Noname manuscript No.
(will be inserted by the editor)

Global Optimization based on sparse grid surrogate models for black-box expensive functions

Frédéric Delbos · Laurent Dumas · Eugenio Echagüe

the date of receipt and acceptance should be inserted later

Abstract In this article we propose a new method to solve a general black-box global optimization problem with bound constraints where function evaluations are expensive. Our work was motivated by many problems in the oil industry, coming from several fields like reservoir engineering, molecular modeling, engine calibration and inverse problems in geosciences. In such cases, classical derivative free optimization methods often need too many function evaluations, especially in high-dimension cases. To overcome this difficulty, we propose here a new optimization approach, called GOSGrid (Global Optimization based on Sparse Grid), using successive sparse grid interpolation as surrogate models.

Keywords global optimization · expensive functions · surrogate models · sparse grid interpolation

1 Introduction

In the context of oil industry, many problems consist in a minimization process of a computationally expensive function with bound constraints ([24]):

Frédéric Delbos
IFPEN, 1 et 4 avenue du Bois Préau, 92500 Rueil-Malmaison, France
Tel.: +33-1-47526626
Fax: +33-1-47527022
E-mail: frederic.delbos@ifpen.fr

Laurent Dumas
Université de Versailles S.Q, 45 Avenue des États-Unis, 78035 Versailles, France
Tel.: +33-1-39253066
Fax: +33-1-39254645
E-mail: laurent.dumas@uvsq.fr

Eugenio Echagüe
Université de Versailles S.Q, 45 Avenue des États-Unis, 78035 Versailles, France
Tel.: +33-1-39253629
Fax: +33-1-39254645
E-mail: eugenio.echague@uvsq.fr

$$\begin{gathered}\text{Minimize } f(x)\\ x_l \leq x \leq x_u\\ x \in \mathbb{R}^n\end{gathered}$$

where x_l, $x_u \in \mathbb{R}^n$ and $f : \mathbb{R}^d \to \mathbb{R}$ is the computationally expensive function.

In order to speak of a global optimization approach as mentioned in the title, f needs to be at least Lipschitz continuous (see [28] for a discussion on this subject). In practical cases, this hypothesis may fall down so that we can only speak of a minimization process.

The values of f are in general the output of a complex simulator for which we don't have any explicit expression. The absence of any information on the function gradient narrows the resolution field to algorithms using no first or second order derivatives.

There exists many different approaches in derivative free optimization (see [9] for a review of DFO methods), among which the most popular are space partitioning methods like DIRECT ([16]), direct search methods like Nelder Mead or MADS ([1,2]) but also evolutionary algorithms like genetic algorithms ([11,14]), evolution strategies ([15]) or other similar methods. The main drawback for all these methods is that they can suffer from a poor convergence rate and a high computational cost, especially for high dimensional cases. However, they can succeed in finding a global optimum where all other classical methods fail.

A surrogate modeling has been widely used for many years in the kind of optimization considered here. A surrogate model is a framework used to minimize a function by sequentially building and minimizing a simpler model (surrogate) of the original function (see examples in [4,22]).

Surrogate model-based methods can be classified into two groups: non interpolating (quadratic polynomials and other regression models) and interpolating ones. A widely used form of interpolating surrogate models consists in linear combinations of basis functions, for instance Radial Basis Functions [11] or Kriging ([17], [18]).

In this paper, we construct a new surrogate model by using the sparse grid interpolation method. Basically, the sparse grid approach is a grid error-controlled hierarchical approximation method which neglects the basis functions with the smallest supports. This approach was introduced in 1963 by Smolyak [27] in order to evaluate integrals in high dimensions. It was then applied for PDE approximations but also for optimization ([6,19]) and more recently for sensitivity analysis ([8]) . Compared to the first optimization algorithm based on sparse grids proposed by Klimke et al in [19], a local refinement step is constructed here in order to explore the more promising regions. Moreover, no optimization steps are performed over the objective function, which reduces significantly the number of function evaluations employed.

This paper is organized as follows: the basis and the refinement process of the sparse grid interpolation method are developed in section 2, the GOSGrid algorithm is presented in section 3 and in section 4, its performances are compared with other derivative free global algorithms. The comparisons are based first on an analytical benchmark and then on a parameter estimation problem in reservoir engineering.

2 The sparse grid interpolation method

The sparse grid interpolation method that we have implemented uses Lagrange polynomials on Chebyshev points as basis functions in one dimension. This choice is not mandatory, we can also find for instance sparse grids on equidistant points where basis functions are piecewise linear [27]. However, such choice is more suited for the problems that we have treated where functions are supposed to be continous. The expression in dimension d is done by simply tensoring the formulas obtained in one dimension. The hierarchical approach and the sparsity principle are respectively presented in the first two subsections. A model refinement process is then described in subsection 2.3.

2.1 The one-dimensional case and the hierarchical approach

Here, we suppose that we want to model a function

$$f : [0,1] \to \mathbb{R}.$$

For $i \in \mathbb{N}$, we call X^i the set of Chebyshev points on the interval $[0,1]$ of level i. These sets have the property that the current step is included in the next one ($X^i \subset X^{i+1}$).

More precisely, the number of points in the set X^i is given by:

$$n_i = \begin{cases} 1 & \text{if } i = 1 \\ 2^{i-1}+1 & \text{if } i > 1, \end{cases}$$

and each $x_j^i \in X^i$ reads:

$$x_j^i = \begin{cases} \frac{1}{2} & \text{for } j = 1 \text{ and } n_i = 1 \\ \frac{1}{2}(-\cos(\pi \frac{j-1}{n_i-1})+1) & \text{for } j = 1,\ldots,n_i \text{ and } n_i > 1. \end{cases}$$

Denote by a_j^i the Lagrange polynomial associated to each $x_j^i \in X^i$:

$$a_j^i(x) = \begin{cases} 1 & \text{for } j = 1 \text{ and } n_i = 1 \\ \prod_{k=1,k\neq j}^{n_i} \frac{x-x_k^i}{x_j^i-x_k^i} & \text{for } j = 1,\ldots,n_i \text{ and } n_i > 1. \end{cases}.$$

Then, the interpolation model of level i of f, called $m_i(f)$, is equal to

$$m_i(f) = \sum_{x_j^i \in X^i} f(x_j^i) a_j^i. \tag{1}$$

In order to explore the hierarchical aspect of the method, we define Δ^i as the difference between models of level i and $i-1$:

$$\Delta^i = m_i(f) - m_{i-1}(f) = \textstyle\sum_{x_j^i \in X^i} (f(x_j^i) - m_{i-1}(f)(x_j^i)) \,.\, a_j^i.$$

If we set $X_\Delta^i = X^i \backslash X^{i-1}$, we get

$$\Delta^i = \sum_{x_j^i \in X_\Delta^i} \underbrace{(f(x_j^i) - m_{i-1}(f)(x_j^i))}_{w_j^i} \cdot a_j^i, \tag{2}$$

where w_j^i is called j-th hierarchical surplus of level i. Formula (2) means that for computing Δ^i we only evaluate the function on the points that don't belong to the previous set.
The telescopic sum principle and the convention $m_0(f) = 0$ give us the hierarchical approach

$$\begin{aligned} m_i(f) &= m_{i-1}(f) + \Delta^i \\ &= \textstyle\sum_{k=1}^i \Delta^k. \end{aligned}$$

In other words, in order to get the next approximation of level $i+1$ we will only need to compute the function values at the new interpolation points X_Δ^{i+1}.

2.2 The general case and the sparsity principle

In the general case, we want to model a function

$$f : [0,1]^d \to \mathbb{R},$$

For $k = 1, \ldots, d$, let X^{i_k} be a set of Chebyshev points of some level i_k. By simply tensoring (1) we get the Lagrange interpolation formula on the set $X = \prod_{k=1}^d X^{i_k}$ as:

$$m_{(i_1,i_2,\ldots i_d)}(f) = \sum_{x_{j_1}^{i_1} \in X^{i_1}} \cdots \sum_{x_{j_d}^{i_d} \in X^{i_d}} f(x_{j_1}^{i_1}, \ldots, x_{j_d}^{i_d})(a_{j_1}^{i_1} \otimes \ldots \otimes a_{j_d}^{i_d}).$$

With the same hierarchical approach done in dimension one, we get the hierarchical approach in dimension d over the full grid X:

$$m_{(i_1,i_2,\ldots i_d)}(f) = \sum_{k_1=1}^{i_1} \cdots \sum_{k_l=1}^{i_l} \cdots \sum_{k_d=1}^{i_d} \underbrace{(\Delta^{k_1} \otimes \ldots \otimes \Delta^{k_l} \otimes \ldots \otimes \Delta^{k_d})}_{\Delta^{\mathbf{k}}},$$

Denote $\mathbf{k} = (k_1, \ldots, k_l, \ldots, k_d)$ and $\Delta^{\mathbf{k}} = \prod_{i=1}^d (m_{\mathbf{k}}(f) - m_{\mathbf{k}-\mathbf{e}_i}(f))$ with $\mathbf{e}_i$ the i-th canonical vector in $\mathbb{R}^d$ and $m_{\mathbf{k}}(f) = 0$ if there exists l such that $k_l = 0$. In the case where $i_j = N$ for every $j = 1, \ldots, d$, and $n_N = N$, it can be showed that the number of grid points is of order $\mathscr{O}(N^d)$ and the interpolation error is of order $\mathscr{O}(N^{-2})$. When dimension grows, the number of grid points, and consequently the number of function evaluations thus becomes too large to handle. In order to overcome this problem, without loosing approximation precision, Smolyak proposed in [27] a new approach: if we only apply the summation on the indexes $\mathbf{k} = (k_1, \ldots, k_d)$ such that

$$|\mathbf{k}|_1 = \sum_{i=1}^d k_i \leq d + N - 1,$$

we get the *sparse grid interpolation formula* of level N, which neglects the smallest support basis functions:

$$SG_N(f) = \sum_{|\mathbf{k}|_1 \leq d+N-1} (\Delta^{k_1} \otimes \ldots \otimes \Delta^{k_d}).$$

With this new formula it can be proved that for a sufficiently smooth function f, the approximation $SG_N(f)$ is of order $\mathscr{O}(N^{-2}(\log N)^{d-1})$ with only $\mathscr{O}(N(\log N)^{d-1})$ points if we use piecewise linear basis functions [5]. The desired effect is thus achieved, that is an efficient reduction of number of grid points without loosing much precision (see table 1). The interpolation points of a sparse grid of level $N = 4$ in dimension 2 is depicted on Figure 1(a). Compared to a full grid of the same level which would contain 81 points, it is only made of 29 points.

Table 1 Number of grid points and associated interpolation error in dimension d.

Level N	Number of grid points	L^2 error norm
Full grid	$\mathscr{O}(N^d)$	$\mathscr{O}((N)^{-2})$
sparse grid	$\mathscr{O}(N(\log N)^{d-1})$	$\mathscr{O}(N^{-2}(\log N)^{d-1})$

2.3 The refinement process

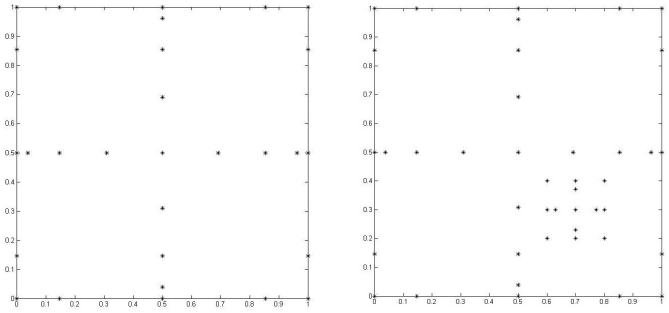
In global optimization, we need to explore the whole domain but we also need sometimes to focus our attention to special promising zones (the exploitation phase). To do so, a refinement process of the sparse grid interpolation model is used to construct a new and more accurate model around a promising point. A local sparse grid model interpolates the error function (difference between the function and the current global model) in this area (see Figure 1(b)) and can thus locally improve the current global model. The refinement domain is defined as an hypercube of the whole domain defined by a center (the promising point) and its edge length. In order to get a continuous model function, basis functions are slightly modified to make them vanish at the boundary of the refinement domain. As all basis functions vanish at the boundary from level 3, only those corresponding to levels 1 and 2 need to be modified. The difference between the basis function of level 1 and the two basis functions from level 2 gives us a function that vanishes at the boundary. This new function is taken as the refinement basis function of level 1 in dimension one: let b_j^i be the j-th refinement basis function in 1 dimension. Then, for $j = 1, \ldots, m_i$ where

$$m_i = \begin{cases} 1 & \text{if } k = 1 \\ n_{i+1} & \text{if } i > 1, \end{cases},$$

we have

$$\begin{cases} b_1^1 = a_1^1 - a_1^2 - a_2^2 \\ b_j^i = a_j^{i+1} \qquad \text{if } i > 1, j = 1, \ldots, m_i. \end{cases}$$

We get the refinement basis functions in dimension d by a simple tensorization of the refinement basis functions in dimension one. Even if this approach provides a continuous refinement model, note that it does not provide a derivable one: the refinement model is not derivable at the points lying on the boundary of the refinement domain.



(a) Sparse grid interpolation points for N=4. Number of points: 29

(b) Sparse grid in 2-D for N=4 and a refinement process of order 3. Number of points: 42

Fig. 1 Sparse grid and refined sparse grid of level 4.

3 Global Optimization based on sparse grid models (GOSGrid)

Given a maximum budget of function evaluations and bound constraints, the proposed method, called GOSGrid, sequentially minimizes sparse grid models of the objective function f. In particular, during the process, it refines some potentially interesting zones to get a lower-value of the cost function. The hierarchical principle allows us to improve the global model without throwing away the previous function evaluations. Moreover, the sparsity greatly reduces the number of exact evaluations, especially in high dimensions.

The GOSGrid and DIRECT algorithms ([16]) share some similarities in the space partitioning principle. However, note that in the GOSGrid algorithm, the local and global searches are done sequentially even though the same reconstruction principle is used for both steps. Moreover, optimization sequences on the surrogate models are included in GOSGrid which is not the case for the DIRECT algorithm.

The algorithm starts by constructing the global model of level 1 (build from the central grid point) in the area described by the bound constraints. Global model minimization is performed from level 2, the refinement process and the minimization of

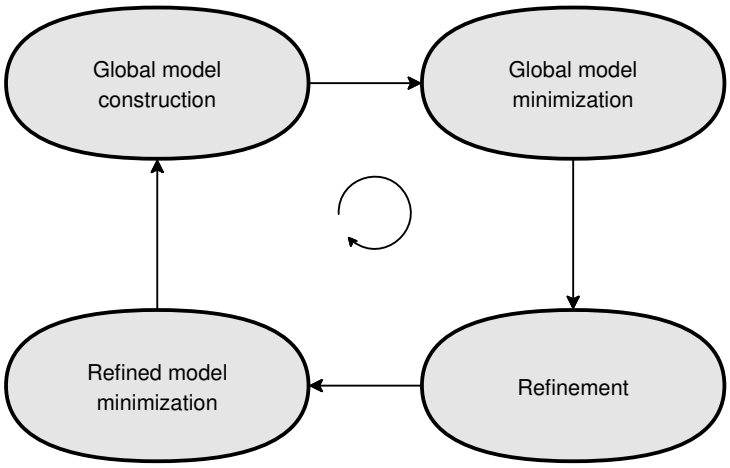


Fig. 2 GOSGrid algorithm diagram

the refined model from level 3. The general steps of the GOSGrid algorithm are given in figure 2 , these steps are detailed below and the complete algorithm is synthesized in Algorithm 1 of Appendix A.

I.1. Model construction If the maximum number of function evaluations is not yet reached, the interpolated sparse grid model is constructed in this step, according to the hierarchical approach : if the current level is greater than 1 then the new model is updated from the previous one by adding new terms coming from the new points included to the current interpolation set.

I.2. Best grid point In this step we choose the best function value between all the objective function evaluations already performed on the global grid. We save this value and the point where the function achieve it.

II.1. Model minimization If the current level is greater than 1, a global optimization of the model is performed in this step. As the model function is computationally inexpensive to compute and smooth, it is minimized by a multistart gradient type method, where the number of initial points is taken here equal to $10d$. These initial points are created using a Latin Hypercube design [20].

II.2. Value's comparison The objective function is evaluated at the minimum point found in II.1. and this value is compared with the value of the best grid point (already computed in I.2). We save the best of them and the point where the function achieves it. This point will be used as the center of the refinement domain in the next step.

III. Model refinement If the maximal number of function evaluations is not yet reached and if the global level is greater than 3, a refinement process is performed in this step. The refinement domain is an hypercube in the normalized domain, centered at the point found in step II.2. Its size is the 10% of the global domain.
If the point found in II.2. is too close from the boundary of the global domain, the refinement area may be not included in the domain. Then some points of the refinement grid may lie outside of the domain. For some industrial application (like the one presented in section 4.2) the evaluation of the function at these points may lead to a non physical value (or even to a crash). In this case, function values can be replaced by global model values at non admissible points.

IV.1 Refined model minimization The refined model is minimized in a slightly smaller area than the refinement domain in order to avoid bound irregularity effects. Again, as model evaluations are computationally inexpensive, the refined model is minimized by a multistart gradient type method including the minimum point of the refined grid.

IV.2 Value's comparison The best value of the function on the refined grid is compared with the minimum value found in IV.1. The best objective function value and the point associated to it are saved. If the cumulative number of performed function evaluations is not greater than the maximal number of evaluations and if the refinement level is strictly lower than the global level, then the refinement level is increased by one and the algorithm is restarted from step III. For other cases the global level is increased by one and the algorithm is restarted from step I.

Note that a convergence proof to the global optima of a Lipschitz continuous function f on an hypercube can be performed for the GOSGrid algorithm when the interpolation level goes to infinity. This property comes from the sparse grid construction itself and not from the optimization process that is done at each step: when the global interpolation level goes to infinity, it can be proven that the interpolation points will fill the whole hypercube for any given tolerance $\delta > 0$.

4 Numerical applications

In this section, the numerical results obtained when applying the new optimization strategy GOSGrid are shown, first on an analytical benchmark and then on an applied problem in oil engineering. This approach is compared with an evolution strategy, a space partitioning method and the first sparse grid based optimization algorithm where no refinement is performed. In order to compare the efficiency of the different methods, the results are depicted by using data profile plots.

4.1 Benchmark validation

In order to validate the algorithm, GOSGrid has been tested on a benchmark containing many global optimization problems with bound constraints (see a detailed

description of the benchmark in Appendix B). The results obtained with GOSGrid are compared with:

(i) a classic evolution strategy called CMA-ES ([15]),
(ii) the space partitioning DIRECT algorithm ([16]),
(iii) the first Sparse Grid optimization method proposed by Klimke ([19]).

All these tools have been used with their nominal parameters.

Figure 3 gives a comparison between these four optimization methods for the Michalewicz function in dimension $d = 5$. Note that in this case, the global minimum is approximately equal to -4.687.

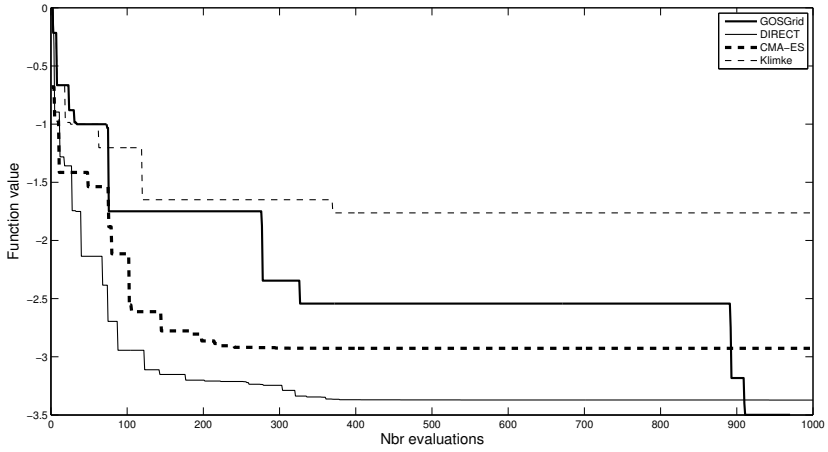


Fig. 3 Michalewicz function minimization in 5 dimensions

After 1000 function evaluations, it can be seen that the lowest function value is found by GOSGrid. Actually, this example has been chosen to illustrate the impact of the refinement process when comparing GOSGrid and the algorithm from Klimke where no refinement is performed.

Data profile comparison

In order to compare GOSGrid with the other methods, a data profile plot, a well-known comparison method in derivative-free optimization ([23]), is chosen. Data profiles indicate the efficiency of a solver (for a given number of functions evaluations) to reach a specific reduction in function value, measured by

$$f(x_0) - f(x^*) \geq (1 - \tau)(f(x_0) - f_L), \tag{3}$$

where x_0 is the initial point, f_L is the best value found by all solvers for the same problem and τ is the reduction rate. For a given problem p and a solver s, we call $t_{p,s}$ the number of functions evaluations needed to find x^* in order to satisfy inequality (3).

The data profile of a solver s is then defined as the distribution function of $\frac{t_{p,s}}{d+1}$, where d is the problem dimension. If we note $\mathscr{P}$ the set of problems, for a given number of function evaluations α we define:

$$d_s(\alpha) = \frac{1}{|\mathscr{P}|}\#\{p \in \mathscr{P} | \frac{t_{p,s}}{d+1} \leq \alpha\}, \tag{4}$$

which is the percentage of problems that can be solved (for a given tolerance τ) with the α evaluations of the function. So, $d_s(\alpha)$ is the percentage of problems that can be solved with the equivalent to α gradient estimation by finite differences.

Figures 4 and 5 show four data profiles for the whole benchmark functions or a part of it.

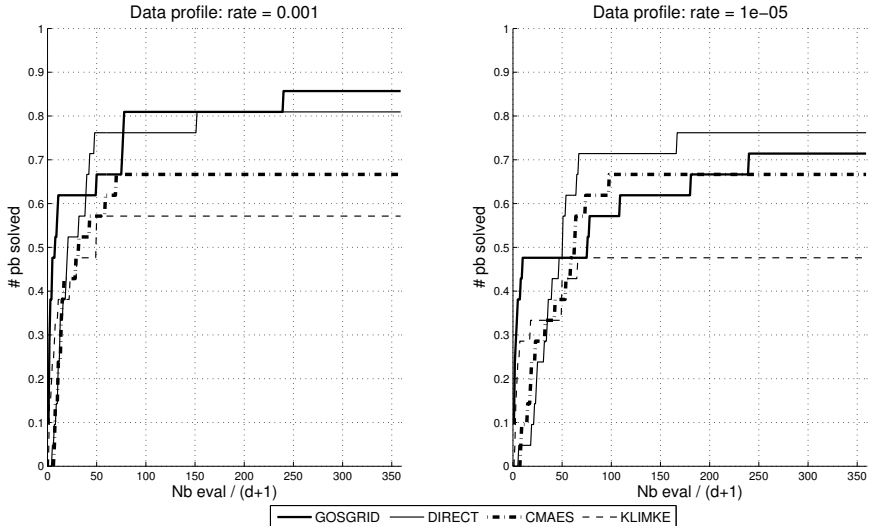


Fig. 4 Data profiles for $\tau = 10^{-3}$ and $\tau = 10^{-5}$ and benchmark functions of dimension $d = 2$

In figure 4, we consider only the 21 benchmark functions where $d = 2$. For a rate $\tau = 10^{-3}$, the best solver appears to be GOSGrid whereas for $\tau = 10^{-5}$ it is DIRECT. However, in the two cases, note that for a small number of functions evaluations, GOSGrid performs better than DIRECT.

In figure 5 we consider the whole 68 benchmark functions with dimensions ranging from $d = 2$ to $d = 25$. For the two rates, if we accept a high computational budget, the best method appears to be CMA-ES, followed by GOSGrid, DIRECT and the Klimke algorithm. However, as before, for a small number of function evaluations and for the two rates, GOSGrid overperforms all the other methods.

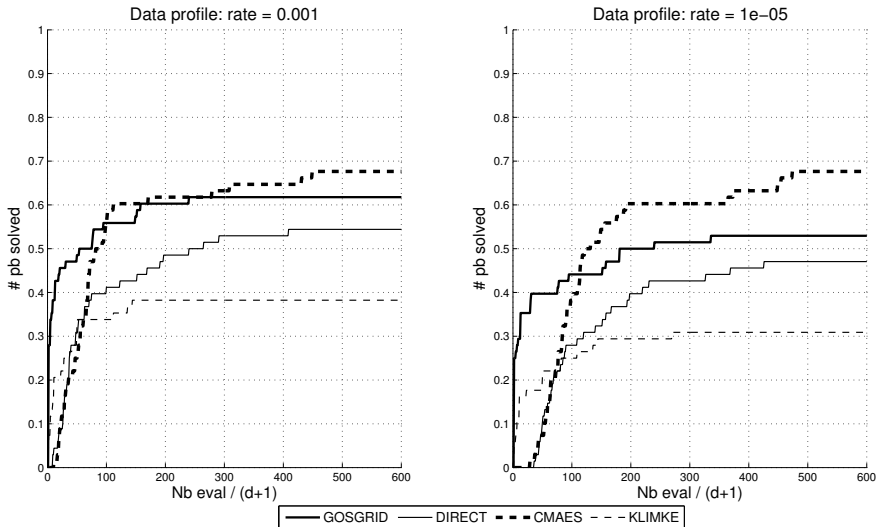


Fig. 5 Data profiles for $\tau = 10^{-3}$ and $\tau = 10^{-5}$ and all benchmark functions

4.2 A reservoir characterization problem

The goal of reservoir characterization is the estimation of reservoir parameters in order to take decisions for future production scheme. There are two types of parameters: those related to the geological modeling (spatial distribution of permeability, porosity, faults) and those related to fluid flow modeling (productivity index of wells, relative permeability curves). These parameters cannot by determined by local measurements on the field. On the other hand, some functions of these parameters can be measured, like bottom-hole pressure, gas-oil ratio, oil rate at the wells or compressional and shear wave impedance for seismic campaigns at different calendar times during production. These function values can also be computed by a simulator, which is expensive in terms of CPU time and do not provide any information about the derivatives.

In this setting, reservoir characterization is an inverse problem, formulated as the minimization of a least-square objective function where each residual is the difference between simulated and observed values. This problem is also referred as history-matching.

The global minimization of this problem yields the parameter values which best fit the data of the problem, and then, the presumed reservoir parameters. Some weights can be introduced to normalize and incorporate confidence information in the data points; the confidence in the experimental data is typically taken into account by making the weights equal to the reciprocal of the standard deviation of the experimental data at each time point and state, while the maximum value of the data or simulations for each state is typically used when the values of the states differ significantly in magnitude.

In general, these problems can be solved by non-linear optimization methods like SQP [3,26], where derivatives are approximated by finite differences. Even if these methods are local, in practice they are quite efficient and they can be easily coupled with a multistart algorithm to seek for a global solution. But in a multistart approach, the number of function evaluations needed to compute derivatives is too high. This fact makes the resolution very expensive, and then, cheaper strategies are needed. Derivative-free techniques seem to be the best alternative for this problem, specially those with a low-cost in terms of function evaluations. This is the reason why GOSGrid is presented as a potential solution strategy for this problem. Note that some previous application studies based on derivative-free methods have already been realized for a close but different application concerning groundwater supply and hydraulic capture [13].

The PUNQ test case The PUNQ test case was already used for comparative inversion studies in the European PUNQ project ([12]) and for validation of constrained modeling and optimization scheme development methods ([25]). In this test case, depicted on figure 6, the reservoir is surrounded by an aquifer in the north and the west and delimited by a fault in the south and the east. A small gas cap is initially present. The geological model is composed of five independent layers. Layers 1, 3, 4 and 5 are presumed to be of good quality, while layer 2 is of poorer quality. The synthetic production data are calculated using the PumaFlow IFPEN flow simulator with a black oil model over an eight-year period. The production results selected are the gas oil ratio (GOR), the cumulative oil production (CUM OIL SURF) and the water cut (WTC) value at the six production wells (PRO-1, 4, 5, 11, 12 and 15), and the total cumulative oil production (CUM PROD OIL SURF).

For history matching, the parameters of the simulation model are constrained by production data. An optimal matching is sought by minimization of the following objective function

$$f(x) = \frac{1}{2}\sum_{i=1}^{6}\sum_{j=1}^{n_p}(d_{P_i}^{sim}(x,t_j) - d_{P_i}^{obs}(t_j))^2, \tag{5}$$

where $i = 1,\ldots,6$ is the well index; and t_j, $j = 1,\ldots,n_p$ are the measurements times of production data. The inversion parameters are four transmissivity multipliers which model the fluid migration ability (MPH2, MPH1, MPV2, MPV1), the residual oil saturation after water sweeping (SORW), the residual oil saturation after gas sweeping (SORG) and the aquifer permeability (AQUI).

The analyzed case is in fact noisy. It was constructed by adding a relative noise of 4% to the production values, which introduces new local minima. The reference values of the unknown parameters and their bounds are given in table 2. The value of the objective function at the reference values of the unknown parameters is equal to 1.794.

As we desire a global solution, GOSGrid was compared to the most performant solver in the data profile : the evolutionary strategy CMA-ES. Given the stochastic characteristic of this last method, it was run 10 times in order to estimate a mean-standard deviation curve. Modeling with the sparse grid technique was done as follows: each residual of the mismatch term was modeled, and the global function model

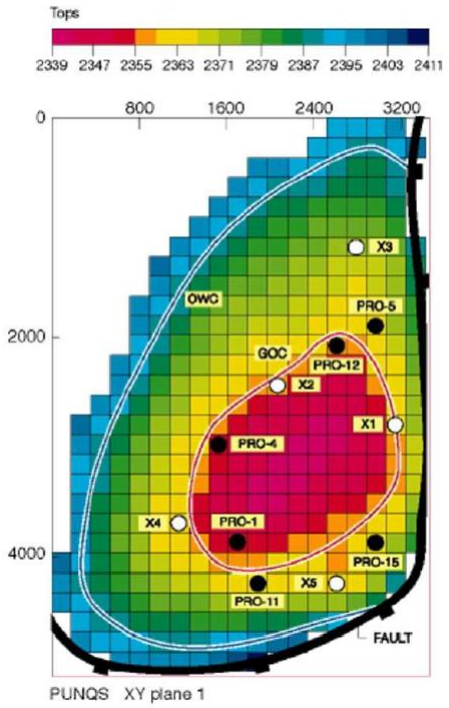


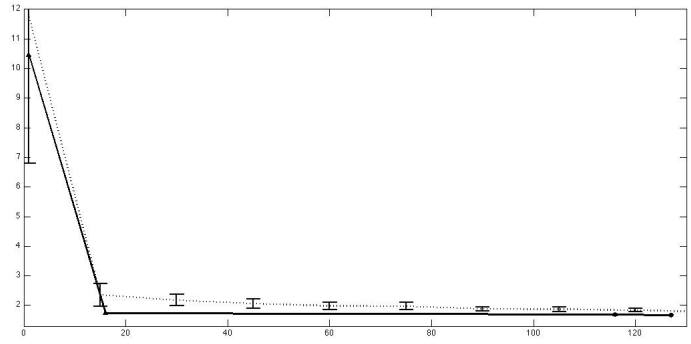
Fig. 6 Top structure of the PUNQ test case extracted from [12]

Table 2 Reference values and bounds for the inversion parameters

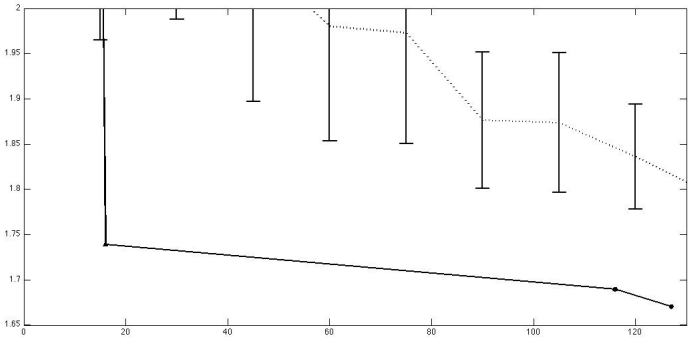
	Ref. value	Min. value	Max. value
MPH2	1.1	0.8	1.2
MPH1	0.9	0.8	1.2
MPV2	1.1	0.8	1.2
SORW	0.16	0.15	0.25
MPV1	1.1	0.8	1.2
SORG	0.19	0.15	0.2
AQUI	0.22	0.2	0.3

was then the sum of squares of each residual model. This approach is particularly suited for least-square functions, the global model being more accurate. It implies an extra computational cost due to the construction of each residual model (compared to the construction of one global model). But, in this application, this computational time is negligible compared to the cost of computing the synthetic production data (i.e. the residual values).

Figure 7(a) compares the two optimization methods. The abscissa axis is the number of function evaluations and the ordinate axis is the objective function value. The dotted line corresponds to the evolution strategy (mean-standard deviation) and the continuous line to GOSGrid method where each of the mismatch terms was modeled.



(a) GOSGrid (continuous line) and an evolution strategy (mean and standard deviation) for the PUNQ case (function values vs. number of function evaluations)



(b) Zoom of the previous figure on the axis of function values

Fig. 7 Comparison between GOSGrid and an evolution strategy for the reservoir characterization problem corresponding to the PUNQ case.

It can be seen that the lower function value is obtained with the GOSGrid algorithm. The refinement process corresponds to the continuous lines that leave the main curve (around 120 function evaluations) and its impact can be appreciated in figure 7(b). The best function value (around 1.67052 for 127 evaluations), and the nearest parameters values to the reference's ones are found as the result of the refinement process of level 3. In order to get a smaller function value than the one found by GOSGrid the evolution strategy would need to perform 650 additional function evaluations and the corresponding function value is 1.6702.

5 Conclusions and perspectives

A new global optimization tool for computationally expensive functions, called GOSGrid, is presented here. It is based on the sparse grid interpolation method with a

sparse grid refinement process. The hierarchical construction of the surrogate model and its sparsity allows to reach a fast decrease in a minimization process. After a benchmark validation including a comparison with various well known optimization methods (among them, CMAES and DIRECT), the GOSGrid algorithm has been successfully applied on a practical reservoir engineering case. The next step will consist of improving the refinement strategy by making it more adaptive, for example by taking into account more promising points at the same time.

Acknowledgements The authors would like to thank the anonymous referees for their useful comments and references.

References

1. Audet, C.; Dennis Jr, J.E. *Mesh adaptive direct search algorithms for constraint optimization*, SIAM J.Optim, Vol 17, pp. 188-217 (2006)
2. Audet, C.; Dennis Jr, J.E. *A progressive barrier for derivative-free nonlinear programming*. SIAM J.Optim, Vol 20(4), pp.445-472 (2009)
3. Bonnans, J.F.; Gilbert, J.C.; Lemarechal, C.; Sagastizabal, C. *Numerical Optimization: Theoretical And Practical Aspects*. Springer-Verlag (2003)
4. A.J. Booker, J.E. Dennis, Jr., P.D. Frank, D.B. Sera
ni, V. Torczon, and M.W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. Structural and Multidisciplinary Optimization, 17(1):1-13, 1999.
5. Bungatz, H-J; Griebel, M. *Sparse grids*, Acta numerica, Vol 13, pp. 1-123 (2004)
6. Buzzard, G.; Donahue, M.; Rundell, A. *Parameter Identification with Adaptive sparse grid-based Optimization for Models of Cellular Processes*, Methods in Bioengineering: Systems Analysis of Biological Networks, pp. 211-232. (2009)
7. Brachetti, P.; De Felice Ciccoli, M.; Di Pillo, G. ; S. Lucidi *A New Version of the Price's Algorithm for Global Optimization*, Journal of Global Optimization, Vol 10, pp. 165-184 (1997)
8. Causse, M. *Sparse grid : de l'approximation des EDP au calcul de sensibilité*. Ph.D thesis, Toulouse (2010)
9. Conn, A.; Scheinberg, K.; Vicente, L. *Introduction to derivative-free optimization*. MPS-SIAM Series on Optimization (2009)
10. Custodio, A. L.; Madeira, J.F.A. *GLODS: Global and Local Optimization using Direct Search*, submitted.
11. Dumas, L.; Herbert, V.; Muyl, F. *Comparison of global optimization methods for drag reduction in the automotive industry*, Lecture Notes in Computer Science, Vol 3483, pp. 948-957 (2005)
12. Floris, F.J.T. *Methods for quantifying the uncertainty of production forecasts; a comparative study*, J. Petroleum Geoscience, Vol. 7. (2001)
13. Fowler, K.R.; Reese, J.P.; Kees, C.E.; Dennis Jr, J.E.; Kelley, C.T.; Miller, C.T; Audet, C; Booker, A.J.; Couture, G.; Darwin, R.W.; Farthing, M.W.; Finkel, D.E.; Gablonsky, J.M.; Gray, G.; Kolda, T.G. *Comparison of derivative-free optimization methods for groundwater supply and hydraulic capture community problems*, Advances in Water Resources, Vol 31, pp. 743-757 (2008)
14. Goldberg, D. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Adison Wesley (1989)
15. Hansen, N.; Ostermeier, A. *Completely derandomized self-adaptation in evolution strategies*, Evolutionary Computation, Vol 9 (2), pp. 159-195 (2001)
16. Jones, D.R.; Perttunen C. D.; Stuckman B.E. *Lipschitzian Optimization without the Lipschitz Constant*, Journal of Optimization Theory and Application, Vol 79 (1), pp.157-181 (1993)
17. Jones, D.R.; Schonlau, M.;Welch, W.J. *Efficient Global Optimization of Expensive Black-Box Functions*, Journal of Global Optimization, Vol 13 (4), pp. 455-492 (1998)
18. Jones, D.R. *A taxonomy of global optimization methods based on response surfaces*, Journal of Global Optimization, Vol. 21, pp. 345-383 (2001)
19. Klimke, A. *Sparse Grid Interpolation Toolbox User's Guide*. Institut fur Angewandte Analysis und Numerische Simulation, Universitat Stuttgart (2008)

20. Mc Kay, M.D.; Conover, W.J.; Beckman, R.J. *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code*, Technometrics, Vol 21, pp. 239-245 (1979)
21. Montaz Ali, M.; Kharoenchai, C.; Zabinsky, Z.B. *A numerical evaluation of several stochastic algorithms on selected continuous global optimization test*, Journal of Global Optimization, Vol 31, pp. 635-672 (2005).
22. Marsden, A.L. ; Wang, M.; Dennis Jr, J.E.; Moin, P. *Optimal aeroacoustic shape design using the surrogate management framework*, Optimization and Engineering, Vol. 5(2), p. 235-262 (2004)
23. Moré, J.J.; Wild S. *Benchmarking derivative free optimization algorithms*, SIAM Journal of Optimization,Vol 20, 172-191 (2009)
24. Oliver, D.; Reynolds, A.; Liu, N. *Inverse theory for petroleum characterization and history matching*, Cambridge University Press (2008)
25. Roggero, F. *Constraining stochastic reservoir models to dynamic data: an integrated approach*. OAPEC seminar, Rueil, France, (2001)
26. Sinoquet, D.; Delbos, F. *Adapted nonlinear optimization method for production data and 4d seismic inversion*, ECMOR XI, expanded abstract B44 (2008)
27. Smolyak, S.A. *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Soviet. Math. Dokl., Vol.4, pp. 240-243 (1963)
28. Stephens, C.P.; Baritompa, W. *Global optimization requires global information*, Journal of Optimization Theory and Applications, Vol. 96, pp. 575-588 (1998)
29. Storn, R.; Price, K. *Differential Evolution A Simple and Efficient Heuristic for global Optimization over Continuous Spaces*, Journal of Global optimization, Vol 11, pp. 341-359 (1997)

Appendix A: the GOSGrid algorithm

In this appendix we first detail the GOSGrid algorithm. Input and output parameters are specified and the general steps of figure 2 are described.

Algorithm 1 Algorithme *GOSGrid*

Inputs
f: objective function
NevalMax: maximal number of available function evaluations
d: problem dimension
I: $d \times 2$ bound matrix
i **Outpouts:**
min: global minimum
$\mathbf{p}_{min}$: Matrix of optimal points

I & II.Global model construction and minimization
Set N=1; NEval=1; MIN=[]; FMIN=[].
while NEval $\leq$ NEvalMax **do**
Sparse Grid global model construction of level N of f.
if $N \geq 2$ **then**
Choice of the best grid point x_{grid}, computation of $f_{grid} = f(x_{grid})$.
Global minimization of the global model by a multi-start minimization algorithm in order to find x_{glob} and $f_{glob} = f(x_{glob})$.
Comparison between f_{grid} et f_{glob} in order to save the best point x_{min} and the best value f_{min}.
else
x_{min} is the central grid point and $f_{min} = f(x_{min})$.
end if
Update of the numer of performed function evaluations NEval.
Set MIN=[MIN; x_{min}]; FMIN=[FMIN, f_{min}].
III & IV. Refined model construction and minimization
Set M=2; $\Delta_1 = 0$; $\Delta_2 = 0$; MIN_{raff}=[]; FMIN_{raff}=[].
Calculation of the number of points NEval_{raff} needed to construct the refinement grid of level M.
while $\text{NEval}_{raff} \leq$ NEval **or** $\Delta_2 \geq 0.1 * \Delta_1$ **do**.
Refinement around x_{min}. Creation of the Sparse Grid refined model of level M.
Choice of the best refined grid point x_{raff}, computation of $f_{raff} = f(x_{raff})$.
Global minimization of the refined model by a multi-start minimization algorithm
in order to find x_{rep} et $f_{rep} = f(x_{rep})$.
Comparison between f_{rep} and f_{raff} to save the best point $x_{min_{raff}}$ and the best value $f_{min_{raff}}$.
Set MIN_{raff}=[MIN_{raff}; $x_{min_{raff}}$]; FMIN_{raff}=[FMIN_{raff}, $f_{min_{raff}}$].
Set MIN=[MIN; $x_{min_{raff}}$]; FMIN=[FMIN, $f_{min_{raff}}$].
if $M \geq 5$ **then**
Calculation of the relative change rate between the four last solutions:
$\Delta_1 = |\text{FMIN}_{raff}(\text{M-3}) - \text{FMIN}_{raff}(\text{M-4})|$,
$\Delta_2 = |\text{FMIN}_{raff}(\text{M-1}) - \text{FMIN}_{raff}(\text{M-2})|$.
end if
Set M $\leftarrow$ M+1.
Update NEval_{raff}.
end while
Set N $\leftarrow$ N+1.
Update NEval
end while

min is the lowest value of FMIN and $\mathbf{p}_{min}$ is a matrix containing in each line the lines of MIN for which the value of f is **min**.

Appendix B: benchmark information

In this appendix, the benchmark functions, their dimension and the search domain are specified. This benchmark is a collection of bound constrained optimization problems for global minimization proposed by A. L. Custodio and J. F. A. Madeira on a paper to appear [10]. It contains classical problems in optimization, with many local minima or particularities like long and narrow flat valleys. The problem dimensions are between 2 and 25. Domains are conceived in a way such that the global minimizer is not on its center. The reference paper for each problem is presented in the last column of table 3.

Function	Dimension	Domain	Reference
Ackley	10	$[-27\ 33]^{10}$	[21]
Aluffi-Pentini	2	$[-10\ 10]^{2}$	[21]
Becker-Lago	2	$[-50\ 50]^{2}$	[21]
Bohachevsky	2	$[-45\ 55]^{2}$	[21]
Branin-Hoo	2	$[-5\ 20]^{2}$	[21]
Cauchy	4	$[3\ 17]^{2}$	[7]
Cauchy	10	$[2\ 26]^{10}$	[7]
Cauchy	25	$[4.1\ 2745.6]^{10}$	[7]
Cosine mixture	2	$[-11]^{2}$	[7]
Cosine mixture	4	$[-11]^{4}$	[7]
Deeker-Aarts	2	$[-20\ 20]^{2}$	[21]
Epistatic Michalewicz	5	$[0\ \pi]^{5}$	[21]
Epistatic Michalewicz	10	$[0\ \pi]^{10}$	[21]
Exponential	2	$[-0.9\ 1.1]^{2}$	[7]
Exponential	4	$[-0.9\ 1.1]^{4}$	[7]
Fifteen local minima	2	$[-10\ 10]^{2}$	[7]
Fifteen local minima	4	$[-10\ 10]^{4}$	[7]
Fifteen local minima	6	$[-10\ 10]^{6}$	[7]
Fifteen local minima	8	$[-10\ 10]^{8}$	[7]
Fifteen local minima	10	$[-10\ 10]^{10}$	[7]
Fletcher-Powell	3	$[-10\ 10]^{3}$	[7]
Goldstein-Price	2	$[-2\ 2]^{2}$	[7]
Griewank	10	$[-360\ 440]^{10}$	[29]
Gulf	3	$[0.1\ 100;\ 0\ 25.6;\ 0\ 5]$	[21]
Hartman 4	3	$[0\ 1]^{3}$	[7]
Hartman 4	6	$[0\ 1]^{6}$	[7]
Hosaki	2	$[0\ 5;\ 0\ 6]$	[21]
Kowalik	4	$[0\ 0.42]^{4}$	[21]
Langerman	10	$[0\ 10]^{10}$	[21]
Mc Cormick	2	$[-1.5\ 4;\ -3\ 3]$	[21]
Meyer-Roth	3	$[-10\ 10]^{3}$	[7]
Miele-Cantrell	4	$[-10\ 10]^{4}$	[7]
Multi Gaussian	2	$[-2\ 2]^{2}$	[7]
Neumaier 2	4	$[0\ 4]^{4}$	[21]
Neumaire 3	10	$[-100\ 100]^{10}$	[21]
Odd Square	20	$[-15\ 15]^{20}$	[21]
Paviani	10	$[2.001\ 9.999]^{10}$	[21]
Periodic	2	$[-9\ 11]^{2}$	[21]
Poissonian	2	$[1\ 21;\ 1\ 8]$	[7]
Powell	4	$[-9\ 11]^{4}$	[21]
Rastrigin	10	$[-4.6\ 5.6]^{2}$	[21]

Rosenbrock	2	$[-5.12\ 5.12]^2$	[7]
Rosenbrock	10	$[-2.048\ 2.048]^{10}$	[7]
Salomon	5	$[-90\ 110]^5$	[21]
Salomon	10	$[-90\ 110]^{10}$	[21]
Schaffer 1	2	$[-90\ 110]^2$	[21]
Schaffer 2	2	$[-90\ 110]^2$	[21]
Schwefel	10	$[-450\ 550]^{10}$	[21]
Shekel 410	4	$[0\ 10]^4$	[7]
Shekel 45	4	$[0\ 10]^4$	[7]
Shekel 47	4	$[0\ 10]^4$	[7]
Shekel fox-holes	5	$[0\ 10]^5$	[21]
Shekel fox-holes	10	$[0\ 10]^{10}$	[21]
Shubert	2	$[-10\ 10]^2$	[21]
Sinusoidal	10	$[0\ 180]^{10}$	[21]
Sinusoidal	20	$[0\ 180]^{20}$	[21]
Six hump camel back	2	$[-2.5\ 1.5; -1.5\ 2.5$	[7]
Sphere	3	$[-4.65.6]^3$	[29]
Storn-Tchebychev	9	$[-128\ 128]^9$	[21]
Tenn local minima	2	$[-10\ 10]^2$	[7]
Tenn local minima	4	$[-10\ 10]^4$	[7]
Tenn local minima	6	$[-10\ 10]^6$	[7]
Tenn local minima	8	$[-10\ 10]^8$	[7]
Three hump camel back	2	$[-4.5\ 5.5]^2$	[21]
Transitor	9	$[-10\ 10]^9$	[21]
Wood	4	$[-10\ 10]^4$	[7]

Table 3 Benchmark functions, dimension, search domain and reference.