Kriging-based generation of optimal databases as forward and inverse surrogate models

Sandor Bilicz, Marc Lambert and Szabolcs Gyimóthy

Inverse Problems 26(7) 2010.

Abstract

Numerical methods are used to simulate the mathematical models of a wide range of engineering problems. The precision provided by such simulators is usually fine but at the price of computational cost. In some applications this cost might be crucial. This leads us to consider cheap surrogate models in order to reduce the computation time still meeting the precision requirements. Among all available surrogate models, we deal herein with the generation of an “optimal” database of pre-calculated results combined with a simple interpolator. A database generation approach is investigated which is intending to achieve an optimal sampling. Such databases can be used for the approximate solution of both forward and inverse problems. Their structure carries some meta-information about the involved physical problem. In the case of the inverse problem, an approach for predicting the uncertainty of the solution (due to the applied surrogate model and/or the uncertainty of the measured data) is presented. All methods are based on kriging—a stochastic tool for function approximation. Illustrative examples are drawn from eddy-current nondestructive evaluation.

1 Introduction

In the engineering practice computer simulations of the mathematical model of the physical phenomena are commonly used. The progress of computation tools makes possible precise simulations in a wide range of domains. However, in a number of applications, computational load and time consumption are crucial, whereas the precision requirements are still demanding. Nevertheless, an end-user can neither be expected to know the mathematical model in details nor have time to run overlong simulations.

So, more and more emphasis is being put on the emulation, or surrogate modeling, for example, of electromagnetic phenomena [1]. A natural approach of surrogate modeling is to build a database, consisting of pre-calculated samples computed by the simulator. Using an appropriate interpolator, the end-user does not need the simulator anymore, but only the database and the interpolator. Most of the computational load is then concentrated into the database generation stage, performed only once and off-line using the simulator. Once the database has been obtained the inverse problem can be solved in real-time.

Building the database is a way to sample the mathematical model of the physical phenomena we are interested in. Both the forward and inverse problems can be solved approximately by using databases (also called training set). In the case of forward interpolation, the values of the input parameters of the mathematical model are known and the aim is to approximate the corresponding output. The inverse interpolation is formally similar to the forward one, but herein the output data is assumed to be known and the related input values are to be found.

In spite of the simplicity of the database approach summarized above, it leads to challenging questions. The precision of the interpolation strongly depends indeed on the choice of the samples to be stored and on the applied interpolation method. The sampling strategy must be adapted both to the mathematical model and to the interpolation method.
Classical Design-Of-Experiments (DOE, see [2, 3, 4] for an overview and [5] with almost 200 references) provides a vast number of sampling strategies, for instance, the well-known Latin hypercube sampling (LHS) [6] or maximin sampling [7]. Several additional criteria can be used, e.g., the orthogonality of the sampling [6].

Many DOE methods treat the sampling (choosing input values) and the observation (running the simulator at the chosen input values) as separated steps. Such a DOE is designed to fit requirements related only to the input domain (like uniformity, space-filling or orthogonality), leading to a non-adaptive sampling method. An approach of sample-pattern optimization is presented in [8], whose goal is to reduce the uncertainty of the kriging prediction in a geostatistical application (kriging being a stochastic tool, initially developed in geostatistics—see [9] for a review in the original domain and [10] in a more general context.). This method can be considered as a quasi-adaptive sampling as it takes into account the interpolator when optimizes the sample pattern. However, once the pattern is chosen, the output values observed at the sampling points do not modify the sampling.

An attractive kind of sampling is the family of sequential strategies. A thorough review is [11], focusing on kriging-based surrogate models. Two sampling strategies are presented. The first aims to achieve maximin sample pattern in the input domain, using a weighted distance. The second one is designed to reduce the uncertainty (in a certain sense) of the kriging prediction. Both methods take into account the observations made during the sampling procedure. More recent contributions are, e.g., [12] and [13] in which a step-by-step uncertainty reduction of kriging interpolators is dealt with. In [12] jackknifing is used, whereas in [13], bootstrapping is applied in discrete-event simulations. Those methods are adaptive, or application-driven, as they account for the specific input–output function at hand.

A fully deterministic manner of adaptive sampling is based on meshing methods inspired by meshes of Finite Element Methods. Basic concepts are discussed in [14], and a more advanced and recent mesh-based sampling method is addressed in [15]. By appropriate mesh refinement, smoothing and deformation—according to the output values being computed during the sampling process—the database can be fully adapted to the mathematical model and to the applied interpolator.

In this paper, a meshless adaptive sampling strategy for the generation of optimal databases is investigated. The approach consists in an iterative optimization procedure, adding samples one-by-one, according to a given criterion of optimality combined with a kriging interpolation. To solve the forward and inverse problems, nearest-neighbour (NN) interpolation is used, i.e., the stochastic way of sampling is combined with a deterministic interpolation, a main difference with the approaches of [11, 12, 13]. The use of the NN interpolator implies that the sampling strategies of the latter papers are not optimal in our case.

To avoid confusion, we have to emphasize that the goal of such sampling is quite different from sequential sampling as is performed by the classical optimization-based inversion algorithms. The latter are basically one-shot approaches, used to solve one single inverse problem. They usually trace back the inverse problem to the optimization of a well-chosen cost function, via an appropriate sequence of forward solutions. From the vast domain of methods we mention as an example only the neighbourhood algorithm [16] and the “expected improvement” algorithm [17]. On the contrary, the sampling strategies of database generation aim to improve the quality of the supported sample-based surrogate model as discussed in detail in [11].

Beyond the generation of optimal databases, the applied tools make possible the characterization of the inverse problem as well. Valuable information about the uncertainty of the solution can also be obtained in an elegant manner.

For the sake of completeness, let us note that several other methods using computer models exist for the solution of the inverse problem. For example a Bayesian approach is discussed in [18] to solve calibration problems (determining a subset of the input parameters given the output data and some other input parameter values). However, the Bayesian method described therein differs from ours, as we use the random process model only in the sampling stage, the solution of the inverse problem being then deterministic.

The paper is organized as follows. In section 2, the forward and inverse problems along with their interpolation-based solutions are formalized. In section 3, the principles of our sampling method are presented. The char-
acterization of the inverse problem by means of the tools involved in the database generation is discussed in section 4. In section 5, the performance of the methods is illustrated on typical examples taken from electromagnetic nondestructive testing, where usually complex EM models are involved—an exhaustive overview of the EM-related inverse problems is found in [19]. For completeness A summarizes the basics of kriging, highlighting important issues related to our application.

2 Database, surrogate model and interpolation

For the sake of generality, the approach is formalized in a general manner. Let us define a continuous, well-behaved forward operator $F$ representing a forward problem with a unique solution, an input vector $x = [x_1, x_2, \ldots, x_p]^T$ of $p$ real parameters, and the corresponding real or complex output scalar function $y(t)$. They are related as

$$F : \mathcal{X} \mapsto \mathcal{Y} \quad x \mapsto y(t) = F(x), \quad t \in T \subset \mathbb{R},$$

(1)

$y(t)$ is square-integrable over $T$, $X \subset \mathbb{R}^p$ being the input space and the output space $\mathcal{Y}$ is defined via the forward operator, as the codomain of $F(x)$:

$$\mathcal{Y} = \{y(t) : y(t) = F(x), \forall x \in \mathcal{X}\}.$$

(2)

$\mathcal{Y}$ is then a subset of the $L^2(T)$ space consisting of all square-integrable functions over $T$.

Let us assume that each $x_k$ ($k = 1, 2, \ldots, p$) has a lower and an upper bound, $x_k^{\text{min}}$ and $x_k^{\text{max}}$, respectively. $\mathcal{X}$ is then a bounded, continuous subset of $\mathbb{R}^p$. Both $\mathcal{X}$ and $\mathcal{Y}$ are equipped with a norm. In the input space, we define

$$||x||_\mathcal{X} = \sqrt{\sum_{k=1}^{p} \eta_k^2 (x_k - x_k^{\text{min}})^2},$$

(3)

$\eta_k$ being a scaling factor enabling to put more or less emphasis on a priori chosen parameters. Equal emphasis on all parameters is obtained if $\eta_k = (x_k^{\text{max}} - x_k^{\text{min}})^{-1}$. In the $L^2(T)$ space (and so, in $\mathcal{Y}$) the classical norm is used:

$$||y(t)||_\mathcal{Y} = \sqrt{\int_T |y(t)|^2 dt}.$$

(4)

The operator $F$ is usually realized via a numerical simulation of the mathematical model of the underlying physical phenomena.

Let us define a database $D_n$ as a set of $n$ samples, each of them being a pair of input sample $(x_i)$ and output sample $(y_i(t) = F(x_i))$, $\forall i \in N$. $N$ is the index set of the database: $N = \{1, 2, \ldots, n\}$. Formally:

$$D_n = \{(x_1, y_1(t)), (x_2, y_2(t)), \ldots, (x_n, y_n(t))\}.$$

(5)

Let us define a forward interpolator or a surrogate model $G_n$ based on $D_n$ and approximating $F$ everywhere in $\mathcal{X}$ as

$$G_n : \mathcal{X} \mapsto \mathcal{Y} \quad x \mapsto \begin{cases} G_n(x) = F(x), & \forall x \in \{x_1, \ldots, x_n\} \\ G_n(x) = F(x), & \forall x \in \mathcal{X}/\{x_1, \ldots, x_n\}. \end{cases}$$

(6)

In the following the so-called Nearest Neighbour (NN) interpolation method is used, formalized as

$$G_n[x] = y_j(t), \quad \text{where } j = \arg \min_{i \in N} ||x - x_i||_\mathcal{X}$$

(7)
The result of the interpolation is not necessarily unique, in the case of equal distances from more than one input sample, one output sample has to be chosen arbitrarily among those samples. The main advantages of NN are easy interpretability (close inputs – supposedly similar outputs) and low computational cost. However, our main reason for preferring the NN interpolation becomes obvious in section 3, as we show that the sampling strategy strongly relies on the properties of the NN interpolation.

The inverse interpolation can be formalized similarly to the forward one. The inverse problem can be ill-posed, however, we might give the formal definition of the inverse interpolation by means of the database $D_n$ and the NN interpolation. Let $G_n^{-1}$ denote the (formal) inverse of the operator $G_n$. For an output $\tilde{y}(t)$ the corresponding approximated input parameter is defined as

$$G_n^{-1}\{\tilde{y}(t)\} = x_j, \text{ where } j = \arg \min_{i \in N} ||\tilde{y}(t) - y_i(t)||_y.$$  (8)

### 3 Sampling strategy for response diversity: output sample spreading

#### 3.1 Theoretical principles

The main idea of our sampling strategy is to build an “optimal” database $D_n$ such as its samples $\{x_i, y_i(t)\}, i \in N$: (i) spread over the output space $Y$; (ii) are evenly distributed in $Y$. For a formal definition, let us introduce the distance function $Q_i(x)$ which measures the distance between any point of the output space $F(x)$ and the $i$-th output sample of the database $D_n$ as

$$Q_i(x) = ||F(x) - y_i(t)||_y, x \in X.$$  (9)

$Q_i(x)$ is then defined over $X$ but related to the distances in $Y$. Other distance functions $Q_i(x)$ might be chosen depending on the context in which the obtained database will be used (see [20] for an example).

Let us also define $d_{min}$ (respectively $d_{max}$) the minimum (resp. maximum) distance between two neighbour samples of $D_n$. The output sampling is uniform and the samples cover $Y$ if the following two conditions are simultaneously fulfilled:

$$\frac{d_{max}}{d_{min}} = \frac{\max_{i \in N} \left[ \min_{j \in N \setminus \{i\}} Q_i(x_j) \right]}{\min_{i \in N} \left[ \min_{j \in N \setminus \{i\}} Q_i(x_j) \right]} \leq 1$$  (10)

and

$$\max_{x \in X} \left[ \min_{i \in N} Q_i(x) \right] \leq \Delta,$$  (11)

where $\Delta$ can be seen as the precision of the database. Let us note that those two criteria are not part of the sampling strategy but are only here to qualify how far/close to the “optimal” a database is and to compare the databases between themselves once they have been built.

A similar idea of such an “output space filling” is introduced in [21] and referred as “response diversity”. However, [21] focuses only on the criterion (11), moreover, a functional output is not considered. When both (10) and (11) are holding, some meta-information about the involved forward operator $F$ is available as well:

1. regions of $X$ with dense distribution of input samples indicate a fast variation of $F$ whereas a sparsely sampled region of $X$ signals the flatness of $F$;

2. any unknown output $y(t) \in Y$ can be found at a maximum distance of $\Delta$ from one of the already computed output samples $y_i(t)$ of the database.

Formally speaking the generation of an optimal database can be recast as an optimization problem which can be solved iteratively. Starting with an initial database $D_n$ with one sample ($n = 1$), the iterative “addition” procedure can be described as follows:
1. find \( x_{n+1} \), the most remote point in terms of the distance function (9) from all points already present in \( D_n \) by solving
\[
x_{n+1} = \arg \max_{x \in \mathcal{X}} \min_{i \in N} Q_i(x)
\] (12)

2. compute the corresponding response \( y_{n+1}(t) = F(x_{n+1}) \)

3. insert the sample \( (x_{n+1}, y_{n+1}(t)) \) in \( D_n \) and increase \( n = n + 1 \)

4. repeat steps (i) to (iii) until the criteria (11) is met.

However, with such a procedure, even if the samples cover the whole output space \( Y \), the equi-spacing of samples –criterion (10)– is not necessarily fulfilled. One way to overcome this is to introduce in the procedure a “removal strategy” which will delete a sample from \( D_n \). Among all possible strategies we choose a simple two-step procedure:

1. find the pair of closest output samples \( y_a(t) \) and \( y_b(t) \) in \( D_n \) defined as
\[
||y_a(t) − y_b(t)||_Y \leq ||y_i(t) − y_j(t)||_Y \quad \forall i, j \in N, i \neq j.
\] (13)

2. between the two selected samples delete the one (here \( y_a(t) \)) for which the distance with the second nearest neighbour is the smallest, formally defined as
\[
\min_{i \in N\setminus\{a,b\}} ||y_a(t) − y_i(t)||_Y < \min_{j \in N\setminus\{a,b\}} ||y_b(t) − y_j(t)||_Y.
\] (14)

The final procedure is arbitrarily defined as an alternating use of two successive “addition” and one “removal” steps. Heuristically, we might expect that such a procedure converges to the “optimal” database, but no formal proof can be proposed, however.

3.2 Practical and numerical considerations

The “removal” procedure involving (13) and (14) is rather simple since it operates on already computed samples. But the “addition” step involves the solutions of (11) and (12), both being computationally expensive since those solutions need the evaluation of the forward operator \( F \) via the distance functions \( Q_i(x), i \in N \). An approximate solution is proposed by using the kriging predictor \( \hat{Q}(x) \) of \( Q(x) \) as
\[
\hat{Q}_i(x) = \sum_{j=1}^n \lambda_j(x) Q_j(x) + \sum_{j=n+1}^{n+n_{\text{rem}}} \lambda_j(x) Q_j(x) + \sum_{j=n+n_{\text{rem}}+n_{\text{init}}}^{n+n_{\text{rem}}+n_{\text{init}}} \lambda_j(x) Q_j(x), \quad i \in N.
\] (15)

\( n \) is the number of points in \( D_n \), \( n_{\text{rem}} \) the number of points which have been removed from \( D_n \) and \( n_{\text{init}} \) the number of initial points used to compute \( \hat{Q}(x) \) (the need of the introduction of \( n_{\text{rem}} \) and \( n_{\text{init}} \) will be explained in the next paragraph). The \((n+n_{\text{rem}}+n_{\text{init}}) \lambda_j(x)\) are estimated in the framework of the ordinary kriging theory (see A) and \( Q_j(x) \) is the exact evaluation of the distance functions at \( x_j \).

In the database generation process, indeed, not only the \( n \) input samples \( x_1, x_2, \ldots, x_n \) but also a set of “hidden” input samples \( (n_{\text{remove}} \text{ and } n_{\text{init}}) \) are used as observation points for the distance functions. Those hidden samples do not appear in \( D_n \), but only support the kriging model. The reason of using them is twofold. First, at the beginning of the generation process, when \( n \) is small, the kriging prediction would not provide acceptable precision so we introduce, in an initialization part, \( n_{\text{init}} \) samples spaced on a regular grid in \( X \). Second, the removed samples remain involved in the kriging model as hidden samples through \( n_{\text{rem}} \).

The original optimization problem can then be solved approximately at a much less computational cost by replacing \( Q_i(x) \) by \( \hat{Q}(x) \) as given by (15) into (11) and (12). The problem is then tractable even when
carrying out an exhaustive search in \( X \) (using Latin hypercube or a fine grid, provided that the number of input parameters \( p \) is small enough).

The stopping criteria to obtain an “optimal” database have been theoretically described in the previous section by equations (10) and (11). However, the number of iterations needed to reach them can be important; so, a third criterion based on the maximum number of samples \( n_{\text{max}} \) is introduced. The procedure is then stopped when one of the three criteria is reached, leading to a “quasi-optimal” database \( D_{n_{\text{max}}} \). Proximity to optimal conditions can be quantified by estimating (10) and (11) and by checking how close to one and to \( \Delta \) they are, respectively.

## 4 Characterization of the inverse problem by means of the database tools

In the following we highlight the advantages of the use of such “quasi-optimal” database \( D_{n} \), which allows us to get valuable information about the underlying operator \( F \), thus, about the inverse problem as well.

### 4.1 Inverse mapping of a noise level

The measured output \( \tilde{y}(t) \) is always corrupted by noise, the latter being either inaccuracy of the measurements, uncertainty of the modeling assumptions, or numerical inaccuracy of the applied forward simulator.

Let us assume that a uniform noise level \( \delta \) affects all measurements. For a given measured output \( \tilde{y}(t) \), one has to consider all functions \( y(t) \), being closer to \( \tilde{y}(t) \) than a distance \( \delta \), as possible real (noise-free) data. For instance, by using a database \( D_{n} \), the sub-region \( \Phi_{\delta} \) of \( X \) consisting of possible inputs \( x \) such as their corresponding outputs are closer to \( y_{i}(t) \) than \( \delta \) can easily be assigned. Formally, let us define the set \( \Phi_{\delta} \in X \) so that

\[
\Phi_{\delta} = \{ x \in X : \| F(x) - \tilde{y}(t) \|_{y} \leq \delta \} = \{ x \in X : Q_{\delta}(x) \leq \delta \}. \tag{16}
\]

This is the inverse mapping of a noise level \( \delta \), a sub-region of \( Y \) is projected back onto a sub-region of \( X \). The shape and the dimensions of the different \( \Phi_{\delta} \) cells provide valuable information about the underlying problem, since they show how the noise influences the uncertainty of the solution of the inverse problem. Where \( F \) varies rapidly, a smaller \( \Phi_{\delta} \) is expected than in the regions where \( F \) is flat, i.e., when the problem tends to be ill-posed. This formalization gives an explicit expression of the within reach precision at a given noise level.

Let us note that if the noise level \( \delta \) is known in advance, the generation of the database can be optimized by using \( \Delta = 2\delta \) in (11).

In the realization, due to the high computational cost, not the exact distance functions but their kriging predictions are used. Thus, the approximate noise-cells \( \hat{\Phi}_{\delta} \) can be determined as

\[
\hat{\Phi}_{\delta} = \{ x \in X : \tilde{Q}_{\delta}(x) \leq \delta \}. \tag{17}
\]

### 4.2 Inverse mapping of Voronoi cells

The database \( D_{n} \) can also be applied for inverse interpolation using (8). Let us define \( \Omega_{i} \) the output Voronoi cell [22] related to the \( i \)-th output sample \( y_{i}(t) \) of \( D_{n} \) as

\[
\Omega_{i} = \{ y(t) \in Y : \| y(t) - y_{i}(t) \|_{y} \leq \| y(t) - y_{a}(t) \|_{y} \forall u \in N \}. \tag{18}
\]

The inverse mapping of \( \Omega_{i} \) for each \( i \in N \) to the corresponding subset \( \Psi_{i} \) of the input space \( X \) can then be performed using the distance function. Formally, let \( \Psi_{i} \) be a subset of \( X \), so that

\[
\Psi_{i} = \{ x \in X : F(x) \in \Omega_{i} \} = \{ x \in X : Q_{\delta}(x) \leq \tilde{Q}_{\delta}(x), \forall u \in N \}. \tag{19}
\]
This inverse mapping can be very effective for the characterization of the inverse problem. Provided that the database $D_n$ is approximately output-equidistant and the distance between the output samples is roughly $2\delta$, the dimensions of $\Psi_i$ represent the attainable precision of the solution in each region of $\mathcal{X}$.

Such a Voronoi cell mapping is similar to the noise mapping (subsection 4.1) except that it does not need to impose any $\delta$ explicitly. Moreover, $\Psi_i$ are (almost) disjoint sets (except for the common boundaries) and their union gives $\mathcal{X}$:

$$\Psi_i \bigcap \Psi_j = \emptyset, \forall i, j \in N, i \neq j \quad \text{and} \quad \bigcup_{i \in N} \Psi_i = \mathcal{X}. \quad (20)$$

Here again the kriging prediction of $Q_i(x)$ is used to reduce the computational burden and the approximate images are then determined as

$$\hat{\Psi}_i = \{x \in \mathcal{X} : \hat{Q}_i(x) \leq \hat{Q}_u(x), \forall u \in N\}. \quad (21)$$

Some meta-information can then be provided by the knowledge of $\hat{\Psi}_i$ via the estimation of their dimensions. Formally, let us define $\Delta x_i$ as the vector of the maximum distance between the $i$-th input sample $x_i$ and all the possible $x \in \hat{\Psi}_i$, defined as

$$\Delta x_i = [\Delta x_{i,1}, \Delta x_{i,2}, \ldots, \Delta x_{i,p}]^T, \text{ with } \Delta x_{i,k} = \max_{x \in \hat{\Psi}_i} |x_k - x_{i,k}|, \quad (22)$$

$x_k$ and $x_{i,k}$ being the $k$-th component (parameter) of the inputs $x$ and $x_i$, respectively. The components of $\Delta x_i$ are related to the confidence in the solution of the inverse problem.

5 Application in nondestructive eddy current testing

5.1 Introduction

The illustrative examples are chosen from the domain of Eddy-Current Testing (ECT), a widely used technique for revealing and characterizing material flaws within conductive specimens. The essence of the method is to generate eddy-currents within the examined specimen by exciting with alternating magnetic field. The magnetic field due to the eddy-currents is measured, and might vary near the damaged zones. Effective numerical simulators of ECT are available ([23, 24, 25], . . . ), at a quite high computational cost, however.

In our examples, a simple ECT setup is considered (Fig. 1). An infinitesimally thin crack affects a non-magnetic, homogeneous infinite metal plate with a thickness of $d = 1.25$ mm and an electrical conductivity of $\sigma = 10^6$ S/m. An air-cored probe coil (driven by a time-harmonic current of frequency $f = 150$ kHz) is scanning a centered rectangular surface of $5 \times 20$ mm above the damaged zone. The impedance change of the coil $\Delta Z$ (influenced by the crack) is measured at $11 \times 41$ regularly spaced positions along $\alpha$ and $\beta$. The output signal $y(t)$ is then the impedance of the coil, $t$ being related to the position of the coil.

The EM phenomenon is modeled using a surface integral approach. The numerical simulation –representing $\mathcal{F}$– is based on the Method-of-Moments (for details, see [26]). The average simulation time for one crack is less than 2 minutes.

<table>
<thead>
<tr>
<th>Ex.</th>
<th>$A$ (mm)</th>
<th>$B$ (mm)</th>
<th>$L$ (mm)</th>
<th>$D$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>min</td>
<td>max</td>
<td>min</td>
<td>max</td>
</tr>
<tr>
<td>2D</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>3D</td>
<td>-1.5</td>
<td>1.5</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>4D</td>
<td>-1.5</td>
<td>1.5</td>
<td>-1.5</td>
<td>1.5</td>
</tr>
</tbody>
</table>

Table 1: Bounds of the input parameters.
The crack is rectangular-shaped, perpendicular to the surface of the plate and it opens to the side opposite with the coil. The configuration can be described using four parameters: \( L \) and \( D \) are the length and the depth (in \% of \( d \)) of the crack, respectively; \( A \) and \( B \) are the co-ordinates of the center of the crack along \( \alpha \) and \( \beta \). Three configurations are then used to illustrate our approach:

- **2D example**, where only two parameters (\( L \) and \( D \)) are varying, \( A \) and \( B \) being both 0.
- **3D example**, the \( x \) co-ordinate of the center of the crack (\( A \)) is added as input parameter.
- **4D example**, all four parameters are now considered as input parameters.

The minimum and the maximum values of each parameter are defined in Table 1.

The criterion of sample insertion (12) –approximated by using the predicted distance functions (15)– is evaluated by performing an exhaustive search in \( \mathcal{X} \). In the 2D example, a fine regular grid of \( 100 \times 100 = 10\,000 \) nodes is used. In the 3D and 4D examples, a point-set generated by Latin hypercube sampling [4] is used, with \( 22^3 = 10\,648 \) and \( 12^4 = 20\,736 \) points, respectively. The initial sample in the database \( D_1 \) corresponds to the largest crack in all examples. In 3D and 4D cases, its position is \( A = 1.5 \) \( \text{mm} \) and \( A = B = 1.5 \) \( \text{mm} \), respectively. The stopping criterion is simply an upper limit for \( n \).

### 5.2 Characterization of the sampling and the space filling of the database

The properties of the optimal databases can be evaluated via the minimal distance \( d_{\text{min}} \) and the maximal distance \( d_{\text{max}} \) between an output sample and its nearest neighbour defined in (10), the ratio \( d_{\text{max}} / d_{\text{min}} \) measuring the uniformity of the sampling. In Table 2 a comparison is carried out between the optimal and regular databases. From now on, the term “regular” refers to a database in which the input samples are spaced at the nodes of a regular grid (covering the whole input space). As expected, the regular sampling yields much higher values of \( d_{\text{max}} / d_{\text{min}} \) than our adaptive strategy in all examples. It is also to be noticed that increasing the number of the samples decreases \( d_{\text{min}} \) but increases the ratio \( d_{\text{max}} / d_{\text{min}} \) due to the faster decrease of \( d_{\text{min}} \) compared to \( d_{\text{max}} \).
As expected, our sampling strategy concentrates the input samples in the region of high values of \( \Delta Z \) respectively in the three examples. where the reference crack is \( x \). The optimal databases are evaluated and compared to regular databases also via the normalized interpolation error defined as

\[
\varepsilon_n(x) = \frac{\| F(x) - G_n(x) \|}{\max_{x \in \mathcal{X}} \left( \left\| F(x^*) \right\| \right)^{-1}}.
\]  (23)

where the reference crack is \( x^* = [4.5 \text{ mm}, 50 \%] \), \( x^* = [0 \text{ mm}, 4.5 \text{ mm}, 48 \%] \) and \( x^* = [0 \text{ mm}, 0 \text{ mm}, 4.5 \text{ mm}, 48 \%] \), respectively in the three examples.

Table 2 provides a comparison between the size of the optimal database for a given \( n \) and the number of forward simulations needed to build it. The maximal normalized error, \( \max_{x \in \mathcal{X}} \varepsilon_n(x) \), is determined by using a fine grid and given in Table 4 for three different \( \mathcal{D}_n \) with increasing number of samples \( n \). As expected, for a given \( n \), (i) the optimal databases outperform the regular ones in all studied cases, (ii) increasing the number of samples in \( \mathcal{D}_n \) leads to a decrease of \( \varepsilon_n(x) \). Table 4 illustrates also the fact that, if the regular database was build with \( n + n_{\text{rem}} + n_{\text{init}} \) samples (instead of only \( n \)) this would lead to a smaller maximal \( \varepsilon_n(x) \) but at the price of a much larger database, and the maximal \( \varepsilon_n(x) \) is still larger than the one obtained using the optimal one with a smaller number of samples.

Table 3: Properties of the regular and optimal databases.

<table>
<thead>
<tr>
<th>Ex.</th>
<th>( d_{\text{min}} )</th>
<th>( d_{\text{max}}/d_{\text{min}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>4D</td>
<td>( 1.09 \times 10^{-3} )</td>
<td>( 3.96 \times 10^{-4} )</td>
</tr>
<tr>
<td>2D</td>
<td>( 8.34 \times 10^{-6} )</td>
<td>( 2.99 \times 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>( 6.75 \times 10^{-6} )</td>
<td>( 2.33 \times 10^{-4} )</td>
</tr>
<tr>
<td>3D</td>
<td>( 1.20 \times 10^{-6} )</td>
<td>( 4.30 \times 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>( 8.93 \times 10^{-7} )</td>
<td>( 3.13 \times 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>( 7.15 \times 10^{-7} )</td>
<td>( 2.54 \times 10^{-4} )</td>
</tr>
<tr>
<td>4D</td>
<td>( 1.20 \times 10^{-6} )</td>
<td>( 4.62 \times 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>( 8.75 \times 10^{-7} )</td>
<td>( 3.15 \times 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>( 7.01 \times 10^{-7} )</td>
<td>( 2.28 \times 10^{-4} )</td>
</tr>
</tbody>
</table>

5.3 Forward interpolation

The optimal databases are evaluated and compared to regular databases also via the normalized interpolation error defined as

\[
\varepsilon_n(x) = \frac{\| F(x) - G_n(x) \|}{\max_{x \in \mathcal{X}} \left( \left\| F(x^*) \right\| \right)^{-1}}.
\]  (23)

where the reference crack is \( x^* = [4.5 \text{ mm}, 50 \%] \), \( x^* = [0 \text{ mm}, 4.5 \text{ mm}, 48 \%] \) and \( x^* = [0 \text{ mm}, 0 \text{ mm}, 4.5 \text{ mm}, 48 \%] \), respectively in the three examples.

Table 3 provides a comparison between the size of the optimal database for a given \( n \) and the number of forward simulations needed to build it. The maximal normalized error, \( \max_{x \in \mathcal{X}} \varepsilon_n(x) \), is determined by using a fine grid and given in Table 4 for three different \( \mathcal{D}_n \) with increasing number of samples \( n \). As expected, for a given \( n \), (i) the optimal databases outperform the regular ones in all studied cases, (ii) increasing the number of samples in \( \mathcal{D}_n \) leads to a decrease of \( \varepsilon_n(x) \). Table 4 illustrates also the fact that, if the regular database was build with \( n + n_{\text{rem}} + n_{\text{init}} \) samples (instead of only \( n \)) this would lead to a smaller maximal \( \varepsilon_n(x) \) but at the price of a much larger database, and the maximal \( \varepsilon_n(x) \) is still larger than the one obtained using the optimal one with a smaller number of samples.

Table 3: Comparison of the size of the optimal database (\( n \)) and the number of forward simulations needed to build it (\( n + n_{\text{rem}} + n_{\text{init}} \)).

<table>
<thead>
<tr>
<th></th>
<th>( n + n_{\text{rem}} + n_{\text{init}} )</th>
<th>( n + n_{\text{rem}} + n_{\text{init}} )</th>
<th>( n + n_{\text{rem}} + n_{\text{init}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>( 4^2 ) 50</td>
<td>( 4^3 ) 232</td>
<td>( 4^4 ) 1060</td>
</tr>
<tr>
<td></td>
<td>( 5^2 ) 64</td>
<td>( 5^3 ) 337</td>
<td>( 5^4 ) 1578</td>
</tr>
<tr>
<td></td>
<td>( 6^2 ) 86</td>
<td>( 6^3 ) 462</td>
<td>( 6^4 ) 2466</td>
</tr>
</tbody>
</table>

For the 2D case the distribution of \( \varepsilon_n(x) \) over \( \mathcal{X} \) and the input sample spacing can easily be depicted (Fig. 2). As expected, our sampling strategy concentrates the input samples in the region of high values of \( L \) and \( D \) (large cracks) for which the values of \( \Delta Z \) are known to vary rapidly. In addition, the interpolation error yielded by this optimal database is lower than the one obtained using regular sampling.
Table 4: Maximal normalized interpolation error $e_n(x)$.

(a) 2D.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Regular</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$4^2$</td>
<td>4.132</td>
<td>2.683</td>
</tr>
<tr>
<td>$5^2$</td>
<td>3.399</td>
<td>2.042</td>
</tr>
<tr>
<td>$6^2$</td>
<td>2.726</td>
<td>1.842</td>
</tr>
<tr>
<td>$7^2$</td>
<td>2.495</td>
<td></td>
</tr>
<tr>
<td>$8^2$</td>
<td>2.254</td>
<td></td>
</tr>
<tr>
<td>$9^2$</td>
<td>1.994</td>
<td></td>
</tr>
<tr>
<td>$10^2$</td>
<td>1.668</td>
<td></td>
</tr>
</tbody>
</table>

(b) 3D.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Regular</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$4^3$</td>
<td>5.009</td>
<td>3.623</td>
</tr>
<tr>
<td>$5^3$</td>
<td>3.900</td>
<td>2.858</td>
</tr>
<tr>
<td>$6^3$</td>
<td>3.411</td>
<td>2.207</td>
</tr>
<tr>
<td>$7^3$</td>
<td>3.050</td>
<td></td>
</tr>
<tr>
<td>$8^3$</td>
<td>2.597</td>
<td></td>
</tr>
</tbody>
</table>

(c) 4D.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Regular</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$4^4$</td>
<td>5.112</td>
<td>3.917</td>
</tr>
<tr>
<td>$5^4$</td>
<td>4.037</td>
<td>2.928</td>
</tr>
<tr>
<td>$6^4$</td>
<td>3.405</td>
<td>2.918</td>
</tr>
<tr>
<td>$7^4$</td>
<td>2.941</td>
<td></td>
</tr>
<tr>
<td>$8^4$</td>
<td>2.664</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2: 2D: Input samples (dots) and the interpolation error $e_n(x)$ (map) in regular and optimal databases ($n = 25$).

5.4 Characterization of the inverse problem

In the following the use of optimal databases for the inverse mappings introduced in section 4 is illustrated.

2D-example: the predicted inverse-mapped Voronoi cells $\hat{\Psi}$ in Fig. 3a and the noise levels $\hat{\Phi}$ for three different $\delta$ in Fig. 3b are shown. Let us note that $D_{25}$ is quasi output-equidistant ($d_{\text{max}}/d_{\text{min}} = 1.66$), even so the shape and the dimensions of the different cells $\hat{\Psi}$ in the input-space are very diverse and illustrate the behaviour of $F$ in the different regions of $X$. The large cells in the region of small cracks highlight that a finer sampling is not needed and how difficult it will be to invert such a kind of defects using our measurement configuration. On the contrary, the small cells in the region of large cracks justify the concentration of input samples.

3D-example: predicted inverse-mapped noise levels $\hat{\Phi}$ are now depicted (Fig. 4). For the sake of clarity only two different levels $\delta = 0.1 d_{\text{min}}, 0.3 d_{\text{min}}$ are drawn for two different view angles and for three samples ($x_a = [-1.48 \text{ mm}; 8.34 \text{ mm}; 68.1 \%]$; $x_b = [0.56 \text{ mm}; 6.12 \text{ mm}; 51.2 \%]$; $x_c = [-0.86 \text{ mm}; 9.59 \text{ mm}; 86.4 \%]$ characterized by $[A; L; D]$). This configuration is an extension of the previous 2D case and, as expected, the distribution of the input samples leads to the same conclusion for the two common parameters $L$ and $D$: a quite
Figure 3: 2D: characterization of the inverse problem using $D_n$ with $n = 25$ samples (dots). The inverse mapping of (a) the Voronoi cells and, (b) three different noise levels $\delta$. Each sample has three cells $\Phi_{\delta}^{i}$, the largest ones are related to $\delta = 0.5d_{\text{min}}$, including the cells corresponding to smaller $\delta$ as subsets.

4D-example: in such a case to visualize the results is rather difficult, so only the inverse mapping of the noise levels for one sample among all those of the database $D_{256}$ is shown. This point –$[A = 1.118 \text{ mm}, B = 0.877 \text{ mm}, L = 8.788 \text{ mm}, D = 78.76 \%]$– is chosen within the dense part of the database. Fig. 5 shows the six 2D-cuts obtained by keeping two of the four parameters constant and letting the two other vary from their minimum to their maximum values. As illustrated in Fig. 5d and Fig. 5e by the extension of the surface in the $A$ and $B$ directions $\Delta Z$ is not very sensitive to the position of the center of the crack $(A, B)$ whereas Fig. 5f clearly shows how $L$ and $D$ are of importance.

6 Conclusion

The use of a surrogate model combining an optimal database and a simple nearest neighbour (NN) interpolator as a tool for the forward and inverse problems has been presented. The optimal database is a surrogate model linking an input space to an output space and is built such that the latter is evenly filled by samples. The generation of the database is recast as an optimization problem, which is solved iteratively, by inserting the next input sample with respect to the distance between points in the output space. This is an expensive-to-compute optimization problem, thus, the use of a kriging-based interpolator is proposed. The efficiency of such an approach is exemplified by an eddy-current non-destructive application involving two, three and four parameters.

The database combined with a simple interpolator (here a nearest neighbour) can then be used as an improved fast forward surrogate model. Though the preliminary results are encouraging, such a basic interpolator is however not yet optimal and some other techniques should be used [27].

Beyond the sought-after forward and inverse interpolation, the structure of the databases yields meta-information about the underlying problem to be emulated. Two inverse mappings are then proposed, one is
Figure 4: 3D: characterization of the inverse problem with the inverse mapping of $\hat{\Phi}^\delta_i$ for two different noise levels ($\delta = 0.1 \delta_{\text{min}}, 0.3 \delta_{\text{min}}$). Only three points are presented for two different view angles. $x_a = [A = -1.48 \text{ mm}; L = 8.34 \text{ mm}; D = 68.1 \%]$, $x_b = [A = 0.56 \text{ mm}; L = 6.12 \text{ mm}; D = 51.2 \%]$, $x_c = [A = -0.86 \text{ mm}; L = 9.59 \text{ mm}; D = 86.4 \%]$.

Figure 5: 4D: characterization of the inverse problem with the inverse mapping of three different noise levels $\delta (\delta/\delta_{\text{min}} = 0.1, 0.3, 0.5)$ for one point [$A = 1.118 \text{ mm}$, $B = 0.877 \text{ mm}$, $L = 8.788 \text{ mm}$, $D = 78.76 \%$] using $D_n$ with $n = 256$. 
related to the NN interpolator via the Voronoi cell mapping, the second links to a noise-level mapping. Both contribute to quantify the precision that can be reached in the solution of the corresponding inverse problem.

A further challenge arises due to the well-known “curse of dimensionality”. The evaluation of the criterion functions during the generation process becomes computationally difficult when the number of input parameters increases and the exhaustive search currently used becomes intractable. It could also be interesting to combine the classical “space-filling” methods [4] and our approach to be able to treat more complex forward operators.

Acknowledgments

The authors are grateful to Dominique Lesselier for the advices and comments concerning the manuscript. The research is partially supported by the Research and Technology Innovation Fund of the Hungarian Government in the frame of the French-Hungarian Bilateral Intergovernmental S&T Cooperation (FR-1/2008) and by DIGITEO cluster’s project.

The authors would like to thank the guest editors and the anonymous reviewers for their helpful comments and suggestions.

A Summary of ordinary kriging

Let us assume that a distance function \( Q_i(x) \) is evaluated at \( m \) points \( x_1, x_2, \ldots, x_m \), i.e., the values \( Q_i(x_1), Q_i(x_2), \ldots, Q_i(x_m) \) are known. The aim of kriging is to find a predictor \( \hat{Q}_i(x) \) of \( Q_i(x) \), based on the observations. The basic idea is to model \( Q_i(x) \) by a Gaussian random process \( \xi(x) \), usually assumed to be stationary, i.e., its statistical properties are invariant to shifting in \( x \). Formally, the covariance function \( k(h) \) does not depend on \( x \) and the mean is an unknown constant \( c \):

\[
k(h) = \text{cov} [\xi(x), \xi(x + h)] \quad \text{and} \quad \mathbb{E} [\xi(x)] = c.
\]

(24)

Kriging computes the Best Linear Unbiased Predictor (BLUP) of \( \xi(x) \) in the space spanned by the random variables \( \xi(x_1), \xi(x_2), \ldots, \xi(x_m) \). The predictor is sought in the linear form

\[
\hat{\xi}(x) = \sum_{j=1}^{m} \lambda_j(x) \xi(x_j).
\]

(25)

The error of the prediction is \( \chi(x) = \hat{\xi}(x) - \xi(x) \). The coefficients \( \lambda_j(x) \) are chosen so that the mean of \( \chi(x) \) is zero (unbiased prediction) and the variance of \( \chi(x) \) is the smallest possible (best prediction). These requirements are satisfied if the coefficients \( \lambda_j(x) \) are the solutions of the linear system of equations:

\[
\begin{bmatrix}
  k(x_1 - x_1) & \cdots & k(x_1 - x_m) \\
  \vdots & \ddots & \vdots \\
  k(x_m - x_1) & \cdots & k(x_m - x_m)
\end{bmatrix}
\begin{bmatrix}
  \lambda_1(x) \\
  \vdots \\
  \lambda_m(x)
\end{bmatrix}
= 
\begin{bmatrix}
  k(x-x_1) \\
  \vdots \\
  k(x-x_m)
\end{bmatrix},
\]

with the Lagrange multiplier \( \mu \) to enforce the unbiasedness. The covariance function is not known explicitly, but one has to predict it from the observed data. Usually a parameterized class of functions is chosen and its parameters are estimated, e.g., by maximum likelihood method. This part is out of the scope of this article. We only mention that our covariance functions are chosen from the so-called Matérn-class and their parameters are estimated by using the Restricted Maximum Likelihood Method (see, e.g., [28]).

Once the coefficients \( \lambda_j(x) \) are computed by solving (26), \( Q_i(x) \) can be predicted as well:

\[
\hat{Q}_i(x) = \sum_{j=1}^{m} \lambda_j(x) Q_i(x_j).
\]

(27)
In most cases, not only one but $n$ distance functions have to be predicted. A favorable property of the kriging prediction is that the same covariance model can be used to model all distance functions, since all of them inherit the same behaviour from $F$. Moreover, the observation points can easily be chosen the same for all scalar functions. Let us define the matrix $Q$ as

$$
Q = \begin{bmatrix}
Q_1(x_1) & \cdots & Q_1(x_m) \\
\vdots & \ddots & \vdots \\
Q_n(x_1) & \cdots & Q_n(x_m)
\end{bmatrix}.
$$

(28)

The entries of $Q$ are easy-to-compute, as $Q_i(x_j) = \|y_j(t) - y_i(t)\|$. By using $Q$, the prediction (27) can be extended for all scalar functions as

$$
[\hat{Q}_1(x), \hat{Q}_2(x), \ldots, \hat{Q}_n(x)]^T = Q [\lambda_1(x), \lambda_2(x), \ldots, \lambda_m(x)]^T.
$$

(29)

References


