The Kalai-Smorodinsky solution for many-objective Bayesian optimization

M. Binois MBINOIS@MCS.ANL.GOV
Mathematics and Computer Science Division, Argonne National Laboratory, Lemont, USA

V. Picheny VICTOR@PROWLER.IO
PROWLER.io, Cambridge, UK and
MIAT, Université de Toulouse, INRA, Castanet-Tolosan, France

P. Taillandier PATRICK.TAILLANDIER@INRA.FR
MIAT, Université de Toulouse, INRA, Castanet-Tolosan, France

A. Habbal HABBAL@UNICE.FR
Université Côte d'Azur, Inria, CNRS, LJAD, UMR 7351, Parc Valrose, 06108 Nice, France

Abstract

An ongoing aim of research in multiobjective Bayesian optimization is to extend its applicability to a large number of objectives. While coping with a limited budget of evaluations, recovering the set of optimal compromise solutions generally requires numerous observations and is less interpretable since this set tends to grow larger with the number of objectives. We thus propose to focus on a specific solution originating from game theory, the Kalai-Smorodinsky solution, which possesses attractive properties. In particular, it ensures equal marginal gains over all objectives. We further make it insensitive to a monotonic transformation of the objectives by considering the objectives in the copula space. A novel tailored algorithm is proposed to search for the solution, in the form of a Bayesian optimization algorithm: sequential sampling decisions are made based on acquisition functions that derive from an instrumental Gaussian process prior. Our approach is tested on three problems with respectively four, six, and ten objectives. The method is available in the R package `GPGame` available on CRAN at `https://cran.r-project.org/package=GPGame`.

Keywords: Gaussian process, Game theory, Stepwise uncertainty reduction

1. Introduction

Bayesian optimization (BO) is recognized as a powerful tool for global optimization of expensive objective functions and now has a broad range of applications in engineering and machine learning (see, for instance, Shahriari et al., 2016). A typical example is the calibration of a complex numerical model: to ensure that the model offers an accurate representation of the system it emulates, some input parameters need to be chosen so that some model outputs match real-life data (Walter and Pronzato, 1997). Another classical application is the optimization of performance of large machine learning systems via the tuning of its hyperparameters (Bergstra et al., 2011). In both cases, the high cost of evaluating performance drastically limits the optimization budget, and the high sample-efficiency of BO compared with alternative black-box optimization algorithms makes it highly competitive.

Many black-box problems, including those cited, involve several (or many) performance metrics that are typically conflicting, so no common minimizer to them exists. This is the

field of multiobjective optimization, where one aims at minimizing simultaneously a set of p objectives with respect to a set of input variables over a bounded domain $\mathbb{X} \subset \mathbb{R}^d$:

$$\min_{\mathbf{x}\in\mathbb{X}} \left\{ y^{(1)}(\mathbf{x}), \ldots, y^{(p)}(\mathbf{x}) \right\}. \tag{1}$$

We assume that the $y^{(i)} : \mathbb{X} \to \mathbb{R}$ functions are expensive to compute, nonlinear, and potentially observed in noise. Defining that a point $\mathbf{x}^*$ dominates another point $\mathbf{x}$ if *all* its objectives are better, the usual goal of multiobjective optimization (MOO) is to uncover the Pareto set, that is, the subset $\mathbb{X}^* \subset \mathbb{X}$ containing all the Pareto nondominated solutions:

$$\forall \mathbf{x}^* \in \mathbb{X}^*, \forall \mathbf{x} \in \mathbb{X}, \exists k \in \{1, \ldots, q\} \text{ such that } y^{(k)}(\mathbf{x}^*) \leq y^{(k)}(\mathbf{x}).$$

The image of the Pareto set in the objective space, $\mathcal{P}^* = \{y^{(1)}(\mathbb{X}^*), \ldots, y^{(q)}(\mathbb{X}^*)\}$, is called the Pareto front. Since $\mathbb{X}^*$ is in general not finite, most MOO algorithms aim at obtaining a discrete representative subset of it.

One way of solving this problem is by scalarization, aggregating all objective functions via weights. This allows the use of any technique dedicated to optimization of expensive black-boxes, generally using surrogates (see, e.g., Knowles, 2006; Zhang et al., 2010). Nevertheless, the aggregated function may become harder to model than its components, and the relation between weights and the corresponding Pareto optimal solution is generally not trivial. It may even be harder for hyperparameter optimization tasks (Smithson et al., 2016) that have no physical intuition backing weights. In addition, these issues worsen with more objectives, advocating taking objectives separately.

Most of the well-established Pareto-based algorithms, such as evolutionary (Deb et al., 2002; Chugh et al., 2017), descent-based (Das and Dennis, 1998), or Bayesian optimization (Wagner et al., 2010; Hernández-Lobato et al., 2016a), perform well on two or three objectives problems but poorly when $p \geq 4$, the so-called many-objective optimization (MaO). Indeed, difficulties inherent to the higher dimension of the objective space (Ishibuchi et al., 2008) arise, such as the exponential increase in the number of points necessary to approximate the –possibly singular– Pareto front hyper-surface and the difficulties in its graphical representation. Moreover, one has to deal with a more MaO intrinsic problem, which is that almost any admissible design becomes nondominated.

To circumvent these issues, Kukkonen and Lampinen (2007) advocated the use of ranks instead of nondomination and Bader and Zitzler (2011) contributions to the hypervolume. However, such algorithms require many objective evaluations and hence do not adapt well to expensive black boxes. In addition, they do not solve the problem of exploiting the resulting very large Pareto set. Some authors proposed methods to reduce the number of objectives to a manageable one (Singh et al., 2011), to use the so-called decomposition-based approaches (Asafuddoula et al., 2015), or to rely on a set of fixed and adaptive reference vectors (Chugh et al., 2018). Remarkably, while the difficulty of representing the Pareto front is highlighted, the question of selecting a particular solution on the Pareto front is mostly left to the user.

Our present proposition amounts to searching for a single, but remarkable in some sense, solution. To do so, we adopt a game-theoretic perspective, where the selected solution arises as an equilibrium in a (non) cooperative game played by p virtual agents who own the respective p objectives (Désidéri et al., 2014). In the following, we show that the so-called Kalai-Smorodinsky (KS) solution (Kalai and Smorodinsky, 1975) is an appealing alternative.

Intuitively, solutions at the "center" of the Pareto front are preferable compared with those at extremities –which is precisely what the KS solution consists of. Yet, the notion of center is arbitrary, since transforming the objectives (nonlinearly, e.g., with a log scale) would modify the Pareto front shape and affect the decision. Still, most MOO methods are sensitive to a rescaling of the objective, which is not desirable (Svenson, 2011). Our second proposition is to make the KS solution insensitive to monotone transformations, by operating in the copula space (Nelsen, 2006; Binois et al., 2015).

Uncovering the KS solution is a nontrivial task for which, to our knowledge, no algorithm is available in an expensive black-box framework. Our third contribution is a novel Gaussian-process-based algorithm, building on the stepwise uncertainty reduction (SUR) paradigm (Bect et al., 2012). SUR, which is closely related to information-based approaches (Hennig and Schuler, 2012; Hernández-Lobato et al., 2016b), has proven to be efficient for solving single- and multiobjective optimization problems (Villemonteix et al., 2009; Picheny, 2013), while enjoying strong asymptotic properties (Bect et al., 2016).

The rest of the paper is organized as follows. Section 2 describes the KS solution and its extension in the copula space. Section 3 presents the Bayesian optimization algorithm developed to find KS solutions. Section 4 reports empirical performances of our approach on three challenging problems with respectively four, six and ten objectives. Section 5 summarizes our conclusions and briefly discusses areas for future work.

2. The Kalai-Smorodinsky solution

2.1 The standard KS solution

The Kalai-Smorodinsky solution was first proposed by Kalai and Smorodinsky in 1975 as an alternative to the Nash bargaining solution in cooperative bargaining. The problem is as follows: Starting from a *disagreement* or *status quo* point $\mathbf{d}$ in the objective space, the players aim at maximizing their own benefit while moving from $\mathbf{d}$ toward the Pareto front (i.e., the efficiency set). The KS solution is of egalitarian inspiration (Conley and Wilkie, 1991) and states that the selected efficient solution should yield equal benefit ratio to all the players. Indeed, given the utopia (or ideal, or shadow) point $\mathbf{u} \in \mathbb{R}^p$ defined by

$$u^{(i)} = \min_{\mathbf{x} \in \mathbb{X}^*} \; y^{(i)}(\mathbf{x}),$$

selecting any compromise solution $\mathbf{s}$ would yield, for objective i, a benefit ratio

$$r^{(i)}(\mathbf{s}) = \frac{d^{(i)} - y^{(i)}(\mathbf{s})}{d^{(i)} - u^{(i)}}.$$

Notice that the benefit from staying at $\mathbf{d}$ is zero, while it is maximal for the generically unfeasible choice $\mathbf{s} = \mathbf{u}$. The KS solution is the Pareto optimal choice $\mathbf{s}^*$ for which all the benefit ratios $r^{(i)}(\mathbf{s})$ are equal. One can easily show that $\mathbf{s}^*$ is the intersection point of the Pareto front and the line $(\mathbf{d}, \mathbf{u})$ (Figure 1, left).

We use here the extension of the KS solution to discontinuous fronts proposed in Hougaard and Tvede (2003) under the name *efficient maxmin solution*. Indeed, for discontinuous fronts the intersection with the $(\mathbf{d}, \mathbf{u})$ line might not be feasible, so there is a necessary trade-off

between Pareto optimality and centrality. The efficient maxmin solution is defined as the Pareto-optimal solution that maximizes the smallest benefit ratio among players, that is:

$$\mathbf{s}^{**} \in \arg\max_{\mathbf{y}\in\mathcal{P}^*} \min_{1\leq i\leq p} r^{(i)}(\mathbf{s}). \tag{2}$$

It is straighforward that when the intersection is feasible, then $\mathbf{s}^*$ and $\mathbf{s}^{**}$ coincide.

Figure 1 shows $\mathbf{s}^{**}$ in two situations when the feasible space is nonconvex. $\mathbf{s}^{**}$ is always on the Pareto front (hence not on the $(\mathbf{d}, \mathbf{u})$ line). In the central graph, it corresponds to the point on the front closest to the $(\mathbf{d}, \mathbf{u})$ line, which is intuitive. In the right graph, the closest point on the front (on the right hand side of the line) is actually a poor solution, as there exists another point with identical performance in y_2 but much better in terms of y_1 (on the right hand side of the line): the latter corresponds to $\mathbf{s}^{**}$.

In the following, we refer indifferently to $\mathbf{s}^*$ (if it exists) and $\mathbf{s}^{**}$ as the KS solution. Note that this definition also extends to discrete Pareto sets, which will prove useful in Section 3.

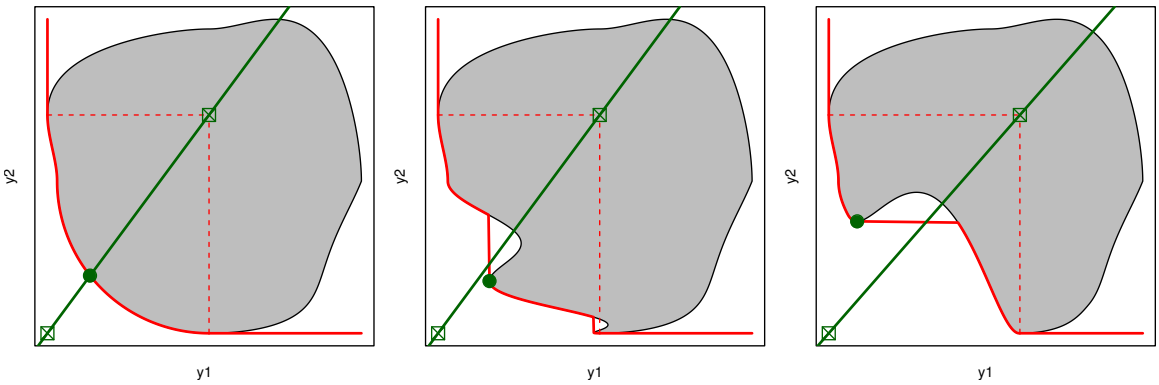

Figure 1: KS solution for a continuous (left) and discontinuous (center and right) $\mathcal{P}^*$. The KS is shown with a green disk, along with the shadow and nadir points (squares). The shaded area shows the feasible space, and the Pareto front is depicted in red.

For $p \geq 3$, the KS solution, defined as the intersection point above, fulfills some of the bargaining axioms: Pareto optimality, affine invariance, and equity in benefit ratio. Moreover, for $p = 2$, KS is the *unique* solution that fulfills all the bargaining solution axioms that are Pareto optimality, symmetry, affine invariance, and restricted monotonicity (Kalai and Smorodinsky, 1975).

It is particularly attractive in a many-objective context since it scales naturally to a large number of objectives and returns a single solution, avoiding the difficulty of exploring and approximating large p-dimensional Pareto fronts–especially with a limited number of observations.

The KS solution is known to depend strongly on the choice of the disagreement point $\mathbf{d}$. A standard choice is the nadir point N given by $N_i = \max_{\mathbf{x}\in\mathbb{X}^*} y^{(i)}(\mathbf{x})$. Some authors introduced alternative definitions, called extended KS, to alleviate the dependence on choice of $\mathbf{d}$ (Bozbay et al., 2012), for instance by taking as disagreement point the Nash equilibrium

arising from a noncooperative game, but such a choice would need a prebargaining split of the decision variable $\mathbf{x}$ among the p players.

Incorporating preferences or constraints In many situations, the end user may discard solutions with extreme values and actually solve the following problem:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{X}} \quad & \left\{y^{(1)}(\mathbf{x}), \dots, y^{(p)}(\mathbf{x})\right\} \\ \text{s.t.} \quad & y^{(i)}(\mathbf{x}) \leq c_i, \qquad i \in J \subset [1, \dots, p]\,, \end{aligned} \tag{3}$$

with c_i's predefined constants. Incorporating those constraints (or *preferences*, Junker et al., 2004; Thiele et al., 2009) is straightforward in our case, simply by using $\mathbf{c}$ as the disagreement point.

2.2 Robust KS using copulas

A drawback of KS is that it is not invariant under a monotonic (nonaffine) transformation of the objectives (this is not the case of the Pareto set, since a monotonic transformation preserves ranks, hence domination relations). In a less-cooperative framework, some players could be tempted to rely on such transformations to influence the choice of a point on the Pareto front. It may even be involuntary, for instance when comparing objectives of different natures.

To circumvent this problem, we use copula theory, which has been linked to Pareto optimality by Binois et al. (2015). In short, from a statistical point of view, the Pareto front is related to the zero level-line ∂F_0 of the multivariate cumulative density function F_Y of the objective vector $Y = (y_1(X), \dots, y_p(X))$ with X any random vector with support equal to $\mathbb{X}$. That is, $\partial F_0 = \lim_{\alpha\to 0^+} \{\mathbf{y} \in \mathbb{R}^p, F_Y(\mathbf{y}) = \mathbb{P}(Y_1 \leq y_1, \dots Y_p \leq y_p) = \alpha\}$. Notice that solving the MaO problem by random sampling (with X uniformly distributed), as used for hyperparameter optimization by Bergstra and Bengio (2012), amounts to sample from F_Y. Extreme-level lines of F_Y actually indicate how likely it is to be close to the Pareto front (or on it if the Pareto set has a nonzero probability mass).

A fundamental tool for studying multivariate random variables is the copula. A copula C_Y is a function linking a multivariate cumulative distribution function (CDF) to its univariate counterparts, such that for $\mathbf{y} \in \mathbb{R}^p$, $F_Y(\mathbf{y}) = C_Y(F_1(y_1), \dots, F_p(y_p))$ with $F_i = \mathbb{P}(Y_i \leq y_i)$, $1 \leq i \leq p$ (Nelsen, 2006). When F_Y is continuous, C_Y is unique, from Sklar's theorem (Nelsen, 2006, Theorem 2.3.3). As shown by Binois et al. (2015), learning the marginal CDFs F_i's and extreme levels of the copula C_Y amounts to learning the Pareto front.

A remarkable property of copulas is their invariance under monotone increasing transformations of univariate CDFs (Nelsen, 2006, Theorem 2.4.3). Extending their proof to the p-dimensional case, suppose that $g_1, \dots, g_p$ are strictly increasing transformations on the ranges of $Y_1, \dots, Y_p$, respectively; hence they are invertible. Denote $G_1, \dots, G_p$ the marginal distribution functions of $\mathbf{g}(Y)$. It then holds that $G_i(y_i) = \mathbb{P}(g_i(Y_i) \leq y_i) = \mathbb{P}(Y_i \leq g_i^{-1}(y_i)) = F_i(g_i^{-1}(y_i))$. Then $C_Y(y_1, \dots, y_p) = \mathbb{P}(g_1(Y_1) \leq y_1, \dots, g_p(Y_p) < y_p) = \mathbb{P}(Y_1 \leq g_1^{-1}(y_1), \dots, Y_p \leq g_p^{-1}(y_p)) = C(F_1(g_1^{-1}(y_1)), \dots, F_p(g_p^{-1}(y_p))) = C_{\mathbf{g}(Y)}(G_1(y_1), \dots, G_p(y_p))$.

Now, our proposition is to consider the KS solution in the copula space, that is, taking $F_1, \dots, F_p$ as objectives instead of $y_1, \dots, y_p$. This "copula-KS solution" (henceforth CKS) is Pareto-efficient and invariant to any monotonic transformation of the objectives. CKS

depends on the instrumental law of X. In the following, we always assume that X is uniformly distributed over $\mathbb{X}$ and defer elements of discussion to the conclusion.

In addition, in the copula space the utopia point is always $(0, \ldots, 0)$. While the nadir remains unknown, the point $(1, \ldots, 1)$ may serve as an alternative disagreement point, since it corresponds to the worst solution for each objective. This removes the difficult task of learning the $(\mathbf{d}, \mathbf{u})$ line, at the expense of learning the marginal distribution and the copula function. Unless additional information is available about the marginal distributions and copula function, empirical estimators can be used; see, for instance, Omelka et al. (2009).

We finally remark that when $\mathbb{X}$ is finite, our proposed solution amounts to work on ranks: the CKS solution is simply the Pareto optimal solution with the closest ranks over all objectives.

2.3 Illustration

Let us consider a classical two-variable, biobjective problem from the multiobjective literature (P1; see Parr (2013)). We first compute the two objectives on $2,000$ uniformly sampled points in $\mathbb{X} := [0, 1]^2$, out of which the feasible space, Pareto front, and KS solution are extracted (Figure 2, left). The KS solution has a visibly central position in the Pareto front, which makes it a well-balanced compromise.

Applying a log transformation of the first objective does not change the Pareto set but modifies here substantially the shape of the Pareto front (from convex to concave) and the KS solution, leading to a different compromise (Figure 2, center).

Both original and rescaled problems share the same image in the copula space (Figure 2, right), which provides a third compromise. Seen from the original space, the CKS solution seems here to favor the first objective: this is due to the high density of points close to the minimum on Y_1 (Figure 2, left, top histogram). It is, however, almost equivalent to the KS solution under a log transformation of Y_1. From a game perspective, the two players agree on a solution with equal ranks: here roughly the 100th best ($F_1 \approx F_2 \approx 0.25$) out of $2,000$, independently of the gains in terms of objective values.

3. A Bayesian optimization algorithm to find KS solutions

Computing the KS and CKS solutions is a challenging problem. It requires for KS learning the ideal and disagreement points $\mathbf{u}$ and $\mathbf{d}$ (which are challenging problems on their own; see, for instance, Bechikh et al., 2010) and for CKS the marginals and copula, as well as the part of the Pareto front that intersects the $(\mathbf{d}, \mathbf{u})$ line.

An additional difficulty arises when the objective functions cannot be computed exactly but only through a noisy process. In this section, we consider that one has access to observations of the form

$$\mathbf{f}_i = \mathbf{y}(\mathbf{x}_i) + \boldsymbol{\varepsilon}_i, \tag{4}$$

where $\boldsymbol{\varepsilon}_i$ is a zero-mean noise vector with independent components of variances $(\tau_i^{(1)}, \ldots, \tau_i^{(p)})$. Our approach readily adapts to the deterministic case by setting $\forall j : \tau_i^{(j)} = 0$.

We assume here that all objectives are collected at the same time (using a single black-box), hence sharing the experimental points $\mathbf{x}_i$. The case of several black-boxes, as presented by Hernández-Lobato et al. (2016b), is not considered here and is deferred to future work.

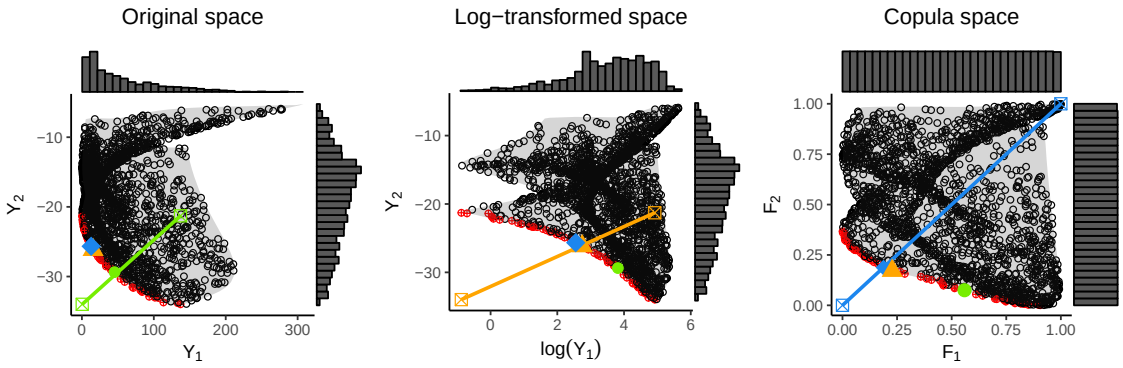

Figure 2: KS (green disk), KS in log-scale (orange triangle) and CKS (blue diamond) solutions for a biobjective problem, based on $2,000$ uniformly sampled designs, shown in the objective space. The black circles show all the dominated values of the grid and the red crossed circles the Pareto-optimal ones. The shaded area shows the feasible space. Marginal objective densities are reported on the corresponding axes. The $(\mathbf{d}, \mathbf{u})$ lines are shown with matching colors.

3.1 Elements of Bayesian optimization

For our algorithm, we consider a classical BO framework, where independent Gaussian process (GP) priors are put on the objectives:

$$\forall i \in 1, \ldots, p, \quad Y^{(i)}(.) \sim \mathcal{GP}\left(\mu^{(i)}(.), \sigma^{(i)}(.,.)\right), \tag{5}$$

where the mean $\mu^{(i)}$ and covariance $\sigma^{(i)}$ have predetermined parametric forms whose parameters are estimated by maximum likelihood (Rasmussen and Williams, 2006). Conditioning on a set of observations $\{\mathbf{f}_1, \ldots, \mathbf{f}_n\}$, GPs provide flexible response fits associated with uncertainty estimates. They enable operating sequential design decisions via an *acquisition function* $J(\mathbf{x})$, which balances between exploration and exploitation in seeking global optima. Hence, the design consists of a first set of n_0 observations generated by using a space-filling design, generally from a variant of Latin hypercube design (LHD, McKay et al., 1979) to obtain a first predictive distribution of $Y^{(i)}(.)$, and a second set of sequential observations chosen as

$$\mathbf{x}_{n+1} \in \arg\max_{\mathbf{x}\in\mathbb{X}} J(\mathbf{x}), \quad (n \geq n_0). \tag{6}$$

GP equations are deferred to Appendix A. In the following, we use the subscript n to denote quantities conditional on the set of n observations (e.g., $Y_n^{(i)}$, $\mu_n^{(i)}$ or $\sigma_n^{(i)}$).

3.2 Stepwise uncertainty reduction

Now, we wish to design an acquisition $J(\mathbf{x})$ tailored to our problem. To do so, we follow a step-wise uncertainty reduction (SUR) approach, as follows. Define first an uncertainty measure $\Gamma(\mathbf{Y}_n)$, which expresses a lack of knowledge regarding a quantity of interest. In the present case, this quantity of interest is the KS solution (either in the objective space or in

the design space). The SUR strategy aims at minimizing greedily, at each step, the expected uncertainty at the next step (so that eventually the quantity of interest becomes exactly known).

More precisely, the SUR sampling criteria is defined as

$$J(\mathbf{x}) = \mathbb{E}_{\mathbf{Y}_n(\mathbf{x})}\left[\Gamma(\mathbf{Y}_{n,\mathbf{x}})\right], \tag{7}$$

where $\mathbf{Y}_{n,\mathbf{x}}$ is the GP conditioned on $\{\mathbf{y}(\mathbf{x}_1), \ldots, \mathbf{y}(\mathbf{x}_n), \mathbf{y}(\mathbf{x})\}$ and $\mathbb{E}_{\mathbf{Y}_n(\mathbf{x})}$ denotes the expectation taken over $\mathbf{Y}_n(\mathbf{x})$. The next observation is the one that minimizes the residual expected uncertainty:

$$\mathbf{x}_{n+1} \in \arg\min_{\mathbf{x}\in\mathbb{X}} J(\mathbf{x}). \tag{8}$$

To choose Γ, we follow an approach similar to the one proposed by Picheny et al. (2016) to solve Nash equilibria problems. Let us first denote by $\Psi : \mathbb{Y} \to \mathbb{R}^p$ the application that associates a KS or CKS solution with any multivariate function $\mathbf{y}$. If we consider the random process $\mathbf{Y}_n$, $\Psi(\mathbf{Y}_n)$ is a random vector (of unknown distribution) with second moment $\mathrm{cov}\left(\Psi(\mathbf{Y}_n)\right)$.

Intuitively, $\mathrm{cov}\left(\Psi(\mathbf{Y}_n)\right)$ tends to the null matrix when all the components of $\Psi(\mathbf{Y}_n)$ become known accurately. This situation may happen only when $\mathbf{Y}_n$ has little variability in the region of the equilibrium (*exploitation* steps have been performed) and when there is no subset B of $\mathbb{X}$ such that $\mathbf{Y}_n(B)$ has a large variability (*exploration* steps have been performed).

A classical measure of variability of a vector is the determinant of its covariance matrix (Fedorov, 1972); hence, we represent the uncertainty regarding our knowledge of the equilibrium as

$$\Gamma(\mathbf{Y}_n) = \det\left[\mathrm{cov}\left(\Psi(\mathbf{Y}_n)\right)\right]. \tag{9}$$

3.3 Computational aspects

3.3.1 Computing and optimizing the SUR criterion

Because of the strong nonlinearity of Ψ, no closed-form expression for $J(\mathbf{x})$ is available, and one must rely on Monte Carlo approaches. We employ here the *fast update of conditional simulation ensemble* algorithm proposed by Chevalier et al. (2015), as detailed below.

We first discretize the design space $\mathbb{X}$, for example using a fine grid or a low discrepancy sequence (Niederreiter, 1988) (see also next subsection). We call $\mathbb{X}_\star$ the discrete set and N its size. Let $\mathbf{x}$ be a candidate observation point.

Let $\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_M$ be independent drawings of $\mathbf{Y}$ ($\{\mathbb{X}_\star, \mathbf{x}\}$) (each $\boldsymbol{\mathcal{Y}}_i \in \mathbb{R}^{(N+1)\times p}$), generated by using the posterior Gaussian distribution of Equation (5). Since $\boldsymbol{\mathcal{Y}}_i(\mathbb{X}_\star)$ is discrete, its corresponding KS solution $\Psi(\boldsymbol{\mathcal{Y}}_i(\mathbb{X}_\star))$ can be computed by exhaustive search. The following empirical estimator of $\Gamma(\mathbf{Y})$ is then available:

$$\hat{\Gamma}\left(\boldsymbol{\mathcal{Y}}_1(\mathbb{X}_\star), \ldots, \boldsymbol{\mathcal{Y}}_M(\mathbb{X}_\star)\right) = \det\left[\mathbf{Q}_{\mathcal{Y}}\right],$$

with $\mathbf{Q}_{\mathcal{Y}}$ the sample covariance of $\Psi(\boldsymbol{\mathcal{Y}}_1(\mathbb{X}_\star)), \ldots, \Psi(\boldsymbol{\mathcal{Y}}_M(\mathbb{X}_\star))$.

Now, let $\boldsymbol{\mathcal{F}}_1, \ldots, \boldsymbol{\mathcal{F}}_K$ be independent drawings of $\mathbf{Y}(\mathbf{x})$ from the posterior Gaussian distribution (5). As shown by Chevalier et al. (2015), drawings of $\mathbf{Y}|\boldsymbol{\mathcal{F}}_i$ can be obtained

efficiently from $\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_M$, using

$$\mathcal{Y}_j^{(i)}|\mathcal{F}_k^{(i)} \quad = \quad \mathcal{Y}_j^{(i)} + \boldsymbol{\lambda}^{(i)}(\mathbf{x}) \left(\mathcal{F}_k^{(i)} - \mathcal{Y}_j^{(i)}(\mathbf{x}) \right), \tag{10}$$

with $1 \leq i \leq p$, $1 \leq j \leq M$, $1 \leq k \leq K$ and

$$\boldsymbol{\lambda}^{(i)}(\mathbf{x}) = \frac{1}{\boldsymbol{\sigma}_n^{(i)}(\mathbf{x}, \mathbf{x})} \left[\boldsymbol{\sigma}_n^{(i)}(\mathbb{X}_{\star 1}, \mathbf{x}), \ldots, \boldsymbol{\sigma}_n^{(i)}(\mathbb{X}_{\star N}, \mathbf{x}) \right].$$

Notice that $\boldsymbol{\lambda}^{(i)}(\mathbf{x})$ may be computed only once for all $\mathcal{Y}_j^{(i)}(\mathbf{x})$. This step is illustrated in Figure 3.

An estimator of $J(\mathbf{x})$ is obtained by using the empirical mean:

$$\hat{J}(\mathbf{x}) = \frac{1}{K} \sum_{i=1}^{K} \hat{\Gamma} \left(\boldsymbol{\mathcal{Y}}_1 | \mathcal{F}^i, \ldots, \boldsymbol{\mathcal{Y}}_M | \mathcal{F}^i \right).$$

A decisive advantage of this approach is that if we restrict $\mathbf{x}$ to belong to $\mathbb{X}_\star$, drawing $\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_M$–which has an $\mathcal{O}(N^3)$ complexity when using the standard decomposition procedure based on Cholesky, (see, e.g., Diggle and Ribeiro, 2007)–needs to be done only once, prior to minimizing of the acquisition function. Hence, although optimizing $J(\mathbf{x})$ over the continuous space $\mathbb{X}$ is definitely feasible, it requires completing the simulation over $\mathbb{X}_\star$ with $\mathbf{x}$, incurring an additional computational cost. We thus restrict the search to $\mathbb{X}_\star$ only.

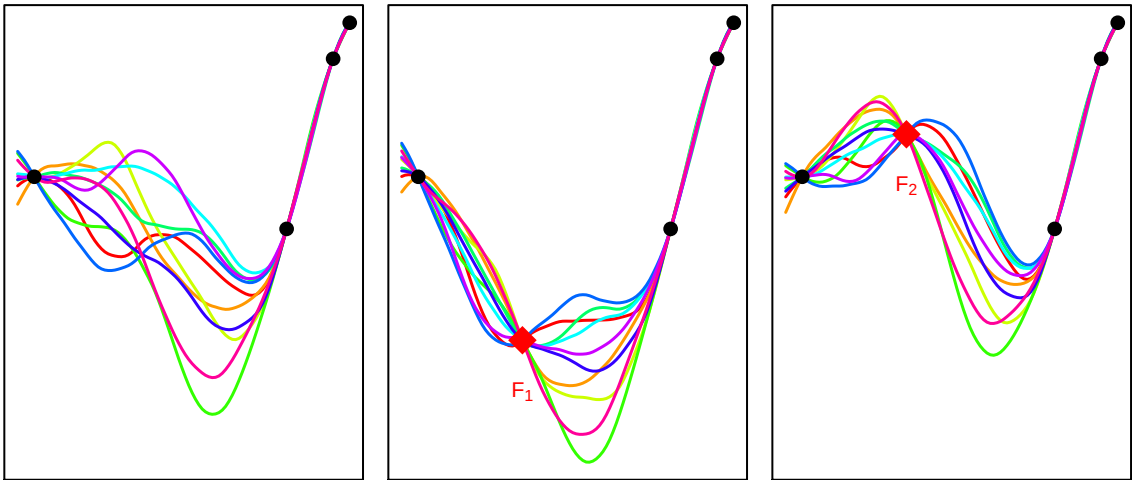

Figure 3: Left: initial drawings $\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_M$. Middle and right: same drawings, but conditioned further on an observation $\mathcal{F}^1$ (middle) and $\mathcal{F}^2$ (right).

3.3.2 Choosing integration points

Generating $\boldsymbol{\mathcal{Y}}_1, \ldots, \boldsymbol{\mathcal{Y}}_M$ limits $\mathbb{X}_\star$ in practice to at most a couple thousand points. In small dimension (say, $1 \leq d \leq 3$), $\mathbb{X}_\star$ may consist simply of a Cartesian grid or a dense space-filling

design (Niederreiter, 1988). In higher dimension, $\mathbb{X}_\star$ may not be large enough to cover $\mathbb{X}$ sufficiently. Accurate approximations of $J(\mathbf{x})$ can still be obtained, however as long as $\mathbb{X}_\star$ covers the influential parts of $\mathbb{X}$ with respect to J, that is, regions where (a) the variance of $\mathbf{Y}$ is large, (b) the KS or CKS solution is likely to be or (c) extremal Pareto optimal values of $\mathbf{Y}$ are likely (to estimate the nadir and shadow). Note that this last aspect is not necessary for CKS.

To design a set $\mathbb{X}_\star$ of limited size that contains those three components, we proceed as follow. First, a large space-filling set $\mathbb{X}_{\text{large}}$ is generated, and the GP posterior distribution is computed for each element of $\mathbb{X}_{\text{large}}$. Since we do not consider the joint distribution, this has a cost of only $\mathcal{O}(N^2)$. At the initial stage, we compute the KS solution of the posterior GP mean to obtain a crude estimate of the KS solution of the problem. For the other iterations, the set of simulated KS solutions used for the computation of J are available: $\Psi(\boldsymbol{\mathcal{Y}}_1), \ldots, \Psi(\boldsymbol{\mathcal{Y}}_M)$. Then, using the GP distribution, we compute the probability p_{box} for each element of $\mathbb{X}_{\text{large}}$ to belong to a box defined by the extremal values of the simulated KS (see Appendix A for formulas). A first set $\mathbb{X}_\star$ is obtained by sampling from $\mathbb{X}_{\text{large}}$ randomly with probabilities proportional to p_{box}. The set is likely to contain points in the vicinity of the KS solution.

To capture the extremal regions, we use the GP posteriors to compute for each objective the expected improvement (EI, Jones et al., 1998, see Appendix A) for all elements in $\mathbb{X}_{\text{large}}$. We add the p EI maximizers (i.e., points where the individual minima are most likely) to $\mathbb{X}_\star$ in order to emulate the shadow. We also compute the expected improvement of minus the objective, which we multiply by the probability of nondomination. The p maximizers of this criterion are also added to $\mathbb{X}_\star$ to emulate the nadir (KS only).

Both p_{box} and EI are likely to be high when the predictive variance is high. Hence, regions with high $\mathbf{Y}$ variability are accounted for indirectly.

$\mathbb{X}_\star$ is renewed at each iteration, firstly to include sequentially new information provided by the observations, but also to improve robustness (a critical region can be missed at an iteration but accounted for during the next).

For CKS, empirical estimates of the marginal distributions and copula may be too coarse with a limited sample such as $\mathbb{X}_\star$. in order to overcome this problem while limiting the computational cost, auxiliary points are added with values taken from the GP means conditioned on pseudo-observations. Details are deferred to Appendix A.

4. Results

This section details numerical experiments on three test problems: one classical toy problem from the multiobjective literature, a problem of hyperparameter tuning and the calibration of a complex simulator. All experiments were conducted in `R` (R Core Team, 2018), by using the dedicated package `GPGame`; see the work of Picheny and Binois (2018) for details.

4.1 Synthetic problems

As a proof of concept, we consider the DTLZ2 function (Deb et al., 2002), with five variables and four objectives, that has a concave dome-shaped Pareto front. We consider a finite candidate set $\mathbb{X}$ with 10^7 elements (uniformly distributed in $[0, 1]^5$), which allows the computation of reference KS and CKS solutions.

For both KS and CKS, we use $n_0 = 50$ and an optimized LHD, followed by 50 infill points. The set of potential candidate $\mathbb{X}_{\text{large}}$ is of size 10^5, renewed every iteration, out of which 800 points are selected as described in Section 3.3.2 for $J(\mathbf{x})$ computation and optimization, which was found as a satisfactory trade-off between speed and accuracy.

Results for one run are given in Figure 4 in the form of projections on the marginal 2D spaces. Notice the central location of the KS point on this problem (Figure 4, first row), while the CKS point leans toward areas of larger densities (for instance, second line, third and fifth plots, CKS is close to the upper left corner). For KS, new points are added close to the reference solution, but some are also more exploratory, in particular near the individual minima to reduce uncertainty on the $(\mathbf{d}, \mathbf{u})$ line. For CKS, the behavior is more local, with points added mostly around the reference solution.

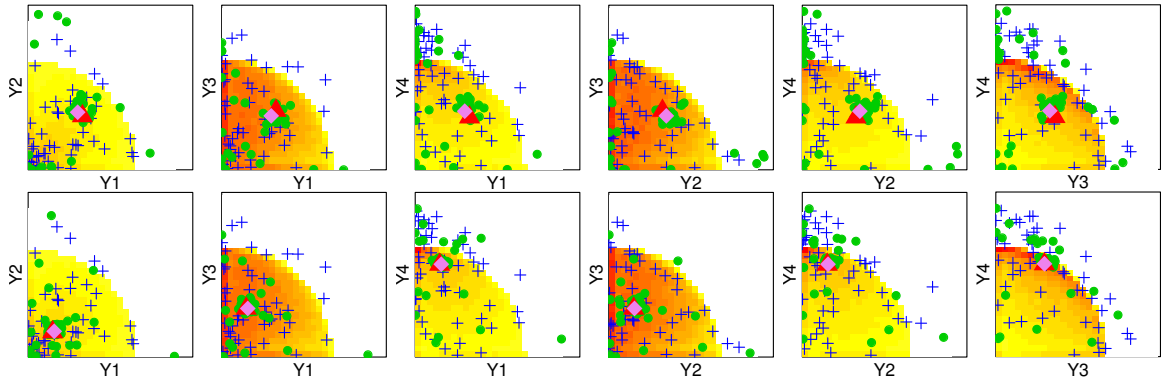

Figure 4: Results of one run for KS (top) and CKS (bottom), represented by using marginal 2D projections. The blue crosses are the initial design points, the green points the added designs, the red triangles the target true equilibria and pink diamonds the predicted ones. The heatmap represents the density of Pareto front points on the projected spaces.

Convergence results are provided in Figure 5 in terms of Euclidean distance to the actual equilibrium, computed by using a 10^7-point grid. The decrease is sharper for CKS than KS, indicating that estimating the KS solution is harder, presumably since it requires estimated extremes of the Pareto front. In both cases and all runs, a solution close to the reference one is found despite the restricted budget. Fine convergence is not yet achieved, however which may require additional budget as well as a finer discretization of the search space.

4.2 Training of a convolutional neural network

A growing need for BO methods emerges from machine learning applications, to replace manual tuning of hyperparameters of increasingly complex methods. One such example is with hyperparameters controlling the structure of a neural network. Since such methods are integrated in products, accuracy is not the only concern; and additional objectives have to be taken into account, such as prediction times.

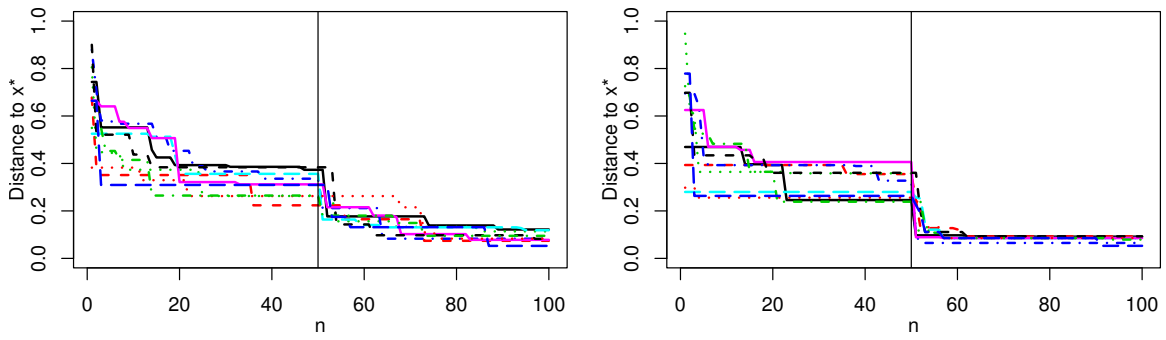

Figure 5: Results of 10 runs of KS and CKS, Euclidean distance to the reference solution as a function of number of evaluations. Thin vertical lines mark the start of the sequential procedure.

4.2.1 Problem description

We consider here the training of a convolutional neural network (CNN) on the classical MNIST data (LeCun et al., 1998), with 60,000 handwritten digits for training and an extra 10,000 for testing. We use the `keras` package (Allaire and Chollet, 2018) to interface with the high-level neural networks API Keras (Chollet et al., 2015) to create and train a CNN. We follow a common structure for such a task, represented in Figure 6, with a first 2D convolutional layer, a first max pooling layer, then a second 2D convolutional layer and a second max pooling layer. Max pooling consists in keeping only the max over a window, introducing a small amount of translational invariance. Dropout, that is, randomly cutting off some neurons to increase robustness, is then applied before flattening to a dense layer, followed by another dropout before the final dense layer. Because of the dropout phases, the performance is random. This is handled by repeating five times each experiment, which takes up to 30 minutes on a desktop with a 3.2 Ghz quad-core processor and 4 Go of RAM.

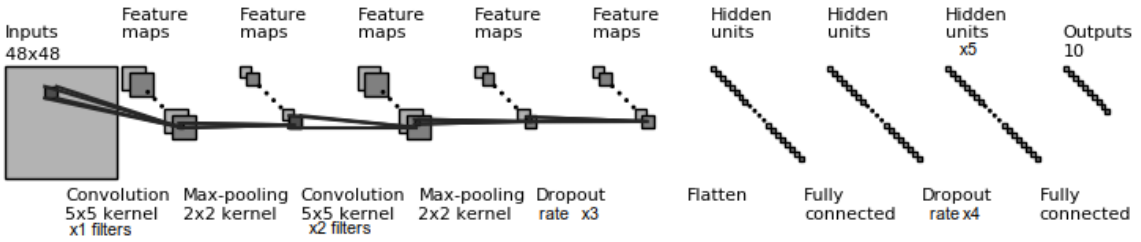

Figure 6: Architecture of the CNN on the MNIST data.

The five hyperparameters to tune, detailed in Table 1 in Appendix C, along with their range of variation, are the number of filters and dropout rates of each layer, plus the number of units of the last hidden layer.

The training of the CNN is performed based on the categorical cross-entropy, over 10 epochs. A validation data set is extracted from the training data to monitor overfitting, representing 20% of the initial size of the training data. The progress can be monitored by taking into account the accuracy (i.e., proportion of properly classified data) and cross-entropy on the training data, on the validation data or on the test data. Of these six possible objectives, we dropped training and validation accuracy since they are extremely correlated with the cross-entropy. We replaced them by the training time of the CNN, as well as the prediction time on the testing data. This latter is relevant, for instance, when using a pretrained CNN for a given task. The six objectives are summarized in Table 2 in Appendix C.

We take a total budget of 100 evaluations, split in half between initial LHD and sequential optimization using KS or CKS. GPs are trained by using Matérn 5/2 kernels with an estimated linear trend. In this case, 500 integration points were selected out of 10^5 possible candidates, renewed every iteration. The resulting time of each iteration is under a minute for KS and less than 10 minutes for CKS.

4.2.2 Results

Figure 7 shows the distribution of objective values explored during optimization. For the error and accuracy metrics, we observe first that most of the initial set of observations consist of values close to the optimum. Both KS and CKS searches attributed most of the sampling effort to such values, although CKS search also sampled over the whole range of variation of the objectives, which can be attributed to the learning of the marginals. For the training and test times, initial values are more uniformly spread, and both strategies concentrated their effort on the best half of the objective ranges. The KS solution dominates the CKS on the first four objectives, and conversely on the last two.

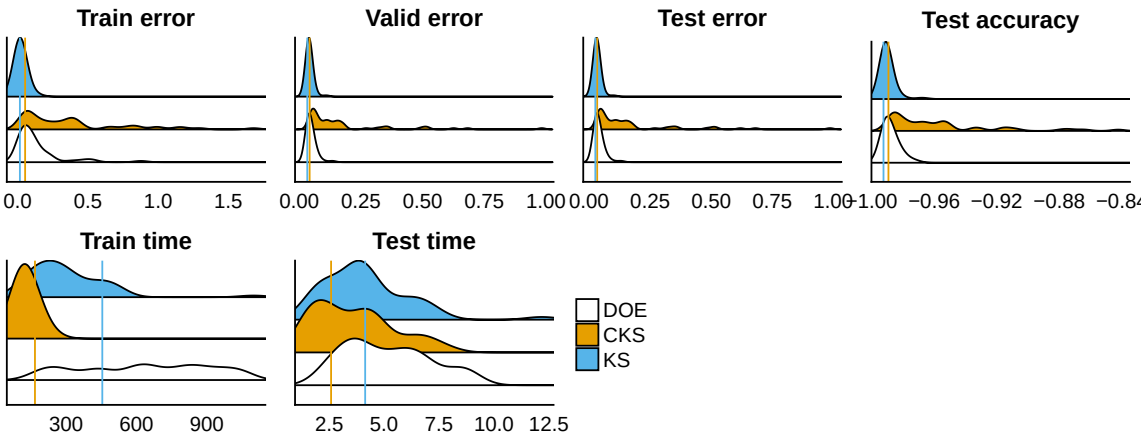

Figure 7: Marginal distributions of the 6 objectives of the CNN problem. Each density corresponds to 50 values. The vertical bars represent the identified solutions.

Figure 8 shows the two solutions in two radar plots (that can be seen as parallel coordinates, Li et al., 2017) using the original scale (left) and the scale of ranks (right), that is, the rank of the solution compared with the initial 50 observations. On the original scale, CKS clearly appears as a better trade-off even if it dominates on two-thirds of the objectives, since the performance gap (i.e., difference between the objective mininmum and solution value) is negligible on four objectives and substantial on the other two. Looking at ranks, we see that CKS offers a more balanced solution that accounts better for the heavy left tail of the first four objectives.

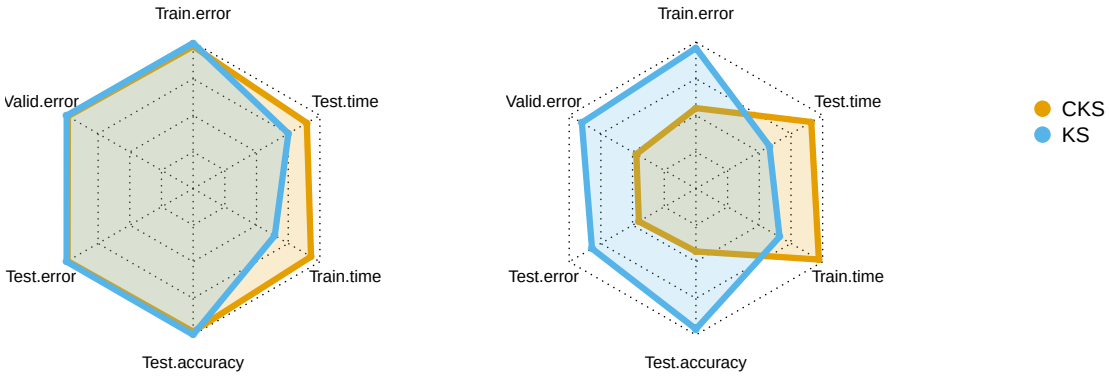

Figure 8: Comparison of the KS and CKS solutions on the CNN problem. Left: original scale; right: rank scale (with respect to the initial 50-point design only). The outer dotted line corresponds to the best performance, the center to the worst.

For now, the five variables are taken as continuous. But these results could be extended by increasing the number of variables, including categorical variables affecting more profoundly the structure of the network, for instance relying on the work of Roustant et al. (2018).

4.3 Calibration of an agent-based behavioral model

Model calibration (sometimes referred to as inverse problem) consists of adjusting input parameters so that the model outputs match real-life data. In this experiment, we consider the calibration of the li-BIM model (Taillandier et al., 2017), implemented under the GAMA platform (Taillandier et al., 2018), which exhibits several challenging features: stochasticity, high numerical cost (approximately 30 minutes per run on a desktop computer with a 3.60 GHz eight-core processor and 32 Go RAM), and a large number of outputs.

4.3.1 Problem description

The Li-BIM model simulates the behavior of occupants in a building. It is structured around the numerical modeling of the building and an evolved occupational cognitive model developed with a belief-desire-intention (BDI) architecture (Bourgais et al., 2017). It simulates several quantities that strongly depend on the occupants, such as thermal conditions, air quality,

lighting, etc. In the configuration considered here, three occupants are simulated over a period of one year.

In order to reproduce realistic conditions, 13 parameters can be tuned, related to either the occupant behavior or building characteristics (see Table 3 in Appendix C for details). Nine outputs ($G_1 \dots G_9$) should match some target values ($T_1 \dots T_9$), chosen based on records or surveys (see Table 4 in Appendix C). Since the model is stochastic, we consider as objectives the squared expected relative differences between the outputs and targets:

$$y_i = \log\left(\left[\mathbb{E}\left(\frac{G_i - T_i}{T_i}\right)\right]^2 + \delta\right) \tag{11}$$

The logarithm transformation is useful here to bring more constrast for values close to zero, attenuated by a small δ (in our experiments, $\delta = 0.01$). In practice, we use estimates based on eight repeated runs.

Note that such an objective focusses on the average behavior without considering the variability. As an alternative criterion, one may invert the square and expectation.

To solve this 13-variable 9-objective problem, we proceed as follow. An initial 100-point optimized LHD is generated, which is used to fit GPs (constant trend, Matérn 5/2 anisotropic covariance). From this initial design, both KS and CKS SUR strategies are conveyed independently with 100 additional points for each.

In addition, a third solution is sought by using KS with a partly prespecified disagreement point $\mathbf{d}$ to account for preferences (see Eq. 3), so that the average error on five of the outputs does not exceed a certain percentage (either 50% or 30%), the other outputs being unconstrained. To do so, we use

$$\tilde{d}_i = \min(N_i, c_i), \quad 1 \leq i \leq 9,$$

$N_i = \max_{\mathbf{x} \in \mathbb{X}^*} \; y^{(i)}(\mathbf{x})$ being the nadir i-th coordinate, and

$$c = \log([0.5, 0.5, +\infty, +\infty, 0.3, 0.5, +\infty, 0.5, +\infty]^2).$$

Importantly, the GPs are used to fit the expected values of the outputs of the model ($\mathbb{E}G_i$) instead of the objectives (y_i). This greatly improves the prediction quality of the GPs (as G_i is smoother than $(G_i - T_i)^2$), while allowing us to convey our strategy almost without modification: on Sections 3.2 and 3.3, the drawings $\boldsymbol{\mathcal{Y}}$ and $\boldsymbol{\mathcal{F}}$ are obtained by first generating drawings $\boldsymbol{\mathcal{G}}$ of G, then transforming them ($\log \boldsymbol{\mathcal{G}}^2$).

We used $1,000$ integration points, chosen from a 2×10^5 space-filling design, renewed at each iteration.

4.3.2 Results

The resulting designs of experiments and solutions are reported in graphical form in Figures 9 and 10. As a preliminary observation, of the 400 points computed during this experiment, only five were dominated. This result illustrates the exponential growth of Pareto sets with the number of objectives.

Figure 9 shows the distribution of objective values explored during optimization, along with the distribution corresponding to the initial (space-filling) set of experiments. On most

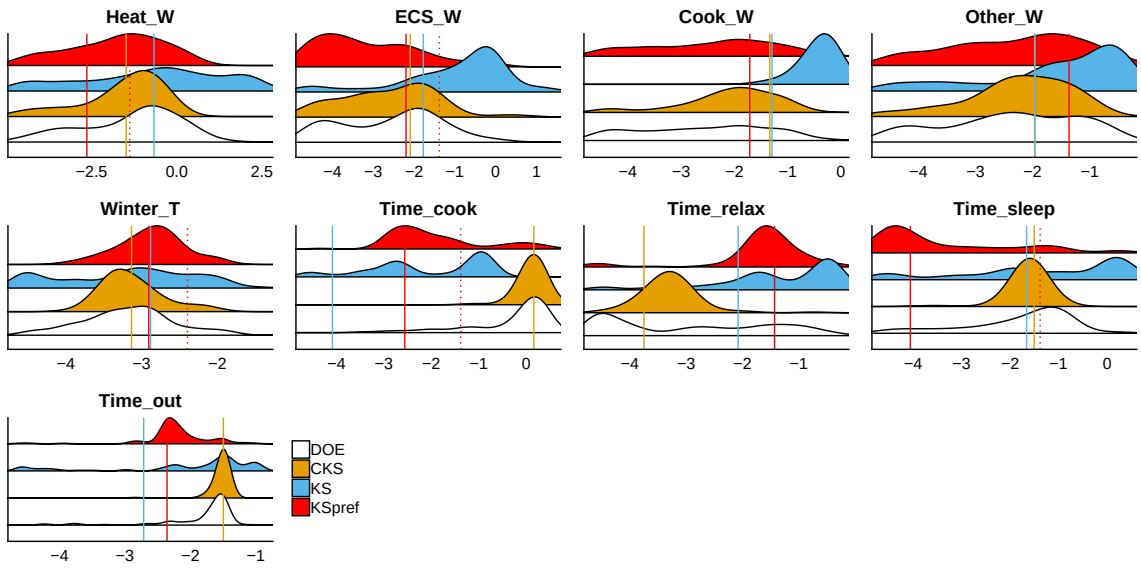

Figure 9: Marginal distributions of the 10 objectives of the calibration problem. Each density corresponds to 100 values. The vertical bars represent the identified solutions. The dotted lines represent the preferences imposed on five of the objectives.

cases, we observe that the entire range of variation is explored, which indicates that either exploration steps or steps aiming at learning the $(\mathbf{d}, \mathbf{u})$ were performed. A higher density of points is visible around the final solutions, which indicates exploitation steps. In general, we observe a more uniform distribution for KS (in particular for the last two objectives), which can be attributed to the task of learning extremal values, while CKS focusses directly on the equilibrium.

We see that the solutions are similar for some objectives (ECS_W, Cook_W, Winter_T), they differ substantially for others (in particular Heat_W, Time_cook, Time_out). For all the objectives, the KS with predefined disagreement point satisfy the constraints. For Heat_W and Time_sleep for instance, the predefined disagreement point clearly shifted the solution to the left, which was desired. This is at the price of balance between the objectives, as the solution at the unconstrained objectives is in general worse than at the KS one (e.g., Other_W, Time_relax). The CKS solution clearly follows the distribution of the initial experiments, while the KS solution follows the range of the objectives. The difference between KS and CKS appears most clearly on objectives Heat_W, Time_cook and Time_out. For Heat_W, the heavy right-tail "pushes" the KS solution to the right, while CKS accounts for the high density of solutions on the left. Conversely, for Time_cook and Time_out the density peak on the right "pushes" the CKS solution to the right.

Figure 10 illustrates the different trade-offs achieved by our three solutions. Using the original scale of the objectives, KS appears as the most balanced solution, while CKS seems to "sacrifice" two objectives (Time_cook and Time_out). However, if we look at the ranks of

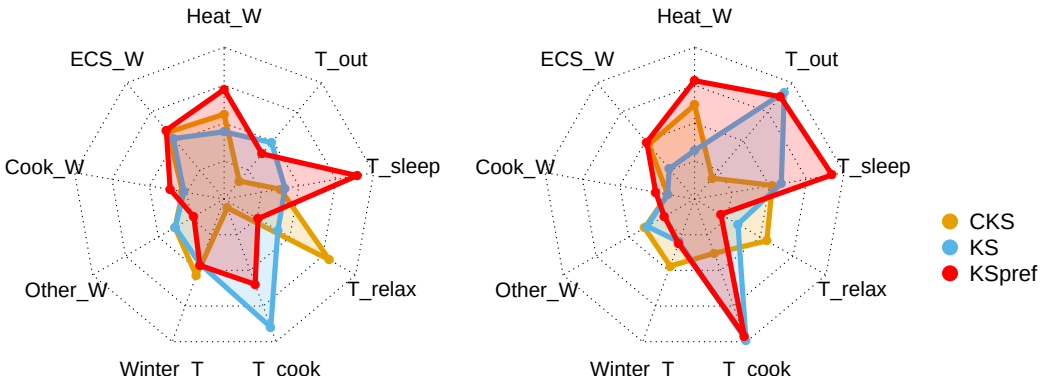

Figure 10: Comparison of the three solutions of the calibration problem. Left: original scale; right: rank scale (with respect to the initial 100-point design only). The outer dotted line corresponds to the best performance, the center to the worst.

solutions with respect to the initial 100 points, we see that CKS is the most central solution, while KS and KS with preferences are biased towards some objectives.

5. Conclusion and future work

In this article, we tackled many-objective problems by looking for a single well-balanced solution, originating from game theory. Two alternatives to this solution have been proposed, either by imposing preferences on some objective values (by specifying a disagreement point) or by working in the space of copulas, the latter solution being insensitive to monotonic transformations of the objectives. Looking for these solutions is in general a complex learning task. We proposed a tailored algorithm based on the stepwise uncertainty reduction paradigm, that automatically performs a trade- off between the different learning tasks (estimating the ideal point, the nadir point, and the marginals or exploring locally the space next to the estimated solution). We tested our algorithm on three different problems with growing complexity and found that well-balanced solutions could be obtained despite severely restricted budgets.

Choosing between the alternatives seem highly problem-dependent. On DTLZ2, KS appears as a more "central" (hence desirable) solution, while on the CNN tuning CKS is clearly a better choice. This difference may be imputed to scaling of the objectives prior to optimization. On DTLZ2, all objectives behave similarly, while on CNN some are strongly heavy-tailed. Hence, a reasonable choice would be CKS for more exploratory studies, and KS for a finer design on a pre-explored problem. Incorporating user preferences, as in the calibration problem, was proved easy and efficient and may direct the choice toward KS in such a case.

Many potential lines of future works remain. First, we may consider batch- sequential strategies instead of one observation at a time. This approach was not necessary in our test problems, since parallel computation was used to repeat simulations and average out noise. The SUR strategy naturally adapts to this case (Chevalier et al., 2015). In practice, one

may also have to deal with asynchronous returns of batches, and possibly the use of several black-boxes with different costs, as in (Hernández-Lobato et al., 2016b).

An unused degree of freedom here is the number of replications to handle the noise, which might improve substantially the practical efficiency (Jalali et al., 2017). To do so, one may combine the presented approach with an efficient scheme for designing replications and estimating noise, as done by Binois et al. (2016).

Another direction is to consider alternative equilibria. A promising idea is to use a set of disagreement points instead of a single one: this could result in more robust solutions and potentially a small set of Pareto-optimal solutions, which might be preferred by decision-makers. Combining the KS solution with Nash games has been suggested in the game theory literature. Conley and Wilkie (1991) proposed using Nash equilibria as disagreement points, provided there exists a natural, or some relevant, splitting of the decision variable among the players. In the multicriteria Nash game framework (Ghose and Prasad, 1989), noncooperative players have to handle individual vector payoffs, so that to each request by other players, they have to respond with some payoff, rationally selected from their own Pareto front; a good candidate would be the KS solution. Both alternatives were poorly investigated in practice, mainly because their potential of application seems to be hindered by the lack of efficient tools that allow for an acceptable implementation. We think that our algorithmic framework could apply to both cases and provide a first solution to such problems.

Copula spaces have been used here mostly for rescaling. Taking advantage of the predictive capacity of copulas to accelerate the estimation of the Pareto front might accelerate substantially the search for the CKS solution. Furthermore, combining efficiently the two types of metamodels (GPs and copulas), as done by Wilson and Ghahramani (2010), may lead to new theoretical advances and algorithms.

We leave to future work theoretical considerations on the convergence of the approach, following for instance the recent works on information- directed sampling by Russo and Van Roy (2014) or on SUR by Bect et al. (2016).

Acknowledgments

The work of MB is supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Contract No. DE-AC02-06CH11357. We thank Gail Pieper for her useful language editing.

Appendix A: Quantities related to Gaussian processes

GP moments We provide below the equations of the moments of a GP conditioned on n (noisy) observations $\mathbf{f} = (f_1, \ldots, f_n)$. Assuming a kernel function σ and a mean function $m(\mathbf{x})$, we have

$$\begin{aligned} \mu_n(\mathbf{x}) &= m(\mathbf{x}) + \lambda(\mathbf{x})\left(\mathbf{f} - m(\mathbf{x})\right), \\ \sigma_n^2(\mathbf{x}, \mathbf{x}') &= \sigma(\mathbf{x}, \mathbf{x}') - \lambda(\mathbf{x})\sigma(\mathbf{x}', \mathbf{X}_n), \end{aligned}$$

where

- $\lambda(\mathbf{x}) = \sigma(\mathbf{x}, \mathbf{X}_n)^\top \sigma(\mathbf{X}_n, \mathbf{X}_n)^{-1}$,

- $\sigma(\mathbf{x}, \mathbf{X}_n) := (\sigma(\mathbf{x}, \mathbf{x}_1), \ldots, \sigma(\mathbf{x}, \mathbf{x}_n))^\top$ and
- $\sigma(\mathbf{X}_n, \mathbf{X}_n) := (\sigma(\mathbf{x}_i, \mathbf{x}_j) + \tau_i^2 \delta_{i=j})_{1 \leq i,j \leq n}$,

δ standing for the Kronecker function.

Commonly, σ belongs to a parametric family of covariance functions such as the Gaussian and Matérn kernels, based on hypotheses about the smoothness of y. Corresponding hyperparameters are often obtained as maximum likelihood estimates; see e.g., Rasmussen and Williams (2006) for the corresponding details.

Expected improvement Denote $f_{\min} = \min_{1 \leq i \leq n}(f_i)$ the minimum of the observed values. The expected improvement is the expected positive difference between $f_{\min}$ and the new potential observation $Y_n(\mathbf{x})$:

$$\begin{aligned} EI(\mathbf{x}) &= \mathbb{E}\left(\max\left((0, f_{\min} - Y_n(\mathbf{x})\right)\right) \\ &= (f_{\min} - \mu_n(\mathbf{x}))\Phi\left(\frac{f_{\min} - \mu_n(\mathbf{x})}{\sigma_n(\mathbf{x})}\right) + \sigma_n^2(\mathbf{x})\phi\left(\frac{f_{\min} - \mu_n(\mathbf{x})}{\sigma_n(\mathbf{x})}\right), \end{aligned}$$

where ϕ and Φ are respectively the PDF and CDF of the standard Gaussian variable.

$p_{\mathbf{box}}$ Let $LB \in \mathbb{R}^p$ and $UB \in \mathbb{R}^p$ such that $\forall 1 \leq i \leq p, LBi < UB_i$ define a box in the objective space. Defining $\mathbf{\Psi} = [\Psi(\boldsymbol{\mathcal{Y}}_1), \ldots, \Psi(\boldsymbol{\mathcal{Y}}_M)]$ the $p \times M$ matrix of simulated KS solutions, we use

$$\forall 1 \leq i \leq p \qquad LB_i = \min \mathbf{\Psi}_{i,1\ldots M} \quad \text{and} \quad UB_i = \max \mathbf{\Psi}_{i,1\ldots M}.$$

Then, the probability to belong to the box is

$$p_{\text{box}}(\mathbf{x}) = \prod_{i=1}^{p} \left[\Phi\left(\frac{UB_i - \mu_i(\mathbf{x})}{\sigma_i(\mathbf{x})}\right) - \Phi\left(\frac{\mu_i(\mathbf{x}) - LB_i}{\sigma_i(\mathbf{x})}\right)\right].$$

Probability of nondomination Let $\mathbb{X}_n^*$ be the subset of nondominated observations. The probability of non-domination is

$$p_{ND}(\mathbf{x}) = \mathbb{P}\left(\forall \mathbf{x}^* \in \mathbb{X}_n^*, \exists k \in \{1, \ldots, q\} \text{ such that } Y_n^{(k)}(\mathbf{x}) \leq Y_n^{(k)}(\mathbf{x}^*)\right).$$

Using the GP equations for Y_n, one can compute $p_{ND}(\mathbf{x})$ in closed form. We refer to the work of Couckuyt et al. (2014) for the formulas expressed in an efficient form.

Copula estimation The copula and marginals need to be estimated based on a sample of $\mathbf{Y}(X)$, with X i.i.d. This prevents directly using the observations, since the $\mathbf{x}_i$'s are chosen to target specific regions. This also applies to the integration points $\mathbb{X}_\star$. To avoid such bias, we use a set of a large auxiliary i.i.d. sample $\mathbb{X}_\dagger = X_1, \ldots, X_P$, $P \in \mathbf{N}$. Since conditional simulation is out of reach for a large sample (Section 3.3.2), we follow Oakley (2004) and use the conditional simulations $\boldsymbol{\mathcal{Y}}_j$ on $\mathbb{X}_\star$ as pseudo-observations to update the GP predictive mean, and we take our sample $\mathbf{Y}(\mathbb{X}_\dagger)$ for this updated mean.The empirical marginals and copula are estimated on $\mathbf{Y}(\mathbb{X}_\dagger)$.

Appendix B: CNN lists of inputs and outputs

Appendix C: li-BIM list of inputs and outputs

Table 1: List of inputs for the CNN training problem.

	Description	Min	Max
x_1	number of filters of first convolutional layer	1	100
x_2	number of filters of second convolutional layer	1	100
x_3	first dropout rate	0	0.9
x_4	second dropout rate	0	0.9
x_5	number of units of dense hidden layer	1	200

Table 2: List of outputs for the CNN training problem

Name	Unit
Training cross-entropy	-
Validation cross-entropy	-
Testing cross-entropy	-
Testing accuracy	-
Training time	s
Prediction time	s

Table 3: List of inputs of the li-BIM model.

Name	Unit	Meaning	Min	Max
Cth_h	J/K	Thermal capacity of the dwelling	10^6	10^7
RD	W	Average power of the relaxing devices	150	1000
HW	W	Power of the boiler to produce hot water	500	3000
CD	W	Average power of the cooking devices	100	300
Pmaxchaud	kW	Maximum power of the boiler to heat dwelling	500	3000
SensitiveCold	[0-1]	Sensitivity of the occupant to cold temperature	0.2	1.0
SensitiveWarm	[0-1]	Sensitivity of the occupant to warm temperature	0.5	1
NbHfreshair	hours	Average number of hours between two outings	16	36
NbHtire	hours	Average number of hours between two sleeps	8	60
NbHhungry	hours	Average number of hours between two meals	6	16
NbHdirty	hours	Average number of hours between two showers/baths	20	48
Deltamodif	hours	Time before new action if previous action insufficient	5	30
Thermal_effort	Celcius	Max difference to ideal temperature before acting	2	15

Table 4: List of outputs of the li-BIM model.

Name	Unit	Meaning	Target
Heat_W	kWh	Total energetic consumption of heating devices	1384
ECS_W	kWh	Total energetic consumption of hot water devices	1198
Cook_W	kWh	Total energetic consumption of cooking devices	306
Other_W	kWh	Total energetic consumption of other devices	1751
Winter_T	°C	Average temperature during winter	21.8
Time_cook	hours	Average time spent cooking	0.9
Time_relax	hours	Average time spent relaxing	3.7
Time_sleep	hours	Average time spent sleeping	8.35
Time_out	hours	Average time spent outside the building	0.58

References

JJ Allaire and François Chollet. *keras: R Interface to 'Keras'*, 2018. URL `https://keras.rstudio.com`. R package version 2.1.4.

Md Asafuddoula, Tapabrata Ray, and Ruhul Sarker. A decomposition-based evolutionary algorithm for many objective optimization. *IEEE Transactions on Evolutionary Computation*, 19(3):445–460, 2015.

Johannes Bader and Eckart Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary Computation*, 19(1):45–76, 2011.

Slim Bechikh, Lamjed Ben Said, and Khaled Ghedira. Estimating nadir point in multi-objective optimization using mobile reference points. In *2010 IEEE congress on Evolutionary computation (CEC)*, pages 1–9. IEEE, 2010.

Julien Bect, David Ginsbourger, Ling Li, Victor Picheny, and Emmanuel Vazquez. Sequential design of computer experiments for the estimation of a probability of failure. *Statistics and Computing*, 22(3):773–793, 2012.

Julien Bect, François Bachoc, and David Ginsbourger. A supermartingale approach to Gaussian process based sequential design of experiments. *arXiv preprint arXiv:1608.01118*, 2016.

James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13:281–305, 2012.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011.

Mickaël Binois, Didier Rullière, and Olivier Roustant. On the estimation of Pareto fronts from the point of view of copula theory. *Information Sciences*, 324:270 – 285, 2015.

Mickael Binois, Robert B Gramacy, and Michael Ludkovski. Practical heteroskedastic Gaussian process modeling for large simulation experiments. *arXiv preprint arXiv:1611.05902*, 2016.

Mathieu Bourgais, Patrick Taillandier, and Laurent Vercouter. Enhancing the behavior of agents in social simulations with emotions and social relations. In *18th workshop on Multi-Agent-Based Simulation-MABS 2017*, 2017.

Irem Bozbay, Franz Dietrich, and Hans Peters. Bargaining with endogenous disagreement: The extended Kalai–Smorodinsky solution. *Games and Economic Behavior*, 74(1):407–417, 2012.

Clément Chevalier, Xavier Emery, and David Ginsbourger. Fast update of conditional simulation ensembles. *Mathematical Geosciences*, 47(7):771–789, 2015.

François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

Tinkle Chugh, Karthik Sindhya, Jussi Hakanen, and Kaisa Miettinen. A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Computing*, pages 1–30, 2017.

Tinkle Chugh, Yaochu Jin, Kaisa Miettinen, Jussi Hakanen, and Karthik Sindhya. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 22(1):129–143, 2018.

John P Conley and Simon Wilkie. The bargaining problem without convexity: Extending the egalitarian and Kalai-Smorodinsky solutions. *Economics Letters*, 36(4):365–369, 1991.

Ivo Couckuyt, Dirk Deschrijver, and Tom Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60(3):575–594, 2014.

Indraneel Das and John E Dennis. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8(3):631–657, 1998.

Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on Evolutionary Computation*, 6(2):182–197, 2002.

J.-A. Désidéri, R. Duvigneau, and A. Habbal. *Computational Intelligence in Aerospace Sciences, V. M. Becerra and M. Vassile Eds.*, volume 244 of *Progress in Astronautics and Aeronautics*, chapter Multi-Objective Design Optimization Using Nash Games. AIAA, 2014.

Peter Diggle and Paulo Justiniano Ribeiro. *Model-Based Geostatistics*. Springer, 2007.

Valerii Vadimovich Fedorov. *Theory of Optimal Experiments*. Elsevier, 1972.

D Ghose and UR Prasad. Solution concepts in two-person multicriteria games. *Journal of Optimization Theory and Applications*, 63(2):167–189, 1989.

Philipp Hennig and Christian J Schuler. Entropy search for information-efficient global optimization. *The Journal of Machine Learning Research*, 13:1809–1837, 2012.

Daniel Hernández-Lobato, Jose Hernandez-Lobato, Amar Shah, and Ryan Adams. Predictive entropy search for multi-objective Bayesian optimization. In *International Conference on Machine Learning*, pages 1492–1501, 2016a.

José Miguel Hernández-Lobato, Michael A Gelbart, Ryan P Adams, Matthew W Hoffman, and Zoubin Ghahramani. A general framework for constrained Bayesian optimization using information-based search. *Journal of Machine Learning Research*, 17(160):1–53, 2016b.

Jens Leth Hougaard and Mich Tvede. Nonconvex n-person bargaining: efficient maxmin solutions. *Economic Theory*, 21(1):81–95, 2003.

Hisao Ishibuchi, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization: A short review. In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.*, pages 2419–2426. IEEE, 2008.

Hamed Jalali, Inneke Van Nieuwenhuyse, and Victor Picheny. Comparison of kriging-based algorithms for simulation optimization with heterogeneous noise. *European Journal of Operational Research*, 261(1):279–301, 2017.

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

Ulrich Junker et al. Preference-based search and multi-criteria optimization. *Annals of Operations Research*, 130(1-4):75–115, 2004.

E. Kalai and M. Smorodinsky. Other solutions to Nash's bargaining problem. *Econometrica*, 43:513–518, 1975.

J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, February 2006.

Saku Kukkonen and Jouni Lampinen. Ranking-dominance and many-objective optimization. In *IEEE Congress on Evolutionary Computation, 2007. CEC 2007.*, pages 3983–3990. IEEE, 2007.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Miqing Li, Liangli Zhen, and Xin Yao. How to read many-objective solution sets in parallel coordinates. *arXiv preprint arXiv:1705.00368*, 2017.

Michael D McKay, Richard J Beckman, and William J Conover. Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245, 1979.

Roger B Nelsen. *An Introduction to Copulas*. Springer, 2006.

Harald Niederreiter. Low-discrepancy and low-dispersion sequences. *Journal of Number Theory*, 30(1):51–70, 1988.

Jeremy Oakley. Estimating percentiles of uncertain computer code outputs. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 53(1):83–93, 2004.

Marek Omelka, Irène Gijbels, Noël Veraverbeke, et al. Improved kernel estimation of copulas: weak convergence and goodness-of-fit testing. *The Annals of Statistics*, 37(5B):3023–3058, 2009.

James Parr. *Improvement criteria for constraint handling and multiobjective optimization*. PhD thesis, University of Southampton, 2013.

Victor Picheny. Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, pages 1–16, 2013.

Victor Picheny and Mickael Binois. *GPGame: Solving Complex Game Problems using Gaussian Processes*, 2018. URL `http://CRAN.R-project.org/package=GPGame`. R package version 1.1.0.

Victor Picheny, Mickael Binois, and Abderrahmane Habbal. A Bayesian optimization approach to find Nash equilibria. *arXiv preprint arXiv:1611.02440*, 2016.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL `https://www.R-project.org/`.

Carl E. Rasmussen and Christopher Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. URL `http://www.gaussianprocess.org/gpml/`.

Olivier Roustant, Esperan Padonou, Yves Deville, Aloïs Clément, Guillaume Perrin, Jean Giorla, and Henry Wynn. Group kernels for Gaussian process metamodels with categorical inputs. *arXiv preprint arXiv:1802.02368*, 2018.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. In *Advances in Neural Information Processing Systems*, pages 1583–1591, 2014.

Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando de Freitas. Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2016.

Hemant Kumar Singh, Amitay Isaacs, and Tapabrata Ray. A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, 15(4):539–556, 2011.

Sean C Smithson, Guang Yang, Warren J Gross, and Brett H Meyer. Neural networks designing neural networks: Multi-objective hyper-parameter optimization. In *2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 1–8. IEEE, 2016.

Joshua D. Svenson. *Computer Experiments: Multiobjective Optimization and Sensitivity Analysis*. PhD thesis, The Ohio State University, 2011.

Franck Taillandier, Alice Micolier, and Patrick Taillandier. Li-bim (version 1.0.0), 2017.

Patrick Taillandier, Benoit Gaudou, Arnaud Grignard, Quang-Nghi Huynh, Nicolas Marilleau, Philippe Caillou, Damien Philippon, and Alexis Drogoul. Building, composing and experimenting complex spatial models with the gama platform. *GeoInformatica*, pages 1–24, 2018.

Lothar Thiele, Kaisa Miettinen, Pekka J Korhonen, and Julian Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolutionary computation*, 17(3): 411–436, 2009.

Julien Villemonteix, Emmanuel Vazquez, and Eric Walter. An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, 44 (4):509–534, 2009.

Tobias Wagner, Michael Emmerich, André Deutz, and Wolfgang Ponweiser. On expected-improvement criteria for model-based multi-objective optimization. In *International Conference on Parallel Problem Solving from Nature*, pages 718–727. Springer, 2010.

Eric Walter and Luc Pronzato. *Identification of Parametric Models from Experimental Data*. Springer Verlag, 1997.

Andrew G Wilson and Zoubin Ghahramani. Copula processes. In *Advances in Neural Information Processing Systems*, pages 2460–2468, 2010.

Qingfu Zhang, Wudong Liu, E. Tsang, and B. Virginas. Expensive multiobjective optimization by MOEA/D with Gaussian process model. *IEEE Transactions on Evolutionary Computation*, 14(3):456–474, 2010.