Walsh functions as surrogate model for pseudo-boolean optimization problems

Florian Leprêtre, Sébastien Verel, Cyril Fonlupt, Virginie Marion

HAL Id: hal-02190141
https://hal.science/hal-02190141

Submitted on 22 Jul 2019

Walsh Functions as Surrogate Model
for Pseudo-Boolean Optimization Problems

Florian Leprêtre
Univ. Littoral Côte d'Opale, LISIC
Calais, France
florian.lepretre@univ-littoral.fr

Sébastien Verel
Univ. Littoral Côte d'Opale, LISIC
Calais, France
verel@univ-littoral.fr

Cyril Fonlupt
Univ. Littoral Côte d'Opale, LISIC
Calais, France
fonlupt@univ-littoral.fr

Virginie Marion
Univ. Littoral Côte d'Opale, LISIC
Calais, France
virginie.marion@univ-littoral.fr

ABSTRACT

Surrogate-modeling is about formulating quick-to-evaluate mathematical models, to approximate black-box and time-consuming computations or simulation tasks. Although such models are well-established to solve continuous optimization problems, very few investigations regard the optimization of combinatorial structures. These structures deal for instance with binary variables, allowing each compound in the representation of a solution to be activated or not. Still, this field of research is experiencing a sudden renewed interest, bringing to the community fresh algorithmic ideas for growing these particular surrogate models. This article proposes the first surrogate-assisted optimization algorithm (WSaO) based on the mathematical foundations of discrete Walsh functions, combined with the powerful grey-box optimization techniques in order to solve pseudo-boolean optimization problems. We conduct our experiments on a benchmark of combinatorial structures and demonstrate the accuracy, and the optimization efficiency of the proposed model. We finally highlight how Walsh surrogates may outperform the state-of-the-art surrogate models for pseudo-boolean functions.

CCS CONCEPTS

• **Computing methodologies** → **Search methodologies**.

KEYWORDS

Local search, Surrogate model/fitness approximation, Combinatorial optimization, Empirical study.

1 INTRODUCTION

1.1 Motivations

The optimization or the tuning of numerous systems is based on numerical simulations. As a consequence, the associated optimization problem is black-box, *i.e.* only the design variables and the value of the fitness function are known. Moreover, it is often computationally time expensive (from minutes to hours) to get the fitness value of one single solution [1, 5]. To face such optimization problem, one classical solution in *surrogate-assisted optimization* (SaO) is to learn a surrogate model to approximate the fitness function and then reduce the number of sampled potential solutions during the search process.

Despite the number of research works dealing with surrogate models for numerical optimization problems [14], SaO for combinatorial optimization received little attention [3]. However, design problems with discrete variables can be met in different fields such as bike sharing system [24], bus stop position in public transportation system [23], drug discovery [22] or more generally problems with design variables to represent the presence/absence of a component. Typically, on such kind of real-world applications, the optimization problems have hundred of binary decision variables. That seems far from the actual surrogate-assisted methods [3]. In this work, we target optimization problems considering the search space of binary strings, *i.e.* pseudo-boolean functions with a dimension of the order of one hundred.

A surrogate-assisted method is the combination of three components (see Algorithm 1). The first component is the surrogate model itself which is a regression model of the fitness function. The model must be enough expressive to catch the complexity of the fitness function, but at the same time with low complexity in order to be easier to learn with a small sample of solutions. The second component is an acquisition function defined from the surrogate model. This acquisition function can be directly the surrogate model or a trade-off between the prediction and the estimation error of the surrogate model. The goal is to guide the search keeping the balance between exploration that increases the quality of the surrogate model and exploitation that pushes toward high-quality solutions according to the surrogate model. The last component is the algorithm to optimize the acquisition function. This algorithm has to be efficient in time and in quality to converge quickly to

the promising solutions given by the acquisition function. An efficient surrogate-assisted optimizer for combinatorial problems is a relevant combination of these three components.

Some recent works renew the interest of surrogate models for combinatorial optimization [2, 3, 26]. A recent approach [26] proposes the orthogonal basis of Walsh functions as a surrogate model for pseudo-boolean functions. On pseudo-boolean functions, the regression quality of this surrogate model outperforms the previous models. Although the model seems to be promising, a complete surrogate-assisted algorithm has not been proposed. Indeed, such model must be combined with a relevant optimization algorithm to be efficient in an optimization context. We bring a step forward in this approach by combining Walsh functions surrogate models with powerful techniques of the grey-box optimization [6, 7] dedicated to Walsh functions decompositions of pseudo-boolean functions. In this article, we propose a way to couple both techniques in an operational manner. We also test the proposed Walsh functions Surrogate-assisted Optimization algorithm (WSaO) against state-of-art methods on a benchmark of pseudo-boolean functions up to dimension one hundred.

1.2 Related works

Both numerical and combinatorial surrogate-assisted optimization algorithm follow a similar approach. A sample of evaluated solutions builds a surrogate model. Based on this model and an acquisition function, a new promising solution is selected, evaluated and added to the sample for the next iteration of the search algorithm (see Algorithm 1). However, numerical and combinatorial domains mainly differ on the model of the fitness function approximation used for the surrogate. Although there are models for mixed representation with continuous and discrete variables in the context of parameter tuning such as SMAC [13], only few works are dedicated to combinatorial, even less for pseudo-boolean optimization problems [3]. Indeed, to our best knowledge, only 3 main alternatives have been proposed for pseudo-boolean optimization problems: Radial Basis Function model [21], Kriging approach [31], and Bayesian approach [2].

Algorithm 1: Surrogate-assisted optimization framework.

---

1   $S \leftarrow$ Initial sample $\{(x, f(x)), \ldots\}$
2   **while** *computational budget is not spent* **do**
3      $M \leftarrow$ Build model S
4      $x \leftarrow$ Optimize M w.r.t. an acquisition function
5   Evaluate x using f
6      $S \leftarrow S \cup \{(x, f(x)\}$
7   **end**

One state-of-art approach for combinatorial optimization uses Kriging approach, based on Gaussian Process (GP) [30, 31]. This approach is a direct extension of the numerical surrogate approach where the Euclidean distance is replaced by the Hamming distance for pseudo-boolean functions, or more sophisticated discrete distance for the search space of permutations. Kriging is then embedded within the well-known Efficient Global Optimizer framework

(EGO) [15]. In this framework, the solution with the highest Expected Improvement (EI) is selected with an evolutionary algorithm. The EI takes into account the uncertainty of the approximation given by the GP. This is one advantage of the EGO algorithm. It selects relevant solutions to increase the accuracy of the model toward promising solutions. We can mention that the computational complexity of the function EI is high, and can not be reduce by some classic techniques in combinatorial optimization such as incremental evaluation. Besides, the authors use a genetic algorithm, a black-box optimizer, to maximize the EI function. However, this surrogate-assisted optimization has been shown to outperform Radial Basis Function model [21], one of few others related works in combinatorial surrogate optimization. For a complete overview of this approach see [29]. Notice that such approaches have been tested on small dimensions up to $n = 25$.

Bayesian Optimization of Combinatorial Structures (BOCS) [2] is another recent state-of-art surrogate-assisted algorithm. The statistical model of BOCS is the standard multilinear polynomial of binary variables:

$$\forall x \in \{0, 1\}^n, \ P(x) = \sum_{T \subset N} a_T \prod_{i \in T} x_i, \tag{1}$$

where $N = \{1, \ldots, n\}$. With this model, Baptista *et. al.* argue that interactions between variables can be taken into account.[1] In their article, only a quadratic polynomial model is used:

$$\forall x \in \{0, 1\}^n, \ P_2(x) = a_0 + \sum_{i \in N} a_i x_i + \sum_{i < j \in N} a_{ij} \ x_i x_j. \tag{2}$$

To face the quadratic number of polynomial terms (see Table 1), the regression technique of the algorithm is the Sparse Bayesian Linear Regression [20]. The optimizer is a basic simulated annealing that minimizes the approximation of the fitness function provided by the surrogate with a regularization term. On low-dimensional quadratic problems (up to $n = 24$), BOCS outperforms Kriging approach. A useful python code of BOCS is available on the web.[2]

A new surrogate model for pseudo-boolean functions based on Walsh functions has been proposed [26]. The Walsh functions are a normal orthogonal basis of the pseudo-boolean functions space. Although they have been used in Evolutionary Computation in schemata theory [10], and also in grey-box optimization [6, 8], the authors proposed to use such functions as a surrogate model. Using the sparse regression technique LARS (Least-Angle Regression), the accuracy of the regression model has been demonstrated higher than GP models on low-dimension functions up to $n = 25$. However, these Walsh functions have not been used in a SaO scenario following the framework of Algorithm 1.

In this work, we show how this novel idea of Walsh functions for surrogate modeling can be combined with the efficient grey-box optimization techniques into a surrogate-assisted optimization algorithm. Specifically, our contributions consist in proposing and describing the first Walsh Surrogate-assisted Optimization framework based on grey-box optimization methods. We compare the

---

[1]Indeed, such multilinear approximation has been already used for the approximation of pseudo-boolean functions in another context [11].
[2]https://github.com/baptistar/BOCS

presented method with the aforementioned state-of-the-art algorithms on large size problems up to $n = 100$. Besides, we open the discussion to understand the pros and cons of the Walsh basis and to show how the interactions between variables can be interpreted in the surrogate models.

In section 2, we recall the mathematical foundations of Walsh functions as well as their application to design surrogate models and present the first Walsh Surrogate-assisted Optimization algorithm. Section 3 is devoted to experiments specifications and their analyses. In section 4, we conclude the paper and discuss future works.

2 WALSH FUNCTIONS SURROGATE-ASSISTED OPTIMIZATION

In this section, we present the Walsh Surrogate-assisted Optimization Algorithm (WSaO) based on the Walsh functions. We first recall how surrogate models can be designed with Walsh functions. Then we propose how these models can be employed in an SaO approach.

2.1 Walsh Functions as Surrogate Models

Initially defined by the author who gave them his name, Walsh functions [27] describe a normal and orthogonal basis of discontinuous functions that can be employed to decompose any function of the Hilbert space. As far as we are concerned in the context of combinatorial structures, we employ such functions as a normal orthogonal basis for the space of pseudo-boolean problems. More formally, for any integer l represented in its binary form, the Walsh function φ_ℓ is defined for any binary string $x = (x_1, \ldots, x_n)$ of size n as:

$$\varphi_\ell(x) = (-1)^{\sum_{i=1}^{n} \ell_i x_i}, \qquad (3)$$

where ℓ_i and x_i stand for the i-th bit in the binary strings ℓ and x. In a surrogate-modeling perspective and following Verel *et. al.* works [26], we assume that the pseudo-boolean function f we want to approximate can be substituted by the following polynomial $\hat{f}$, for every binary string $x \in \{0, 1\}^n$:

$$\hat{f}_L(x) = \sum_{\ell \ s.t. \ o(\varphi_\ell) \leqslant L} \hat{w}_\ell \cdot \varphi_\ell(x), \qquad (4)$$

where o is the order of the Walsh function, *i.e.* the number of binary digits equals to 1 in the binary representation of ℓ. We restrain the expansion of the polynomial to a given order L. For example, the Walsh function of order 2 can be written as:

$$\hat{f}_2(x) = w_0 + \sum_{i=1}^{n} w_i \ (-1)^{x_i} + \sum_{i < j \in N} w_{ij} \ (-1)^{x_i + x_j}. \qquad (5)$$

One should notice the potential of approximation for such a formulation grows when L increases. All things considered, we thus aim to estimate the values of the coefficients $\hat{w}$: the better the quality of the estimation, the more accurate the model. This estimation can be performed thanks to classical linear regression methods, as the Walsh decomposition of Equation 4 can be interpreted as a linear model whose predictors are the Walsh functions' values. Furthermore, when the number of predictors is getting large (see Table 1), *i.e.* when the selected order L increases, as shown in [26],

Table 1: Number of coefficients both in multilinear and Walsh decompositions, according to problem dimension n, and order L.

L	n					
	10	15	20	25	50	100
0	1	1	1	1	1	1
1	11	16	21	26	51	101
2	56	121	211	326	1,276	5,051
3	176	576	1351	2,626	20,876	166,751

sparse linear regression techniques should be considered in order to minimize the number of non-zero coefficients. In this work, we propose to use the Lasso linear model [25], as it tends to favor solutions with fewer parameter values, *i.e.* to maximize the number of zero coefficients in the model. Thus, it appears as a convenient method in our context, since the number of L-order Walsh functions, *i.e.* the number of $\hat{w}$ coefficients to be estimated, might be greater than the budget dedicated to train the Walsh surrogate. Mathematically, such a model is trained with ℓ_1-norm as regularizer, often weighted by a constant α (see section 3). This geometrical property makes Lasso a method of choice for training the proposed surrogate, as the latter relies on a normal and orthogonal basis of functions. Finally, this surrogate-building phase can be related to line 3 of Algorithm 1.

2.2 Surrogate-Assisted Optimization

As discussed in section 1.2, SaO algorithm consists in optimizing the quick-to-evaluate surrogate model instead of the expensive real one. The optimal solution found by the surrogate is expected to be promising when evaluated with the true model. Then, this optimal solution and its associated true fitness value are added to the surrogate's learning sample for the next iteration. In this article, we propose to use the so-called Efficient Hill-climber (EH) from Chicano *et. al.* [9], in order to optimize the Walsh surrogates (line 4 of Algorithm 1). Indeed, this EH is extremely fitted for pseudo-boolean functions expressed as a Walsh decomposition. Without loss of generalization, we will consider the case of maximization. The optimization algorithm, summarized in Algorithm 2, consists in identifying improving moves in a Hamming-ball, so as to focus the optimizer on flipping promising variables in the solution. In this work, we define a score for each move in a ball of radius 1, *i.e.* each possible bit-flip in a given solution:

$$\delta_m(x) = \hat{f}(x \oplus m) - \hat{f}(x), \qquad (6)$$

where x is a solution represented as a binary string, m is a potential move and $\oplus$ denotes the bit-flip operation. This step is related to line 3 of Algorithm 2. The next step consists in flipping the m-th bit in x that maximizes Equation 6, which is related to lines 5 and 7. Actually, it can be shown that only some scores are affected by this flip, depending on the interactions of variables [9]. As a consequence, only specific scores are updated in line 8. This process is iterated until the computation budget is spent or no more improving moves are possible, *i.e.* all δ_m are lower or equal to zero ($\delta_i^\star$ is the best move so far). Finally, if the proposed optimal solution is

already known in the surrogate's learning sample from previous optimizations, then the new point to be added to the learning set is sampled randomly (line 13), in order to provide diversity and ease the learning phase. As a result, we expect this algorithm to tackle large size combinatorial optimization problems in a fast and computationally-efficient way.

Algorithm 2: Maximization of the Walsh surrogate model.

1 while *restarts budget is not spent* **do**
2 $x \leftarrow$ Random solution
3 $\delta \leftarrow$ Compute scores for each move
4   **while** *iterations budget is not spent* **do**
5 $i^\star \leftarrow$ Find arg max δ_i
6   **if** $\delta_{i^\star} > 0$ **then**
7 $x \leftarrow$ Flip $i^\star$-th bit in x
8 $\delta \leftarrow$ Update affected scores
9   **else**
10   **break**
11 $x^\star \leftarrow$ Update best solution if x is an improving solution
12 **if** $x^\star$ *is already known* **then**
13 $x^\star \leftarrow$ Sample random solution

---

3 EXPERIMENTAL ANALYSIS

We first introduce the two combinatorial problems studied and their respective experimental setup. Then, the regression accuracy of the Walsh surrogates and the performances of the proposed optimization algorithm are compared to other methods.

3.1 Benchmark

Unconstrained Binary Quadratic Programming. The Unconstrained Binary Quadratic Programming (UBQP) problems are defined by the minimization of the function:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} q_{ij}\, x_i x_j, \tag{7}$$

where x_i denotes the i-th bit in the binary string x of size n and $Q = (q_{ij})$ is a n-by-n square, symmetric matrix of random coefficients. In the optimization field, this matrix is usually sparse, *i.e.* some coefficients are set to zero, according to a density parameter d. Otherwise, coefficients are set uniformly in the range $[-100, 100]$ [19]. These models, although simple in their definitions, actually represent challenging computational questions and embrace a wide variety of combinatorial optimization problems, such as traffic management, facility positionings, resources allocation, or even economic analysis, as surveyed by Kochenberger *et. al.* [17]. Due to its NP-hard nature and its large application range, UBQP has continually motivated the scientific community to propose novel metaheuristics, mostly based on tabu search as pioneered by Beasley [4].

nk-Landscapes. The *nk*-landscapes (NKL) are mathematical models proposed by Kauffman [16]. They involve fitness landscapes of varied sizes and ruggedness that are tunable thanks to the parameters n and k, respectively. For every binary string $x = (x_1, \ldots, x_n)$ of size n, the fitness of x is based on the sum of fitness contributions g_i. These component functions take into account the value of each variable x_i in the string and the values of its k interacting variables $(x_{i_1}, \ldots, x_{i_k})$. The function f to maximize is hence defined as follows:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} g_i(x_i, x_{i_1}, \ldots, x_{i_k}). \tag{8}$$

Typically, the component functions g_i assign an uniformly distributed real-valued contribution in $[0, 1[$ for every possible combination of x_i and its k *epistatic* interactions. In some sense, the component functions g_i are peek at random in the space of pseudo-boolean functions with codomain $[0, 1[$. The interactions are chosen uniformly at random among the $(n - 1)$ variables other than x_i. Therefore, the *nk*-landscapes functions describe gradually challenging optimization problems when k increases, providing a variety of landscapes from smooth to hilly ones. For instance, functions are linear when $k = 0$, whereas they are quadratic when $k = 1$. In an optimization perspective, it is usual to search for the binary strings x that maximize these functions. Considering these blueprints, NKL may abstractly portray a wide range of real-world optimization problems [28].

Walsh vs. Multilinear basis. Walsh coefficients, and multilinear coefficients can be computed for *nk*-landscapes [12] and UBQP problems. For *nk*-landscapes, due to the linearity of both decompositions, the decomposition is the average of the decomposition of each component function. For example, when $k = 1$, the dimension of the component functions is 2. In this case there are 4 coefficients for both decompositions. The values of Walsh and multilinear coefficients can be computed respectively with $W_2\, g$ and $L_2\, g$, where

$$W_2 = \frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \text{ and } L_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

with $g = [g(00), g(01), g(10), g(11)]^t$ is the vector with the component function's values. As a consequence, when g is a vector of random values following independent random distributions of mean μ and variance σ^2, then the mean of the first Walsh coefficient is $\frac{\mu}{4}$ and the mean of the others coefficients is 0. Moreover, the variance of all Walsh coefficients is equal to $\frac{\sigma^2}{4}$. For multilinear basis, the mean of the first coefficient is μ and the mean of the other ones is 0. Yet, the variance of the multilinear coefficients increases with the order, respectively σ^2, $2\sigma^2$, and $4\sigma^2$ for the coefficients of order 0, 1, and 2. Concerning *nk*-landscapes, the number of null coefficients is the same for both Walsh and multilinear decompositions, but the distribution of coefficients is not the same. UBQP problems are directly defined in the multilinear basis. Table 2 gives the transformation between multilinear and Walsh bases. The bilinear coefficient q_{ij} with $i \neq j$ gives the coefficient $\frac{1}{4}q_{ij}$ in the Walsh

Table 2: Multilinear *vs.* Walsh bases for linear and quadratic functions.

Multilinear basis	Walsh basis
x_i $x_i x_j$	$\frac{1}{2} - \frac{1}{2}(-1)^{x_i}$ $\frac{1}{4} - \frac{1}{4}(-1)^{x_i} - \frac{1}{4}(-1)^{x_j} + \frac{1}{4}(-1)^{x_i+x_j}$
$1 - 2x_i$	$(-1)^{x_i}$
$1 - 2x_i - 2x_j + 4x_i x_j$	$(-1)^{x_i+x_j}$

basis and adds some linear terms. Thus, for UBQP problems, the decomposition has the same null coefficients for the quadratic terms, but not for the linear ones. Again, the distribution of the coefficients will be different in Walsh and multilinear bases. This suggests that the quality of the regression for nk-landscapes and UBQP will not be the same for different surrogate bases of functions.

Experimental Setup. We consider both of the aforementioned combinatorial problems for our empirical studies. We involve binary strings of size $n \in \{10, 25, 50, 100\}$. For UBQP, according to [19], we set the density of zero parameter d to 90%. For nk-landscapes, we vary $k \in \{1, 2\}$ which represents quadratic ($k = 1$), and cubic interaction ($k = 2$). For every combination of parameters (4 for UBQP, $4 \times 2 = 8$ for NKL), five instances of the corresponding problem are generated. For both combinatorial problems, the regularization parameter α of the LASSO regression method is chosen empirically from the order of the fitness value of random solutions. The regularization parameter α is set to 10^{-1} and 10^{-5} for UBQP and nk-landscapes, respectively. We validate the accuracy of the so-trained models against a test-set of 1000 solutions generated uniformly at random. Competing learning methods are run 20 times on each instance. Competing surrogate-assisted optimization algorithms are restarted and iterated 100 times each, related to lines 1 and 4 in Algorithm 2, respectively. The following results report the accuracy of the trained surrogates and their performances in a surrogate-assisted optimization approach. Algorithms and experiments are fully implemented in Python, using standard machine learning and optimization packages. The code is available on the web[3].

3.2 Accuracy of Surrogate Models

At first, we simply experiment the accuracy of three surrogate models, based on gaussian process (GP), multilinear functions basis used in BOCS, and Walsh functions basis (see sections 1 and 2). No optimization algorithm is involved yet. Solutions selected in order to learn the model are sampled randomly from $\{0, 1\}^n$. In Figures 1, 2 and 3, we compare the mean absolute error made by the models as a function of the random samples dedicated to their learning. For both benchmark problems, each surrogate model converges towards a null error when the sample size grows, yet with different convergence speeds. Overall, gaussian process appears as the slowest of the three compared methods. For UBQP instances reported in Figure 1, the accuracy of multilinear and Walsh surrogates evolves in essentially the same way, regardless of the considered sizes of the problems. Nevertheless, Walsh surrogates always reach an absolute-zero error, while multilinear ones need about twice the
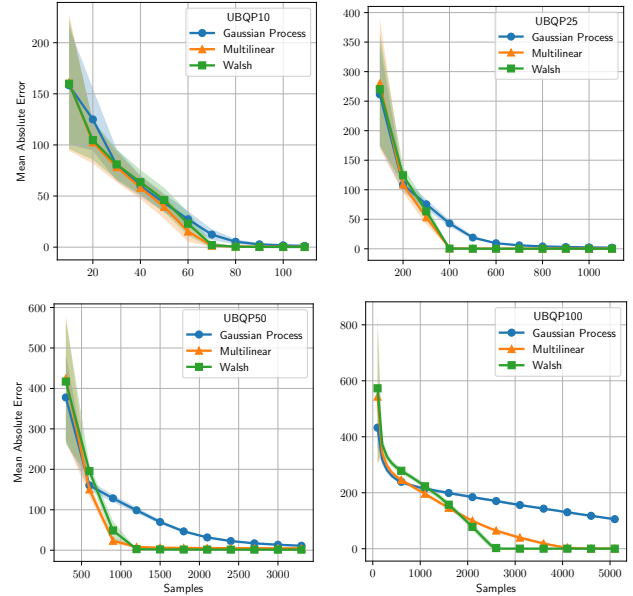
---
[3]https://gitlab.com/florianlprt/wsao

Figure 1: Mean absolute error with confidence interval as a function of random samples for UBQP problems.

sample size required by Walsh surrogates to achieve this accuracy when $n = 100$. In the other hand, Figures 2 and 3 show significant differences in convergence speed for these same two methods applied to NKL problems. The sample size required to achieve a zero absolute error is substantially lower for Walsh surrogates, about twice as fast as multilinear ones when $n = 50$ or $n = 100$. This contrast is even more noticeable when the size of the problem increases or when the epistatic interactions per variable is set to $k = 2$ which is the cubic interaction. Indeed, we clearly notice in Figure 4 that the sample size required for the Walsh surrogates to reach a null error grows rapidly with the number of epistatic interactions.

3.3 Performances of Optimization Algorithms

Secondly, we experiment the Walsh-based surrogate-assisted optimization algorithm (WSaO) proposed in Algorithm 2. We compare its performances with the state-of-the-art algorithms discussed in section 1.2: EGO based on gaussian process and BOCS for optimizing models based on a decomposition of multilinear functions. In addition, we also compare the performances of Algorithm 2 when applied to a multilinear-based surrogate learned with Lasso regression method. Unlike the regression analysis of the previous section 3.2, the solution added each iteration to the surrogate's learning sample is now the solution that maximizes (or minimizes) the surrogate model, according to Algorithm 1. Due to a lack of computational time, the results presented in the following only consider UBQP problems and NKL problems where k is set to 1 (quadratic interaction). Further analyses involving more epistatic interactions are left for future works. Though, in order to provide exploitable results to the reader, Table 3 records the average quality of solutions found by these four optimization algorithms, regarding two computational budgets. We define high budgets approximately
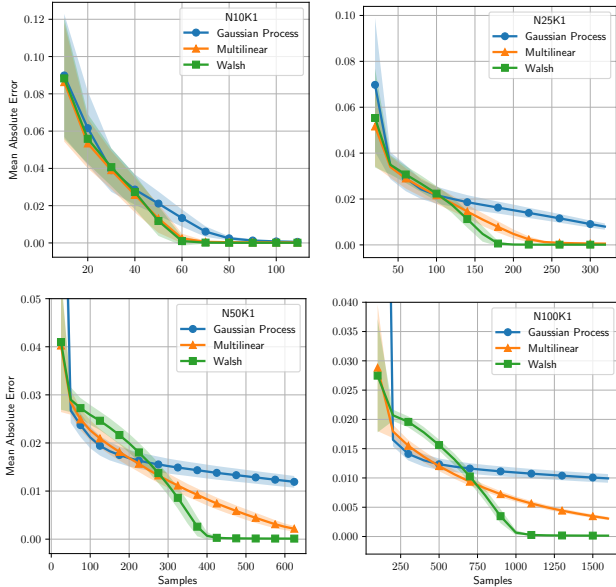
Figure 2: Mean absolute error with confidence interval as a function of random samples for NKL problems. k is set to 1.
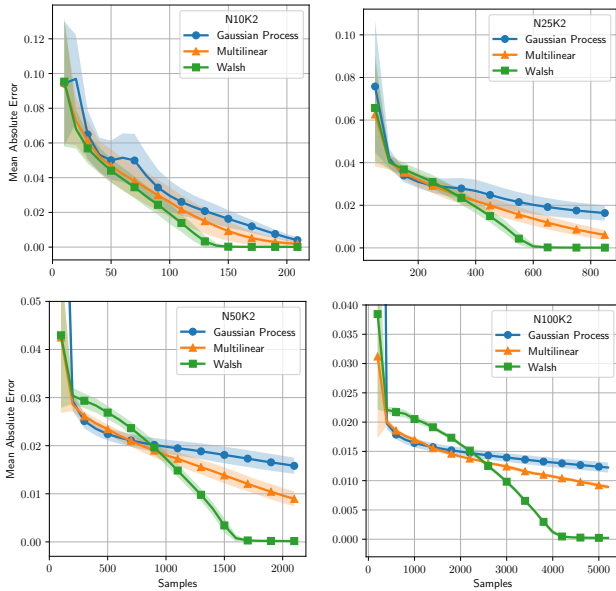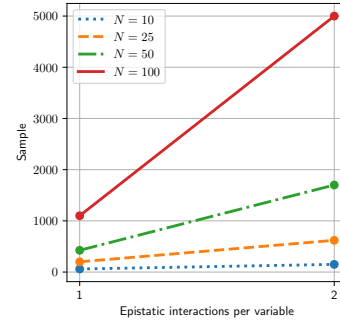

Figure 4: Required sample to reach a null error for Walsh surrogates, according to the size of the problem and the number of epistatic interactions.

difference between those presented averages. Regarding UBQP instances, EGO and a multilinear model optimized with EH provide lower quality solutions. Both get stuck in local optima, all the more the size of the problems increases. BOCS algorithm appears to be very efficient when the dimension is low (up to $n = 25$), as it requires low budget to converge to optimal solutions. However, the algorithm drastically stalls as soon as the size of the problem gets larger. This explains the abrupt stops in Figures 5 and 6: the algorithm could not complete in the allowed computational time (*i.e.* 96 hours parallelized on 4 CPU cores) for sizes $n = 50$ and $n = 100$. In contrast, only the proposed WSaO manage to find high-quality solutions, regardless of the dimension of the problem. Concerning NKL problems, performances are quite similar for every algorithm when dimension is low, although EGO appears less effective. Once again, BOCS algorithm gets hampered when the dimension increases. Nevertheless, Walsh surrogates-assisted optimized with EH (WSaO) reach solutions with the highest quality on large size NKL, even with a low computational budget (Walsh provides slightly better-quality solutions from 500 samples when $n = 100$). As a conclusion, it appears that BOCS provides efficient optimization for both low-dimension problems up to $n = 25$. However, these performances do not scale with higher dimensions, unlike WSaO. The latter are effective independently of the size of the problem until at least size $n = 100$. From our experimental analysis, BOCS algorithm can not be used when the problem dimension is larger than 50, whereas WSaO can scale at least up to 100 dimensions.

Overall, we notice the sample size needed to reach optimal values is larger for UBQP problems than the one required for NKL problems. For instance, when $n = 100$, a sample size of 3000 is required for UBQP, whereas the required sample size is 1000 to reach an optimum for NKL.

4 DISCUSSIONS

Walsh Surrogate-assisted Optimization algorithm (WSaO) combines a relevant basis orthogonal of Walsh functions, and a very efficient optimization method dedicated for this Walsh decomposition from grey-box optimization. It has been compared with two state-of-art algorithms based on Gaussian Process, on sparse bayesian estimation of the coefficient multilinear polynomial model (BOCS). Note

Figure 3: Mean absolute error with confidence interval as a function of random samples for NKL problems. k is set to 2. $\alpha = 10^{-5}$ for Lasso.

equal to the number of samples that are required for the fastest surrogate models to plenty converge to an absolute-zero error in section 3.2. Then, low budgets are defined as the third of these high budgets. Statistical tests are employed to determine a significant

Table 3: Average quality of solutions found by the four optimization algorithms, with low or high budget dedicated to the search. For UBQP problems, the lower the better. For NKL problems, the higher the better. EH stands for Efficient Hill-climber. The * symbol stands when the method has not been completed for the considered problem. Averages appear in bold when the difference is statistically significant with Mann-Whitney tests at level 5%, according to problems and computational budgets.

Problems	Low budget				High budget			
	EGO	Multilinear (EH)	WSaO	BOCS	EGO	Multilinear (EH)	WSaO	BOCS
UBQP10	-104.67±27.62	**-178.00±0.00**	-176.67±5.16	**-178.00±0.00**	-146.27±23.81	**-178.00±0.00**	**-178.00±0.00**	**-178.00±0.00**
UBQP25	-539.27±71.34	-679.07±50.91	-831.80±30.50	**-845.00±0.00**	-586.13±59.27	-707.73±34.83	**-845.00±0.00**	**-845.00±0.00**
UBQP50	-1727.14±197.87	-2069.60±133.17	**-3080.73±70.58**	*	-1924.67±237.10	-2161.93±122.73	**-3141.00±0.00**	*
UBQP100	*	-3756.50±152.03	**-6909.50±120.40**	*	*	-3811.00±74.95	**-7522.00±0.00**	*
N10K1	0.75±0.04	0.80±0.00	0.79±0.01	**0.80±0.00**	0.80±0.00	0.80±0.00	0.80±0.00	0.80±0.00
N25K1	0.59±0.01	**0.62±0.01**	**0.62±0.01**	**0.62±0.00**	0.62±0.01	**0.63±0.00**	**0.63±0.00**	**0.63±0.00**
N50K1	0.70±0.01	**0.72±0.01**	**0.71±0.01**	*	0.72±0.00	**0.73±0.00**	**0.73±0.00**	*
N100K1	0.67±0.01	0.69±0.01	**0.69±0.01**	*	0.67±0.01	0.70±0.00	**0.71±0.00**	*

that this model is able to represent any pseudo-boolean function [11], but is not an orthogonal basis. The results show that WSaO is competitive on small dimension problems although BOCS approach can be outperformed by WSaO, but WSaO can scale to middle size problems up to $n = 100$ which is targeted dimension for the future applications of the algorithm.

It opens many directions of future research. First, we would like to use the WSaO on real-world problems related to city mobility. One target optimization problem is the setting of the bus stop position in the city. This problem is then an expensive pseudo-boolean problem of dimension of the order of 100. The running time of simulator used to test the position is around few minutes [18]. WSaO is then a good candidate to face such real-world problem. According to the complexity of the fitness functions that could be different as in our experiments, one optimization process to find one high quality solution can take between 7 days and 20 days of computation on a single core machine. From an algorithm side of research, it will push us to design a parallel version of the WSaO. In this work, we use a simple Iterated Local Search with a pure random restart. We also intend to improve the efficiency of the optimization algorithm by using a population-based algorithm and the cross-over based on articulation points recently proposed in grey-box optimization [6]. Finally, the performance gap of different surrogate models and regression methods according to the problem dimension, and complexity suggest to design an ensemble of surrogate models that combines the different approaches.

ACKNOWLEDGMENTS

REFERENCES

[1] Rolando Armas, Hernán Aguirre, Saúl Zapotecas-Martínez, and Kiyoshi Tanaka. 2015. Traffic signal optimization: minimizing travel time and fuel consumption. In *International Conference on Artificial Evolution (Evolution Artificielle)*. Springer, 29–43.

[2] Ricardo Baptista and Matthias Poloczek. 2018. Bayesian Optimization of Combinatorial Structures. In *International Conference on Machine Learning (ICML)*. 462–471.

[3] Thomas Bartz-Beielstein and Martin Zaefferer. 2017. Model-based methods for continuous and discrete global optimization. *Applied Soft Computing* 55 (2017), 154–167.

[4] John E Beasley. 1998. Heuristic algorithms for the unconstrained binary quadratic programming problem. *London, England* (1998).

[5] Juergen Branke. 2018. Simulation Optimization Tutorial. In *Proceedings of the Genetic and Evolutionary Computation Conference Compagnion*. ACM.

[6] Francisco Chicano, Gabriela Ochoa, Darrell Whitley, and Renato Tinós. 2018. Enhancing partition crossover with articulation points analysis. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 269–276.

[7] Francisco Chicano, Darrell Whitley, and Enrique Alba. 2014. Exact computation of the expectation surfaces for uniform crossover along with bit-flip mutation. *Theoretical Computer Science* 545 (2014), 76–93.

[8] Francisco Chicano, Darrell Whitley, Gabriela Ochoa, and Renato Tinós. 2017. Optimizing one million variable NK landscapes by hybridizing deterministic recombination and local search. In *GECCO*. 753–760.

[9] Francisco Chicano, Darrell Whitley, and Andrew M. Sutton. 2014. Efficient Identification of Improving Moves in a Ball for Pseudo-boolean Problems. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14)*. ACM, New York, NY, USA, 437–444. https://doi.org/10.1145/2576768.2598304

[10] David E Goldberg. 1989. Genetic algorithms and Walsh functions: Part I, a gentle introduction. *Complex systems* 3, 2 (1989), 129–152.

[11] Peter L Hammer and Ron Holzman. 1992. Approximations of pseudo-Boolean functions; applications to game theory. *Zeitschrift für Operations Research* 36, 1 (1992), 3–21.

[12] Robert B. Heckendorn. 2002. Embedded Landscapes. *Evo. Comp.* 10, 4 (2002), 345–369. https://doi.org/10.1162/106365602760972758 arXiv:https://doi.org/10.1162/106365602760972758

[13] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. 2011. Sequential model-based optimization for general algorithm configuration. In *International Conference on Learning and Intelligent Optimization*. Springer, 507–523.

[14] Yaochu Jin. 2011. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation* 1, 2 (2011), 61–70.

[15] Donald R Jones, Matthias Schonlau, and William J Welch. 1998. Efficient global optimization of expensive black-box functions. *Journal of Global optimization* 13, 4 (1998), 455–492.

[16] S. A. Kauffman. 1993. *The Origins of Order*. Oxford University Press.

[17] Gary Kochenberger, Jin-Kao Hao, Fred Glover, Mark Lewis, Zhipeng Lü, Haibo Wang, and Yang Wang. 2014. The unconstrained binary quadratic programming problem: A survey. *Journal of Combinatorial Optimization* 28 (07 2014). https://doi.org/10.1007/s10878-014-9734-0

[18] Florian Leprêtre, Cyril Fonlupt, Sébastien Verel, and Virginie Marion. 2018. SIALAC benchmark: on the design of adaptive algorithms for traffic lights problems. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, 288–289.

[19] Arnaud Liefooghe, Sébastien Verel, Luis Paquete, and Jin-Kao Hao. 2015. Experiments on local search for bi-objective unconstrained binary quadratic programming. In *EMO - 2015 8th International Conference on Evolutionary Multi-Criterion Optimization (Lecture Notes in Computer Science)*, Vol. 9018. Guimarães, Portugal, 171–186. https://doi.org/10.1007/978-3-319-15934-8_12

[20] Enes Makalic and Daniel F Schmidt. 2016. A simple sampler for the horseshoe estimator. *IEEE Signal Processing Letters* 23, 1 (2016), 179–182.

[21] Alberto Moraglio and Ahmed Kattan. 2011. Geometric generalisation of surrogate model based optimisation to combinatorial spaces. In *European Conference on Evolutionary Computation in Combinatorial Optimization*. Springer, 142–154.

[22] Diana M Negoescu, Peter I Frazier, and Warren B Powell. 2011. The knowledge-gradient algorithm for sequencing experiments in drug discovery. *INFORMS*
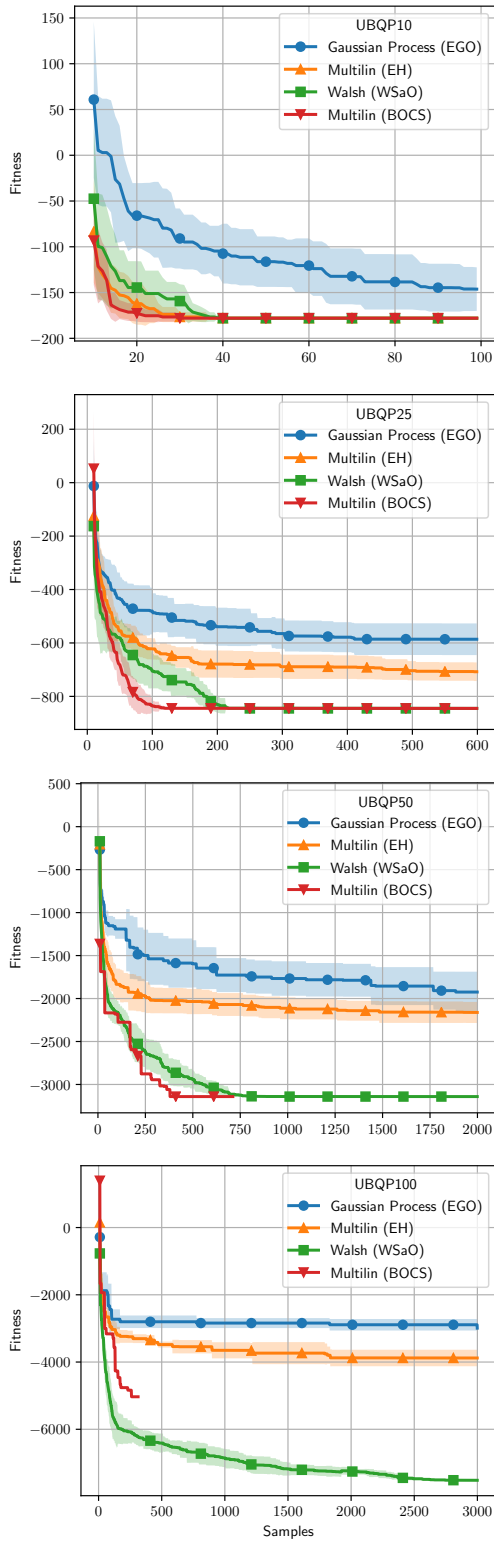
Figure 5: Minimization of fitness according to sample size for UBQP problems. Four optimization algorithms are compared. EH stands for Efficient Hill-climber.
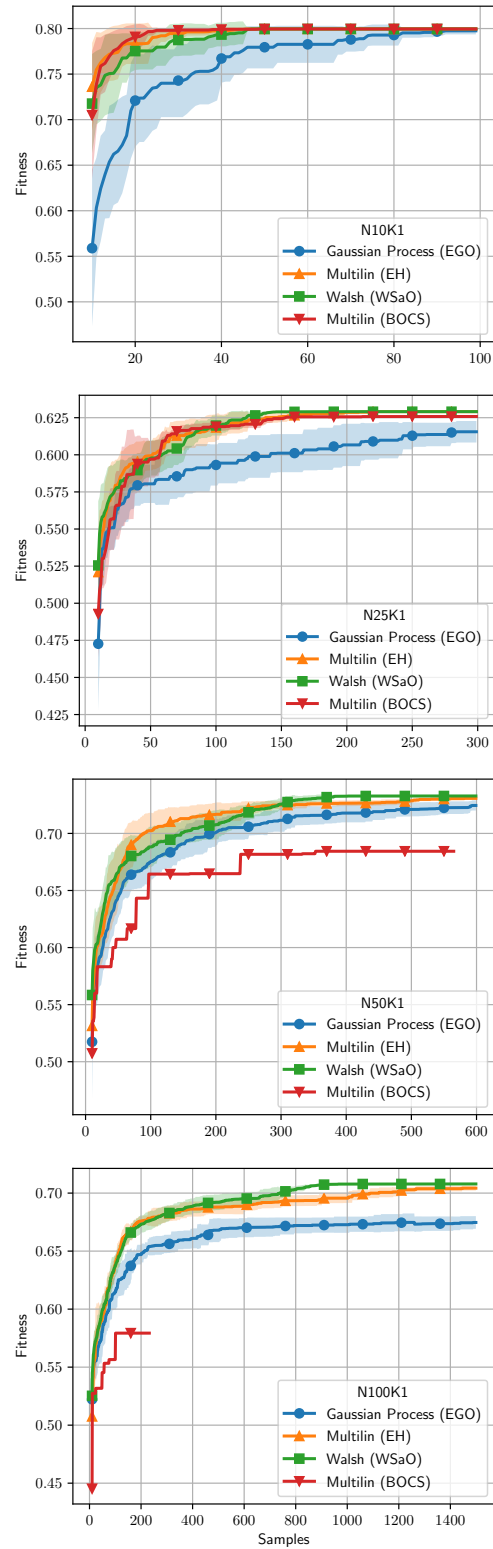
Figure 6: Maximization of fitness according to sample size for NKL problems, with k fixed to 1. Four optimization algorithms are compared. EH stands for Efficient Hill-climber.

Journal on Computing 23, 3 (2011), 346–363.

[23] Marcel Rieser. 2010. *Adding Transit to an Agent-Based Transportation Simulation: Concepts and Implementation.* Ph.D. Dissertation. der Technischen Universitat Berlin.

[24] Juan P Romero, Angel Ibeas, Jose L Moura, Juan Benavente, and Borja Alonso. 2012. A simulation-optimization approach to design efficient systems of bike-sharing. *Procedia-Social and Behavioral Sciences* 54 (2012), 646–655.

[25] Robert Tibshirani, Martin Wainwright, and Trevor Hastie. 2015. *Statistical learning with sparsity: the lasso and generalizations.* Chapman and Hall/CRC.

[26] Sébastien Verel, Bilel Derbel, Arnaud Liefooghe, Hernan Aguirre, and Kiyoshi Tanaka. 2018. A surrogate model based on Walsh decomposition for pseudo-boolean functions. In *International Conference on Parallel Problem Solving from Nature.* Springer, 181–193.

[27] J. L. Walsh. 1923. A Closed Set of Normal Orthogonal Functions. *American Journal of Mathematics* 45, 1 (1923), 5–24.

[28] Darrell Whitley, Renato Tinós, and Francisco Chicano. 2018. Optimal Neuron Selection and Generalization: NK Ensemble Neural Networks. In *Parallel Problem Solving from Nature – PPSN XV*, Anne Auger, Carlos M. Fonseca, Nuno Lourenço, Penousal Machado, Luís Paquete, and Darrell Whitley (Eds.). Springer International Publishing, Cham, 449–460.

[29] Martin Zaefferer. 2018. *Surrogate Models for Discrete Optimization Problems.* Ph.D. Dissertation. Technischen Universitat Dortmund.

[30] Martin Zaefferer, Jörg Stork, and Thomas Bartz-Beielstein. 2014. Distance measures for permutations in combinatorial efficient global optimization. In *International Conference on Parallel Problem Solving from Nature.* Springer, 373–383.

[31] Martin Zaefferer, Jörg Stork, Martina Friese, Andreas Fischbach, Boris Naujoks, and Thomas Bartz-Beielstein. 2014. Efficient Global Optimization for Combinatorial Problems. In *GECCO.*