mathematics

MDPI

*Article*

# On a Parallelised Diffusion Induced Stochastic Algorithm with Pure Random Search Steps for Global Optimisation

**Manuel L. Esquível [1,*], Nadezhda P. Krasii [2], Pedro P. Mota [1] and Nélio Machado [3]**

[1] Department of Mathematics, Centre for Mathematics and Applications, NOVA School of Science and Technology, New University of Lisbon, 2829-516 Caparica, Portugal; pjpm@fct.unl.pt
[2] Department of Higher Mathematics, Faculty of Informatics and Computer Engineering, Don State Technical University, 344003 Rostov-on-Don, Russia; krasnad@yandex.ru
[3] Department of Mathematics, Faculty of Science and Technology, New University of Lisbon, 2829-516 Caparica, Portugal; neliomachado@gmail.com
* Correspondence: mle@fct.unl.pt

**Abstract:** We propose a stochastic algorithm for global optimisation of a regular function, possibly unbounded, defined on a bounded set with regular boundary; a function that attains its extremum in the boundary of its domain of definition. The algorithm is determined by a diffusion process that is associated with the function by means of a strictly elliptic operator that ensures an adequate maximum principle. In order to preclude the algorithm to be trapped in a local extremum, we add a pure random search step to the algorithm. We show that an adequate procedure of parallelisation of the algorithm can increase the rate of convergence, thus superseding the main drawback of the addition of the pure random search step.

**Keywords:** global optimisation; stochastic algorithms; pure random search; rate of convergence; parallelisation of algorithms



**Citation:** Esquível, M.L.; Krasii, N.P.; Mota, P.P.; Machado, N. On a Parallelised Diffusion Induced Stochastic Algorithm with Pure Random Search Steps for Global Optimisation. *Mathematics* **2021**, *9*, 3043. https://doi.org/10.3390/math9233043

Academic Editor: Alexander Grigoriev

Received: 26 October 2021
Accepted: 20 November 2021
Published: 26 November 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



## 1. Introduction

If we consider the partition of random type algorithms for global optimisation between those that are adaptive, that is, those that have a search distribution, at a given step, to depend on the distributions of previous steps (see [1–3] for comprehensive approaches and, for yet another example, [4]) and those algorithms with steps being independent distributed search variables, it seems that only with those of the second class can we expect to escape the curse of being trapped in local extrema. This curse is surely related to the well known slogan *global optimisation requires global information* first proposed by Stephens and Baritompa (see [5] and illustrated in [6]). One obvious way to counteract the pernicious effect of adaptivity of the algorithms is to include an intermediate *pure random search* (PRS) step (see [7]) that will overcame the proclivity of the algorithm to be trapped on a possible local extremum. However, this introduction comes with a price, namely the slowing down of the rate of convergence of the adaptive algorithm. As a countermeasure, we can consider the parallelisation of the algorithm at the pure random stage. Let us briefly state our main objectives in order to give some context to the remainder of this introduction. Our first objective in this work is to study the effect of parallelisation on the rate of convergence of a parallelizable stochastic algorithm for global optimisation with no constraints in the search for the maximum of a function. Our second objective is to study a random adaptive algorithm for global optimisation of functions that may be assumed to attain its extremums at the border of the bounded set where these functions are defined. Thorough and fundamental expositions on the general subject of random algorithms requiring familiarity with probability theory, as well as some knowledge of deterministic and stochastic differential equations and Markov chains, are found in [8,9]. A broad introduction to stochastic algorithms for global optimisation is found in [10]. An

excellent reference treaty on stochastic methods for global optimisation is found in [11]; the authors lay a solid ground for further studies, for instance, in detailing extreme value theory and its applications to statistical inference in some stochastic optimisation algorithms. The quest for better performing algorithms for stochastic optimisation continues to develop newer approaches aiming at more general applicability, provable convergence, and stability in regard to small changes in the parameters. An interesting example of such a work is [12], where the objective functions are supposed to be weakly convex, having as a particular case compositions of convex losses with smooth functions seen in machine learning algorithms. Another example may be found in the recent work [13]; the algorithm proposed is akin to the *zig-zag* algorithm first introduced in [14]—further studied in [15] and with its convergence also studied in [6]—and a thorough comparison of the performance of the introduced algorithm with other well known stochastic optimisation algorithms is detailed. Further results on the convergence of global stochastic optimisation algorithms using perturbed Liapunov function methods are presented in [16]. Let us stress that the subject of *stochastic optimisation* is broad, encompassing many different types of algorithms such as simulated annealing, genetic algorithms, and tabu search; a review of these different types is provided in [17], detailing advantages and disadvantages and summarising the literature on optimal values of the inputs. There are many instances of the idea that the algorithm may, in itself, guide the search into more promising regions; for instance, in [18], modifications of the pure random search algorithm are proposed following this idea—implemented using the regularity of the function to determine the more promising regions—and we quote, *...We think that if the objective function satisfies some conditions such as continuity, the Lipschitz condition or differentiability etc., the regions containing the minimal points are easy to be determined. If the algorithm generates many points in the regions of this kind, the probability of success will be large.*

Fruitful ideas to build global optimisation algorithms with stochastic differential equations were previously published in, for instance, [19–26]. For numerical methods in stochastic differential equations, a fundamental reference is [27]. The splendid book [19] provides a well-founded approach to the convergence of algorithms either using functional compactness arguments or weak convergence results; some of the algorithms studied are taken as perturbations of ideal deterministic algorithms by an additive noise, not necessarily Gaussian, satisfying regularity properties. The idea of defining the intensity of the additive noise, the volatility of the stochastic differential equation by an annealing rate decreasing to zero with increasing time, is proposed in [21], and weak convergence is studied. In [25], the convergence of the studied algorithm for global optimisation with constraints is established for general nonconvex, smooth functions; the stochastic differential equation underpinning the algorithm has as a drift the symmetric of the gradient of the function—a well known approach for obtaining a solution to an unconstrained optimisation problem considering the ordinary differential equation—and for the volatility, the square root of a positive function denominated *annealing schedule* with a parameter to be chosen according to the problem under analysis. This approach was already studied in [20], where a detailed justification for the choice of the stochastic differential equation is given, alongside many interesting related results. In [26], the authors use Euler discretisation of a stochastic differential equation with mean reversion, in which the volatility is suitably modified in order to reduce the Gaussian noise contribution as the number of iterations increase; the method is tested against 14 classical functions, and the results indicate that the random search guided by the trajectory of a discretised diffusion perform well. The fine study of stochastic algorithms based on stochastic differential equations can be read in [28], where, using perturbed Liapunov function methods, stability results of the algorithms are established. Furthermore, in [16], an algorithm of simulated annealing-type procedure is studied and the following recommendation is stated: *...whenever possible, one can and should use parallel processors....*

In this work, we do not need much of a theoretical background on parallel computing, as we use a standard computation tool to take advantage of the processors being able

to compute either with the specification of working independently or not. Some general references for the study of parallel algorithms are [29–31]. Early efforts on the parallelisation of those stochastic algorithms that may be called by the *random search* type can be read in [32], where a particular structure of the algorithm is designed using several units capable of processing data independently, notwithstanding communication with the units during the computation. In [29], a collective work, there is an excellent review of the main ideas of parallelisation techniques for global optimisation algorithms. More recently, in [33], the authors propose a framework to estimate the parallel performance of a given algorithm by analysing the runtime behaviour of its sequential version; the goal is achieved by approximating the runtime distribution of the sequential process with statistical methods, and then the runtime behaviour of the parallel process can be predicted by a model based on order statistics. The method we develop in this work bears some resemblance with this general idea.

As we previously observed, some fundamental random search algorithms for global optimisation built using discretised stochastic differential equations have as volatilities—that is, variances—parametrised quantities derived from simulated annealing principles. Parallelisation of simulated annealing type algorithms were studied by several authors. A precursor work is [34], where the simulated annealing algorithm is mapped onto a dynamically structured tree of processors; the algorithm presented achieves *speedups* between $\log_2 N$ and $(N + \log_2 N)/2$, $N$ being the number of processors. Another important work is a volume edited by Robert Azencott, from which [35] is a paradigmatic chapter. In [36], five parallel algorithms for global optimisation are studied; these algorithms are categorised by the amount of information exchanged among the different parallel runs with the *asynchronous* approach, being such that no information is exchanged among parallel runs. The parallelisation we propose is asynchronous, and the parallel runs are independent. In [37], there are three proposals of parallel algorithms for simulated annealing that fall in a similar categorisation for the the amount of information exchanged among the different parallel runs leading to independent, semi-independent, and co-operating searches. An important reference on the same vein is [38], reporting on five conversions of simulated annealing algorithm from sequential-to-parallel forms on high-performance computers and their application to a set of standard function optimisation problems. In the algorithm proposal of [39], a common feature of simulated annealing algorithms, that is, adaptive cooling based on variance, is abandoned; instead, the algorithm resamples the states across processors periodically, and the resampling interval is tuned according to the success rate for each specific number of processors.

We now present in greater detail the contents and main contributions of this work. The first one is to develop and study a novel formal description of parallelisation of random algorithms for global optimisation with no constraints, and the second one is to introduce and study a novel stochastic differential equation-based algorithm suited to functions attaining the extremum at some points of the border of the domain of definition, for instance, unbounded functions. On the first contribution, in Theorem 1, we detail a proof of the convergence of the *pure random search* algorithm for measurable functions, using simple results of martingale theory in discrete time, with the main purpose of highlighting the perspective used in this work, to wit, a stochastic algorithm may be seen as a sequence of random variables. With this perspective, in Theorem 2, we show that two convergent pure random search algorithms associated with the same function converge to random variables with the same law. We define, in Definition 3, the rate of convergence of a random algorithm and, in Definition 4, the parallelisation of such an algorithm, following the MPI standard, which is designed for distributed memory with multiple kernels of computation. Theorem 3 is another of the main contributions of this work, since we prove an estimate showing the improvement of the rate of convergence of a parallelised algorithm measured with respect to the nonparallelised version of the algorithm. We show that the parallelisation of the pure random search algorithm behaves in conformity with the results of Theorem 3 with examples of standard test functions for global

optimisation. For the second main contribution of this work, in Section 4, we introduce a novel algorithm using stochastic differential equations based on a maximum principle, which is specially suited for unbounded functions; we consider an example that we treat both in the parallelised and nonparallelised versions of the algorithm, thus showing again the advantage of parallelising a random algorithm. The proposed algorithm is different from the algorithms proposed in the literature, in particular, in the works referred above in the paragraph on global optimisation algorithms with stochastic differential equations, and we intend to further explore the properties of this new algorithm in future work. Broadly stated, the main difference stems from the fact that the stochastic differential equation is built in order for the algorithm to be a martingale and for a maximum principle to be applied.

## 2. On the Parallelisation of Random Algorithms

The introduction of a pure random search step in an adaptive stochastic algorithm for global optimisation is an auxiliary device to avoid missing the global extremum. However, keeping in mind that adaptive algorithms are introduced to increase the convergence rates, the pure random search step is counterproductive, since it slows down the whole algorithm.

Thus, we propose parallelisation of the algorithm at the pure random search step level in order to improve the global rate of convergence of the algorithm. For completeness, we first describe a martingale approach to pure random search. Next, in Section 2.2, we show that, with an appropriate definition of the rate of convergence, parallelisation of a stochastic algorithm improves the global rate of convergence.

### *2.1. Pure Random Search Revisited*

A probabilistic description of the pure random search may be defined as follows.

**Definition 1** (Pure random search for global optimisation)**.** *A pure random search algorithm for the global maximisation of a function $f$ on a compact set $K$ is a sequence $(f(Y_n))_{n\geq 0}$ of random variables, such that:*

***(i)*** *The function $f : K \subset \mathbb{R}^d \longrightarrow \mathbb{R}$, defined on $K$ a compact set of $\mathbb{R}^d$, is measurable.*

***(ii)*** *The sequence $X_0, X_1, \ldots X_n, \ldots$ is a sequence of independent and identically distributed (IID) random variables with $X \frown \mathcal{U}(K)$, that is, $X$ is uniformly distributed in $K$. We recall that, with $\lambda$ denoting the Lebesgue measure over $\mathbb{R}^d$ we have—with the symbol $\frown$ denoting **with probability law**—that:*

$$X \frown \mathcal{U}(K) \Leftrightarrow \forall B \in \mathcal{B}(K)\ , \mathbb{P}[B] = \frac{1}{\lambda(K)} \int_B d\lambda\ . \tag{1}$$

***(iii)*** *The sequence $Y_0, Y_1, \ldots Y_n, \ldots$ is a sequence of random variables with values in $K$ defined for almost all $\omega \in \Omega$ by $Y_0 = X_0$ and for $n \geq 1$:*

$$Y_{n+1}(\omega) = \begin{cases} X_{n+1}(\omega) & \text{if } f(X_{n+1}(\omega)) > f(Y_n(\omega)) \\ Y_n(\omega) & \text{if } f(X_{n+1}(\omega)) \leq f(Y_n(\omega))\ , \end{cases} \tag{2}$$

*with the consequence that $(f(Y_n))_{n\geq 0}$ is a nondecreasing sequence for almost all $\omega \in \Omega$.*

Four our purposes, we recall an essential definition.

**Definition 2** (The essential supremum of a function over a compact set)**.** *The **essential supremum** $\alpha_f$ of the function $f$ over $K$ is given by:*

$$\begin{aligned} \alpha_f &:= \inf\{t : \lambda(\{x \in K : f(x) > t\}) = 0\} = \\ &= \inf\{t : f(x) \leq t\ ,\ \lambda \text{ almost everywhere on } K\} = \\ &= \sup\{t : \lambda(\{x \in K : f(x) > t\}) > 0\}\ . \end{aligned} \tag{3}$$

We next single out Hypothesis 1 with the purpose of focusing on the most interesting cases for our study. In Remark 1, we identify the extension of Theorem 1 to the excluded cases of this hypothesis.

**Hypothesis 1.** *We will suppose that $\alpha_f \in \mathbb{R}$. This hypothesis implies that, for all $p \geq 1$, we have that $f$ is of p-power integrable (see for instance ([40], p. 190)).*

We now present the convergence result for the random algorithm of pure random search for global optimisation.

**Theorem 1** (On the convergence of the pure random search algorithm)**.** *Under Hypothesis 1, we have that, for the pure random search algorithm in Definition 1:*

1. *The sequence $(f(Y_n))_{n\geq 0}$ is a **submartingale** with respect to its natural filtration $\mathbb{F} = (\mathcal{F}_n)_{n\geq 0}$, that is, with: $\mathcal{F}_n := \sigma(f(Y_0), f(Y_1), \ldots, f(Y_n))$.*
2. *The sequence $(f(Y_n))_{n\geq 0}$ converges **almost surely** to a random variable $\mathbb{Y}_\infty^f$ such that, with $\alpha_f$ the essential supremum of $f$ over $K$ in Definition 2,*

$$\mathbb{P}\left[\mathbb{Y}_\infty^f \geq \alpha_f\right] = 1\,, \tag{4}$$

*that is, the random variable $\mathbb{Y}_\infty^f$ is a **strong** solution of the global optimisation problem of $f$ over $K$.*

**Proof.** We first observe that the random variables of the sequence $(f(Y_n))_{n\geq 0}$ are integrable. In fact as we have, by Formula (2) and by the righthand side of Formula (1), that:

$$\begin{aligned}\mathbb{E}[|f(Y_{n+1})|] = \mathbb{E}\left[\left|f(X_{n+1})\mathbb{1}_{\{f(X_{n+1})>f(Y_n)\}} + f(Y_n)\mathbb{1}_{\{f(X_{n+1})\leq f(Y_n)\}}\right|\right] \leq \\ \leq \mathbb{E}[|f(X_{n+1})|] + \mathbb{E}[|f(Y_n)|] = \frac{1}{\lambda(K)}\int_K |f(x)|d\lambda + \mathbb{E}[|f(Y_n)|]\,,\end{aligned} \tag{5}$$

and we may conclude by induction. Now, since the conditional expectations are well defined and the sequence $(f(Y_n))_{n\geq 0}$ is a nondecreasing sequence for almost all $\omega \in \Omega$, we have that:

$$0 \leq \mathbb{E}[f(Y_{n+1}) - f(Y_n)|\,\mathcal{F}_n] = \mathbb{E}[f(Y_{n+1})|\,\mathcal{F}_n] - f(Y_n)\,,$$

and so we have the first statement in Theorem 1. In order to prove that the submartingale $(f(Y_n))_{n\geq 0}$ converges, we will take advantage of Hypothesis 1 to obtain an uniform bound on the terms in the furthest left-hand side of Formula (5). Since we have that:

$$\alpha_f = \inf\{t : f(x) \leq t\,,\ \lambda \text{ almost everywhere on } K\} < +\infty\,,$$

we have that, for every $\epsilon > 0$, there exists $t_\epsilon$, such that $\alpha_f \leq t_\epsilon < \alpha_f + \epsilon$, and such that $f(x) \leq t_\epsilon$ almost everywhere on $K$ with respect to $\lambda$. This implies the uniform bound:

$$\sup_{n\geq 0} \mathbb{E}[|f(Y_n)|] \leq t_\epsilon < +\infty\,. \tag{6}$$

As a consequence of the bound in Formula (6), we have, by a standard result on martingale theory (see ([41], p. 50)), that the submartingale $(f(Y_n))_{n\geq 0}$ converges almost surely to a random variable that we denote by $\mathbb{Y}_\infty^f$. In order to prove that $\mathbb{Y}_\infty^f$ is a strong solution of the maximisation problem for $f$ over $K$—defined to be a solution such that Formula (4) is satisfied—we consider, for an arbitrary $\epsilon > 0$, the fixed set defined by:

$$E_{\alpha_f - \epsilon} := \left\{x \in K : f(x) > \alpha_f - \epsilon\right\}\,, \tag{7}$$

and we proceed with the following sequence of observations. Without loss of generality, we may assume, as an Hypothesis 2, that $\alpha_f > 0$, since, by Hypothesis 1, if it is otherwise, we can always consider a translation of $f$.

**(j)** For $\epsilon < \alpha_f$ we have that:

$$\mathbb{P}\left[\limsup_{n\to+\infty}\left\{X_n \in E_{\alpha_f-\epsilon}\right\}\right] = 1\,. \tag{8}$$

We first observe that, by the fact that $\alpha_f = \sup\{t : \lambda(\{x \in K : f(x) > t\}) > 0\}$, and as a consequence of a standard *supremum* argument, we have that

$$\begin{aligned}\mathbb{P}\left[X_n \in E_{\alpha_f-\epsilon}\right] = \mathbb{P}\left[f(x) > \alpha_f - \epsilon\right] \geq \frac{1}{\lambda(K)}\int_{\{f(x)>\alpha_f-\epsilon\}} f(x)d\lambda(x) \geq \\ \geq (\alpha_f-\epsilon)\frac{\lambda(\{x : f(x) > \alpha_f - \epsilon\})}{\lambda(K)} > 0\,.\end{aligned} \tag{9}$$

Now, since the sequence $(X_n)_{n\geq 0}$ is an IID sequence, by the converse of Borel–Cantelli lemma, as a consequence of Formula (9), $\sum_{n=0}^{+\infty}\mathbb{P}\left[X_n \in E_{\alpha_f-\epsilon}\right] = +\infty$, and so the announced result in Formula (8) follows.

**(jj)** Almost surely, we will have $Y_n \in E_{\alpha_f-\epsilon}$ for some integer $n \geq 0$, that is, more precisely,

$$\exists\Omega_0 \in \mathcal{F}\,, \mathbb{P}[\Omega_0] = 1\,,\ \forall\omega_0 \in \Omega_0\ \ \exists n_0 = n_0(\omega_0)\,,\ \ Y_{n_0}(\omega_0) \in E_{\alpha_f-\epsilon}\,.$$

Let us suppose the contrary, that is,

$$\forall\Omega_0 \in \mathcal{F}\,, \mathbb{P}[\Omega_0] = 1\,,\ \exists\omega_0 \in \Omega_0\ \ \forall n\,,\ \ Y_n(\omega_0) \notin E_{\alpha_f-\epsilon}\,. \tag{10}$$

Consider, for $\epsilon < \alpha_f$, the set,

$$\Omega_0 := \limsup_{n\to+\infty}\{X_n \in E_{\alpha_f-\epsilon}\} = \bigcap_{m\geq 0}\bigcup_{n\geq m}\{X_n \in E_{\alpha_f-\epsilon}\}\,, \tag{11}$$

which we know, by Formula (8), to have full probability and select $\omega_0 \in \Omega_0$ according to the condition in Formula (10). Now, by the definition of $\Omega_0$ in Formula (11), there exists $n_0 > 1$, such that $X_{n_0}(\omega_0) \in E_{\alpha_f-\epsilon}$, that is such that $f(X_{n_0}(\omega_0)) > \alpha_f - \epsilon$. However, by the condition in Formula (10), we have that $f(Y_{n_0-1}(\omega_0)) \leq \alpha_f - \epsilon$, and so, by the definition of the sequence $(Y_n)_{n\geq 0}$ in Formula (2), we should have that $Y_{n_0}(\omega_0) = X_{n_0}(\omega_0)$ and so $f(Y_{n_0}(\omega_0)) > \alpha_f - \epsilon$, that is, $Y_{n_0}(\omega_0) \in E_{\alpha_f-\epsilon}$, a statement that contradicts the condition in Formula (10).

**(jjj)** We have the final conclusion of the second statement in the theorem, Formula (4), that is:

$$\mathbb{P}\left[\mathbb{Y}_\infty^f \geq \alpha_f\right] = 1\,.$$

Take $\epsilon < \alpha_f$ and $\Omega_0 \in \mathcal{F}$ satisfying Observation **(jj)**, that is, a set of full probability and such that,

$$\forall\omega_0 \in \Omega_0\ \ \exists n_0 = n_0(\omega_0)\,,\ \ Y_{n_0}(\omega_0) \in E_{\alpha_f-\epsilon}\,.$$

Now, since the sequence $(f(Y_n))_{n\geq 0}$ is almost surely nondecreasing, for all $\omega_0 \in \Omega_0$ and the adequate $n_0 = n_0(\omega_0)$, we can write:

$$\forall n \geq n_0\,,\ f(Y_n(\omega_0)) \geq f(Y_{n_0}(\omega_0)) > \alpha_f - \epsilon\,,$$

a condition that implies $\lim_{n\to+\infty} f(Y_n(\omega_0)) =: \mathbb{Y}_\infty^f(\omega_0) \geq \alpha_f - \epsilon$. We have thus shown that

$$\forall 0 < \epsilon < \alpha_f\,,\ \mathbb{P}\left[\mathbb{Y}_\infty^f \geq \alpha_f - \epsilon\right] = 1\,,$$

a condition that, by a standard argument, implies Formula (4).

Being so, the proof of the theorem is now complete. □

**Remark 1** (On the case of an infinite essential supremum)**.** *In order to extend Theorem 1 to the case $\alpha_f = +\infty$, we may consider that*

$$\sup_{n\geq 0} \mathbb{E}\left[f^+(Y_n)\right] < +\infty\,,$$

*a condition that, for any submartingale, is equivalent to $\sup_{n\geq 0} \mathbb{E}[|f(Y_n)|] < +\infty$, that is, the sequence $(f(Y_n))_{n\geq 0}$ is bounded in $L^1$ (see ([41], pp. 50–51)). Furthermore, by observing that, if*

$$\alpha_f = \sup\{t : \lambda(\{x \in K : f(x) > t\}) > 0\} = +\infty\,,$$

*we have for arbitrary $M > 0$ that $\lambda(\{x \in K : f(x) > M\}) > 0$, and so we can consider, instead of $E_{\alpha_f - \epsilon}$, the set $E_{\alpha_f, M} = \{x \in K : f(x) > M\}$. We then show that $\mathbb{P}\left[\mathbb{Y}_\infty^f \geq M\right] = 1$ with a similar proof, a condition that, in turn, implies that $\mathbb{Y}_\infty^f = +\infty$ almost surely.*

We now state a remark and an important result on the laws of the solution of the global optimisation problem by means of *pure random search* that we will use in the sequel.

**Remark 2** (On the laws of the variables of the algorithm)**.** *Consider the two sequences $X_0, X_1, \ldots X_n, \ldots$, and $X'_0, X'_1, \ldots X'_n, \ldots$ of IID random variables with $X \frown \mathcal{U}(K)$, and let $f(Y_0), f(Y_1), \ldots f(Y_n), \ldots$ and $f(Y'_0), f(Y'_1), \ldots f(Y'_n), \ldots$ be the corresponding pure random search algorithms according to Definition 1. Then, for all $n \geq 0$, we have that $f(Y_n)$ and $f(Y'_n)$ have the same law. In fact, a simple proof by induction shows that, for every $n \geq 0$, the law of $Y_n$—and consequently, the law of $f(Y_n)$—only depends on the law of $(X_0, X_1, \ldots X_n)$.*

**Theorem 2** (On the unicity of the global optimisation solution of pure random search)**.** *Suppose that $\alpha_f$, the essential supremum of $f$ over the compact $K$, is finite. Let $(f(Y_n))_{n\geq 0}$ and $(f(Y'_n))_{n\geq 0}$ be two pure random search algorithms associated with $X_0, X_1, \ldots X_n, \ldots$ and $X'_0, X'_1, \ldots X'_n, \ldots$, two sequences of IID random variables with $X \frown \mathcal{U}(K)$, as in Remark 2. Suppose that these random algorithms converge to $\mathbb{Y}_\infty$ and $\mathbb{Y}'_\infty$, respectively. Then, we have that $\mathbb{P}[\mathbb{Y}_\infty \neq \mathbb{Y}'_\infty] = 0$, and so $\mathbb{Y}_\infty$ and $\mathbb{Y}'_\infty$ have the same law.*

**Proof.** We can consider the sequence defined for $n \geq 0$ by:

$$Z_n := \begin{cases} Y_n & \text{if } f(Y_n) \geq f(Y'_n) \\ Y'_n & \text{if } f(Y_n) < f(Y'_n)\,. \end{cases} \tag{12}$$

Now, we may observe the following. Firstly, we have that, almost surely for $n \geq 0$, $f(Z_n) = \max(f(Y_n), f(Y'_n))$ as an immediate consequence of the definition in Formula (12). Then, we have that, almost surely, $(f(Z_n))_{n\geq 0}$ is a nondecreasing sequence, as it is defined as the pointwise maximum of two nondecreasing sequences. Lastly, since, for all $n \geq 0$, we have that $Y_n$ and $Y'_n$ have the same law (see Remark 2), and since, by Formula (2), we have that:

$$Z_{n+1} = \begin{cases} Y_{n+1} = \begin{cases} X_{n+1}(\omega) & \text{if } f(X_{n+1}(\omega)) > f(Y_n(\omega)) \\ Y_n(\omega) & \text{if } f(X_{n+1}(\omega)) \leq f(Y_n(\omega))\,, \end{cases} & \text{if } f(Y_n) \geq f(Y'_n) \\ Y'_{n+1} = \begin{cases} X'_{n+1}(\omega) & \text{if } f(X'_{n+1}(\omega)) > f(Y'_n(\omega)) \\ Y'_n(\omega) & \text{if } f(X'_{n+1}(\omega)) \leq f(Y'_n(\omega))\,, \end{cases} & \text{if } f(Y_n) < f(Y'_n)\,, \end{cases} \tag{13}$$

then $(f(Z_n))_{n\geq 0}$ is a pure random search algorithm for global optimisation of $f$ over $K$ that, by Theorem 1, converges almost surely to some random variable $\mathbb{X}_\infty$. Since we have that,

for almost all $\omega \in \Omega$, the sequences $(f(Y_n(\omega)))_{n\geq 0}$ and $(f(Y'_n(\omega)))_{n\geq 0}$ are subsequences of $(f(Z_n(\omega)))_{n\geq 0}$, we immediately have that $\mathbb{Y}_\infty = \mathbb{X}_\infty = \mathbb{Y}'_\infty$ almost surely, as claimed. □

*2.2. On the Rate of Convergence of a Parallelisation of a Random Algorithm*

In this section, we use the Ky Fan distance to define the rate of convergence of a generic random algorithm, and we obtain natural bounds for the rate of convergence of a parallelisation of this random algorithm. The *pure random search* developed in Section 2.1 is a basic algorithm, to which the following results apply.

Consider a stochastic algorithm for global optimisation—for instance, on what follows, maximisation—of a measurable function $f$ on a regular domain $D \in \mathcal{B}(\mathbb{R}^{\mathrm{d}})$ given by the sequence of random variables $(f(Y_n))_{n\geq 0}$. We will suppose that $(f(Y_n))_{n\geq 0}$ converges, at least in probability, to some random variable $\mathbb{Y}_\infty$. We recall that $d_{KF}$, the Ky Fan metric (see ([42], p. 289)), is defined for two random variables $X$ and $Y$ to be:

$$d_{KF}(X,Y) := \inf\{\epsilon > 0 : \mathbb{P}[|X-Y| > \epsilon] \leq \epsilon\}\ ,$$

and that this distance is the distance of the convergence in probability. Furthermore, we have that $\lim_{n\to+\infty} d_{KF}(f(Y_n), \mathbb{Y}_\infty) = 0$.

We now introduce a quantitative notion for the rate of convergence of an algorithm.

**Definition 3** (The rate of convergence of a random algorithm)**.** *Suppose that the random algorithm $(f(Y_n))_{n\geq 0}$ converges in probability to some random variable $\mathbb{Y}_\infty$. The rate of convergence of the algorithm is given by the numerical sequence:*

$$(d_{KF}(f(Y_n), \mathbb{Y}_\infty))_{n\geq 0}\ .$$

*that is, the rate of convergence of the algorithm is the rate of convergence to zero of the numerical sequence $\left(d_{KF,n}^{\mathbb{Y}_\infty}\right)_{n\geq 0}$, with, by definition, $d_{KF,n}^{\mathbb{Y}_\infty} := d_{KF}(f(Y_n), \mathbb{Y}_\infty)$.*

We now consider a definition of parallelisation of a random algorithm.

**Definition 4** (Parallelisation of a random algorithm)**.** *Given a random algorithm $(f(Y_n))_{n\geq 0}$ for global optimisation of a measurable function $f$ on a regular domain $D \in \mathcal{B}(\mathbb{R}^{\mathrm{d}})$, converging in probability to some random variable $\mathbb{Y}_\infty$, we say that this algorithm is parallelisable if, for every integer $r > 1$ and for every $n \geq 0$, there exist $r$ independent copies of the terms $f(Y_n)$ of the algorithm sequence $(f(Y_n))_{n\geq 0}$ and of the random variable $\mathbb{Y}_\infty$—namely $f(Y_n^j)$ and $\mathbb{Y}_\infty^j$ for $j = 1, \ldots, r$ and every $n \geq 0$—such that the following two conditions are verified:*

1. *For all $j = 1, \ldots, r$, we have that $\lim_{n\to+\infty} d_{KF}(f(Y_n^j), \mathbb{Y}_\infty^j) = 0$.*
2. *For all $j = 1, \ldots, r$, the random variable $\mathbb{Y}_\infty^j$ is a solution of the **weak** optimisation problem for $f$ on $D$, that is, $\alpha_f \leq \mathbb{E}\left[\mathbb{Y}_\infty^j\right]$, with $\alpha_f$ the essential supremum of $f$ (see Definition 2).*

**Remark 3** (PRS as an example of a parallelizable algorithm)**.** *The pure random search of a measurable function $f$ on a compact set $K$ studied in Section 2.1 is parallelisable. In fact, given any two samples of $X \frown \mathcal{U}(K)$, as in Remark 2 and Theorem 2, the correspondent terms of the two algorithms generated by the two samples are independent as a consequence of the fact that, as seen in Remark 2, the law of $f(Y_n)$ only depends on the law of the initial segment $(X_0, X_1, \ldots X_n)$ of the sample of $X \frown \mathcal{U}(K)$ used to define it. It thus follows firstly that the limit random variables are independent, because almost sure limits of term-by-term independent sequences are independent by an application of a standard reasoning using characteristic functions. Moreover, as by Theorem 2, both limits of the two algorithms are equal almost surely if one of these limits is a solution of the (strong) optimisation problem.*

We now state the result that compares the parallelisation of a random algorithm with the version of the algorithm without parallelisation. This result can be used to determine stopping criteria for parallelised random algorithms, as detailed in Remark 5.

**Theorem 3** (On the rate of convergence of a parallelised random algorithm)**.** *Let $(f(Y_n))_{n\geq 0}$ be a parallelizable random algorithm for global optimisation of a measurable function $f$ on a regular domain $D \in \mathcal{B}(\mathbb{R}^{\mathrm{d}})$ and let the limit in probability of the algorithm and solution of the global optimisation problem be the $\mathbb{Y}_\infty$ random variable. Let $r > 1$ be an integer and let $(f(Y_n^j))_{n\geq 0}$ and $\mathbb{Y}_\infty^j$, for $j = 1, \ldots, r$, be the independent copies of the algorithm and of the solution of the optimisation problem according to Definition 4. We then have for all $n \geq 0$:*

$$\mathbb{P}\left[\min_{1\leq j\leq r}\left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| \geq d_{KF,n}^{\mathbb{Y}_\infty}\right] \leq \left(d_{KF,n}^{\mathbb{Y}_\infty}\right)^r , \tag{14}$$

*and also, for all $n \geq 0$ such that $d_{KF,n}^{\mathbb{Y}_\infty} < 1$,*

$$\mathbb{P}\left[\max_{1\leq j\leq r}\left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| \geq d_{KF,n}^{\mathbb{Y}_\infty}\right] \leq 1 - \left(1 - d_{KF,n}^{\mathbb{Y}_\infty}\right)^r \leq r \cdot d_{KF,n}^{\mathbb{Y}_\infty} . \tag{15}$$

**Proof.** For the proof, we will use Lemma 1, which is well known, but which we reproduce next for the reader's convenience. Since we have, by definition, that:

$$d_{KF,n}^{\mathbb{Y}_\infty} = \inf\{\epsilon > 0 : \mathbb{P}[|f(Y_n) - \mathbb{Y}_\infty| > \epsilon] \leq \epsilon\} , \tag{16}$$

we also have that, for every integer $k \geq 1$,

$$\exists \epsilon_k > 0 \ , \ d_{KF,n}^{\mathbb{Y}_\infty} \leq \epsilon_k < d_{KF,n}^{\mathbb{Y}_\infty} + \frac{1}{k} \ \text{ and } \ \mathbb{P}[|f(Y_n) - \mathbb{Y}_\infty| > \epsilon_k] \leq \epsilon_k \ .$$

Without loss of generality, we may assume that the sequence $(\epsilon_k)_{k\geq 1}$ is nondecreasing. Now, the algorithm being parallelisable, by Definition 4, we have the necessary conditions to apply the estimate in Formula **(a)** of Lemma 1, and so we obtain

$$\mathbb{P}\left[\min_{1\leq j\leq r}\left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| > \epsilon_k\right] = \mathbb{P}[|f(Y_n) - \mathbb{Y}_\infty| > \epsilon_k]^r \leq (\epsilon_k)^r \leq \left(d_{KF,n}^{\mathbb{Y}_\infty} + \frac{1}{k}\right)^r .$$

The bound in Formula (14) now follows by taking the limit as $k$ goes to infinity, since $\lim_{k\to+\infty} \epsilon_k = d_{KF,n}^{\mathbb{Y}_\infty}$. The proof of Formula (15) is a consequence of applying the estimate in Formula **(b)** of Lemma 1 to obtain

$$\mathbb{P}\left[\max_{1\leq j\leq r}\left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| > \epsilon_k\right] = 1 - (1 - \mathbb{P}[|f(Y_n) - \mathbb{Y}_\infty| > \epsilon_k])^r ,$$

and then by taking the limit as $k$ goes to infinity. □

For the reader's convenience, we recall the following well known elementary result that was used in the proof of Theorem 3.

**Lemma 1.** *Let $(Z_i)_{1\leq i\leq r}$ be a sequence of non-negative, independent and identically distributed with a random variable $Z$. We then have, for all $\epsilon$:*

***(a)*** $\mathbb{P}[\min_{1\leq i\leq r} Z_i > \epsilon] = \mathbb{P}[Z > \epsilon]^r$

***(b)*** $\mathbb{P}[\max_{1\leq i\leq r} Z_i > \epsilon] = 1 - (1 - \mathbb{P}[Z > \epsilon])^r$

**Proof.** We have that

$$\bigcup_{1\leq i\leq r} \{Z_i \leq \epsilon\} = \left\{\min_{1\leq i\leq r} Z_i \leq \epsilon\right\} \Leftrightarrow \bigcap_{1\leq i\leq r} \{Z_i > \epsilon\} = \left\{\min_{1\leq i\leq r} Z_i > \epsilon\right\} ,$$

and so it follows, by the independence and the identical distributions of $Z_i$, that:

$$\mathbb{P}\left[\min_{1\leq i\leq r} Z_i > \epsilon\right] = \mathbb{P}\left[\bigcap_{1\leq i\leq r} \{Z_i > \epsilon\}\right] = \prod_{1\leq i\leq r} \mathbb{P}[Z_i > \epsilon] = \mathbb{P}[Z > \epsilon]^r \,.$$

The proof of the second assertion is similar by observing that

$$\left\{\max_{1\leq i\leq r} Z_i \leq \epsilon\right\} = \bigcap_{1\leq i\leq r} \{Z_i \leq \epsilon\} \Rightarrow \mathbb{P}\left[\max_{1\leq i\leq r} Z_i \leq \epsilon\right] = \mathbb{P}[Z \leq \epsilon]^r \,,$$

which is equivalent to the second stated formula. □

**Remark 4** (On the rate of convergence of a parallelised random algorithm)**.** *The rate of convergence of an algorithm* $(f(Y_n))_{n\geq 0}$*, as in Definition 3, is the rate of convergence to zero of the sequence* $\left(d_{KF,n}^{\mathbb{Y}_\infty}\right)_{n\geq 0}$*, with the terms of the sequence being as in Formula (16). In this formula, the infimum is attained (see again ([42], p. 289)), and so we have that:*

$$\mathbb{P}\left[|f(Y_n) - \mathbb{Y}_\infty| > d_{KF,n}^{\mathbb{Y}_\infty}\right] \leq d_{KF,n}^{\mathbb{Y}_\infty} \,. \tag{17}$$

*If this random algorithm is parallelisable, as in Theorem 3, we may consider that the gain in performance with this parallelisation is observed on the behaviour of*

$$\min_{1\leq j\leq r} \left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| \,, \tag{18}$$

*a behaviour that can be assessed in Formula (14), that is,*

$$\mathbb{P}\left[\min_{1\leq j\leq r} \left|f(Y_n^j) - \mathbb{Y}_\infty^j\right| \geq d_{KF,n}^{\mathbb{Y}_\infty}\right] \leq \left(d_{KF,n}^{\mathbb{Y}_\infty}\right)^r \,.$$

*As a consequence, we should compare this behaviour with the behaviour of the original algorithm in Formula (17). It is clear that, when* $d_{KF,n}^{\mathbb{Y}_\infty} < 1$*, there is, for the parallelised algorithm, a remarkable improvement in the probability of a deviation from zero of the random variable in Formula (18) that measures the gain in performance with the parallelisation.*

**Remark 5** (On the actual implementation of a parallelisation of a random algorithm)**.** *One way for an actual implementation of a random algorithm to take advantage of the improvement in the performance gained with the parallelisation, observed in Remark 4, is the following. Let* $M_j^n$ *for* $j = 1, \ldots, r$ *be the current maximum at step n for each one of the parallel runs of the algorithm. Let* $n + m$ *be such that* $m \geq 1$ *is the smallest integer, such that at least one of the current maxima changes. Let* $M_{i_j}^{n+m}$ *for* $j = 1, \ldots, s \leq r$ *be the changed current maxima at step* $n + m$*. Applying the stopping criteria to* $M^{n+m} = \max_{j=1,\ldots,s} M_{i_j}^{n+m}$ *will accelerate the algorithm stopping.*

## 3. Some Parallel Computations with the Pure Random Search Algorithm

This section is devoted to present a computational study aiming at exhibiting the effects of parallel computing with the pure random search algorithm. Intuitively, asynchronous parallelisation of an algorithm with the parallel runs running independently is tantamount to performing the evaluations of the algorithm in several different machines, not necessarily simultaneously; with this perspective, it seems equivalent to perform four thousand search steps in a single machine or one thousand search steps in each one of four similar distinct machines and then collect the results. It may be observed that, in order to consider the possible performance improvements with the parallelisation of an algorithm, we have to account for the execution times. Moreover, there is evidence of an upper limit for the gain obtained by the parallelisation of an algorithm, namely Amdhal's

Law (see ([43], p. 65)) or the latter refined Gustafson's Law (see [44]). In Remark 5, we suggested a procedure to take into account the execution time of the best performing copy of the parallelised algorithm in order to stop the algorithm. In the following, we will not stop the algorithm whenever there is a copy of the algorithm satisfying the stopping criteria. Instead, we will stop the parallelised algorithm only when all the copies satisfy the stopping criteria and take the result of the copy—both in the variables time of execution and value attained—having the value attained when stopped that is closest to the maximum. This approach also allows us to study the improvement in the time variable in a simple way.

From Formula (14), in Theorem 3, we have a theoretical upper bound on a tail probability of the parallelised algorithm and the rate of convergence of the nonparallelised algorithm that depends on $r$, the number of independent copies of the algorithm in the parallelisation. In the study presented in the following, we intend to illustrate a determined quantitative improvement with the parallelisation of the pure random search algorithm.

The proposed method for the study is described in the following steps.

**1.** For each of the functions of two real variables studied (see Appendix A for additional information on these functions)—Ackley's, drop-wave, Keane's, Goldstein-Price's, Himmelblau's, and a *nearly unbounded* function—we determined, by Monte Carlo simulation, two lists of pairs of numbers, **UU** and **VV**. The list **VV** has 400 pairs of numbers, the first number of the pair being the time in seconds that the nonparallelised PRS algorithm overcomes 95% of the known maximum of the function, at which point the simulation stopped, and the second number being the value attained at the moment the algorithm stopped. The list **UU** is similar to the list **VV**, but it contains the pair corresponding to the best performing kernel—among the four kernels of the machine used in the parallelised algorithm—regarding the value attained. In Appendix B, we detail the machine characteristics as well as the main code used in the computations both of the parallelised and the simple algorithms.

**2.** We fitted a continuous bivariate distribution to both samples **UU** and **VV**, truncated to exact intervals of both time, the first marginal variate, and value, the second marginal variate. Then, using the probability density functions (PDF) of the marginals, denoted $f^{\mathbf{UU}}_{\text{time}}$ and $f^{\mathbf{VV}}_{\text{time}}$ for the first marginal and $f^{\mathbf{UU}}_{\text{value}}$ and $f^{\mathbf{VV}}_{\text{value}}$ for the second marginal of each of the two samples **UU** and **VV**, we determined two values of reference one $t_{\text{ref}}$ the *time reference* and the other $v_{\text{ref}}$ the *value reference*, such that:

$$f^{\mathbf{UU}}_{\text{value}}(v_{\text{ref}}) = f^{\mathbf{VV}}_{\text{value}}(v_{\text{ref}}) \text{ and } f^{\mathbf{UU}}_{\text{time}}(t_{\text{ref}}) = f^{\mathbf{VV}}_{\text{time}}(t_{\text{ref}}) \, . \tag{19}$$

We observe that, in some cases, $t_{\text{ref}}$ and $v_{\text{ref}}$ were not unique solutions of the equations $f^{\mathbf{UU}}_{\text{value}}(v) = f^{\mathbf{VV}}_{\text{value}}(v)$ in the variable $v$ and $f^{\mathbf{UU}}_{\text{time}}(t) = f^{\mathbf{VV}}_{\text{time}}(t)$ in the variable $t$; whenever this situation occurred, we chose the solutions closest to the medians of the two distributions. Furthermore, in at least one of the cases, there was no solution of the equations with the PDF, and we used a solution of a similar equation but with the cumulative distribution functions.

**3.** With the determined values of $t_{\text{ref}}$ and $v_{\text{ref}}$, we can immediately compute the conditional probabilities given by $\mathbb{P}_{\parallel}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}]$ for the parallelised algorithm, $V$ being the value variable and $T$ the time marginal variable, and similarly, $\mathbb{P}_{|}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}]$ for the nonparallelised algorithm. By analogy with Formula (14) in Theorem 3 and due to the choice of $t_{\text{ref}}$ and $v_{\text{ref}}$, we would expect to have:

$$\mathbb{P}_{\parallel}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}] \approx \mathbb{P}_{|}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}]^{r} \, ,$$

where $r$ is the number of copies in the parallelised algorithm. As a consequence, with:

$$\widehat{r} := \frac{\log\Big(\mathbb{P}_{\parallel}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}]\Big)}{\log\Big(\mathbb{P}_{|}[V \geq v_{\text{ref}} | T \leq t_{\text{ref}}]\Big)}, \tag{20}$$

we expect to have $\widehat{r} \approx r$ in the examples given.

In Table 1, we present the results of the computations just described, with $r = 4$ kernels, using the machine and code both presented in Appendix B. A first observation is that the result for the nearly unbounded function is the best among all functions tested. This kind of function will be the object of the results presented in Section 4. A second observation is that the median time values come from samples of repetitions of the algorithm both in the parallelised and nonparallelised versions; in the parallelised version, there are four times more repetitions than in the nonparallelised version, and being so, it is possible for the median times obtained with parallel algorithms to be sometimes worse than with the nonparallelised version.

**Table 1.** Results comparing the parallelised and the nonparallelised algorithms.

| Function | $t_{\text{ref}}$ | Med. Time UU; VV | $v_{\text{ref}}$ | Med. Value UU; VV | $\widehat{r}$ |
|---|---|---|---|---|---|
| Ackley | 1.01974 | 2.315; 0.320748 | 22.1859 | 22.3949; 22.0619 | 3.2257 |
| Drop-wave | 1.09633 | 0.791884; 0.67545 | 0.981912 | 0.990949; 0.974948 | 3.65206 |
| Keane | 0.0121573 | 0.0114686; 0.00892817 | 0.659555 | 0.66714; 0.656837 | 3.35966 |
| Nearly unbounded | 2.97751 | 1.35259; 6.05625. | 9784.74 | 9901.14; 9746.28 | 3.89735 |
| Goldstein-Price | 3.52532 | 1.25178; 6.00467 | 6.87869 | 6.93984; 6.82673 | 3.34803 |
| Himmelblau | 0.25 | 0.0163107; 0.0135014. | 9.82452 | 9.91097; 9.73618. | 3.45038 |

**Remark 6** (Quantitative experimental validation of the advantage of a parallelisation of a pure random search)**.** *The results presented in Table 1 seem to substantiate the claim presented in Theorem 3, in Formula (14). In fact, it seems clear that, both for the examples of slightly modified well known test functions having several local maximums—Ackley's, drop-wave, and Keane's—and the functions attaining their maximums on a plateau-like surface—Goldstein–Price's and Himmelblau's—we obtain estimates of $r = 4$, given by $\widehat{r}$, that can be considered acceptable, taking into account that, in the parallelised algorithm, efficiency losses occur in the distribution of tasks among kernels.*

**Remark 7** (Simulated data representations for examples of two types of functions)**.** *The simulated data that allowed us to determine the values of Table 1 can be visualised to provide a better insight of their characteristics. In Figures 1 and 2, we compare the same data for functions that clearly differ in the maximum localisation characteristics. As already mentioned, the drop-wave function presents several local maxima, with some of these local maxima agglomerated together with the global maximum, and the Himmelblau's function has its maximum—with four distinct maximisers—in a surface plateau.*

*There is a noticeable similarity between the parallelised data of both functions, as well as between the nonparallelised data of both functions.*

*However, there is a noticeable difference, for each function, between the parallelised data and the nonparallelised data. The parallelised data for both functions show a concentration on the upper left corner—less accentuated for Ackley's function—that is, surrounding the best values in a way that can be described as: closest to the true maximum and smallest times—more noticeable for Himmelblau's function—while the nonparallelised data, for both functions, appear to be uniformly distributed in the interval* $[95\%\max, \max]$*, also being more scattered in the time variable.*

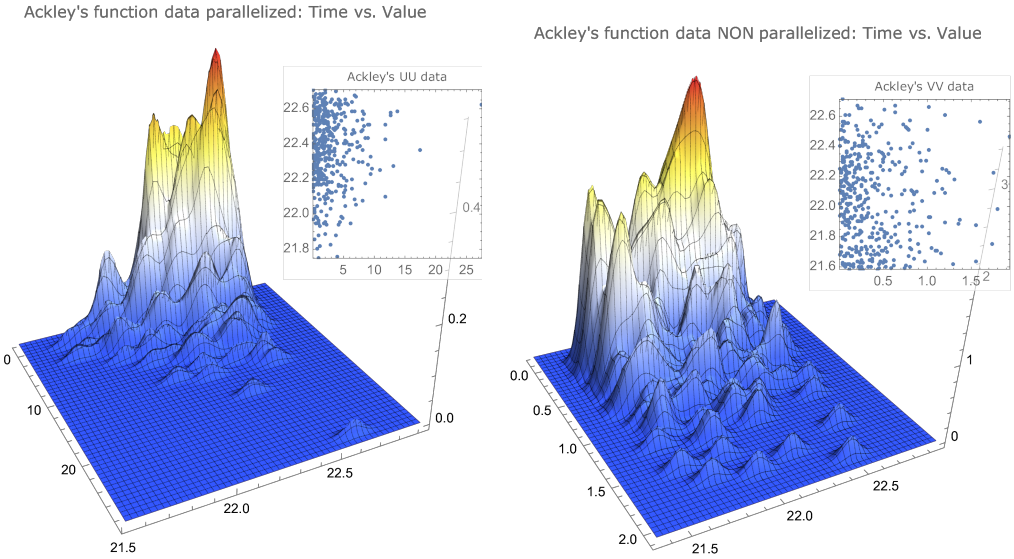


**Figure 1.** Smoothed histograms and simulation data for Ackley's function.

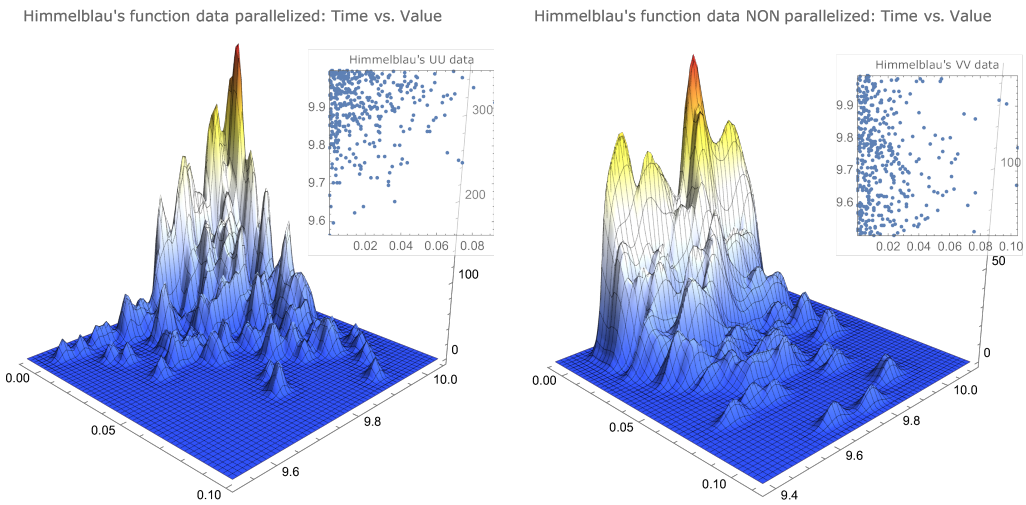


**Figure 2.** Smoothed histograms and simulation data for Himmelblau's function.

## 4. On a Martingale Diffusion Type Algorithm for Global Optimisation

In this section, we propose an algorithm for global optimisation based on the maximum principle for strictly elliptic operators. Let us take as an example a harmonic function on an open and bounded domain with regular boundary; this function belongs to the kernel of the Laplacian operator. It is well known that such a function attains its maximum on the boundary. Consider the diffusion naturally associated with the Laplacian. The algorithm we propose consists of running the diffusion on the domain until it touches the boundary; then, by a Feynman–Kăc-type result, we can obtain the maximum value of the function. This algorithm is suited for nearly unbounded functions. We next detail the general idea of our approach.

### *4.1. On the Maximum Principle for Strictly Elliptic Operators*

Consider a function $f : \overline{\mathcal{D}} \subset \mathbb{R}^2 \longrightarrow \mathbb{R}$, where $\mathcal{D}$ is a bounded open set with a regular boundary $\partial\mathcal{D}$. We consider the following two global optimisation goals for the function $f$ over the compact $\overline{\mathcal{D}}$.

**(a)** Determination of $\sup_{(x_1,x_2)\in\overline{\mathcal{D}}} f(x_1,x_2) =: M_{f,\overline{\mathcal{D}}}$.

**(b)** Determination of the set of maximisers $\left\{(x_1,x_2) \in \overline{\mathcal{D}} : f(x_1,x_2) =: M_{f,\overline{\mathcal{D}}}\right\}$.

We will consider the following regularity hypothesis of $f$ over $\overline{\mathcal{D}}$.

**(A1)** $f \in \mathcal{C}^2(\mathcal{D})$ ($f$ is twice differentiable on $\mathcal{D}$, and the second differential is continuous) and $f \in \mathcal{C}^0\left(\overline{\mathcal{D}}\right)$ ($f$ is continuous in $\overline{\mathcal{D}}$).

We consider $(\Omega, \mathcal{F}, \mathbb{P})$ a complete probability space and $\mathbb{B}^1 = (B_t^1)_{t\geq 0}$ and $\mathbb{B}^2 = (B_t^2)_{t\geq 0}$ two **independent** Brownian processes on $(\Omega, \mathcal{F}, \mathbb{P})$. Let $\mathbb{F} = (\mathcal{F}_t)_{t\geq 0}$ be the natural filtration generated by both $\mathbb{B}^1$ and $\mathbb{B}^2$, which we assume to be complete and right-continuous. The first ingredient in the elaboration of our algorithm is a two-dimensional diffusion process $(Y_t)_{t\geq 0} \equiv (Y_t^1, Y_t^2)_{t\geq 0}$, that is, a strong solution of the following system of stochastic differential equations:

$$\begin{cases} dY_t^1 = \mu_1(Y_t)dt + \sigma_1(Y_t)dB_t^1 \ , \ Y_0^1 \in \mathbb{R} \\ dY_t^2 = \mu_2(Y_t)dt + \sigma_2(Y_t)dB_t^2 \ , \ Y_0^2 \in \mathbb{R} \ . \end{cases} \tag{21}$$

We will suppose that the coefficients $\mu_1, \sigma_1$ and $\mu_2, \sigma_2$ have the necessary regularity to ensure the existence of a strong solution (see Remark 9 on validity of this hypothesis).

In order to guarantee some useful convergence properties, we assume the following hypothesis.

**(A2)** The process $(f(Y_t))_{t\geq 0}$ is a $\mathbb{F}$-**martingale**.

As a result of an application of Ito's formula (as in ([45], pp. 32–33)), we can identify a partial differential operator that will impose further constraints on the coefficients of the diffusions in Formula (21). In fact, we have:

$$\begin{aligned} df(Y_t) = &\left[\mu_1(Y_t)\frac{\partial f}{\partial x_1}(Y_t) + \mu_2(Y_t)\frac{\partial f}{\partial x_2}(Y_t) + \frac{1}{2}\left(\frac{\partial^2 f}{\partial x_1^2}(Y_t)\sigma_1^2(Y_t) + \frac{\partial^2 f}{\partial x_2^2}(Y_t)\sigma_2^2(Y_t)\right)\right]dt + \\ &+ \left(\frac{\partial f}{\partial x_1}(Y_t)\sigma_1(Y_t)dB_t^1 + \frac{\partial f}{\partial x_2}(Y_t)\sigma_2(Y_t)dB_t^2\right) , \end{aligned} \tag{22}$$

and in order for $(f(Y_t))_{t\geq 0}$ to be a martingale, the sum of the terms that constitute the $dt$ coefficient in Formula (22) has to vanish. Furthermore, if we consider the operator defined for $h : (x_1, x_2) \in \mathcal{D} \mapsto h(x_1, x_2) \in \mathbb{R}$, $h \in \mathcal{C}^2(\mathcal{D})$, by:

$$\mathcal{L}(h)(x_1, x_2) := \sum_{i=1}^{2} \mu_i(x_1, x_2)\frac{\partial h}{\partial x_i}(x_1, x_2) + \frac{1}{2}\sum_{i=1}^{2} \sigma_i^2(x_1, x_2)\frac{\partial^2 h}{\partial x_i^2}(x_1, x_2) \ , \tag{23}$$

we should have $\mathcal{L}(f) \equiv 0$ in $\mathcal{D}$; a condition denominated $f$ is $\mathcal{L}$ harmonic on $\mathcal{D}$ (see [46], p. 47).

Now, in order to ensure a Feynman–Kăc-type representation result, we assume the following hypothesis:

**(A3)** The operator $\mathcal{L}$ is **strictly elliptic** on $\mathcal{D}$, that is,

$$\forall \boldsymbol{x} = (x_1, x_2) \in \mathcal{D} \ \exists \Lambda(\boldsymbol{x}) > 0 \ \forall z_1, z_2 \in \mathbb{R} \ , \ \sum_{i,j=1}^{2} \sigma_i(\boldsymbol{x})\sigma_j(\boldsymbol{x})z_i z_j \geq \Lambda(\boldsymbol{x}) \sum_{i=1}^{2} z_i^2 \ . \tag{24}$$

The following result, which we quote from ([46], p. 46), gives a representation of the solutions of a well posed problem with operator $\mathcal{L}$ using the diffusion $(Y_t)_{t\geq 0}$.

**Theorem 4** (A Feynman–Kăc-type representation of the solution of a Dirichlet problem)**.** *Let $u$ be a solution of the **Dirichlet problem** for the operator $\mathcal{L}$ in Formula (23) associated with the function $f$ in $\mathcal{D}$, that is:*

- $u \in \mathcal{C}^2(\mathcal{D})$ *and* $u \in \mathcal{C}^0\left(\overline{\mathcal{D}}\right)$.
- $\mathcal{L}(u) \equiv 0$ *in* $\mathcal{D}$ *and* $u \equiv f$ *in* $\partial\mathcal{D}$.

*Then, with $\tau_{\mathcal{D}}$, the **exit time** of $(Y_t)_{t\geq 0}$ from $\mathcal{D}$, that is, $\tau_{\mathcal{D}} := \inf\{t > 0 : Y_t \notin \mathcal{D}\}$, we have the following representation:*

***(a)*** *$\tau_{\mathcal{D}} < +\infty$ almost surely (see also ([47], p. 324)).*

***(b)*** *For all $\boldsymbol{x} \in \mathcal{D}$, we have that $u(\boldsymbol{x}) = \mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})]$, with $\mathbb{E}^{\boldsymbol{x}}$ being the expectation with respect of the law of $(Y_t)_{t\geq 0}$ given as a solution of the diffusion system in Formula (21) with initial condition $\boldsymbol{x}$.*

As a consequence of the statement **(b)** of Theorem 4, we have the following maximum principle that is crucial for our algorithm proposal.

**Theorem 5** (A maximum principle application)**.** *Consider the notations and conditions of Theorem 4, and suppose that $f$ is $\mathcal{L}$ harmonic in $\mathcal{D}$, and moreover, that $f$ is a solution of the **Dirichlet problem** for the operator $\mathcal{L}$, associated with the function $f$ in $\mathcal{D}$, that is:*

- *$f \in \mathcal{C}^2(\mathcal{D})$ and $f \in \mathcal{C}^0\left(\overline{\mathcal{D}}\right)$.*
- *$\mathcal{L}(f) \equiv 0$ in $\mathcal{D}$, and obviously, $f \equiv f$ in $\partial\mathcal{D}$.*

*We then have, by the conclusion of Theorem 4:*

$$\forall \boldsymbol{x} \in \overline{\mathcal{D}} \;\; |f(\boldsymbol{x})| \leq \sup_{\boldsymbol{y}\in\partial\mathcal{D}} |f(\boldsymbol{y})| \, .$$

**Proof.** Since, by the representation formula, we have for all $\boldsymbol{x} \in \mathcal{D}$ that $f(\boldsymbol{x}) = \mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})]$, we can conclude that:

$$|f(\boldsymbol{x})| = |\mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})]| \leq \mathbb{E}^{\boldsymbol{x}}[|f(Y_{\tau_{\mathcal{D}}})|] \leq \mathbb{E}^{\boldsymbol{x}}\left[\sup_{\boldsymbol{y}\in\partial\mathcal{D}} |f(\boldsymbol{y})|\right] = \sup_{\boldsymbol{y}\in\partial\mathcal{D}} |f(\boldsymbol{y})| \, ,$$

and the conclusion now follows the hypothesis $f \in \mathcal{C}^0\left(\overline{\mathcal{D}}\right)$. □

**Remark 8.** *Let us summarise the main ideas we are pursuing. We link the function $f$ to a two-dimensional diffusion by means of a strictly elliptic differential operator of second order having as coefficients the diffusion coefficients, for which $f$ is a solution of a Dirichlet problem. The representation Theorem 4 shows that, if we start the diffusion anywhere on the domain of $f$, and then we wait until the diffusion hits the boundary, we will have that the value of the function $f$ at this point is a candidate to the maximum of the function in the closure of its definition domain.*

### *4.2. The Practical Implementation of the Algorithm*

Following the main idea exposed in Remark 8, we need to determine the coefficients of the diffusions in such a way that we have $\mathcal{L}(f) \equiv 0$. For that purpose, we do the following.

1. We assume that, as $f$ grows, the volatilities of the diffusions should decrease, and so we consider, for instance,

$$\sigma_i = e^{-\alpha_i |f|^{\beta_i}} \;, \; i = 1, 2 \; ; \alpha_i \; , \; \beta_i > 0 \; , \tag{25}$$

   with $\alpha_i$ and $\beta_i$ to be chosen appropriately depending on $f$, its domain of definition $\mathcal{D}$, and possibly also in an adaptive way in the algorithm.
2. We consider two functions $g_i : (x_1, x_2) \in \mathcal{D} \mapsto g_i(x_1, x_2) \in \mathbb{R}$, $g_i \in \mathcal{C}^1(\mathcal{D})$, for $i = 1, 2$, such that:

$$\frac{\partial f}{\partial x_i} = g_i f \; . \tag{26}$$

   We observe that, by Schwarz's theorem, we have that $\partial g_1/\partial x_2 = \partial g_2/\partial x_1$.
3. In order to determine the drift coefficients of the diffusions, we may observe that, for $f \neq 0$, we have that $\mathcal{L}(f) = 0$ if and only if:

$$\mu_1 g_1 + \mu_2 g_2 + \left(\frac{\partial g_1}{\partial x_1} + g_1^2\right)\frac{\sigma_1^2}{2} + \left(\frac{\partial g_2}{\partial x_2} + g_2^2\right)\frac{\sigma_2^2}{2} = 0 \, ,$$

as a result of Formula (23) and of the previous determinations. This equality can be achieved by taking, for instance, for $i = 1, 2$,

$$\mu_i = -\left(\frac{\partial g_i}{\partial x_i} + g_i^2\right)\frac{\sigma_i^2}{2g_i}\,, \tag{27}$$

an expression that is determined as soon as the functions $g_i$ are determined.

4. We can consider the equally spaced Euler–Maruyama discretisation (see ([48], p. 305) or ([49], p. 152)) of the diffusions in Formula (21), that is, for $i = 1, 2$,

$$\begin{cases} Y^i_{n+1} = Y^i_n + \mu_i((Y^1_n, Y^2_n))\Delta t + \sigma_i((Y^1_n, Y^2_n))\sqrt{\Delta t}Z^i_n \quad n \geq 0 \\ Y^i_0 = x_i\,, \end{cases} \tag{28}$$

$\Delta t$ being the discretisation step, with $Y^i_n \equiv Y^i_{\Delta t n}$ and the sequence $(Z^i_n)_{n\geq 0}$ being a sequence of independent and identically distributed random variables with a standard Gaussian random variable, that is, $Z \frown \mathcal{N}(0, 1)$.

The effectiveness of the practical implementation of the algorithm is conditioned to the existence of strong solutions of the system of SDE given in Formula (21), having the drifts given by Formula (27) and volatilities given by Formula (25). A first observation is that, as a consequence of Formula (26) and of Formula (27), we have, for $i = 1, 2$, that:

$$\mu_i = -\frac{\frac{\partial^2 f}{\partial x_i^2}}{\frac{\partial f}{\partial x_i}}\cdot\frac{\sigma_i^2}{2}\,. \tag{29}$$

Thus, if $\frac{\partial f}{\partial x_i}$ is bounded away from zero, that is, if $\frac{\partial f}{\partial x_i} \geq a > 0$, and if $\frac{\partial f}{\partial x_i} \in \mathcal{C}^0(\overline{\mathcal{D}})$, we have that $\mu_i$ is a continuous function over the compact $\overline{\mathcal{D}}$ and so is both uniformly continuous and bounded in $\overline{\mathcal{D}}$. A second observation is that, with the choice $\sigma_1 = \sigma_2 > \epsilon > 0$ in Formula (25) we have that, for

$$\sum_{i,j=1}^{2} z_i z_j \sigma_i \sigma_j = (z_1\sigma_1 + z_2\sigma_2)^2 = \sigma_1^2(z_1 + z_2)^2 \geq \sigma_1^2|z_1 + z_2|^2 \geq \sigma_1^2||z_1| - |z_2||^2 \geq$$

$$\geq \sigma_1^2 \max(|z_1|^2, |z_2|^2) \geq \frac{\sigma_1^2}{2}(z_1^2 + z_2^2) \geq \frac{\epsilon^2}{2}(z_1^2 + z_2^2)\,,$$

and so the operator defined in Formula (23) is strictly elliptic—even uniformly strictly elliptic—according to Formula (24). We now need an existence result for the system of stochastic differential equations given in Formula (21) having the drifts given by Formula (27) and volatilities given by Formula (25). For that purpose, we quote the following version of the celebrated Yamada–Watanabe theorem (see ([50], pp. 40–41)).

**Theorem 6** (Yamada and Watanabe 1971)**.** *Suppose that $\mu$ and $\sigma$ are progressively measurable and bounded and satisfy the following hypothesis.*

***(1)*** *There exists $\rho_1 : [0, +\infty[ \mapsto [0, +\infty[$, an **increasing** continuous function, such that $\rho_1(0) = 0$, such that:*

$$\forall t, x, y \;\; \|\sigma(t, x) - \sigma(t, y)\| \leq \rho_1(|x - y|) \;\; \textit{and} \;\; \lim_{\epsilon\to 0}\int_\epsilon \frac{du}{\rho_1(u)} = +\infty\,.$$

***(2)*** *There exists $\rho_2 : [0, +\infty[ \mapsto [0, +\infty[$, an **increasing concave** function, such that $\rho_2(0) = 0$, such that:*

$$\forall t, x, y \;\; (\mu(t, x) - \mu(t, y)) \leq \rho_2(|x - y|) \;\; \textit{and} \;\; \lim_{\epsilon\to 0}\int_\epsilon \frac{du}{\rho_2(u)} = +\infty\,.$$

*Then, for any random variable $Z$, where $t_0 \in [0,T]$, the following stochastic differential equation*

$$\begin{cases} dX_t = \mu(t,X_t)dt + \sigma(t,X_t)dB_t\,, \quad t \in [t_0,T]\,, \\ X_{t_0} = Z\,, \end{cases} \tag{30}$$

*has a strong solution and uniqueness in law holds.*

**Remark 9** (On the existence and unicity of the process $(Y_t)_{t\geq 0}$ defining the diffusion algorithm)**.** *Despite the fact that, in general, we do not have the sub-linear growth of the coefficients that, together with the Lipschitz character of these coefficients, ensures existence and unicity of a strong solution in the context of the classical Ito's theorem, we may nevertheless conclude that the system of stochastic differential equations given in Formula (21), having the drifts given by Formula (27) and volatilities given by Formula (25), has a strong solution that is unique in law; this happens as long as we force the solution process to stay on a compact set contained in $\mathcal{D}$. In fact, by the regularity properties imposed on $f$, we have that $\sigma_i \in \mathcal{C}^2(\mathcal{D})$ and $\sigma_i \in \mathcal{C}^0(\overline{\mathcal{D}})$; thus, by the mean value theorem, on any compact set contained in $\mathcal{D}$, we have that $\sigma_i$ has a bounded derivative, and so it verifies a Lipschitz condition, which implies that it satisfies hypothesis* ***(1)*** *of Theorem 6. Moreover, as pointed out above, the regularity properties imposed on $f$ also imply that the term not containing $\sigma_i$ in Formula (27) is bounded, and so the drifts are also Lipschitz on any compact set contained in $\mathcal{D}$ by virtue of the same property that the term $\sigma_i$—in Formula (27)—has. The practical implementation of the algorithm in Section 4.3 considers that the diffusion process is restrained to a compact set contained in the maximal open set where the function $f$ is defined.*

**Remark 10** (On the convergence of the Euler–Maruyama discretisation)**.** *The equally spaced Euler–Maruyama discretisation that we propose in Formula (28) converges weakly under a very general hypothesis, as stated in Proposition 4.58 in ([49], p. 146). This is a consequence of the regularity of the coefficients—$\mu_i$ is continuous, and $\sigma_i$ has the same regularity as $f$—and the fact that the strong solution of a SDE given by an Ito integral, with continuous coefficients, is a continuous function of the Brownian process. Nevertheless, the question of the strong convergence and of the rate of convergence of the Euler–Maruyama discretisation should be addressed for each particular case of the function $f$ studied.*

**Remark 11** (The martingale diffusion type algorithm)**.** *A simplified scheme describing the martingale diffusion-type algorithm is the following. The actual implementations in Mathematica ™(see [51]) for both the parallelised and nonparallelised versions are presented in Listing A4 and Listing A3, respectively.*

**STEP 1:**

**CHOOSE** Tolerance error $\epsilon > 0$ and upper bound $A$.

**STEP 2:**

**CHOOSE** $(x_1, x_2) \in \overline{\mathcal{D}}$ at random uniformly ;
**DO:** $Y_0^i = x_i$, for $i = 1, 2$ and $M_1 = f(x_1, x_2)$

**STEP 3:**

**DO:** compute iteratively $(Y_n^1, Y_n^2)$ for $n \geq 1$ according to Formula (28);
**IF** $f(Y_n^1, Y_n^2) > M_1$ **DO:** $M_1 = f(Y_n^1, Y_n^2) > M_1$
**UNTIL** either one of the following stopping criteria is verified:

stop at $n = n_0$ such that: $(Y_{n_0}^1, Y_{n_0}^2) \notin \mathcal{D}$ or,
stop at $n$ such that: $\left|Y_n^1 - Y_{n-1}^1\right| + \left|Y_n^2 - Y_{n-1}^2\right| < \epsilon$ and
either $|f(Y_n) - f(Y_{n-1})| < \epsilon$ or $|f(Y_n)| > A$;

**STOP** or **GO TO STEP 2** with new $M_2$ that must be compared with $M_1$.

*We observe that* ***STEP 2*** *in the algorithm is of the pure random search kind. Being so, if the algorithm is parallelisable, the results of Section 2.2 can be applied. In the parallelised version, there*

*will be an extra step added to the above pseudo-code for collecting the times and values of each of the four copies of the algorithm.*

We now prove that the algorithm we just described in the simplified scheme above is parallelizable according to Definition 4.

**Theorem 7** (Parallelisation of martingale diffusion-type algorithm)**.** *Let the process* $(Y_t)_{t\geq 0}$ *be given by the set of two diffusions in Formula* (21)*. Suppose that the process* $(f(Y_t))_{t\geq 0}$ *is such that Hypothesis* ***(A1)****,* ***(A2)****, and* ***(A3)*** *hold. Then, the process* $(f(Y_t))_{t\geq 0}$ *is a uniformly integrable martingale, and so it converges, almost surely and in* $L^1$*, to a random variable* $\mathbb{Y}_{\infty}^{f}$*. Moreover, if* $f \geq 0$*, we have that:*

- ***(a)*** *the well defined random variable* $f(Y_{\tau_{\mathcal{D}}})$ *is a weak solution of the global optimisation problem for* $f$ *in* $\overline{\mathcal{D}}$*;*
- ***(b)*** *the algorithm described in Remark 11 is parallelizable.*

**Proof.** It is clear that, as $f \in \mathcal{C}^0(\overline{\mathcal{D}})$ that for every $p > 1$, we have that:

$$\sup_{t\geq 0} \mathbb{E}\big[|f(Y_t)|^p\big] < +\infty \, ,$$

and so the process $(f(Y_t))_{t\geq 0}$ is uniformly integrable (see, for instance, ([41], p. 29)); moreover, this process being a martingale in continuous time it converges almost surely and in $L^1$ to a random variable that closes the martingale (see, for instance, ([41], p. 88)). From statement **(b)** in Theorem 4, it follows that the random variable $f(Y_{\tau_{\mathcal{D}}})$ is well defined. Now, without loss of generality, we may suppose that $f \geq 0$. Now, again by Theorem 5 and its proof, we have that:

$$\forall x \in \overline{\mathcal{D}} \;\; f(x) \leq \sup_{y\in\partial\mathcal{D}} f(y) = \mathbb{E}^x[f(Y_{\tau_{\mathcal{D}}})] \, .$$

Since $\mathcal{D}$ is bounded, we recall Definition 2 of the essential supremum of $f$ over the compact set $\overline{\mathcal{D}}$,

$$\alpha_f = \inf\big\{t : f(x) \leq t \, , \; \lambda \text{ almost everywhere on } \overline{\mathcal{D}}\big\} \, ,$$

from which is clear that

$$0 \leq \alpha_f \leq = \sup_{y\in\overline{\mathcal{D}}} f(y) = \sup_{y\in\partial\mathcal{D}} f(y) = \mathbb{E}^x[f(Y_{\tau_{\mathcal{D}}})] \, , \tag{31}$$

and so Assertion **(a)** of the theorem is proved. From Formula (28), it is clear that, for $i = 1, 2$, the law of $Y_{n+1}^i$ depends only on the law of the random variable $(Z_0^i, Z_1^i, \ldots, Z_n^i)$, and so the same happens with the law of the integer valued hitting time of $\partial\mathcal{D}$, that is, $\tau_{\mathcal{D}}^{\Delta} := \inf\big\{n > 0 : (Y_n^1, Y_n^2) \notin \mathcal{D}\big\}$; we observe that, almost surely, we have:

$$\lim_{\Delta t \longrightarrow 0} \tau_{\mathcal{D}}^{\Delta} = \tau_{\mathcal{D}} \, . \tag{32}$$

In fact, since we have that $\tau_{\mathcal{D}} = \inf\{t > 0 : Y_t \in \partial\mathcal{D}\}$, we may state:

- $\forall t < \tau_{\mathcal{D}} \, , \; Y_t \in \mathcal{D}$;
- and for almost all $\omega \in \Omega$, we have that:

$$\forall \Delta t > 0 \, , \; \exists n_{\Delta} = n_{\Delta}(\omega) \, , \; \tau_{\mathcal{D}}(\omega) \in [n_{\Delta}\Delta t, (n_{\Delta} + 1)\Delta t[ \, . \tag{33}$$

Since $\tau_{\mathcal{D}}^{\Delta} = \inf\{m > 1 : Y_{m\Delta t} \notin \mathcal{D}\}$, we now have for the hitting time of $\partial\mathcal{D}$ by the discretised process that $\tau_{\mathcal{D}}^{\Delta} \leq (n_{\Delta} + 1)\Delta t$ by force of Formula (33); moreover, the following dichotomy applies:

$$\begin{cases} \text{if } \tau_{\mathcal{D}}(\omega) = n_{\Delta}\Delta t\,, & \text{then } \tau_{\mathcal{D}}^{\Delta}(\omega) = \tau_{\mathcal{D}}(\omega)\,, \\ \text{if } \tau_{\mathcal{D}}(\omega) > n_{\Delta}\Delta t & \text{then } \tau_{\mathcal{D}}^{\Delta}(\omega) \geq n_{\Delta}\Delta t\,, \end{cases}$$

and in any case, we have $\left|\tau_{\mathcal{D}}(\omega) - \tau_{\mathcal{D}}^{\Delta}(\omega)\right| < \Delta t$, as claimed. Now, if we redefine the discretised algorithm $(\widetilde{Y}_n)_{n\geq 0}$ by:

$$\widetilde{Y}_n = \begin{cases} Y_n & \text{for } n < \tau_{\mathcal{D}}^{\Delta} \\ Y_{\tau_{\mathcal{D}}^{\Delta}} & \text{for } n \geq \tau_{\mathcal{D}}^{\Delta}\,, \end{cases}$$

we will have

$$\lim_{n\to+\infty} d_{KF}(f(\widetilde{Y}_n), f(Y_{\tau_{\mathcal{D}}^{\Delta}})) = 0\,. \tag{34}$$

Moreover, we observe that:

$$\begin{aligned} \left|\left|\mathbb{E}^{\boldsymbol{x}}\left[f(Y_{\tau_{\mathcal{D}}^{\Delta}})\right]\right| - |\mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})]|\right| &\leq \mathbb{E}^{\boldsymbol{x}}\left[\left|f(Y_{\tau_{\mathcal{D}}^{\Delta}})) - f(Y_{\tau_{\mathcal{D}}})\right|\right] \leq \\ &\leq \mathbb{E}^{\boldsymbol{x}}\left[\left|f(Y_{\tau_{\mathcal{D}}^{\Delta}})) - f(\widetilde{Y}_n)\right|\right] + \mathbb{E}^{\boldsymbol{x}}\left[\left|f(\widetilde{Y}_n) - f(Y_{\tau_{\mathcal{D}}})\right|\right]\,. \end{aligned}$$

We have that $(f(Y_n))_{n\geq 0}$ is a uniformly integrable martingale, and so the stopped process $(f(\widetilde{Y}_n))_{n\geq 0}$ is a uniformly integrable martingale converging in probability to the random variable $f(Y_{\tau_{\mathcal{D}}^{\Delta}})$ by reason of Formula (34). As a consequence, we have that:

$$\lim_{\Delta t\longrightarrow 0}\left(\lim_{n\longrightarrow+\infty} \mathbb{E}^{\boldsymbol{x}}\left[\left|f(Y_{\tau_{\mathcal{D}}^{\Delta}})) - f(\widetilde{Y}_n)\right|\right]\right) = 0\,.$$

Furthermore, using Formula (32), we then have that:

$$\lim_{\Delta t\longrightarrow 0}\left(\lim_{n\longrightarrow+\infty} \mathbb{E}^{\boldsymbol{x}}\left[\left|f(\widetilde{Y}_n) - f(Y_{\tau_{\mathcal{D}}})\right|\right]\right) = \lim_{\Delta t\longrightarrow 0} \mathbb{E}^{\boldsymbol{x}}\left[\left|f(Y_{\tau_{\mathcal{D}}^{\Delta}}) - f(Y_{\tau_{\mathcal{D}}})\right|\right] = 0\,,$$

and, as a consequence, we may state that

$$\lim_{\Delta t\longrightarrow 0} \mathbb{E}^{\boldsymbol{x}}\left[f(Y_{\tau_{\mathcal{D}}^{\Delta}})\right] = \mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})]\,,$$

Finally, as a consequence of Formula (31), we have, for sufficiently small $\Delta t$, that

$$0 \leq f(x) = \mathbb{E}^{\boldsymbol{x}}[f(Y_{\tau_{\mathcal{D}}})] \approx \mathbb{E}^{\boldsymbol{x}}\left[f(Y_{\tau_{\mathcal{D}}^{\Delta}})\right]\,,$$

and so the algorithm is parallelisable. □

*4.3. An Example of an Application of the Diffusion Optimisation Algorithm with Parallelisation*

In order to consider an example of a function to apply the diffusion algorithm, we start with

$$(x, y) \in ]-1, 1[\times]-1, 1[ \mapsto \frac{2 + \cos^2(7x + 11y)}{\sqrt{x^2 + y^2}} \in \{+\infty\} \cup \mathbb{R}^+ \setminus \{0\}\,,$$

a function that is unbounded at the point $(0,0)$. The function $f$ that we analyse may be considered an example of the type of *nearly unbounded* functions, such as the one studied in Sections 4.1 and 4.2 if we consider it to be defined in the open set given by:

$$\mathcal{D} := ]-1,1[^2 \setminus \left\{(x,y) : x^2+y^2 \leq 2\times 10^{-8}\right\},$$

represented in Figure 3 for $(x,y) \in \mathcal{D}$, such that $|f(x,y)| \leq 18$ for a better visualisation. It is clearly a function that will attain its maximum at a point of the boundary of $\mathcal{D}$, more precisely on the set $\left\{(x,y) : x^2+y^2 = 2\times 10^{-8}\right\}$. We observe that the point $(0.0001, 0.0001)$ belongs to this circle, and that $f(0.0001, 0.0001) = 21213.2$; this value will be used in the implementation of the algorithm. We observe also that **numerically**, that is, using a standard maximiser function of the software, we have:

$$\begin{aligned}\max_{(x,y)\in\overline{\mathcal{D}}} f(x,y) &= 21213.037869117255 = \\ &= f(0.00013974449759483935, -0.00002172050421622352)\,,\end{aligned}$$

with a unique maximiser in the set $\left\{(x,y) : x^2+y^2 = 2\times 10^{-8}\right\}$, that is, a point in the boundary of $\mathcal{D}$.

The computation of the coefficients according with the Formulas (25)–(27), allows the determination of the coefficients of the two diffusions. The function $g_1$, computed according to Formula (26), is given by:

$$g_1(x,y) = \frac{\sqrt{x^2+y^2}\left(-\frac{x\left(\cos^2(7x+11y)+2\right)}{\left(x^2+y^2\right)^{3/2}} - \frac{14\sin(7x+11y)\cos(7x+11y)}{\sqrt{x^2+y^2}}\right)}{\cos^2(7x+11y)+2}\,.$$

An inspection of Figure 4 and the Taylor series of order one in a neighbourhood of the point $(0,0)$

$$g_1(x,y) \approx \left(-\frac{154y}{3} + O\left(y^2\right)\right) + x\left(-\frac{1}{y^2} - \frac{98}{3} + O\left(y^2\right)\right) + O\left(x^2\right)$$

show that there is a singularity at the point $(0,0)$, and that the function $g_1$ is well defined in the domain $\mathcal{D}$. Similarly, the function $g_2$, given by:

$$g_2(x,y) = \frac{\sqrt{x^2+y^2}\left(-\frac{y\left(\cos^2(7x+11y)+2\right)}{\left(x^2+y^2\right)^{3/2}} - \frac{22\sin(7x+11y)\cos(7x+11y)}{\sqrt{x^2+y^2}}\right)}{\cos^2(7x+11y)+2}\,,$$

and the correspondent Taylor series of order one in a neighbourhood of the point $(0,0)$,

$$g_2(x,y) \approx \left(-\frac{1}{y} - \frac{242y}{3} + O\left(y^2\right)\right) + x\left(-\frac{154}{3} + O\left(y^2\right)\right) + O\left(x^2\right)\,,$$

also show that the function $g_2$ is well defined in the domain $\mathcal{D}$.

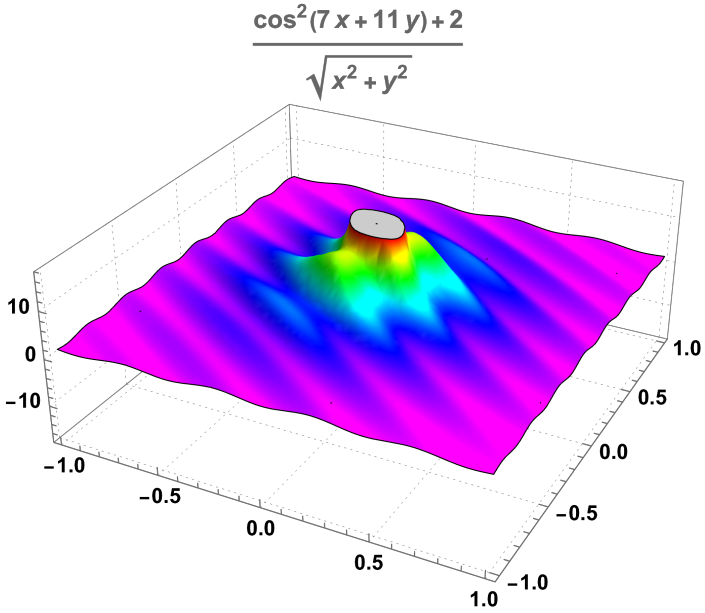


**Figure 3.** A nearly unbounded function having several local maxima.

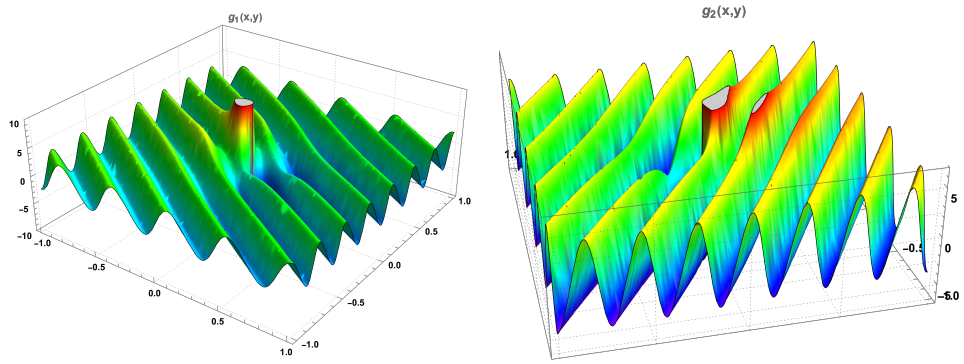


**Figure 4.** The functions $g_1$ and $g_2$.

We now detail some aspects of the coefficients $\mu_1$, $\mu_2$, and $\sigma_1 = \sigma_2$ used in the numerical computations. Using Formula (25) and Formula (27), we determined the coefficients $\widetilde{\mu}_1$ and $\widetilde{\mu}_2$, as presented in Formula (35) and in Formula (36), respectively.

$$\widetilde{\mu}_1(x,y) = \\ = -\frac{5(y^2-2x^2) - 28x(x^2+y^2)\sin(14x+22y) + \left(196x^4 + x^2(392y^2-2) + 196y^4 + y^2\right)\cos(14x+22y)}{2(x^2+y^2)(14(x^2+y^2)\sin(14x+22y) + x\cos(14x+22y) + 5x)} \times \\ \times e^{-0.8\sqrt[4]{\frac{\cos^2(7x+11y)+2}{\sqrt{x^2+y^2}}}}, \tag{35}$$

and

$$\widetilde{\mu}_2(x,y) = \\ = -\frac{5(x^2-2y^2) - 44y(x^2+y^2)\sin(14x+22y) + \left(484x^4 + x^2(968y^2+1) + 484y^4 - 2y^2\right)\cos(14x+22y)}{2(x^2+y^2)(22(x^2+y^2)\sin(14x+22y) + y\cos(14x+22y) + 5y)} \times \\ \times e^{-0.8\sqrt[4]{\frac{\cos^2(7x+11y)+2}{\sqrt{x^2+y^2}}}}. \tag{36}$$

An immediate inspection of Formula (35) and of Formula (36) shows that there is a possibility of large values that will render the Monte-Carlo simulation unstable. So, for computational purposes, we considered, instead of $\widetilde{\mu}_1$ and $\widetilde{\mu}_2$, the following coefficients with $v_\mu = 0.8$.

$$\mu_1(x,y) = \begin{cases} \widetilde{\mu}_1(x,y) & \text{if } |\widetilde{\mu}_1(x,y)| \leq v_\mu \\ v_\mu & \text{if } |\widetilde{\mu}_1(x,y)| > v_\mu \end{cases}, \quad \mu_2(x,y) = \begin{cases} \widetilde{\mu}_2(x,y) & \text{if } |\widetilde{\mu}_2(x,y)| \leq v_\mu \\ v_\mu & \text{if } |\widetilde{\mu}_2(x,y)| > v_\mu \end{cases}. \tag{37}$$

The coefficients of the first diffusion are shown in Figure 5 for $v_\mu = 5$. The value of $v_\mu$ to be taken when applying the method is linked both to the discretisation step and the number of steps of the diffusions, as well as to steepness of the function close to the border where the maximum is attained.

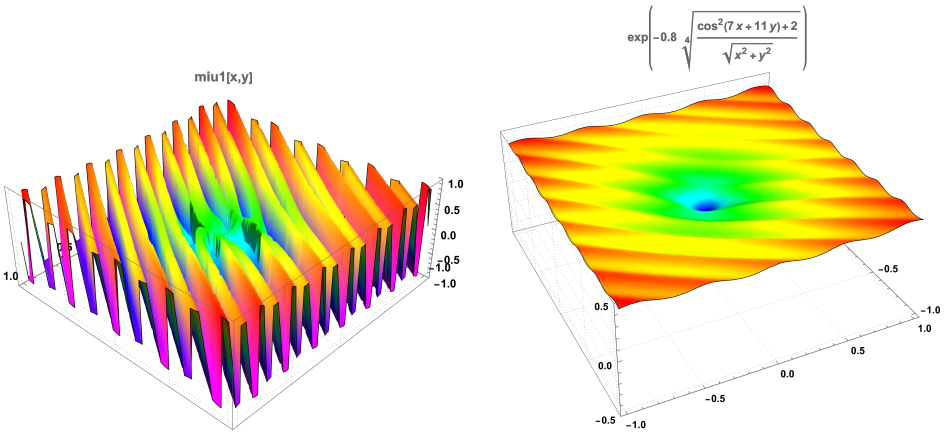


**Figure 5.** The coefficients of the first diffusion $\mu_1$ and $\sigma_1$.

We considered the algorithm prescribed to run on only one kernel, described in Listing A3, and the parallelised algorithm running in four kernels, described in Listing A4. In the study presented in Section 3, we compared, for the pure random search algorithm, the four kernel parallelised algorithm, with the algorithm being processed without the specification of parallelisation, that is, in its normal usual mode. We also performed a trial running the pure random search algorithm parallelised on one kernel only, as done for the diffusion algorithm, but there were no appreciable changes in the results, for instance, for Himmelblau's function, the estimate of $\widehat{r}$ changed from the already reported 3.45038 to the new value 3.60881 if the parallelisation was run with one only kernel specification. The results for the diffusion algorithm are presented in Table 2. We recall that the parallelisation performance indicator $\widehat{r}$ is defined in Formula (20) and that we compare $\widehat{r}$ to 4, the maximum number of kernels used in the parallelisation of the algorithm.

**Table 2.** Diffusion algorithm example: comparing the parallelised and the nonparallelised versions.

| Function | $t_{\text{ref}}$ | Med. Time: UU; VV | $v_{\text{ref}}$ | Med. Value: UU; VV | $\widehat{r}$ |
|---|---|---|---|---|---|
| $\frac{2+\cos^2(7x+11y)}{\sqrt{x^2+y^2}}$ | 45.8033 | 44.0009; 39.91 | 21246.4 | 21268.0; 21239.8 | 5.87919 |

The smoothed bivariate distribution of simulated data for the example function and the diffusion algorithm is presented in Figure 6.

**Remark 12** (Some conclusions on the application example of the diffusion algorithm)**.** *It is clear from the analysis of the median execution times for both the algorithm running in only one kernel and running in parallel with four kernels, reported in Table 2—respectively,* 39.91 *s*

*and* 44.0009 *s—that the implementation of the diffusion algorithm presented is not efficient when compared with the available implementations of global optimisation algorithms that can find the optimum in less than two seconds. Further studies are required to ascertain if there is some specific class of functions for which the proposed diffusion algorithm presents some noticeable advantages over other algorithms. The estimate* $\widehat{r} = 5.87919$ *in Table 2 clearly shows the advantage of running a parallelised algorithm, especially whenever the algorithm is computationally demanding, as is the diffusion algorithm. The possibility of a substancial gain of efficiency in highly time-consuming algorithms by the proposed parallelisation method of the algorithm requires further study.*

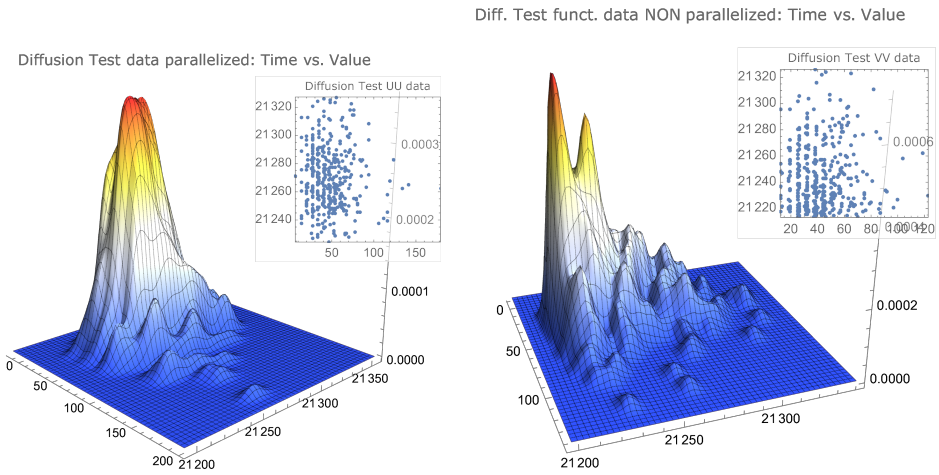


**Figure 6.** Smoothed distribution of parallelised and nonparallelised simulated data.

## 5. Conclusions

In this work, we developed and studied a novel formal description of parallelisation of random algorithms for global optimisation with no constraints and we presented a preliminary study for a novel stochastic differential equation based algorithm suited to functions attaining the extremum at some points of the border of the domain of definition, for instance, unbounded functions. There are many practical applications for which this algorithm may be well suited. For instance, it may be used for the study of the temperature in a mathematical model of the heath diffusion in the core of a nuclear reaction in case of a meltdown as in [52]. The full testing of an improved implementation of the algorithm proposed will be the object of future work, encompassing further testing against well-known and well-performing algorithms and against higher dimensional problems.

We introduced a performance indicator defined in Formula (20) motivated by two reasons: the first reason is the parallelisation method and the implementation of this method used. We chose to study the method of parallelisation consisting of sending the algorithm, simultaneously and independently, to the four kernels available in our computer and stopping each kernel as soon as the running maximum was in a previously prescribed neighbourhood of the known maximum of the function recording both the value and the time taken to attain this value. These recorded values were used to define a sample, from which the indicator of Formula (20) was computed. The second and most important reason is the goal of practically verifying the result in Formula (14) of Theorem 3. The study of the variations of performance, measured by the performance indicator of Formula (20), whenever there is an increase in the number of processors/cores used, will also be a subject of future work. The practical work presented confirms the usefulness of this performance indicator, and this conformation, in turn, confirms the advantages of parallelisation, even with a reduced number of processors.

**Author Contributions:** Conceptualization, M.L.E.; Formal analysis, M.L.E.; Investigation, M.L.E., N.P.K. and P.P.M.; Software, M.L.E. and N.M.; Visualization, M.L.E.; Writing–original draft, M.L.E.; Writing–review and editing, M.L.E., N.P.K., P.P.M. and N.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** For the second author, this work was undertaken with partial financial support of RFBR (Grant n. 19-01-00451). For the first and third author, this work was partially supported through the project of the Centro de Matemática e Aplicações, UID/MAT/00297/2020, financed by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology). The APC was by supported the New University of Lisbon through the PhD program in Statistics and Risk Management of the FCT Nova Faculty.

**Informed Consent Statement:** Not applicable.

**Acknowledgments:** This work was published with financial support from the New University of Lisbon. The authors express gratitude to the comments, corrections and questions of the referees that lead to a revised and better version of this work.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A. The Test Functions for the Pure Random Search Parallelisation Study

In this appendix, we provide information on the test functions studied in Section 3. There are very well-known references on test functions for unconstrained optimisation methods, where a large set of functions—displaying a wide spectrum of features creating obstacles for efficient global optimisation algorithms—can be found (see [53], or [54]). Since our goal is to provide an illustration of the quantitative approach proposed to establish differences with parallelised and nonparallelised random search, we opt to select a set of six functions with essentially three characteristic features: oscillating with several local maxima, a maximum in a plateau like surface, and nearly unbounded. Additional information on this functions is presented in Table A1.

**Table A1.** Some characteristics of the slightly modified test functions used.

| Function | Expression | Maximisers | 95%max |
|---|---|---|---|
| Ackley | $20e^{-0.141421\sqrt{x^2+y^2}} + e^{0.5(\cos(2\pi x)+\cos(2\pi y))}$ | $(0,0)$ | 21.5824 |
| drop-wave | $\frac{\cos\left(12\sqrt{x^2+y^2}\right)+1}{0.5(x^2+y^2)+2}$ | $(0,0)$ | 0.95 |
| Keane | $\frac{\sin^2(x-y)\sin^2(x+y)}{\sqrt{x^2+y^2}}$ | $(1.39324,0)\&(0,1.39324)$ | 0.639984 |
| nearly unbounded | $\frac{1}{x^2+y^2+0.0001}$ | $(0,0)$ | 9500 |
| Goldstein-Price | $f_{\text{G-P}}(x,y)$ | $(0,-1)$ | 6.65 |
| Himmelblau | $10-\left(x^2+y-11\right)^2-\left(x+y^2-7\right)^2$ | $(3,2)\&(-2.8,3.13)\&\ldots.$ | 9.5 |

**Remark A1** (Complements to Table 1)**.** *Himmelblau's function has the following two more maximisers than the ones that figure in Table 1* $(-3.77,-3,28)\&(3.58,-1.85)$*. The modified Goldstein–Price's function we used is given by the following expression.*

$$f_{\text{G-P}}(x,y) := 10-\left(\left(3x^2+6xy-14x+3y^2-14y+19\right)(x+y+1)^2+1\right)\times \\ \times\left(\left(12x^2-36xy-32x+27y^2+48y+18\right)(2x-3y)^2+30\right).$$

We stress that these functions are modified versions of the homonymous functions in the references mentioned above; the modifications are of two sorts: change of sign in order to have critical points as maximums and adding a constant in order to have an essentially

nonnegative function. The graphical representation of these functions is displayed in Figure A1.

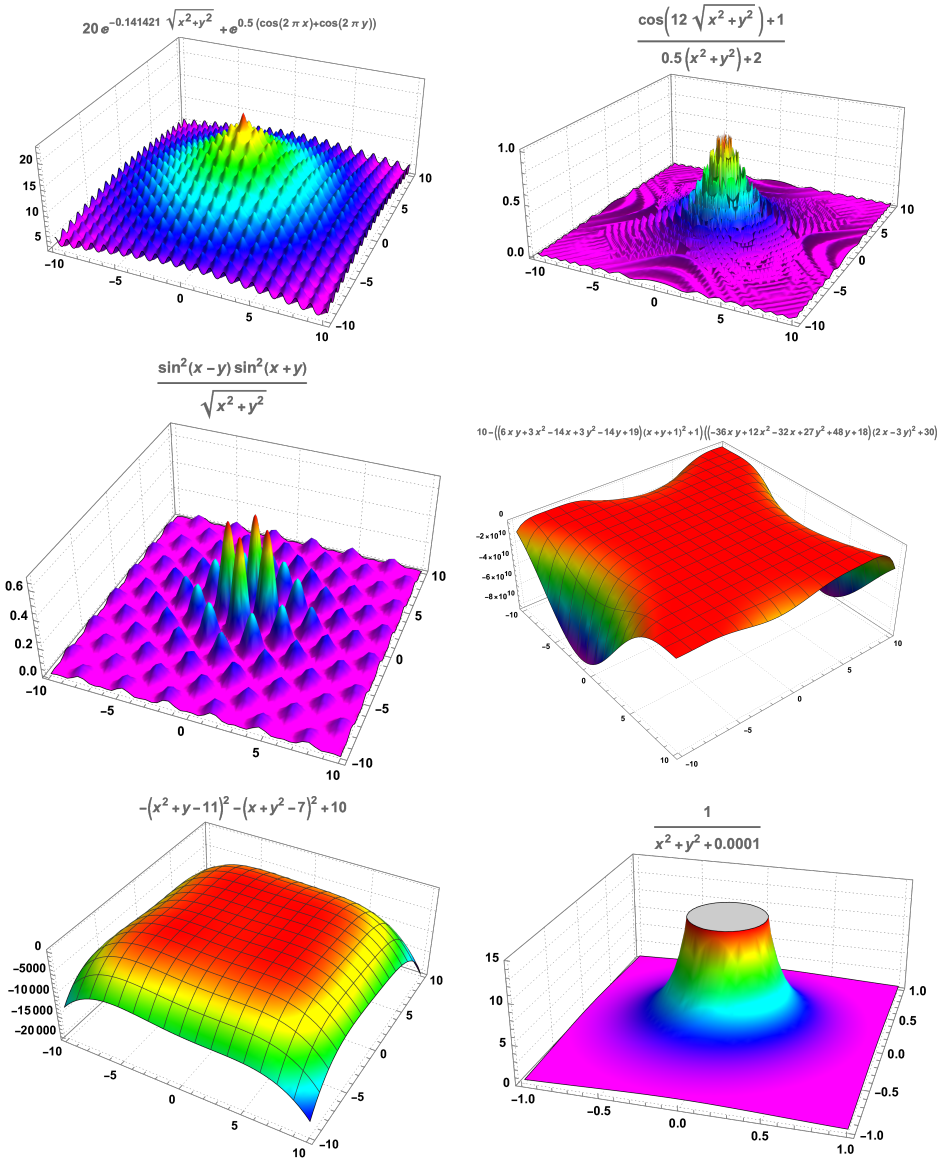

**Figure A1.** Test functions, from left to right and top to down: Ackley's, drop-wave, Keane's, Goldstein-Price's, Himmelblau's and *nearly unbounded*.

## Appendix B. Mathematica ™ Code for Parallel Computation

The details on the machine having four kernels, and on the corresponding operating system (OS), used to perform the parallel computations studies of Section 3, are detailed in Table A2.

**Table A2.** Machine, OS and some of the 4 kernel characteristics.

| KernelID | Machine | OS | Version | Processor | TimeUsed | Memory |
|---|---|---|---|---|---|---|
| 1 | Mac Mini M1 2020 8 Gb | MacOSX-x86-64 | 12.3 | 1289 | 0.809547 | 49401008 |
| 2 | Mac Mini M1 2020 8 Gb | MacOSX-x86-64 | 12.3 | 1290 | 0.809104 | 49401232 |
| 3 | Mac Mini M1 2020 8 Gb | MacOSX-x86-64 | 12.3 | 1291 | 0.773256 | 49401048 |
| 4 | Mac Mini M1 2020 8 Gb | MacOSX-x86-64 | 12.3 | 1292 | 0.828269 | 49403112 |

The determination, in a parallel computation with four kernels, of the table of values **UU** for the best performing kernel was performed using the code in Listing A1.

**Listing A1.** Code for UU table: timing and value of the best performing kernel.

```
UU = {};
For[i = 1, i ≤ 400, i++,
UU = Append[UU,
MaximalBy[
ParallelEvaluate[
AbsoluteTiming[
Function[{eM, Maximz, en},
Module[{eMe = eM, Maximzr = Maximz, ene = en},
While[True,
ene = ene + 1;
eMeT = eMe; MaximzrT = Maximzr;
RdmPt1=RandomVariate[UniformDistribution[{-10, 10}], 1][[1]];
RdmPt2 = RandomVariate[UniformDistribution[{-10, 10}], 1][[1]];
If[ f[RdmPt1, RdmPt2] > eMe, (eMe = f[RdmPt1, RdmPt2])
&&(Maximzr = {RdmPt1, RdmPt2})];
If[(eMe > ptmr), Break[]];]; eMe
]
][0, {RandomVariate[UniformDistribution[{-10, 10}], 1][[1]],1},
RandomVariate[UniformDistribution[{-10, 10}], 1][[1]]]
]  ], Last][[1]]
] ]; UU
```

The definition domain of all numerical valued functions being maximised was $[-10, 10] \times [-10, 10]$ (see lines 12, 13, 18 and 19 of the code in Listing A1).

The determination of the values **VV** of the nonparallelised computation was performed using the code in Listing A2.

**Listing A2.** Code for VV table: timing and value of the nonparallelised computation.

```
VV = {};
For[i = 1, i ≤ 400, i++,
VV = Append[VV, AbsoluteTiming[
Function[{eM, Maximz, en},
Module[{eMe = eM, Maximzr = Maximz, ene = en},
While[True,
ene = ene + 1;
eMeT = eMe; MaximzrT = Maximzr;
RdmPt1 = RandomVariate[UniformDistribution[{-10, 10}], 1][[1]];
RdmPt2 = RandomVariate[UniformDistribution[{-10, 10}], 1][[1]];
If[f[RdmPt1, RdmPt2] > eMe, (eMe = f[RdmPt1, RdmPt2])
&&(Maximzr = {RdmPt1, RdmPt2})];
If[(eMe > ptmr), Break[]];
]; eMe
]
][0, {RandomVariate[UniformDistribution[{-10, 10}], 1][[1]], 1},
RandomVariate[UniformDistribution[{-10, 10}], 1][[1]]]
]
]
]; VV
```

The code function *AbsoluteTiming* returns the absolute number of seconds in real time that have elapsed and measures only the time involved in actually evaluating the expression in its argument that is being determined not time involved in formatting the result.

The implementation of the diffusion algorithm with parallelisation restrained to only one kernel, which is tantamount to an algorithm with no parallelisation, is presented in Listing A3.

**Listing A3.** The diffusion algorithm with one only kernel.

```
Delt=0.0001;ene=100000;
VVV={};
For[i=1,i≤200,i++,
Clear[M,Minit,miu1,miu2];
miu1=Compile[{x,y},Clip[-termo1[x,y],{-0.8,0.8}]*sig1[x,y]^2];
miu2=Compile[{x,y},Clip[-termo2[x,y],{-0.8,0.8}]*sig2[x,y]^2];
VVV=Append[VVV,
ParallelEvaluate[
AbsoluteTiming[
Function[{Minit},
Module[{M=Minit},
Clear[Y1,Y2,Start1,Start2];
brpts1=RandomPoint[dom,1];
Start1=brpts1[[1,1]];Start2=brpts1[[1,2]];
Y1[0]=Start1;Y2[0]=Start2;
M=f[Y1[0],Y2[0]];  The initialization of the Maximum
eme=30;
For[r=1,r≤eme,r++,
brptFu[ene_]:=RandomPoint[dom,ene];
brpts=brptFu[ene];
Z1=Table[brpts[[k,1]],{k,1,Length[brpts]}];
Z2=Table[brpts[[k,2]],{k,1,Length[brpts]}];
Y1[0]=Start1;Y2[0]=Start2;
Y1[n_Integer]:=Y1[n]=Y1[n-1]+miu1[Y1[n-1],Y2[n-1]]*Delt+
sig1[Y1[n-1],Y2[n-1]]*Z1[[n]]*Delt^0.5;
Y2[n_Integer]:=Y2[n]=Y2[n-1]+miu2[Y1[n-1],Y2[n-1]]*Delt+
sig2[Y1[n-1],Y2[n-1]]*Z2[[n]]*Delt^0.5;
Table[{Y1[k],Y2[k]},{k,1,ene}];
k=1;
While[And[++k≤ene-1,-1<Y1[k]<1,-1<Y2[k]<1],
If[M>f[0.0001,0.0001],And[r=eme,Break[],Return[r]]];
If[f[Y1[k],Y2[k]]>M,And[M=f[Y1[k],Y2[k]],Start1=Y1[k],Start2=Y2[k]]]
];
Clear[Y1,Y2]
];M
]
][0]
],
First[Kernels[]]]
]
];
VVV
```

The parallelised diffusion algorithm with four kernels was implemented according to Listing A4. We recall that the computations were achieved with the four kernels running independently, by force of the algorithm, and the result given correspond to the output of the kernel for which the achieved value was higher.

**Listing A4.** The diffusion algorithm with result given by the best of 4 kernels.

```
Delt=0.0001;ene=100000;
UU={};
For[i=1,i≤200,i++,
Clear[M,Minit,miu1,miu2];
miu1=Compile[{x,y},Clip[-termo1[x,y],{-0.8,0.8}]*sig1[x,y]^2];
miu2=Compile[{x,y},Clip[-termo2[x,y],{-0.8,0.8}]*sig2[x,y]^2];
UU=Append[UU,MaximalBy[
ParallelEvaluate[
AbsoluteTiming[
Function[{Minit},
Module[{M=Minit},
Clear[Y1,Y2,Start1,Start2];
brpts1=RandomPoint[dom,1];
Start1=brpts1[[1,1]];Start2=brpts1[[1,2]];
Y1[0]=Start1;Y2[0]=Start2;
M=f[Y1[0],Y2[0]];  The initialization of the Maximum
eme=30;
For[r=1,r≤eme,r++,
brptFu[ene_]:=RandomPoint[dom,ene];
brpts=brptFu[ene];
Z1=Table[brpts[[k,1]],{k,1,Length[brpts]}];
Z2=Table[brpts[[k,2]],{k,1,Length[brpts]}];
Y1[0]=Start1;Y2[0]=Start2;
Y1[n_Integer]:=Y1[n]=Y1[n-1]+miu1[Y1[n-1],Y2[n-1]]*Delt+
sig1[Y1[n-1],Y2[n-1]]*Z1[[n]]*Delt^0.5;
Y2[n_Integer]:=Y2[n]=Y2[n-1]+miu2[Y1[n-1],Y2[n-1]]*Delt+
sig2[Y1[n-1],Y2[n-1]]*Z2[[n]]*Delt^0.5;
Table[{Y1[k],Y2[k]},{k,1,ene}];
k=1;
While[And[++k≤ene-1,-1<Y1[k]<1,-1<Y2[k]<1],
If[M>f[0.0001,0.0001],And[r=eme,Break[],Return[r]]];
If[f[Y1[k],Y2[k]]>M,And[M=f[Y1[k],Y2[k]],Start1=Y1[k],Start2=Y2[k]]]
];
Clear[Y1,Y2]
];M
]
][0]
]
],Last][[1]]
]
];UU
```

## References


1. Zabinsky, Z.B. *Stochastic Adaptive Search for Global Optimization*; *Nonconvex Optimization and its Applications*; Kluwer Academic Publishers: Boston, MA, USA, 2003; Volume 72, pp. xviii+224.
2. Spall, J.C. Stochastic optimization. In *Handbook of Computational Statistics*; Springer: Berlin, Germany, 2004; pp. 169–197.
3. Rubinstein, R.Y.; Kroese, D.P. *Simulation and the Monte Carlo Method*, 2nd ed.; Wiley Series in Probability and Statistics; Wiley-Interscience [John Wiley & Sons]: Hoboken, NJ, USA, 2008; pp. xviii+345.
4. Esquível, M.L. A conditional Gaussian martingale algorithm for global optimization. In *Computational Science and Its Applications—ICCSA 2006, PT 3*; Gavrilova, M., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Lagana, A., Mun, Y., Choo, H., Eds.; Lecture Notes in Computer Science; Springer: Berlin, Germany, 2006; Volume 3982, pp. 841–851.
5. Stephens, C.P.; Baritompa, W. Global optimization requires global information. *J. Optim. Theory Appl.* **1998**, *96*, 575–588. [CrossRef]
6. Esquível, M.L.; Machado, N.; Krasii, N.P.; Mota, P.P. On the Information Content of some Stochastic Algorithms. In *Recent Developments in Stochastic Methods and Applications*; Albert, N.S., Konstantin, E.S., Dmitry, V.K., Eds.; Springer Proceedings in Mathematics & Statistics; Springer International Publishing: Berlin, Germany; Cham, Switzerland, 2021; pp. 57–75.

7. Spall, J.C. *Introduction to Stochastic Search and Optimization*; Wiley-Interscience Series in Discrete Mathematics and Optimization; Estimation, Simulation, and Control; Wiley-Interscience [John Wiley & Sons]: Hoboken, NJ, USA, 2003; pp. xx+595. [CrossRef]
8. Duflo, M. *Algorithmes Stochastiques*; Mathématiques & Applications (Berlin) [Mathematics & Applications]; Springer: Berlin, Germany, 1996; Volume 23, pp. xiv+319.
9. Duflo, M. *Random Iterative Models*; *Applications of Mathematics (New York)*; Stephen, S.W., Translator; Springer: Berlin, Germany, 1997; Volume 34, pp. xviii+385. [CrossRef]
10. Bartholomew-Biggs, M.C.; Parkhurst, S.C.; Wilson, S.P. Global Optimization—Stochastic or Deterministic? In *Stochastic Algorithms: Foundations and Applications*; Albrecht, A., Steinhöfel, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2003; pp. 125–137.
11. Zhigljavsky, A.; Žilinskas, A. *Stochastic Global Optimization*; Springer Optimization and Its Applications; Springer: New York, NY, USA, 2008; Volume 9, pp. x+262.
12. Asi, H.; Duchi, J.C. The importance of better models in stochastic optimization. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 22924–22930. [CrossRef] [PubMed]
13. Plevris, V.; Bakas, N.P.; Solorzano, G. Pure Random Orthogonal Search (PROS): A Plain and Elegant Parameterless Algorithm for Global Optimization. *Appl. Sci.* **2021**, *11*, 5053. [CrossRef]
14. Mexia, J.a.T.; Pereira, D.; Baeta, J. $L_2$ environmental indexes. *Listy Biom.* **1999**, *36*, 137–143.
15. Pereira, D.G.; Mexia, J.T. Comparing double minimization and zigzag algorithms in joint regression analysis: The complete case. *J. Stat. Comput. Simul.* **2010**, *80*, 133–141. [CrossRef]
16. Yin, G. Convergence of a global stochastic optimization algorithm with partial step size restarting. *Adv. Appl. Probab.* **2000**, *32*, 480–498. [CrossRef]
17. Fouskakis, D.; Draper, D. Stochastic Optimization: A Review. *Int. Stat. Rev./Rev. Int. De Stat.* **2002**, *70*, 315–349. [CrossRef]
18. Peng, J.p.; Shi, D.h. Improvement of pure random search in global optimization. *J. Shanghai Univ.* **2000**, *4*, 92–95. [CrossRef]
19. Kushner, H.J.; Clark, D.S. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*; Applied Mathematical Sciences, Springer: New York, NY, USA; Berlin, Germany, 1978; Volume 26, pp. x+261.
20. Aluffi-Pentini, F.; Parisi, V.; Zirilli, F. Global optimization and stochastic differential equations. *J. Optim. Theory Appl.* **1985**, *47*, 1–16. [CrossRef]
21. Chiang, T.S.; Hwang, C.R.; Sheu, S.J. Diffusion for global optimization in $\mathbf{R}^n$. *Siam J. Control Optim.* **1987**, *25*, 737–753. [CrossRef]
22. Aluffi-Pentini, F.; Parisi, V.; Zirilli, F. A global optimization algorithm using stochastic differential equations. *ACM Trans. Math. Softw.* **1988**, *14*, 345–365. [CrossRef]
23. Mohan, C.; Shankar, K. A numerical study of some modified versions of controlled random search method for global optimization. *Int. J. Comput. Math.* **1988**, *23*, 325–341. [CrossRef]
24. Parpas, P.; Rustem, B.; Pistikopoulos, E.N. Linearly constrained global optimization and stochastic differential equations. *J. Glob. Optim.* **2006**, *36*, 191–217. [CrossRef]
25. Parpas, P.; Rustem, B. Convergence analysis of a global optimization algorithm using stochastic differential equations. *J. Glob. Optim.* **2009**, *45*, 95–110. [CrossRef]
26. Vegh, V.; Tieng, Q. Unconstrained real valued optimization based on stochastic differential equations. *Int. J. Innov. Comput. Inf. Control* **2011**, *7*, 6235–6246.
27. Glasserman, P. *Monte Carlo Methods in Financial Engineering*; Applications of Mathematics (New York); Stochastic Modelling and Applied Probability; Springer: New York, NY, USA, 2004; Volume 53, pp. xiv+596.
28. Yin, G. Rates of convergence for a class of global stochastic optimization algorithms. *Siam J. Optim.* **1999**, *10*, 99–120. [CrossRef]
29. Kontoghiorghes, E.J. (Ed.) *Handbook of Parallel Computing and Statistics*; Statistics: Textbooks and Monographs; Chapman & Hall/CRC: Boca Raton, FL, USA, 2006; Volume 184, pp. xvi+530.
30. Rajasekaran, S.; Reif, J. *Handbook of Parallel Computing: Models, Algorithms and Applications*; Chapman & Hall/CRC Computer and Information Science Series; CRC Press: Boca Raton, FL, USA, 2008.
31. Hamadi, Y.; Sais, L. (Eds.) *Handbook of Parallel Constraint Reasoning*; Springer: Cham, Switzerland, 2018; pp. xxvi+677. [CrossRef]
32. Price, W.L. Global optimization algorithms for a CAD workstation. *J. Optim. Theory Appl.* **1987**, *55*, 133–146. [CrossRef]
33. Truchet, C.; Arbelaez, A.; Richoux, F.; Codognet, P. Estimating parallel runtimes for randomized algorithms in constraint solving. *J. Heuristics* **2016**, *22*, 613–648. [CrossRef]
34. Chamberlain, R.; Edelman, M.; Franklin, M.; Witte, E. Simulated annealing on a multiprocessor. In Proceedings of the 1988 IEEE International Conference on Computer Design: VLSI, Rye Brook, NY, USA, 3–5 October 1988; pp. 540–544. [CrossRef]
35. Azencott, R. Parallel simulated annealing: An overview of basic techniques. In *Simulated Annealing*; Wiley-Interscience Series in Discrete Mathematics; Wiley: New York, NY, USA, 1992; pp. 37–46.
36. Onbaşoğlu, E.; Özdamar, L. Parallel simulated annealing algorithms in global optimization. *J. Glob. Optim.* **2001**, *19*, 27–50. [CrossRef]
37. Czech, Z.J. Three Parallel Algorithms for Simulated Annealing. In *Parallel Processing and Applied Mathematics*; Wyrzykowski, R., Dongarra, J., Paprzycki, M., Waśniewski, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 210–217.
38. Chen, D.J.; Lee, C.Y.; Park, C.H.; Mendes, P. Parallelizing simulated annealing algorithms based on high-performance computer. *J. Glob. Optim.* **2007**, *39*, 261–289. [CrossRef]
39. Lou, Z.; Reinitz, J. Parallel simulated annealing using an adaptive resampling interval. *Parallel Comput.* **2016**, *53*, 23–31. [CrossRef] [PubMed]

40. Genet, J. *Mesure et Intégration*; Librairie Vuibert: Paris, France, 1976; p. 323.
41. Kopp, P.E. *Martingales and Stochastic Integrals*; Cambridge University Press: Cambridge, UK, 1984; pp. xi+202.
42. Dudley, R.M. *Real Analysis and Probability*; Cambridge Studies in Advanced Mathematics; Revised Reprint of the 1989 Original; Cambridge University Press: Cambridge, UK, 2002; Volume 74, pp. x+555. [CrossRef]
43. Pacheco, P.S.; Malensek, M. *An Introduction to Parallel Programming*, 2nd ed.; Morgan Kaufmann Publishers; Elsevier: Cambridge, MA, USA, 2022; pp. xix+468.
44. Gustafson, J.L. Reevaluating Amdahl's Law. *Commun. ACM* **1988**, *31*, 532–533. [CrossRef]
45. McKean, H.P. *Stochastic Integrals*; Reprint of the 1969 Edition, with Errata; AMS Chelsea Publishing: Providence, RI, USA, 2005; pp. xvi+141. [CrossRef]
46. Bass, R.F. *Diffusions and Elliptic Operators*; Probability and Its Applications (New York); Springer: New York, NY, USA, 1998; pp. xiv+232.
47. Bass, R.F. *Stochastic Processes*; Cambridge Series in Statistical and Probabilistic Mathematics; Cambridge University Press: Cambridge, UK, 2011; Volume 33, pp. xvi+390. [CrossRef]
48. Kloeden, P.E.; Platen, E. *Numerical Solution of Stochastic Differential Equations*; Applications of Mathematics (New York); Springer: Berlin, Germany, 1992; Volume 23, pp. xxxvi+632. [CrossRef]
49. Korn, R.; Korn, E.; Kroisandt, G. *Monte Carlo Methods and Models in Finance and Insurance*; Chapman & Hall/CRC Financial Mathematics Series; CRC Press: Boca Raton, FL, USA, 2010; pp. xiv+470. [CrossRef]
50. *Probability Theory III: Stochastic Calculus*, Encyclopaedia of Mathematical Sciences, Volume 45; Prokhorov, Y.V., Shiryaev, A.N., Eds.; Springer: Berlin, Germany, 1998; Volume 45, pp. iv+253.
51. Wolfram Research, Inc. *Mathematica, Version 12.3.1*; Wolfram Research, Inc.: Champaign, IL, USA, 2021. [CrossRef]
52. Daniele, C.; Da Vià, R.; Manservisi, S.; Zunino, P. A multiscale heat transfer model for nuclear reactor assemblies. *Nucl. Eng. Des.* **2020**, *367*, 110794. [CrossRef]
53. Moré, J.J.; Garbow, B.S.; Hillstrom, K.E. Testing Unconstrained Optimization Software. *ACM Trans. Math. Softw.* **1981**, *7*, 17–41. [CrossRef]
54. Appel, M.J.; LaBarre, R.; Radulovic, D. On Accelerated Random Search. *SIAM J. Optim.* **2004**, *14*, 708–731. [CrossRef]