SUQ^2: Uncertainty Quantification Queries over Large Spatio-temporal Simulations

Noel Moreno Lemus [a], Fabio Porto [a], Yania M. Souto [a], Rafael S. Pereira [a], Ji Liu[c], Esther Pacciti[b], and Patrick Valduriez [b]
[a]LNCC, DEXL, Petropolis, Brazil
[b]Inria and LIRMM, University of Montpellier, France
[c]Big Data Laboratory, Baidu Research, Beijing, China

Abstract

The combination of high-performance computing towards Exascale power and numerical techniques enables exploring complex physical phenomena using large-scale spatio-temporal modeling and simulation. The improvements on the fidelity of phenomena simulation require more sophisticated uncertainty quantification analysis, leaving behind measurements restricted to low order statistical moments and moving towards more expressive probability density functions models of uncertainty. In this paper, we consider the problem of answering uncertainty quantification queries over large spatio-temporal simulation results. We propose the SUQ^2 method based on the Generalized Lambda Distribution (GLD) function. GLD fitting is an embarrassingly parallel process that scales linearly to the number of available cores on the number of simulation points. Furthermore, the answer of queries is entirely based on computed GLDs and the corresponding clusters, which enables trading the huge amount of simulation output data by 4 values in the GLD parametrization per simulation point. The methodology presented in this paper becomes an important ingredient in converging simulations improvements to the Exascale computational power.

1 Introduction

The rapid growth of high-performance computing combined with recent advances in numerical techniques increases the accuracy of numerical simulations. This leads to practical applicability in models for predicting the behavior of weather, hurricane forecasts [1] and subsurface hydrology [2], just to name a few, positioning simulations as increasingly important tools for high-impact predictions and decision-making applications.

In order to reach higher simulation accuracy of reproduced phenomena, the scientific community is leaving behind the traditional deterministic approach, which offers point predictions with no associated uncertainty [3], to move towards Uncertainty Quantification (*UQ*) as a common practice over simulation results analysis. Arguing for improvements in simulation accuracy, by the assessment of uncertainty quantification of simulation results consider the extra knowledge a scientist acquires whenever the simulation behaviours become more

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

evident over different scenarios. The increase in simulation scenarios call for more computing power. In a subsurface seismic domain, for example, a simulation computes the wave velocity at each point of an area. A scientist is interested in a particular region of the space where a salt dome is located. She may issue a query filtering only that spatial region and checking how precise is the velocity field in that area. In order to achieve that, she could quantify the uncertainty involved in the region of interest. This type of simulation that evaluates the uncertainty by analyzing its output is referred to as *forward propagation*. In this paper, we focus on answering uncertainty quantification queries over large spatio-temporal simulations (UQ^2STS). The UQ^2STS problem is challenging as: (1) it requires analyzing large amounts of simulation output data; (2) the uncertainty at each point may exhibit patterns too complex to be captured by low order statistical moments (such as mean and standard deviation); (3) the uncertainty behavior may vary along a simulation spatio-temporal region leading to a complex data pattern to be modelled and an uncertainty expression of difficult interpretation. Thus, the problem involves conceiving a method to accurately and efficiently solve the UQ^2STS.

We propose the SUQ^2 method to solve this problem. SUQ^2 is based on the adoption of the generalized lambda distribution (*GLD*) PDF type as a model of the uncertainty at each simulation computation point, which solves issue (2). Its uniform representation reduces significantly the computation of the necessary fitting function. Furthermore, by adopting a single function type, we could run clustering algorithms on the GLD parametrization, electing a representative data distribution for a large set of simulation points. This represents a huge saving in data storage, which solves issue (1). Finally, the cluster representatives are used to composed a mixture of GLDs and to measure the information entropy of the UQ^2STS, which solves issue (3). We illustrate the adoption of the SUQ^2 method with a case study in seismology.

In our previous work [12], we designed a system to efficiently compute PDFs in large saptial datasets. The system implements a Spark dataflow to streamline the huge amount of PDF fitting computation. Our work extends the results of this work by adopting the GLDs as a generic model for data distribution, avoiding testing for different distribution types and uniformizing the computation of mixed PDFs in spatial-temporal regions.

To the best of our knowledge, the first effort to use the *GLD* to model uncertainty in data is the work of Lampasi et. al. [8], followed by Movahedi et. al. [9] for a task involving the computation of results reliability. The adoption of mixture of *GLDs* is motivated by the work of Ning et. al. [10]. Algorithms to use the mixture of *GLDs* to model datasets have been deployed with the **GLDEX** R package. Wellmann et al. [11] propose to use information entropy as an objective measure to compare and evaluate model and observational results. Our SUQ^2 method combines these techniques.

The rest of the paper is organized as follows: Section 2 gives the problem formalization and introduces the *GLD* function. Section 3 presents the SUQ^2 method and the workflow to solve the UQ^2STS problem. Section 4 gives an experimental evaluation with a use case in seismology. Section 5 concludes.

2 Preliminaries

In this section we define some basic concepts needed for the rest of the paper. We first formalize the problem. Next, we present the Generalized Lambda Distribution function, including a discussion on its shape and the mixture of GLDs.

A simulation is a combination of a numerical method implementing a mathematical model and a discretization that enables to approximate the solution in points of space-time. A simulation can be used for two different types of problems: *forward* or *inverse*. *Forward problems* study how uncertainty propagates through a mathematical model. In a simulation, a spatio-temporal domain is represented by a grid of positions $(s_i, t_j) \in \mathcal{S} \times \mathcal{T} \subseteq \mathbb{R}^3 \times \mathbb{R}$, where values of a quantity of interest (QoI), such as velocity, are computed. In a parameter sweep application, a simulation is executed multiple times, each with a different initial configuration, leading to multiple occurrences for a given domain position, in order to explore the simulation behavior under different scenarios.

A simulation can be formally expressed as $\boldsymbol{q} = \mathcal{M}(\boldsymbol{\theta})$ where: $\boldsymbol{\theta} \in \mathbf{R^n}$ is a vector of input parameters of the model; $\mathcal{M}$ is a computational model, and $\boldsymbol{q} \in \mathbf{R^k}$ is a vector that represents quantities of interest (*QoI*). In a *forward problem*, the parameters $\boldsymbol{\theta}$ are given and the quantities of interest $\boldsymbol{q}$ need to be computed. In stochastic models, at least one parameter is assigned to a probability density function (*PDF*) or it is related to the parameterization of a random variable (*RV*) or field, causing $\boldsymbol{q}$ to become a random variable as well.

In order to estimate a stochastic behavior of the output solution $\boldsymbol{q}$ in terms of input uncertainties $\boldsymbol{\theta}$, sampling methods analyze the values of $\mathcal{M}(\boldsymbol{\theta})$ at multiple sampled conditions in the Θ space (called stochastic space) directly from numerical simulations. Methods like Monte Carlo *(MC)* are used to randomly sample in the stochastic space, and hence many sample calculations are required to achieve a convergence of stochastic estimations. As a result, the method returns multiple realizations of $\boldsymbol{q}$. Then, other methods to measure the uncertainty need to be applied.

In a more general case, the computational model $\boldsymbol{q} = \mathcal{M}(\boldsymbol{\theta})$ represents the spatio-temporal evolution of a complex systems, and the *QoI* $\boldsymbol{q}$ can be represented as $\mathbf{Q} = (\mathbf{q}(s_1, t_1), \mathbf{q}(s_2, t_2), ..., \mathbf{q}(s_n, t_n))$, where: $(s_1, t_1), (s_2, t_2), ..., (s_n, t_n) \in \mathcal{S} \times \mathcal{T} \subseteq \mathbb{R}^3 \times \mathbb{R}$ represents a set of distinct spatio-temporal locations, and $\mathbf{q}(s_i, t_j)$ represents a value of the *QoI* at the spatio-temporal location (s_i, t_j).

In a stochastic problem, on each spatio-temporal location (s_i, t_j) we have many realizations of $q(s_i, t_j)$ that can be represented as a vector $< q(s_i, t_j) >$. In this context, it is frequent that more than 10^4 simulations are performed while exploring the model parameter space, which leads the output dataset to have a size of order $N_s \times N_t \times N_{sim}$, where N_s is the number of spatial locations, N_t is the number of time steps, and N_{sim} is the number of simulations. An example of the volume of data generated by these simulations is given in the experimental evaluation (see Section 4), where the output dataset is about 2.4 TB.

A simple approach to solve a spatio-temporal UQ query is to consider a simple aggregation query, computing the mean and standard deviation on the selected spatio-temporal region. This approach, albeit being simple and fast, is unable to capture patterns exhibited by the data distribution of complex phenomena. The solution we propose in [12] adopts probability density function (PDF) as a more accurate modeling data distribution at each point. However, the adoption of PDFs brings its own challenges. First, as the uncertainty may vary in different regions of the simulation, one needs to try multiple function types, such as Gaussian, Logarithm, Exponential, etc, at each spatial position to find the one closest to its data distribution. This leads to a huge computation cost for each simulation spatial position. Moreover, as a region may be defined by different PDF types, answering solving a (UQ^2STS) requires dealing with heterogeneous function types, making it more costly and harder to interpret the results. Thus, at the basis of the SUQ^2 method is the adoption of the GLD PDF type, which is presented in the next section.

2.1 Generalized Lambda Distribution

The Generalized Lambda Distribution (GLD) has been applied to fitting phenomena in many fields with very good results. It was proposed by Ramberg and Schmeiser in 1974 [14] as an extention of the Tukey's distribution, and it is tuned to represent different data distributions through the specification of λ parameter, where λ_1 and λ_2 determine location and scale parameters, while λ_3 and λ_4 determine the skewness and kurtosis of the $GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$. The Ramberg and Schmeiser proposal is known as *RS* parameterization. The *RS* parametrization has some constraints with respect to the values of (λ_3, λ_4), see [4]. To circumvent those constraints, Freimer et al. [5] introduced a new parameterization called *FKML*: $Q_{FMKL}(y|\lambda_1, \lambda_2, \lambda_3, \lambda_4) = \lambda_1 + \frac{1}{\lambda_2}\left[\frac{y^{\lambda_3}-1}{\lambda_3} - \frac{(1-y)^{\lambda_4}-1}{\lambda_4}\right]$. As in the previous parameterization, λ_1 and λ_2 are the location and scale parameters, but in this one λ_3 and λ_4 are the tail index parameters. The advantage over the previous parameterization is that there is only one constraint on the parameters, i.e. λ_2 must be positive.

Both representations (i.e. *RS* and *FMKL*) can present a wide variety of shapes and therefore are utilized in practice; however, generally the *FMKL GLD* is preferred due to the ease in its use [6]. In this paper, we opt for

using the *FMKL GLD* representation.

2.1.1 Shapes of GLD

A *GLD* can describe a variety of shapes, such as U-shaped, bell shaped, triangular, and exponentially [7]. At the same time it provides good fits to many well know distributions.

These *GLD* properties are important to the SUQ^2 method for two reasons. First, no previous knowledge is needed to fit the *GLD* to a dataset. Second, *GLDs* can be comparatively assessed; grouped based on their shapes, which enables running clustering algorithms, electing a representative distribution, and synthesizing the data in a cluster.

The shape of a *GLD* depends on its λ values. In the case of the *FMKL GLD* parameterization, Freimer et al. [5] classify the shapes into five categories depending on the variety of distributions, which can be represented by several combinations of the shape parameters λ_3 and λ_4.

The ability to model different shapes is critical to the SUQ^2 approach as it is the basis for the clustering algorithms (see Section 3.2).

2.1.2 GLD mixture

In general, a *mixture distribution* is the probability distribution of a random variable that is derived from a collection of other random variables. Given a finite set of *PDFs* $\{p_1(x), p_2(x), \ldots, p_n(x)\}$, and weights $\{w_1, w_2, \ldots, w_n\}$ such that $w_i \geq 0$ and $\sum w_i = 1$, the mixture distribution can be represented by writing the density $f(x)$ as a sum (which is a convex combination): $f(x) = \sum_i^n w_i p_i(x)$. Extending this concept to *GLD*, the mixture distribution can be represented as: $f(x) = \sum_i^n w_i GLD_i(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$. This model is used in Section 3.3 to characterize the uncertainty in a spatio-temporal region.

3 Simulation Uncertainty Quantification Querying (SUQ^2)

In a stochastic problem, on each spatio-temporal location (s_i, t_j) we have many realizations of $q(s_i, t_j)$. A schema to store this information in a relational database can be:

$$S(s_i, t_j, simId, q(s_i, t_j)) \tag{1}$$

where $simId$ represents the *id* of one simulation (realization).

In this section, we first show how to fit a GLD to a spatio-temporal dataset. Next, we present the clustering of GLDs using the lambda parameters. Then, we present how to compute the uncertainty in a region using a mixture of GLDs and information entropy. Figure 1 shows the SUQ^2 method represented by a workflow and divided into three main steps: fitting process, clustering and UQ analysis.

3.1 Fitting a GLD to a spatio-temporal dataset

Given a random sample $q_1, q_2, q_3, ..., q_n$, the basic problem in fitting a statistical distribution to these data is the distribution from which the sample was obtained. In our approach we divide this process into three steps: (1) fiting the *GLD* to the data; (2) validating the resulting *GLD*; (3) evaluating the quality of the fit.

Algorithm 1 realizes Step 1. Before starting the fitting process, we need to group all the simulation values that correspond to the same spatio-temporal location (s_i, t_j). As a result, we get a new dataset $S^*(s_i, t_j, < q_1, q_2, .., q_n >)$, where $q_i, 1 \leq i \leq n$, represents a vector of all the values of q at point (s_i, t_j). This process is efficiently computed according to the approach developed in [12].

For each spatio-temporal location $(s_i, t_j) \in \mathcal{S} \times \mathcal{T}$, we use a function of the GLDEX R package [7], to fit the *GLD* to a vector $< q_1, q_2, .., q_n >$, line 2. This is an embarrassingly parallel computation method, which we adopt in [12].

Once the fitting process in Step 1 has been applied, a fitted GLD is associated to each simulation spatio-temporal position. The schema in Equation 1 is modified to accommodate the GLD parameters in place of the list of simulation values: $S(s_i, t_j, GLD_i, j(\lambda_1, \lambda_2, \lambda_3, \lambda_4))$.

Finally, we need to evaluate the fit quality, which assesses whether the *GLD* probability density function (PDF) correctly describes the dataset. We use here the Kolmogorov-Smirnov test (KS-test), that determines if two datasets differ significantly. In this case, the datasets are the original dataset and a second one generated using the fitted GLD. As a result, this test returns the p-value, line 5. If the p-value is bigger than 0.05, lines 6-7, we store the lambda values of those *GLDs*.

Algorithm 1 Fitting the GLD to a spatio-temporal dataset

```
1: function GLDFIT(S(s_i, t_j, < q_1, q_2, ..., q_n >))
2:     < λ_1, λ_2, λ_3, λ_4 > ← FIT.GLD.LM(< q_1, q_2, ..., q_n >)
3:     isValid_(s_i,t_j) ← VALIDITYCHECK(< λ_3, λ_4 >)
4:     if isValid_(s_i,t_j) then
5:         [pvalue, D]_(s_i,t_j) ← KS(< λ_1, λ_2, λ_3, λ_4 >_(s_i,t_j))
6:     end if
7:     if pvalue_(s_i,t_j) > 0.05 then
8:         STORELAMBDAS(< λ_1, λ_2, λ_3, λ_4 >, s_i, t_j)
9:     end if
10: end function
```

3.2 Clustering the GLD based on its lambda values

In Section 2.1.1, we discussed the two most important parameterizations of the *GLD* and selected *FMKL* to be used for the rest of the paper. In this parametrization, λ_1 represents the location of the *GLD* and is directly related to the mean of the distribution. λ_2 is the scale, directly related to the standard deviation, and λ_3 and λ_4 represent the left and right tails of the distribution. Combinations of λ_3 and λ_4 can be used to estimate the skewness and kurtosis of the distribution.

As λ_2 defines the dispersion, and λ_3 and λ_4 the shape of a *GLD*, the combination of these parameters determine the quantification of the uncertainty, from the *GLD* point of view.

According to Lampasi et al. [8], a particular $GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ can be rewritten as:

$$GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4) = \lambda_1 + \frac{1}{\lambda_2} GLD(0, 1, \lambda_3, \lambda_4) \tag{2}$$

In Equation 2, the first term applies λ_1 while the second involves the remaining parameters. Then, we can apply clustering algorithms only considering parameters in the second term: λ_2, λ_3 and λ_4. The clustering algorithm would be applied in two steps 1. The first clusters based on the λ_2 values, according to the curve dispersion. Next, for each cluster obtained in the first step, we cluster again according to parameters λ_3 and λ_4, which are the parameters that define the shape of the distribution.

Then, in this step of our workflow, we cluster the *GLDs* using λ_2, λ_3 and λ_4 values. The final result of this step is:

$$S_{\mathcal{C}}(s_i, t_j, GLD_k, clusterID) \tag{3}$$

where $clusterID$ represents the ID of the cluster to which the *GLD* at the spatio-temporal location (s_i, t_j) belongs. With the domain modeled by clustered *GLDs*, we can use this result to characterize the uncertainty in a

Algorithm 2 Clustering the GLD based on its $\lambda_{(2,3,4)}$ values.

1: **function** GLDCLUSTERING($S(s_i, t_j, < 0, \lambda_2, \lambda_3, \lambda_4 >)$)
2: $S(s_i, t_j, clusterID_I) \leftarrow$ FIRSTSTEP($S(s_i, t_j, \lambda_2)$)
3: **for each** $clusterID_I$ **do**
4: $S(s_i, t_j, clusterID_{II}) \leftarrow$ SECONDSTEP($S(s_i, t_j, < \lambda_3, \lambda_4 >), S(s_i, t_j, clusterID_I)$)
5: **end for**
6: **end function**

Algorithm 3 GLD mixture in a region $(\mathcal{S}_i \times \mathcal{T}_j)$

1: **function** GLDMIXTURE($\mathcal{S}_i \times \mathcal{T}_j, C_{\mathcal{S}_i \times \mathcal{T}_j}$)
2: **for each** p_i in $(\mathcal{S}_i \times \mathcal{T}_j)$ **do**
3: $c \leftarrow cluster(p_i)$
4: $w_c = w_c + 1$
5: $N = N + 1$
6: **end for**
7: **return** $\frac{1}{N} \sum_{c}^{C_{(\mathcal{S}_i \times \mathcal{T}_j)}} w_c * c.getGLD()$

Algorithm 4 Information Entropy in a region $(\mathcal{S}_i \times \mathcal{T}_j)$

1: **function** GLDMIXTURE($\mathcal{S}_i \times \mathcal{T}_j, C_{\mathcal{S}_i \times \mathcal{T}_j}$)
2: **for each** p_i in $(\mathcal{S}_i \times \mathcal{T}_j)$ **do**
3: $c \leftarrow cluster(p_i)$
4: $w_c = w_c + 1$
5: $N = N + 1$
6: **end for**
7: $p_c(s,t) = \frac{w_c}{N}$
8: $H(s,t) \leftarrow -\sum_{c=1}^{C} p_c(s,t) \log p_c(s,t)$
9: **return** $H(s,t)$

particular spatio-temporal region or to measure numerically the corresponding uncertainty. In Sections 3.3 and 3.4, we describe how these approaches are implemented (see Figure 1).

3.3 Use of GLD mixture to characterize uncertainty in a spatio-temporal region

One of the main advantages of assessing the complete probability distribution of the outputs is that we can use the *PDFs* to answer queries. If we consider that the clustering of GLD has good quality, we can pick the GLD at the centroid of each cluster as a representative of all its members. In this context, in a particular spatio-temporal region, each cluster may be qualified with a weight given by:$w_k = \frac{1}{N} \sum_{i=1}^{S} \sum_{j=1}^{T} w(s_i, t_j)$, where: $w(s_i, t_j) = \begin{cases} 1 & \text{if } clusterID(s_i, t_j) = k \\ 0 & \text{otherwise} \end{cases}$ and *N* is the number of points in the region $(\mathcal{S}_i \times \mathcal{T}_j)$.

The weight w_k is the frequentist probability of occurrence of the cluster *k* in the region, and complies with the conditions defined in Section 2.1.2 that $w_k \geq 0$ and $\sum w_k = 1$.

Remember that the mixture of the GLDs can be written as $f(x) = \sum_{k=1}^{K} w_k GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$. So, if we have the weights and a representative GLD for each cluster, we have the mixture of GLD that characterizes the uncertainty in the spatio-temporal region $(\mathcal{S}_i \times \mathcal{T}_j)$. The GLD mixture process is summarized in Algorithm 3, which receives a spatio-temporal region and a *clusterId* associated to each spatio-temporal point. In the main loop, lines 3 to 5, the algorithm increments the number of occurrences for each clusterId and the total number of points. At line 7, the mixture expression is returned.

3.4 Information entropy as a measure of uncertainty in a spatio-temporal region

Now, what happens if we want to measure the uncertainty quantitatively? The information entropy is useful in this context. We use the different clusters we got in Section 3.2 as the different outcomes of the system. The information entropy is computed as follows $H(s,t) = -\sum_{c=1}^{C} p_c(s,t) \log p_c(s,t)$, where c represent a particular cluster in the set of clusters C, and $p_c(s,t)$ represents the probability of occurrence of the cluster c in the spatio-temporal region (s,t).

Algorithm 4 computes the information entropy in a region $C_{(\mathcal{S}_i \times \mathcal{T}_j)}$. In lines 2 to 7, we assess the probability of each cluster in the region. Using this result, we can evaluate the information entropy $H(s,t)$, line 8, and finally, return the result in line 9.

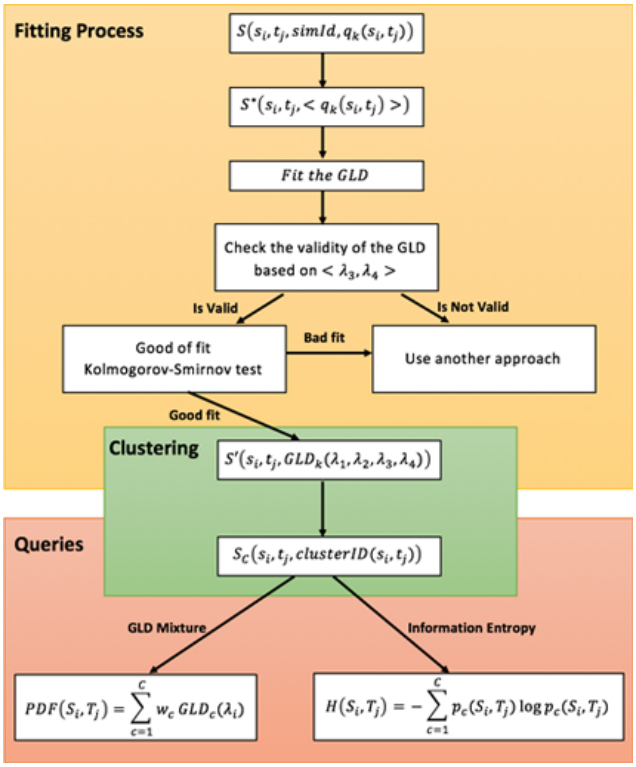

Figure 1: SUQ^2 workflow divided into three steps, (a) fitting process, (b) clustering of the GLDs and, (c) queries over the results of the clustering process.

4 Experimental Evaluation

In this section, we first introduce the data used in our experiments. Next, we discuss how we apply the fitting and clustering techniques over the experiment dataset. Then, we present the queries used in the performance evaluation and discuss the results expressed as a mixture of GLDs and values computed using information entropy.

As a case study, we use the HPC4e seismic benchmark, a collection of four 3D models and sixteen associated tests [1] to generate the data of a cube. The models include simple cases that can be used in the development stage of any geophysical imaging practitioner as well as extremely large cases that can only be solved in a reasonable time using supercomputers. The models are generated based on the required size by means of a Matlab/Octave script. The tests can be used to benchmark and compare the capabilities of different and innovative seismic modelling approaches, thus simplifying the task of assessing the algorithmic and computational advantages.

4.1 Dataset

In the HPC4e benchmark, the models are designed as sets of 16 layers with constant physical properties. The top layer delineates the topography and the other 15 delineate different layer interface surfaces or horizons. To

[1] The benchmark can be freely downloaded from https://hpc4e.eu/downloads/

generate a single cube with dimensions $250 \times 501 \times 501$ we can use the values provided in the benchmark. For example, to generate a cube in the $v_p(m/s)$ variable we can use the fixed values of Table 2. The first slice of this cube is shown in Figure 2.

Layer	$v_p(m/s)$	**Layer**	$v_p(m/s)$
1	1618.92	9	2712.06
2	1684.08	10	2532.2
3	1994.35	11	2841.03
4	2209.71	12	3169.31
5	2305.55	13	3169.31
6	2360.95	14	3642.28
7	2381.95	15	3659.22
8	2223.41	16	4000.00

Table 2: Values of v_p used in the generation of a single velocity field cube.

Layer	**PDF Family**	**Parameters**	**Layer**	**PDF Family**	**Parameters**
1	Gaussian	[1619, 711.2]	9	Exponential	[3949, 394.9]
2	Gaussian	[3368, 711.2]	10	Exponential	[5983, 711.2]
3	Gaussian	[8839, 711.2]	11	Exponential	[3520, 352.0]
4	Gaussian	[7698, 301.5]	12	Exponential	[3155, 315.5]
5	Lognormal	[7723, 294.7]	13	Uniform	[2541, 396.4]
6	Lognormal	[7733, 292.2]	14	Uniform	[2931, 435.3]
7	Lognormal	[7658, 312.1]	15	Uniform	[2948, 437.0]
8	Lognormal	[3687, 368.7]	16	Uniform	[3289, 471.1]

Table 3: PDFs and the parameters used to sample the v_p attribute, to generate n velocity models.

As our purpose is to study the uncertainty in the simulation output, we need the input $v_p(m/s)$ to present a stochastic behavior. We model the input according to the distributions depicted in Table 3. Next, using a Monte Carlo method, we generate a sampling of 1000 realizations of the $v_p(m/s)$ variable and use a Matlab script provided by the HPC4e benchmark to generate the cube data. We perform the simulations 1000 times, one for each realization, and generate 1000 cubes (230 GB) as output. The generated cubes are $250 \times 501 \times 501$ multi-dimensional arrays. In order to simplify the computational process and visualize the results, we select the slice 200 to be used in our experiments. Then, we have 1000 realizations of a slice with size of 250×501. The data schema in Equation 1 can be simplified as we only have two spatial dimensions and no time domain. Thus, the dataset can be represented as $S(x_i, y_j, simId, v_p(x_i, y_j))$. In this new representation, (x_i, y_j) is the 2D coordinates and $v_p(x_i, y_j)$ is the velocity value at point (x_i, y_j). *simId* represents the Id of the simulation, ranging from 1 to 1000.

4.2 Fitting the GLD

The first step is to find the *GLD* that best fits the dataset at each spatial location. Running Algorithm 1, we get a new 2D array:

$$S'(x_i, y_j, GLD(\lambda_1, \lambda_2, \lambda_3, \lambda_4)) \tag{4}$$

The raw data is significantly reduced and the new dataset is characterized by four lambda values at each spatial location.

To check how good is the fit, we use the *ks.test* algorithm included in the R-package *stats* [13], which return the *p-value* at each spatial location. Our results show that the fit of the GLD is acceptable in most cases (*p-value* > 0.05), in 82 % of the spatial locations (see Figure 3). In the 18% regions where the GLD modeling was not acceptable, some *GLD* extensions proposed in [4] could be used. Since the main purpose of this paper is to demonstrate the usefulness of the *GLD* in *UQ*, this particular problem is beyond our scope.

4.3 Clustering

Up to now, the dataset is characterized by the schema depicted by Equation 4. Then, using a clustering algorithm, such as k-means, we group the GLDs based on its $(\lambda_2, \lambda_3, \lambda_4)$ values, as discussed in Section 3.2. In this paper,

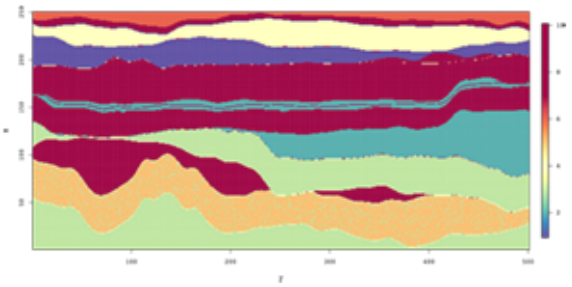

Figure 2: One slice of the $250 \times 501 \times 501$ cube. In the slice, we can distinguish between the different layers.

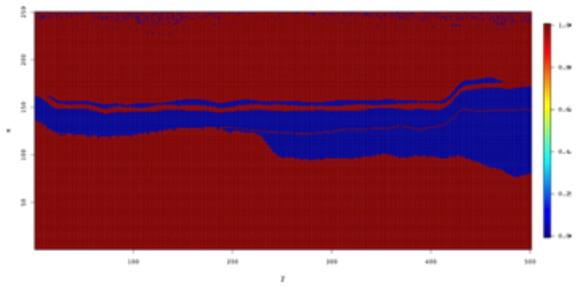

Figure 3: The red color shows where the p-value is greater than 0.05.

we use the k-means algorithm with $k = 10$. The choice of the clustering algorithm and the parameterization is subject for further investigation and is beyond the scope of this paper.

Once the clustering algorithm is applied, a new dataset is produced. In the new dataset, for each spatial location, a label indicates the cluster the *GLD* belongs to, as shown (see the schema at Equation 5) in Figure 4. Note that in Figure 4, the blue region corresponding to cluster 11 is not a cluster itself. It is rather the region where the *GLD* is not valid (see Section 4.2).

$$S_{\mathcal{C}}(x_i, y_j, clusterID, GLD_{x_i,y_j}) \tag{5}$$

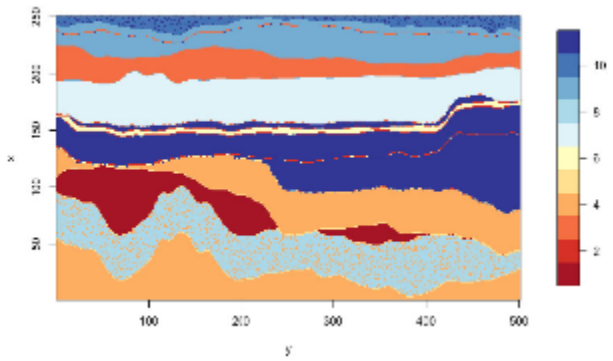

Figure 4: Result of the clustering using k-means with $k = 10$.

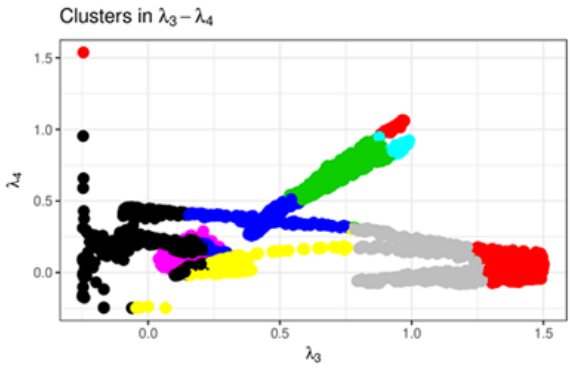

Figure 5: Distribution of the clusters in the (λ_3, λ_4) space. The points that belongs to a same cluster are one near the others, as was expected.

If we visually compare Figure 2 with Figure 4, we observe a close similarity.

Another interesting result is shown in Figure 5, where we plot the clusters in (λ_3, λ_4) space. As we mentioned in Section 2.1.1, the shape of the *GLD* depends on the values of λ_3 and λ_4. In this scenario, the expected result is that the members of the same cluster share similar values of λ_3 and λ_4. This is exactly the result we can observe in Figure 5.

To further corroborate this fact, Figure 6 shows the *PDFs* of 60 members of the 10 clusters. Visually assessing the figures gives an idea of how similar are the shapes of the members of a same cluster and how dissimilar are the shapes of the members of different clusters. This suggests that our approach is valid. A product of these observations is that we can pick one member of each cluster (the centroid) as a representative

of all the members of this cluster, Table 4. The selected member is going to be used to answer the queries in the next Sections.

Cluster	λ_2	λ_3	λ_4	**Cluster**	λ_2	λ_3	λ_4
1	0.0013937313	0.9585829	1.04696461	6	0.0003894541	1.4076354	-0.01925743
2	0.0005291388	1.1633978	-0.07162550	7	0.0021972784	0.3253562	0.01493809
3	0.0020630696	0.1349486	0.17305941	8	0.0015421749	0.9491101	0.86699555
4	0.0016238358	0.8653824	0.83857646	9	0.0018672401	0.2176002	0.17862024
5	0.0027346929	0.5084664	0.39199164	10	0.4856397733	0.1404140	0.14011298

Table 4: Clusters centroids.

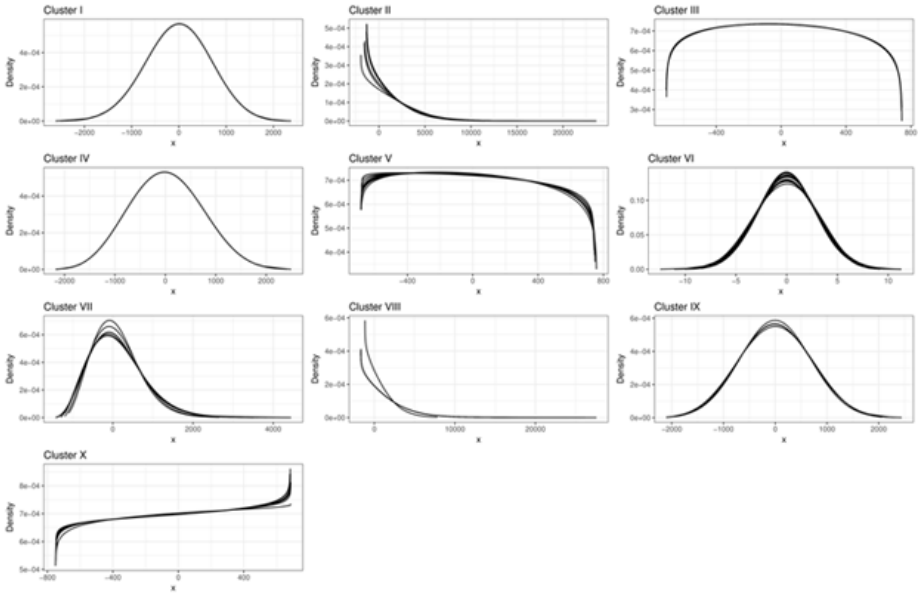

Figure 6: *PDFs* of 60 members of the 10 clusters obtained using k-means over the $(\lambda_2, \lambda_3, \lambda_4)$ values.

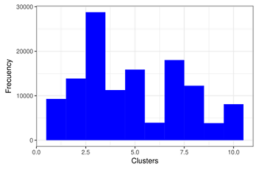

Figure 7: Distribution of the 125250 points by clusters.

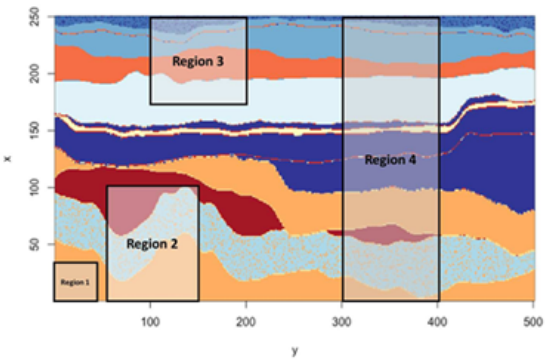

Figure 8: Analysis Regions. The four regions where selected intentionally in this way to warranty different distributions of the clusters inside it.

The 125250 points of the slice are distributed through the clusters following the histogram of Figure 7.

4.4 Spatio-temporal queries

Up to this point, the initial dataset is summarized as depicted by the schema in Equation 5. It can be used to answer queries and validate our approach, by comparing the results with the raw data.

First of all, we select four spatio-temporal regions of the dataset where the clusters suggest us different behaviors. The regions are shown in Figure 8. With these four regions, we assess the adoption of the *GLD* mixture to obtain the *PDF* that characterizes the uncertainty in a specific region (see Sections 4.4.1 and 4.4.2). We use the information entropy to assign a value that measures the uncertainty at each region. In Section 4.4.1, we expect the GLD mixture to characterize well the raw data, and in Section 4.4.2, we hope that the information entropy is zero in region 1 and increases between regions 2, 3 and 4.

4.4.1 GLD mixture

In this experiment, we use the representative *GLDs* at each cluster and the weight associated to it in the region. Using these parameters, we can build a *GLD mixture* that characterizes the uncertainty on that region. We use Algorithm 3 described in Section 2.1.2. First, we query the region to find the clusters represented inside it, and how they are distributed. The retrieved results are shown in Table 5. If we divide the columns of Table 5 by the sum of the elements of each column, we get the weight needed to formulate the *mixed GLDs*. It is clear that the *GLD* in region 1 is represented by the *GLD* of cluster 4. In the other three cases, we get what is shown in the set of equations 6.

Cluster	Region 1	Region 2	Region 3	Region 4
1	0	2250	0	979
2	0	0	0	268
3	0	0	2596	1468
4	1640	4467	0	5173
5	0	149	0	269
6	0	0	0	416
7	0	0	1967	3920
8	0	3335	0	3432
9	0	0	1918	3280
10	0	0	901	583

Table 5: Distribution of the clusters by regions. The four regions are selected intentionally this way to warrant different distributions of the clusters inside it.

$$
\begin{aligned}
GLD_{region2} &= 0.22GLD_{c1} + 0.44GLD_{c4} + 0.014GLD_{c5} + 0.33GLD_{c8} \\
GLD_{region3} &= 0.34GLD_{c3} + 0.26GLD_{c7} + 0.25GLD_{c9} + 0.12GLD_{c10} \\
GLD_{region4} &= 0.22GLD_{c1} + 0.44GLD_{c4} + 0.014GLD_{c5} + 0.33GLD_{c8}
\end{aligned} \tag{6}
$$

Now, we need to evaluate whether the *mixture of GLDs* correctly models the uncertainty in a region. We perform the same *ks-test* used to evaluate the good quality of the fit, described in Section 3.1. Based on the *p-value*, Table 6, we can conclude that in all 4 regions the *mixture of GLDs* is a good fit to the raw data.

Metrics	Region 1	Region 2	Region 3	Region 4
p-value	0.73	0.56	0.34	0.08

Table 6: p-values by regions.

4.4.2 Information Entropy

Based on the distribution of clusters inside the regions (see Table 5), we can compute the entropy.

entropy	Region 1	Region 2	Region 3	Region 4
value	0	1.122243	1.41166	2.024246

Table 7: Information Entropy by regions.

As we expect (see Table 7), the entropy in region 1 is zero, because the region contains only members of the cluster 4. On the other regions the entropy increases from region 2 to region 4, as expected. The information entropy is a very good and simple measure of the uncertainty, and here it is demonstrated its usefulness combined with the *GLD*.

5 Conclusion

In this paper, we proposed SUQ^2, a method to answer uncertainty quantification (UQ) queries over large spatio-temporal simulations. SUQ^2 trades large simulation data by probability density functions (PDFs), thus saving huge amount of storage space and computational cost. It enables complex data distribution representation at each simulation point, as much as a spatio-temporal view of simulation uncertainty computed by mixing spatial point PDFs. We evaluated SUQ^2 using a seismology use case, considering the computation of uncertainty in regions of a slice of the seismic cube. The results show that SUQ^2 method produces an accurate view of the uncertainty in regions of space-time while considerably saving storage space and reducing the cost associated with the PDF modeling of the dataset. To the best of our knowledge, this is the first work to use GLD as the basis for answering UQ queries in spatio-temporal regions and to compile a series of techniques to produce a query answering workflow.

Acknowledgments

This work has been funded by CNPq, CAPES, FAPERJ, the Inria HPDaSc and SciDISC Associated Teams and the European Commission (HPC4E H2020) project.

References

[1] D. R. Tobergte and S. Curtis. Workshop on Quantification, Communication, and Interpretation of Uncertainty in Simulation and Data Science. in *Journal of Chemical Information and Modeling*, 53(9):1689–1699, 2013.

[2] G. Baroni and S. Tarantola. A General Probabilistic Framework for uncertainty and global sensitivity analysis of deterministic models: A hydrological case study. in *Environmental Modelling and Software*, 51:26–34, 2014.

[3] R. H. Johnstone, E. T. Y. Chang, R. Bardenet, T. P. de Boer, D. J. Gavaghan, P. Pathmanathan, R. H. Clayton, and G. R. Mirams. Uncertainty and variability in models of the cardiac action potential: Can we build trustworthy models? in *Journal of Molecular and Cellular Cardiology*, 96:49–62, 2016.

[4] Z. A. Karian and E. J. Dudewicz. *Handbook of fitting statistical distributions with R*. 2011.

[5] M. Freimer, C. T. Lin, and G. S. Mudholkar. A Study Of The Generalized Tukey Lambda Family. in *Communications in Statistics - Theory and Methods*, 17(10):3547–3567, 1988.

[6] C. G. Corlu and M. Meterelliyoz. Estimating the Parameters of the Generalized Lambda Distribution: Which Method Performs Best? in *Communications in Statistics: Simulation and Computation*, 45(7):2276–2296, 2016.

[7] S. Su. Fitting Single and Mixture of Generalized Lambda Distributions to Data via Discretized and Maximum Likelihood Methods: GLDEX in R. *Journal of Statistical Software*, 21(9), 2007.

[8] D. A. Lampasi, F. Di Nicola, and L. Podesta. Generalized lambda distribution for the expression of measurement uncertainty. *IEEE Transactions on Instrumentation and Measurement*, 55(4):1281–1287, 2006.

[9] M. M. Movahedi, M. R. Lotfi, and M. Nayyeri. A solution to determining the reliability of products Using Generalized Lambda Distribution. *Research Journal of Recent Sciences Res.J.Recent Sci*, 2(10):41–47, 2013.

[10] W. Ning, Y. Gao, and E. J. Dudewicz. Fitting mixture distributions using generalized lambda distributions and comparison with normal mixtures. *American Journal of Mathematical and Management Sciences*, 28(1-2):81–99, 2008.

[11] J. F. Wellmann and K. R. Lieb. Uncertainties have a meaning: Information entropy as a quality measure for 3-D geological models. *Tectonophysics*, 526-529:207–216, 2012.

[12] J. Liu,N. Lemus, E. Pacitti, F. Porto, P. Valduriez. Parallel computation of PDFs on big spatial data using spark. in em Distributed and Parallel Databases, pp. 1-28. In press, 10.1007/s10619-019-07260-3, 2019.

[13] R. H. C. Lopes Kolmogorov-Smirnov Test. in Lovric M. (eds) International Encyclopedia of Statistical Science. Springer, Berlin, Heidelberg, 2011.

[14] J. S. Ramberg, and B. W. Schmeiser. An approximate method for generating asymmetric random variables. in em Communications of the ACM, 17(2), 78–82. doi:10.1145/360827.360840, 1974.