.

The first test-case is a simple SoS with two chained solvers each having a single input. The second test-case consists of four solvers, with 8 global inputs. The first 3 solvers are independent and have a single output constituting the inputs of the last solver. The third test-case is composed of four solvers directly chained where a solver has for inputs the output of its upstream solver plus some global inputs (overall 16 global inputs). This structure is representative of many systems of solvers used in industry.

In order to assess the accuracy and robustness of the proposed framework, a systematic comparison is performed with respect to a global GP constructed on the whole SoS considered as a black box.

In particular, we compare the following methods:

- *BB-LHS*: a global GP is built on the whole SoS considered as a black-box, using a LHS-based sampling;
- *BB-MMSPE*: a global GP is built on the whole SoS considered as a black-box, using the MMSPE training strategy;
- *SoGP-LHS*: a SoGP is built, using a LHS-based sampling;
- *GCC*, *LCC* or *SMS*: a SoGP is built, using the GCC, LCC or SMS training strategy (described in Section 4), respectively.

In all cases, the performance of the method is evaluated by computing a normalized L_2-error norm on the global output approximation. Denoting y the exact SoS global output and $\widetilde{y}$ its

Test-case	initial LHS set size	batch size N_{add}	number of repetitions
Test-Case 1	5	5	10
Test-Case 2	150	50	5
Test-Case 3	500	50	5

Table 1: Default parameters for the three test-cases.

approximation (using one of the proposed methods), the error is estimated using N independent Monte Carlo samples of the global inputs, as follows:

$$Err_{L_2}^2 \approx \frac{\sum_{i=1}^{N} \left(\widetilde{y}(x_i) - y(x_i)\right)^2}{\sum_{i=1}^{N} y(x_i)^2}, \tag{26}$$

where the $x_{1 \leq i \leq N}$ are independent Monte Carlo samples of the global inputs.

For each test-case and method, the SoGP are initialized using an initial LHS set in the global uncertain input space. In the case of the GCC, LCC, and SMS adaptive strategies, this initial LHS set is progressively enriched adding a new batch of N_{add} training points following the corresponding strategy, as discussed in section 4 (see also Algorithm 4). In addition, to assess the influence of random generation of the initial LHS sample, the numerical experiments are repeated several times. Table 1 reports the default values of the initial LHS sample size, batch size N_{add}, and number of repetitions used in the 3 test-cases presented below. In all cases the inputs are independently distributed on a uniform distribution. We also recall that the training methods do not require the same number of SoS evaluations: the GCC and LCC methods require the evaluation of the all solvers in the SoS, for each new training point, whereas the SMS only requires only the evaluation of the selected solver.

5.1. Test-Case 1

This first test-case consists in the composition of the two univariate functions f_1 and f_2 presented in Fig. 4. These functions are defined as follows:

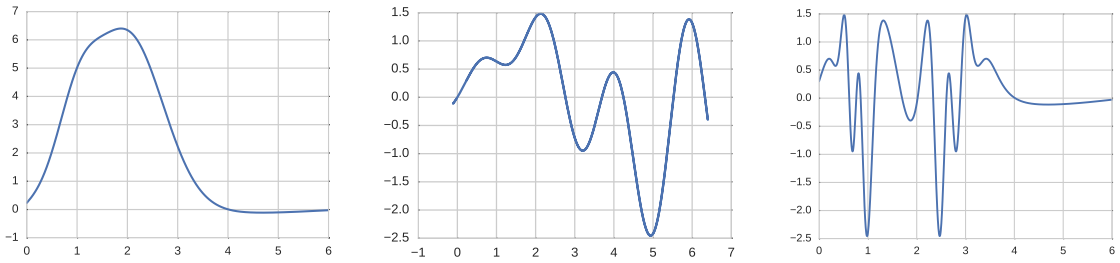
$$f_1 : x \mapsto \exp\left(\sqrt{x}\right)\sin(x) + 6\exp\left(-(x-2)^2\right) + \frac{5}{2}\exp\left(-3(x-1)^2\right), \tag{27}$$

and

$$f_2 : x \mapsto \sin(x) + 0.3 \times x \times \sin(3.4x + 0.5) \tag{28}$$

The global output is defined as $f = f_1 \circ f_2$. The global input is uniformly distributed between 0 and 6.

Figure 5 reports the results obtained for Test-Case 1. As it can be easily observed, the SoGP-based approaches systematically achieve a lower error than the global (BB) approaches, by at least one order of magnitude. Several remarks can explain this result. First, the SoGP-based approach is applied to approximate relatively simple functions, f_1 and f_2, whereas the global approaches are applied to a much more complex function $f_1 \circ f_2$, which features a highly multimodal behavior and a plateau. As a consequence, intuitively, we can expect that the approximation of $f_1 \circ f_2$ should be more challenging than approximating f_1 and f_2 solely. In general, as long as the composing functions are simpler than the final output, it is expected that a SoGP-based approach will perform better. Second, in the global approach, a part of the information available in the training set



(a) First solver ($f_1(x)$) (b) Second solver ($f_2(x)$) (c) SoS ($f_1 \circ f_2$(x))

Figure 4: SoS for the Test-Case 1.

is not used because the evaluations of the first solver are not taken into account. As a general remark, since the SoGP prediction is the composition of multiple GPs, it depends on a larger set of hyperparameters, compared to the global approaches. Having more hyper-parameters to learn could be detrimental to the computational complexity, but this drawback is largely compensated by the improved approximation capabilities brought by the extended set of hyper-parameters and the additional information brought by the intermediate variables.

Differences in performance between the global (BB) and the SoGP-based approaches are even more important for the adaptive strategies. In fact, every adaptive technique formulated in a SoGP-based framework yields better performances than the MMSPE-BB approach (which is doing slightly better than the standard LHS-BB approach). Regarding the performance of the SMS method, we stress that it only requires one solver evaluation per additional training point when the GCC and LCC methods require the evaluation of the whole SoS for each new training point. As a consequence, the results reported in Fig. 5 should be cautiously interpreted since for the SMS method has a lower computational cost compared to the other methods. The computational cost of the solvers must be considered for a fair comparison. For instance, assuming that the two solvers have the same computational cost, the computational complexity of SMS is half that of the GCC and LCC methods, and the SMS efficiency is comparable to the GCC and LCC efficiencies. In fact, in this example, the SMS method focuses essentially on the second solver and does not add many training points for learning the first solver which is much easier to approximate.

5.2. Test-Case 2

In this test-case, the SoS consists of three independent solvers, with independent global inputs, and which scalar outputs are the inputs of the downstream solver. Overall, the SoS has four solvers and eight global inputs as depicted in Fig. 6. As seen from the figure, the SoS of Test-Case 2 has two blocks consisting of solvers 1-3 and solver 4, respectively. The individual solvers are represented by the analytical functions below:

$$f_1 = (x_1, x_2, x_3) \mapsto \sin(2x_1x_2) + x_3^2, \quad f_2 = (x_1, x_2, x_3) \mapsto 3x_1^2x_2^2x_3,$$
$$f_3 = (x_1, x_2) \mapsto 2x_1 + x_2^3, \quad f_4 = (x_1, x_2, x_3) \mapsto x_1 + x_1^2 \sin(x_2)\cos(x_3).$$

The global inputs are uniformly and independently distributed between 0 and 1.

The training strategies formulated in Section 4 are compared with an additional one. Because of the SoS structure, two options are possible in the SMS strategy: i) to identify and train the most

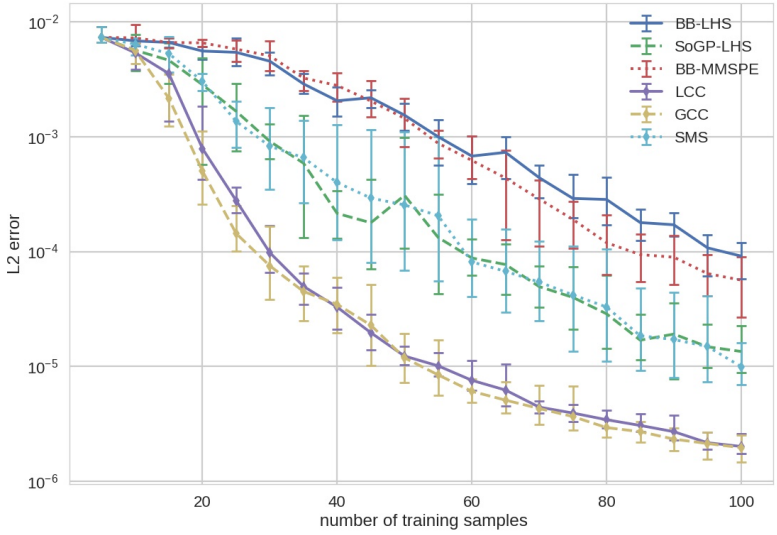


Figure 5: L_2 error norm vs the number of training samples for Test-Case 1.

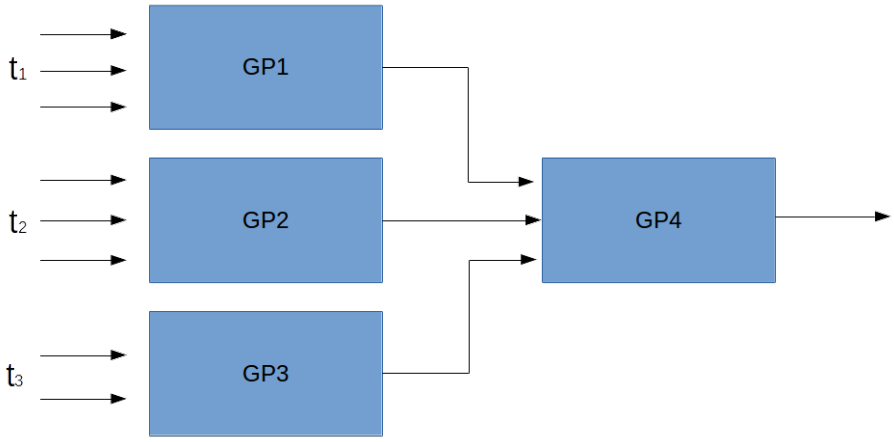


Figure 6: SoS for Test-Case 2.

unreliable solver (which is the original SMS technique); ii) to use SMS to identify and train the most unreliable block (solvers 1-3, or solver 4). This last strategy is denoted in the following as *SMS-block*. For this test case, we compare both approaches together. Figure 7 reports the errors obtained for this test-case and for the different strategies. It is first observed that for the non adapted methods, the simple SoGP method (SoGP-LHS) does not do much better than the global BB method (BB-LHS), in contrast with the previous test-case. This can be explained by the distribution of the output of

the first block solvers $f_{1,2,3}$, shown in Fig. 8, which are highly skewed, such that the training set for solver f_4 is not well adapted.

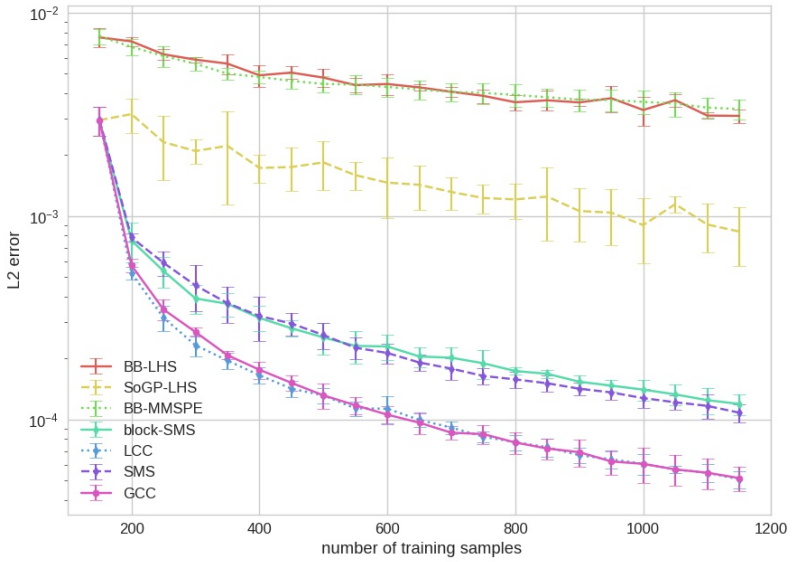


Figure 7: L_2 error norm vs the number of training samples for Test-case 2.

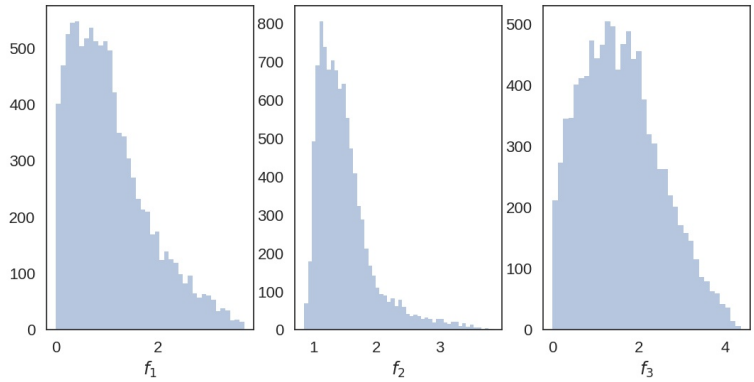


Figure 8: Marginal histograms of the outputs of solver 1, 2 and 3 (from left to right, in arbitrary unit).

Focusing on the performances of the training strategies, it is observed that the global MMSPE-BB method performs particularly bad, on this test-case, since it brings no improvement compared to

the non-adapted global BB-LHS approach. This is a well-known issue of MMSPE adaptivity in high dimension since, as mentioned in [36, 33], the MMSPE tends to place training points at the edge of the domain. This behavior deteriorates the performance of the global surrogate model when the inputs dimension is high. This issue appears to be significantly mitigated for the SoGP methods with MMSPE-based adaptive strategies (GCC, LCC, and SMS), owing to the reduced dimensionality of the individual GPs inputs. In fact, Fig. 7 shows that the SoGP-based approaches outperform the global black-box approaches, with errors reduced by up to two orders of magnitudes. Concerning the relative performances of the SoGP-based adaptive strategies (SMS, SMS-block, LCC, GCC), GCC and LCC are seen to yield similar performances. This is due to the complexity in approximating the last solver whose contribution to the global predictive variance is dominant and the two methods end up selecting the same enrichment point for the second block. Concerning the SMS-based strategies, SMS and SMS-block perform identically because, again, the last solver is the hardest to learn. Looking to Figure 7, where the errors are plotted with respect to the number of training samples, SMS, and SMS-block methods seems to have slightly lower performance than the LCC and GCC method, as this representation does not reveal differences in computational cost. In order to highlight the gain in using SMS-based techniques, which are the most computationally effective techniques for this test-case, we report in Fig. 9 the L_2 error with respect to the computational cost, computed here as the number of calls to a solver (assuming implicitly that each solver has the same evaluation cost).

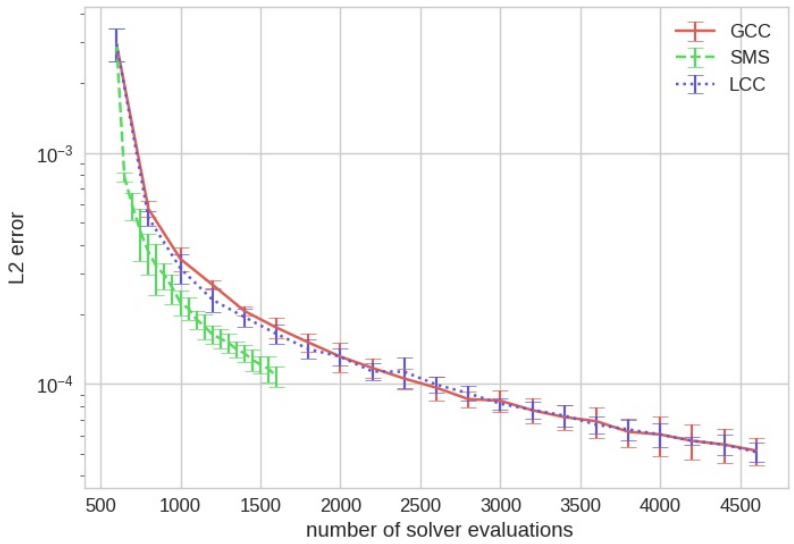


Figure 9: L_2 error norm vs the number of solvers evaluations for Test-case 2.

5.3. Influence of the training batch size N_{add}

As discussed previously, the most effective training strategy should consist in adding one training point at a time. This would oblige to run the SoS in a purely sequential manner, without thereby

exploiting the full potential of parallel computing. By setting $N_{add} > 1$ in algorithm 4, it is possible to evaluate in parallel a whole set of new training points for the same solver. This raises the question of selecting the batch size N_{add} offering the best trade-off between the cost reduction of the solver evaluation, thanks to parallelism, and a less effective adaptation due to a non-optimal sequential choice of the points. In this section, we investigate the influence of the training batch size on the SoGP performance in Test-Case 1 and 2, using the LCC and SMS strategies respectively, since they were found to be the most effective on the two respective test-cases.

The left plot in Fig. 10 shows the convergence of the L2 error in Test-Case 1 for different batch sizes $N_{\text{add}} = 1, 5, 10$ and 20, and the LCC method. Each experiment is repeted 10 times. In this cases, the asymptotic performance of the adaptive methods appears to be essentially insensitive to N_{add}, indicating that one can take advantage of the parallelism over multiple samples without affecting the convergence of the method. This conclusion is also valid for Test-Case 2 when using the SMS strategy with $N_{\text{add}} = 10$ and 50, as shown in the right plot of Fig. 10. This later example also indicates a slightly greater variability with of the error with respect to the initial LHS sample set.

In practical applications, N_{add} should be chosen according to the computational budget and the final number of training samples to be added. It should be as large as possible to exploit parallel solvers evaluations, but not too large, especially in the first stage of the adaptive procedure, to avoid amplifying the initial sample variability.

5.4. Test-Case 3

This test-case consists of a SoS with 16 global inputs and four solvers organized as depicted in Fig. 11. The first solver is a Sobol function [37] depending on five parameters, defined as follows:

$$f_1 = (x_1, x_2, x_3, x_4, x_5) \mapsto \prod_{k=1}^{5} g_k(x_k), \tag{29}$$

where $g_k(x_k) = \frac{|4x_k - 2| + a_k}{1 + a_k}$, $a = (12, 2, 3, 4, 45)$.

The second solver is an Ishigami function [38] defined as:

$$f_2 = (x_1, x_2, x_3) \mapsto \sin(x_1) + a \sin^2(x_2) + b x_3^4 \sin(x_1), \tag{30}$$

with $a = 7$ and $b = 0.1$. The input x_1 of solver f_2 is the output of the first solver, *i.e.* f_1. The other two solvers are products of polynomial functions and trigonometric functions, defined as follows:

$$f_3 = (x_1, x_2, x_3, x_4, x_5, x_6) \mapsto x_2^2 \arctan(1 - x_6) + x_3 x_4 x_5^3 + 3x_1, \tag{31}$$

where x_1 of f_3 is the output of the second solver, *i.e.* f_2;

$$f_4 = (x_1, x_2, x_3, x_4, x_5) \mapsto \sin(x_5) x_4 + x_1 x_2 + x_3, \tag{32}$$

where x_1 of f_4 is the output of the third solver, *i.e.* f_3. The global inputs are uniformly and independently distributed between 0 and 1.

This SoS differs from the previous ones, since each solver only depends on the output of an upstream solver, but has also some additional global inputs. This structure is representative of many industrial SoS, such as for example the one considered in the following section. Figure 12 presents the results for this test-case. Focusing on the errors as functions of the sample set size, shown in the left plot of Fig. 12, it is seen once more that the SoGP framework brings significant gain in accuracy

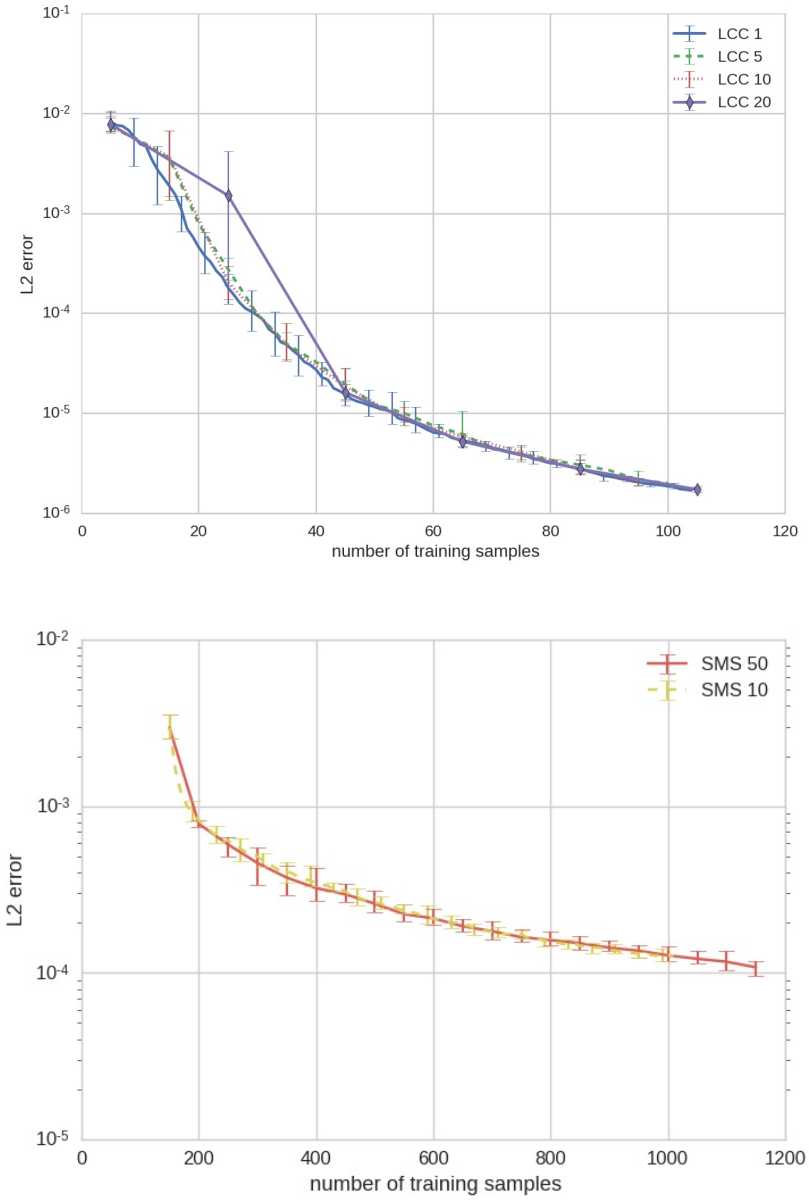


Figure 10: L_2 error norm vs the number of training samples and for different batch size N_{add} as indicated. Left: Test-Case 1 with LCC method. Right: Test-Case 2 with SMS method.

over the global black-box approaches (BB-LHS and BB-MMSPE). We again explain this gain by the relatively lower dimensionality of individual solvers inputs, compared to the global SoS. Also, we remark that the individual solvers are actually complex functions (in particular the Sobol and

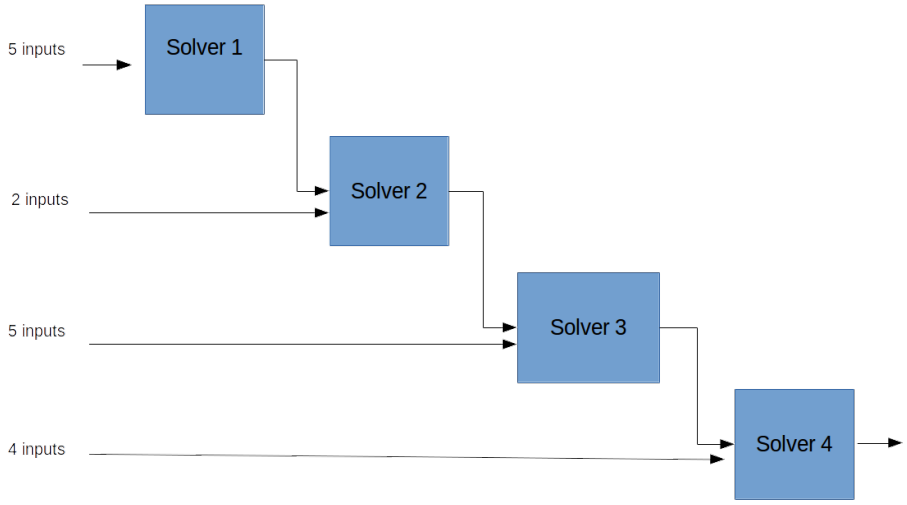


Figure 11: Test-Case 3: cascade-like SoS structure with 16 global inputs and 4 solvers.

Ishigami functions) that yield a very complex output when composed together.

In this test-case, the accuracy gain of adaptive strategies is less significant than in the previous test-cases. In particular, the global BB-MMSPE performs very poorly with a stagnating error after 1500 samples. As before, this behavior is expected in high dimensional problem [33], as in the present test-case. In contrast, the GCC, LCC and SMS strategies perform much better than BB-MMSPE, although they certainly suffer from the relatively high dimensionality of the solver inputs. This translates into a slow decay of the error with the number of training samples. As in the previous test-cases, a fair comparison between SMS, GCC and LCC should consider the lower computational cost of the SMS. This can be appreciated from the right plot of Fig. 12 which depicts the error as a function of the number of calls to a solver, assuming again that all the solvers have the same cost. The SMS is seen to yield the lowest error for given computational cost. The gain of SMS primarily comes from the identification of the most unreliable solver, here the Sobol function, which is responsible for most of the predictive variance. As a consequence, focusing the computational effort on this solver is very efficient and improves the performance of the whole SoS. This example illustrates the interest in identifying the solver yielding the most of variance in a SoS.

5.5. Application to space reentry simulation

We complete this results section by providing an example of application to an engineering application. The SoGP framework is applied to a SoS developed by ArianeGroup (AG) [39]. This SoS is used to predict the trajectory of a reentering space object and consists of two solvers: i) an upper atmosphere trajectory solver; ii) a three degree-of-freedom trajectory solver integrating a thermal module that provides the temperature of the object [40]. The upper atmosphere solver computes the trajectory from 120 km to 80 km of altitude, and the second solver propagates the trajectory from 80 km to 75 km. The output of the first solver is the object position and velocity at an altitude of 80 km, which are the inputs to the second solver. The final quantity of interest is the object temperature at 75 km of altitude, a quantity of interest for the estimation of the breakup risk. We consider a spherical pressure tank as reentering object.

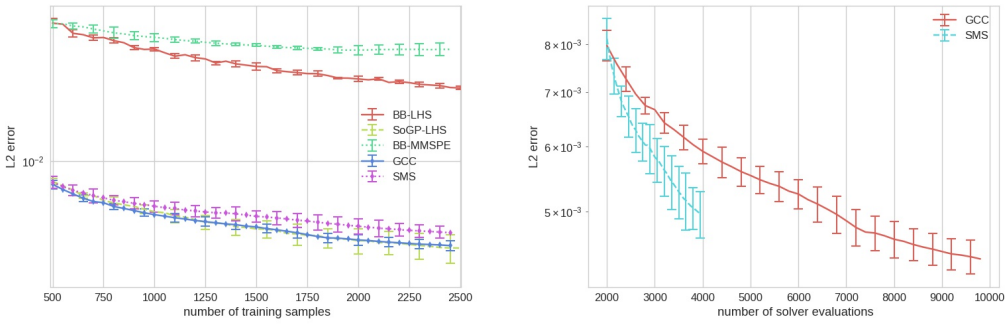


Figure 12: L_2 error norm vs the number of training samples for the Cascade-like SoS.

The structure of the SoS is illustrated in Fig. 13, where the global uncertain inputs are also depicted. these global inputs are the following: i) 6 uncertainties on the initial flight conditions (object initial position and velocity); ii) 2 uncertainties on the atmospheric temperature and density in the upper atmosphere; iii) 3 uncertainties on the material thermal and emissivity properties. In total, 11 global inputs uncertainties are considered and modeled with uniform distributions even for the material characteristic uncertainties that usually Gaussian distributed because in our case the material is not necessarily identified. In this SoS, the two composing solvers present a very different complexity. The first one is an almost linear mapping between the inputs and the outputs. It is therefore extremely easy to learn. On the contrary, the second solver displays a more complex behavior and is so harder to emulate. Finally, we mention that the computational costs of the two solvers are different with an estimated ratio of 86:14 between Solver 1 and Solver 2.

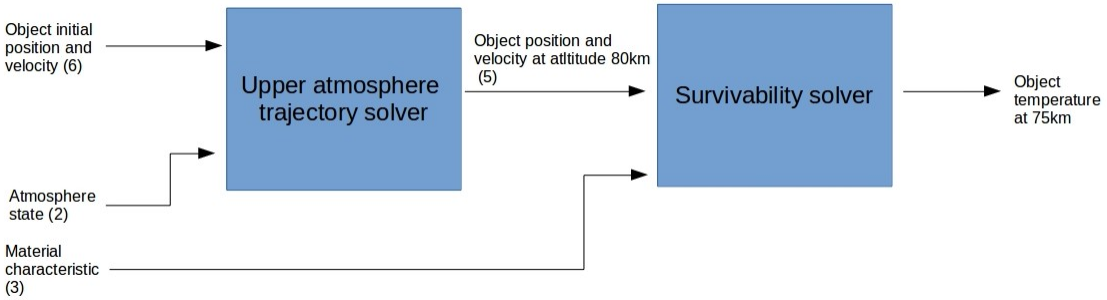


Figure 13: SoS for the space object reentry simulation.

Three approaches are contrasted on this engineering application: the global BB-LHS, the non-adaptive SoGP-LHS, and the adaptive SMS methods. The global BB-LHS and non-adaptive SoGP-LHS methods are applied on four samples LHS sample sets of size 200, 300, 500 and 600 respectively, while the SMS strategy is initialized with the LHS set of dimension 500 before selecting, in batch of size $N_{\text{add}} = 10$ new points, till an equivalent of 600 solver evaluations is reached.

The results are summarized in Fig. 14. The figure depicts the L_2 error norm as a function of the computational cost, reported as the number of solver evaluations scaled by their respective relative

costs (0.86 and 0.14 for the upstream and downstream solvers, respectively). In these experiments, the error is estimated using an independent set of 1000 LHS points. Consistently with the previous test-cases, the SoGP-LHS perform better than global BB-LHS method. This is explained by the number of inputs of the solvers which is less than the dimensionality of the global inputs. Concerning the adaptive strategy, the SMS approach presents an interesting behavior with an initial dramatic decrease in the error, until a computational cost of around 520 (*i.e.* for few adaptive batches). Beyond this point, the error decays at a much slower rate, although it remains lower than for the other LHS-based methods. This behavior can be explained as follows. Initially, the SMS strategy adds sample points to improve exclusively the second solver, which cost is only 14 % of the whole SoS chain. As result, the SMS performs extremely well during this phase. Subsequently, when the contribution of the two solvers to the overall prediction variance is balanced between the two solvers, the advantage of selecting a particular solver is less important and the limits of the MMSPE-based strategy in high dimension [33, 36] are more pronounced with a stagnation of the error decay as a result.

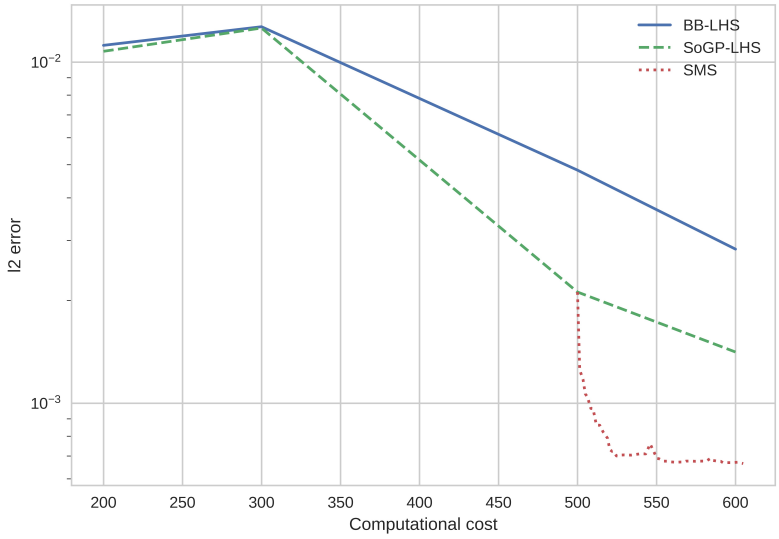


Figure 14: L_2 error norm vs the computational cost for the space object reentry SoS.

6. Conclusion

In this work, we have proposed a framework for constructing a system of Gaussian Processes (SoGP) approximating a directed system of solvers (SoS) having uncertain inputs. The SoGP substitutes GP models to the solvers constituting the SoS. The prediction is then obtained from the composition of the GP models.

We have demonstrated that this approach is potentially more efficient than the direct construction of a global surrogate of the SoS, in particular when the SoS combines individual solvers with low

dimensional inputs and simple mappings from inputs to outputs. In these situations, a reduction of orders of magnitudes in the L_2 error norm can be achieved for the same number of training samples. In addition, the SoGP construction involves an extended set of hyperparameters (typically a set for each model, instead of a unique set for the global mapping from the inputs to the outputs) with improved approximation capabilities as a result.

By design, our proposed SoGP is suited to parsimonious active learning strategies. The active learning strategy is classically based on algorithms requiring a predictive error estimation. In this work, the global predictive variance estimations of the SoGP is used to assess the precision of the prediction. A formal decomposition of the global predictive variance is derived to identify the contribution of each solver. Different approximations of the solver contributions have been proposed to improve the computational efficiency. These estimates have been used to extend the MMSPE-based adaptive algorithm, with improved performance compared to non-adaptive strategies on several test-cases. Specifically, the three training strategies proposed (GCC, LCC, and SMS) have yielded systematically better results, up to one order of magnitude error reduction, compared to the non-adapted SoGP-based approach. In particular, the SMS strategy which selects the specific solver with the highest contribution to the predictive variance is shown to be computationally very effective, in particular when some GP models of the SoGP have a dominant contribution to the prediction variance and low evaluation cost.

Numerical tests also revealed some limitations of the considered active learning strategies which call for improvements in view of applications to engineering problems. These limitations are not related to our SoGP framework but are rather generic to the MMSPE criteria and its lack of robustness, in particular for high-dimensional inputs. Potential improvements of this aspect could involve the extension of the solver (and training point) selection using the *integral* prediction variance reduction criteria, which is known to constitute a more effective approach [33, 34]. This extension would, however, require additional developments to obtain computable estimates of the solver contributions.

Future works could also consider the adaptation of the proposed SoGP framework to task-oriented problems, such as solving optimization problems by Efficient Global Optimization strategies [41], or the simulation of rare events [42]. Such developments will require objective oriented active learning strategies to improve the SoGP predictive capabilities for the input values of interest.

Acknowledgements

The authors acknowledge the financial support of Ariane Group and Région Nouvelle Aquitaine. The authors are also thankful to Charles Bertorello, Jean-Marc Bouilly and the whole Space Debris Team at Ariane Group for their help on the space debris reentry problem.

References

[1] O. P. Le Maître, O. M. Knio, Spectral Methods for Uncertainty Quantification, Scientific Computation, Springer Netherlands, Dordrecht, 2010. doi:10.1007/978-90-481-3520-2. URL http://link.springer.com/10.1007/978-90-481-3520-2

[2] J. Sacks, W. J. Welch, T. J. Mitchell, H. P. Wynn, Design and analysis of computer experiments, Statistical science (1989) 409–423.

[3] D. Xiu, G. E. Karniadakis, The Wiener–Askey Polynomial Chaos for Stochastic Differential Equations, SIAM Journal on Scientific Computing 24 (2) (2002) 619–644. doi:10.1137/S1064827501387826.
URL http://epubs.siam.org/doi/abs/10.1137/S1064827501387826

[4] D. Xiu, Numerical methods for Stochastic Computations: a spectral method approach., Princeton University Press, 2010.

[5] I. Bilionis, N. Zabaras, Multi-output local gaussian process regression: Applications to uncertainty quantification, Journal of Computational Physics 231 (17) (2012) 5718–5746.

[6] J. E. Oakley, A. O'Hagan, Probabilistic sensitivity analysis of complex models: a bayesian approach, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 66 (3) (2004) 751–769. doi:10.1111/j.1467-9868.2004.05304.x.
URL http://dx.doi.org/10.1111/j.1467-9868.2004.05304.x

[7] P. G. Constantine, E. Dow, Q. Wang, Active subspace methods in theory and practice: applications to kriging surfaces, SIAM Journal on Scientific Computing 36 (4) (2014) A1500–A1524.

[8] R. E. Bellman, S. E. Dreyfus, Applied dynamic programming, Princeton university press, 2015.

[9] S. Amaral, D. Allaire, K. Willcox, A decomposition-based approach to uncertainty analysis of feed-forward multicomponent systems, International Journal for Numerical Methods in Engineering 100 (13) (2014) 982–1005.

[10] P. G. Constantine, E. T. Phipps, A lanczos method for approximating composite functions, Applied Mathematics and Computation 218 (24) (2012) 11751–11762.

[11] P. G. Constantine, E. T. Phipps, T. M. Wildey, Efficient uncertainty propagation for network multiphysics systems, International Journal for Numerical Methods in Engineering 99 (3) (2014) 183–202.

[12] M. Arnst, R. Ghanem, E. Phipps, J. Red-Horse, Dimension reduction in stochastic modeling of coupled problems, International Journal for Numerical Methods in Engineering 92 (11) (2012) 940–968.

[13] X. Chen, B. Ng, Y. Sun, C. Tong, A flexible uncertainty quantification method for linearly coupled multi-physics systems, Journal of Computational Physics 248 (2013) 383–401.

[14] A. Mittal, G. Iaccarino, An efficient non-intrusive uncertainty propagation method for stochastic multi-physics models, arXiv preprint arXiv:1410.5419.

[15] A. Mittal, X. Chen, C. Tong, G. Iaccarino, A flexible uncertainty propagation framework for general multiphysics systems, SIAM/ASA Journal on Uncertainty Quantification 4 (1) (2016) 218–243. arXiv:https://doi.org/10.1137/140981411, doi:10.1137/140981411.
URL https://doi.org/10.1137/140981411

[16] S. Sankararaman, Uncertainty quantification and integration in engineering systems, Ph.D. thesis, Vanderbilt University (2012).

[17] A. C. Damianou, N. D. Lawrence, Deep gaussian processes., in: AISTATS, 2013, pp. 207–215.

[18] A. Girard, C. E. Rasmussen, J. Quinonero-Candela, R. Murray-Smith, O. Winther, J. Larsen, Multiple-step ahead prediction for non linear dynamic systems—a gaussian process treatment with propagation of the uncertainty, Advances in neural information processing systems 15 (2003) 529–536.

[19] S. Marque-Pucheu, G. Perrin, J. Garnier, Efficient sequential experimental design for surrogate modeling of nested codes, ArXiv e-printsarXiv:1712.01620.

[20] N. Cressie, The origins of kriging, Mathematical Geology 22 (3) (1990) 239–252. doi:10.1007/BF00889887.
URL http://dx.doi.org/10.1007/BF00889887

[21] P. Chen, N. Zabaras, I. Bilionis, Uncertainty propagation using infinite mixture of gaussian processes and variational bayesian inference, Journal of Computational Physics 284 (2015) 291–333.

[22] C. E. Rasmussen, Gaussian processes for machine learning, MIT Press, 2006.

[23] A. Krause, A. Singh, C. Guestrin, Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies, Journal of Machine Learning Research (JMLR) 9 (2008) 235–284.

[24] M. Stein, Interpolation of spatial data: some theory for kriging, Springer Science & Business Media, 2012.

[25] A. Saltelli, P. Annoni, I. Azzini, F. Campolongo, M. Ratto, S. Tarantola, Variance based sensitivity analysis of model output. design and estimator for the total sensitivity index, Computer Physics Communications 181 (2) (2010) 259 – 270. doi:https://doi.org/10.1016/j.cpc.2009.09.018.
URL http://www.sciencedirect.com/science/article/pii/S0010465509003087

[26] A. Saltelli, Making best use of model evaluations to compute sensitivity indices, Computer Physics Communications 145 (2) (2002) 280 – 297. doi:https://doi.org/10.1016/S0010-4655(02)00280-1.
URL http://www.sciencedirect.com/science/article/pii/S0010465502002801

[27] M. D. McKay, R. J. Beckman, W. J. Conover, Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245.

[28] I. Sobol', On the distribution of points in a cube and the approximate evaluation of integrals, USSR Computational Mathematics and Mathematical Physics 7 (4) (1967) 86 – 112. doi:http://dx.doi.org/10.1016/0041-5553(67)90144-9.
URL http://www.sciencedirect.com/science/article/pii/0041555367901449

[29] T. J. Santner, B. J. Williams, W. I. Notz, The design and analysis of computer experiments, Springer Science & Business Media, 2013.

[30] N. Cressie, Statistics for spatial data, John Wiley & Sons, 2015.

[31] C.-W. Ko, J. Lee, M. Queyranne, An exact algorithm for maximum entropy sampling, Operations Research 43 (4) (1995) 684–691.

[32] J. Sacks, S. B. Schiller, W. J. Welch, Designs for computer experiments, Technometrics 31 (1) (1989) 41–47.

[33] A. Krause, A. Singh, C. Guestrin, Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies, Journal of Machine Learning Research 9 (Feb) (2008) 235–284.

[34] J. Beck, S. Guillas, Sequential design with mutual information for computer experiments (mice): emulation of a tsunami model, SIAM/ASA Journal on Uncertainty Quantification 4 (1) (2016) 739–766.

[35] R. Storn, K. V. Price, Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optimization 11 (1997) 341–359.

[36] N. Ramakrishnan, C. Bailey-Kellogg, S. Tadepalli, V. N. Pandey, Gaussian processes for active data mining of spatial aggregates, in: Proceedings of the 2005 SIAM International Conference on Data Mining, SIAM, 2005, pp. 427–438.

[37] I. Sobol, Theorems and examples on high dimensional model representation, Reliability Engineering and System Safety 79 (2) (2003) 187 – 193, sAMO 2001: Methodological advances and innovative applications of sensitivity analysis. doi:https://doi.org/10.1016/S0951-8320(02)00229-6.
URL http://www.sciencedirect.com/science/article/pii/S0951832002002296

[38] T. Ishigami, T. Homma, An importance quantification technique in uncertainty analysis for computer models, in: [1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis, 1990, pp. 398–403. doi:10.1109/ISUMA.1990.151285.

[39] C. Finzi, C. Bertorello, G. Pinaud, J.-M. Bouilly, Simulation of the ariane 5 epc reentry with the fragmentation tool suite, in: 7th European Conference on Space Debris, ESA, 2017.

[40] C. Bertorello, C. Finzi, G. Pinaud, L. Rhidane, J.-M. Bouilly, Adryans® v5.0 – a 1-dimension object oriented survivability tool, in: 7th European Conference on Space Debris, ESA, 2017.

[41] D. R. Jones, M. Schonlau, W. J. Welch, Efficient global optimization of expensive black-box functions, Journal of Global optimization 13 (4) (1998) 455–492.

[42] V. Dubourg, F. Deheeger, B. Sudret, Metamodel-based importance sampling for the simulation of rare events, Applications of Statistics and Probability in Civil Engineering 26 (2011) 192.