Author(s): Steponavice, Ingrida; Ruuska, Sauli; Miettinen, Kaisa

Title: A solution process for simulation-based multiobjective design optimization with an application in the paper industry

Year: 2014

Version:

Please cite the original version:

Steponavice, I., Ruuska, S., & Miettinen, K. (2014). A solution process for simulation-based multiobjective design optimization with an application in the paper industry. Computer-Aided Design, 47, 45-58. https://doi.org/10.1016/j.cad.2013.08.045

A Solution Process for Simulation-based Multiobjective Design Optimization with an Application in Paper Industry

Ingrida Steponavičė*, Sauli Ruuska, Kaisa Miettinen

Department of Mathematical Information Technology, University of Jyväskylä, Jyväskylä, FINLAND

Abstract

In this paper, we address some computational challenges arising in complex simulation-based design optimization problems. High computational cost, black-box formulation and stochasticity are some of the challenges related to optimization of design problems involving simulation of complex mathematical models. Solving becomes even more challenging in case of multiple conflicting objectives that must be optimized simultaneously. In such cases, application of multiobjective optimization methods is necessary in order to gain an understanding of which design offers the best possible trade-off. We apply a three-stage solution process to meet the challenges mentioned above. As our case study, we consider the integrated design and control problem in paper mill design where the aim is to decrease the investment cost and enhance the quality of paper on the design level and, at the same time, guarantee the smooth performance of the production system on the operational level. In the first stage of the three-stage solution process, a set of solutions involving different trade-offs is generated with a method suited for computationally expensive multiobjective optimization problems using parallel computing. Then, based on the generated solutions an approximation method is applied to create a computationally inexpensive surrogate problem for the

*Corresponding author: Tel.: +358 40 024 7443; fax: +358 14 260 2771,
Postal address: Dept. of Mathematical Information Technology,
P.O. Box 35 (Agora), FI-40014 University of Jyväskylä, FINLAND
Email addresses: ingrida.steponavice@monash.edu (Ingrida Steponavičė),
sauli.ruuska@jyu.fi (Sauli Ruuska), kaisa.miettinen@jyu.fi (Kaisa Miettinen)

design problem and the surrogate problem is solved in the second stage with an interactive multiobjective optimization method. This stage involves a decision maker and her/his preferences to find the most preferred solution to the surrogate problem. In the third stage, the solution best corresponding that of stage two is found for the original problem.

Keywords: Multicriteria decision making, Multiobjective optimization, Pareto optimality, Computational cost, NIMBUS method, PAINT method

---

1. Introduction

The widespread availability of powerful computers has made it possible to obtain detailed analyses of complex systems quickly and at a relatively low cost. Consequently, computer simulation has become a central tool in the design process across the industries. Computer simulation can be readily used to answer questions such as whether or not a system will meet specified requirements. To answer questions such as what is the maximum system performance and how the system should be designed to achieve the maximum performance, simulation must be combined with optimization. Solving an optimization problem that depends on the output of a simulation model is known as *simulation-based optimization*. In this paper, we consider computational challenges of simulation-based optimization encountered with real-life design optimization problems and relate them to a case study in paper industry.

A computer simulation of a physical or some other system of interest typically consists of solving a system of algebraic and differential equations. From the optimization point of view, using a simulator as an external solver for a system of equations is equivalent to dividing the decision variables into two groups, the *dependent* and the *independent* variables, and substituting the dependent variables with functions of the independent ones. The choice between dependent and independent variables is often dictated by the simulator, which has to take the independent variables as input and provide the values of the dependent variables as output. Considering only the independent variables as decision variables reduces the dimensionality of the problem but, on the other hand, makes it a *black-box optimization* problem because the functional relationship between the independent and the dependent variables is known only implicitly. This prevents any algebraic manipulation and makes it very difficult to validate the assumptions of, for example, convexity

and differentiability that many optimization methods rely on.

In a real-life design optimization problem, there is rarely a single performance measure fully appreciating the relative merits of each design. Instead, it is characteristic of a design optimization problem to have multiple, conflicting objectives. When optimization is applied, however, the optimization problem is commonly formulated with a single objective—often by considering a weighted sum of the design objectives or by treating all but one of them as constraints—because most of the optimization algorithms can handle only single-objective problems. A shortcoming of a simplistic single-objective problem formulation is that it provides little support for decision making, often requiring the parameters in the problem formulation to be adjusted by trial and error to achieve the desired outcome. The structure of a design optimization problem can often be reflected more closely by formulating it as a *multiobjective optimization* problem, in which all the objectives are to be optimized simultaneously. A multiobjective formulation comes with a cost, though, as it necessitates the involvement of a decision maker.

With an increasing complexity of design problems, finding an optimal design in real-life applications remains a challenging task [1, 2, 3]. The computational challenges in design optimization that we consider in this paper (and which we have found the most pertinent ones) are the following.

Computational cost In simulation-based optimization, the objective and constraint functions depend on the decision variables not only directly, but also indirectly through the simulation model. Therefore, to calculate the values of those functions, a simulation must be carried out, which may well take from few minutes to several days. Moreover, the simulation must be repeated every time an optimization method needs to evaluate the objective and constraint functions. Thus, the time required for one simulation run on average, or the computational cost of the simulation, is a major factor limiting the practicability of simulation-based optimization.

Conflicting objectives Multiple, conflicting objectives give rise to a set of solutions, called the Pareto optimal set, that correspond to different trade-offs among the objectives and are not self-evidently comparable. This is in contrast to single-objective optimization, in which an optimum, if it exists, is uniquely defined. With multiple objectives, the identification of the preferred solution requires the involvement of a

decision maker and sufficient methodological support to explore the alternative solutions. It is, however, challenging to implement a system that can provide a fast enough response for successful decision making when applied to computationally expensive simulation-based optimization.

Black-box models The lack of closed form expressions for the objective and constraint functions effectively requires a design optimization problem to be treated as a global optimization problem. The necessity of global optimization increases the computational cost of design optimization and limits the size of the design optimization problems that can be solved. Fortunately, it is rarely necessary to guarantee global optimality, but instead, a sufficient improvement over an existing design is acceptable.

Stochasticity In many real-life design optimization problems, the system of interest is best modeled by a stochastic process. In that case, the model output is a random vector, often with an unknown probability distribution. The model output can be sampled by a computer simulation, although the computational cost of simulating the output increases with the sample size. Moreover, unless the sample size is sufficiently large, sampling error introduces noise to the values of the objective and constraint functions that depend on some statistic of the model output.

The above challenges are intertwined in the sense that the presence of each one of them makes the others more difficult to address. For example, global optimization quickly becomes impractical if the computational cost of the design optimization problem increases. Likewise, stochasticity and conflicting objectives both aggravate the difficulties caused by a high computational cost because more computation is required to sample the model output and to assess the trade-offs, respectively.

An overview of optimization methods applied to solving multiobjective engineering problems is given in [4]. Metamodelling techniques have been found to be beneficial tools in supporting design optimization [5, 6]. In multiobjective design optimization, most of the efforts have been devoted to finding a number of Pareto optimal solutions (see, e.g., [7, 8, 9, 10]) without considering support for a decision maker. Only few applications of interactive multiobjective optimization methods to design optimization problems

can be found, e.g., in [11, 12, 13, 14, 15, 16]. For example, Tappeta *et al.* [11] have proposed an approach which differs from ours in three aspects. First, it requires constructing individual metamodels for all objective and constraint functions. Second, a local approximation of the Pareto optimal set is considered. Finally, there is no clear distinction between interaction with a decision maker and the demanding computations which would imply long waiting times in case a decision maker wishes to explore different (other than local) Pareto optimal solutions. To our knowledge, there is no off-the-shelf interactive method which could be directly applied to computationally expensive simulation-based multiobjective optimization problems without creating waiting times for a decison maker.

We present in this paper a three-stage solution process that is designed to address the challenges of computationally expensive multiobjective design optimization. A wide range of optimization algorithms can be integrated with the solution process, which makes it applicable to many real-life design optimization problems. In the first stage, termed the *pre-decision making* stage, sufficient information is gathered about the alternative solutions to the multiobjective design optimization problem. In the second stage, termed the *decision making* stage, a human decision maker is involved by using an interactive method to solve a computationally inexpensive surrogate problem constructed on the basis of the information gathered in the first stage. In the third stage, termed the *post-decision making* stage, the original design optimization problem is solved with the purpose of finding a solution that best matches the preferred solution to the surrogate problem identified in the second stage.

The three-stage solution process has the benefit that it separates the time-consuming simulation-based optimization from the decision making stage. This allows fluent interaction with the decision maker regardless of the computational intensiveness of the simulation model. The solution process is motivated by the PAINT method [17], which can be used to create a surrogate problem for decision making, as well as by the availability of multiobjective optimization methods such as ParEGO [18] and SMS-EGO [19] that provide a finite approximation to the Pareto optimal set of a multiobjective optimization problem.

The paper has the following structure. Section 2 covers main concepts of multiobjective optimization used in this paper. In Section 3, we describe the three-stage solution process we propose for multiobjective design optimization. Section 4 provides a description of a case study of an optimal design

of a paper mill concept. Section 5 demonstrates an application of the three-stage solution process to our case study and reports the numerical results obtained in each stage. The paper is concluded in Section 6.

2. Concepts and background

In this section, we review the concepts and background in multiobjective optimization and design optimization relevant for the rest of the paper. Readers familiar with these topics may wish to proceed directly to Section 3.

2.1. Multiobjective optimization

A multiobjective optimization problem (MOP) has the form

$$\begin{aligned} \min \quad & \mathbf{f}(\mathbf{x}) = \big(f_1(\mathbf{x}), \ldots, f_k(\mathbf{x})\big)^\mathsf{T} \\ \text{subject to} \quad & \mathbf{x} \in S, \end{aligned} \tag{1}$$

where $S \subset \mathbb{R}^n$ is the feasible set and $f_i \colon S \to \mathbb{R}$, $i = 1, \ldots, k$ ($k \geq 2$), are objective functions that are to be minimized simultaneously. All objective functions are conveniently represented by a vector-valued function $\mathbf{f} \colon S \to \mathbb{R}^k$. A vector $\mathbf{x} \in S$ is called a *decision vector* and a vector $\mathbf{z} \in \mathbb{R}^k$ an *objective vector*. Moreover, $\mathbf{z} \in \mathbb{R}^k$ is said to be *attainable* if there exists a decision vector $\mathbf{x} \in S$ such that $\mathbf{z} = \mathbf{f}(\mathbf{x})$ and *unattainable* otherwise. Without loss of generality, we consider only minimization because maximization of an objective function can be converted to minimization by multiplication by minus one.

We assume that there does not exist a decision vector $\bar{\mathbf{x}} \in S$ such that $\bar{\mathbf{x}}$ minimizes f_i in S for all $i = 1, \ldots, k$. In that case, the objective functions $f_1, \ldots, f_k$ in (1) are said to be *conflicting*. A useful notion of optimality for conflicting objectives is given by Pareto optimality: a decision vector $\mathbf{x} \in S$ and the corresponding objective vector $\mathbf{z} = \mathbf{f}(\mathbf{x})$ are said to be *Pareto optimal* or *nondominated* if there does not exist a decision vector $\bar{\mathbf{x}} \in S$ such that $f_i(\bar{\mathbf{x}}) \leq f_i(\mathbf{x})$ for all $i = 1, \ldots, k$ and $\mathbf{f}(\bar{\mathbf{x}}) \neq \mathbf{f}(\mathbf{x})$. On the other hand, if such a vector $\bar{\mathbf{x}} \in S$ does exist, $\mathbf{x}$ and $\mathbf{z}$ are said to be *dominated* by $\bar{\mathbf{x}}$ and $\bar{\mathbf{z}} = \mathbf{f}(\bar{\mathbf{x}})$, respectively. In the following, we use the term *Pareto optimal set* to refer to both the set of all Pareto optimal decision vectors and the set of all Pareto optimal objective vectors.

The order relation appearing in the definition of Pareto optimality is not a total but a partial order, and therefore, the set of all Pareto optimal objective

vectors is not, in general, a singleton. Consequently, an optimal solution to (1) is not well defined without additional information, usually provided by a human *decision maker* (DM). Nonetheless, under the assumption implied by the problem formulation that for all objective functions, the less of any two values is preferred by the DM, it can be shown that the objective vector preferred by the DM to all others is always Pareto optimal. This makes it possible to support the DM in finding the preferred objective vector by excluding from consideration all the dominated ones.

Upper and lower bounds are often determined for the Pareto optimal set in the objective space. The bounds are used, for example, to normalize the objective functions and to help the DM set expectations. A lower bound, known as the *ideal vector*, can be determined by minimizing each of the objective functions $f_1, \ldots, f_k$ independently over S and collecting the outcomes into a vector. In contrast, an analogous upper bound, known as the *nadir vector*, cannot be readily obtained in general. This is because the objective functions would have to be independently maximized over the Pareto optimal set, which is, in general, a nonconvex subset of S that is known only implicitly. Therefore, a rough estimate for the nadir vector is commonly used (see, *e.g.*, [20]).

Multiobjective optimization methods can be classified into *no-preference*, *a priori*, *a posteriori*, and *interactive* methods according to the way they involve the DM in the solution process [20]. No-preference and *a priori* methods produce only a single Pareto optimal decision vector. The latter require sufficient preference information to be provided up front, whereas the former do not take the preferences of the DM into account at all. Both types of methods allow an MOP to be solved with a computational cost comparable to that of solving a single-objective optimization problem, but their main disadvantage is the lack of a systematic procedure in the case that the DM is not satisfied with the obtained solution.

An opposite approach is taken by *a posteriori* methods, which attempt to produce a set of decision vectors so that the corresponding objective vectors represent the Pareto optimal set in the objective space as closely as possible. Instead of eliciting preference information explicitly, these methods rely on the DM to assess the alternative solutions produced and to ultimately select the preferred one. Unlike in the aforementioned methods, all the computation takes place before the DM is involved. The disadvantage of *a posteriori* methods is mainly their inability to support the DM in comparing and ranking the provided alternative solutions. They may also consume a lot of

computational resources producing decision vectors that are of little or no relevance to the DM.

Interactive methods address many of the shortcomings of the other types of methods by eliciting preference information from the DM progressively [21]. At every iteration, the DM is presented with one or more objective vectors and asked to express preferences relative to them. The methods differ in the type of information presented to and requested from the DM, but they commonly tolerate inconsistencies in the expressed preferences as well as preference information that is valid only locally. Interactive methods facilitate exploring the Pareto optimal set and learning about the problem in a way unmatched by any of the other types of methods. Nevertheless, they can only be applied if there is a DM available who is willing to participate in the solution process. In simulation-based optimization, and in design optimization in general, this may become an issue as the interaction between the method and the DM is constantly interrupted by time-consuming computations when new solution(s) based on the preferences are to be calculated.

Many multiobjective optimization methods are based on *scalarization* of problem (1), that is, substituting a real-valued function on S for the vector-valued objective function $\mathbf{f}$. With a properly chosen scalarization, it can be shown that the decision vector obtained by minimizing the scalarized objective function is always Pareto optimal [20]. Scalarization is used in many interactive methods as a way to generate new Pareto optimal solutions based on the preferences specified by the DM.

Scalarization can be used to project vectors in the objective space onto the Pareto optimal set. With a suitable scalarization, it is possible to project attainable and unattainable objective vectors alike. In Section 5, we employ for this purpose an *achievement* (*scalarizing*) *function* $\bar{f}_{\bar{\mathbf{z}}}\colon S \to \mathbb{R}$ defined for all $\bar{\mathbf{z}} \in \mathbb{R}^k$ and all $\mathbf{x} \in S$ as [22, 23]

$$\bar{f}_{\bar{\mathbf{z}}}(\mathbf{x}) = \max_{i=1,\ldots,k} \frac{f_i(\mathbf{x}) - \bar{z}_i}{z_i^{\mathrm{nad}} - z_i^{\star\star}} + \rho \sum_{i=1}^{k} \frac{f_i(\mathbf{x})}{z_i^{\mathrm{nad}} - z_i^{\star\star}}, \tag{2}$$

where $\mathbf{z}^{\mathrm{nad}} \in \mathbb{R}^k$ is the nadir vector of (1) and $\mathbf{z}^{\star\star} \in \mathbb{R}^k$ is a *utopian vector*, that is, a vector that is strictly less than the ideal vector in all components. Vector $\bar{\mathbf{z}} \in \mathbb{R}^k$ is called a *reference point*, and in addition to being the vector to be projected, it can be interpreted as preference information in the form of desirable levels of the objective function values. The second term is called an *augmentation term* and ρ, which is a strictly positive scalar, an *augmentation*

coefficient. The use of the augmentation term ensures that every (global) minimizer of $\bar{f}_{\bar{\mathbf{z}}}$ in S is Pareto optimal for (1) [20].

2.2. Design optimization

Let us consider a physical system and its mathematical model represented by a multivariate, vector-valued function Φ. By multiobjective design optimization we understand the determination of the input to Φ so as to simultaneously minimize two or more objective functions that depend on the output of Φ. The input values that are to be determined by optimization are called *decision* or *design variables*. If the decision variables constitute only a subset of the model input, the rest are considered to be *parameters* of Φ, which must be set to a fixed value before the optimization is carried out. In the terminology of the introduction, the decision variables and parameters constitute the independent variables, and the output of Φ constitutes the dependent variables.

If the model Φ is stochastic, then its output is a random vector. Therefore, the image of the output of Φ under the objective functions is also a random vector. To apply optimization, the random objective vector must be substituted with a vector of moments of the distribution, which can be numerically estimated. For example, a multiobjective design optimization problem concerned with the minimization of the expected value of the objective vector can be written as

$$\min_{\mathbf{x}} \quad E\{\mathbf{f}(\Phi(\mathbf{x}, \omega))\} \quad \text{s. t.} \quad \mathbf{x} \in S, \tag{3}$$

where $E\{\cdot\}$ denotes the expected value, $\mathbf{f}$ is a vector-valued objective function, $\mathbf{x}$ is a decision vector, ω is a random vector which represents the stochasticity in the model, and S is the feasible set.

In practice, a numerical simulation model ϕ that is implemented in terms of pseudorandom numbers is used to sample the output of the mathematical model Φ. This allows, for example, the expected value of the objective vector to be estimated by a sample mean. That is, instead of (3), we consider an MOP

$$\min_{\mathbf{x}} \quad \frac{1}{N} \sum_{i=1}^{N} \mathbf{f}(\phi_i(\mathbf{x})) \quad \text{s. t.} \quad \mathbf{x} \in S, \tag{4}$$

where $\phi_i(\mathbf{x})$ denotes for all $\mathbf{x} \in S$ the output of the ith independent simulation run, or *replication*, at $\mathbf{x}$ and N is the sample size (the number of replications). Problem (4) is a black-box optimization problem because the

simulation output $\phi_i(\mathbf{x})$ is not known in a closed form as a function of $\mathbf{x}$, but must be obtained numerically for each $\mathbf{x} \in S$.

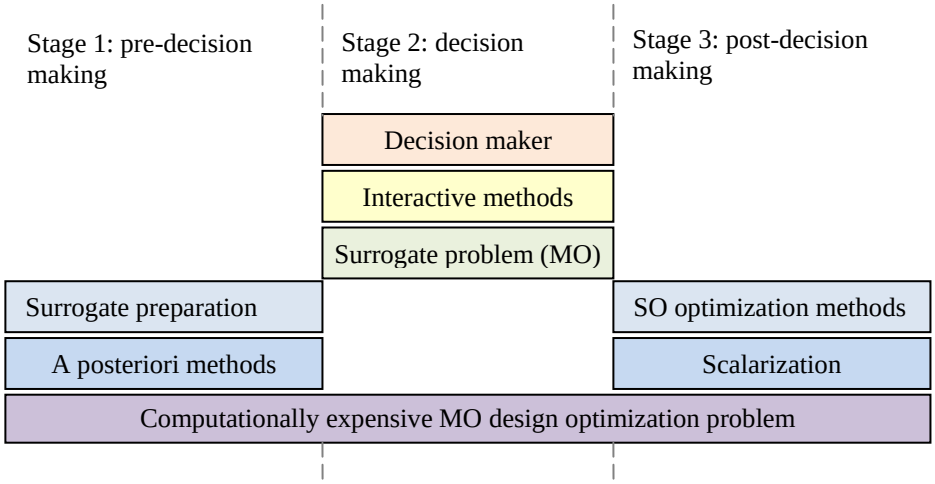
3. A three-stage solution process for multiobjective design optimization

Hartikainen *et al.* [17] proposed a method to approximate the Pareto optimal set of an MOP. Based on the approximation, they formulated a multiobjective surrogate problem with the same number of objectives, and approximately the same Pareto optimal set, as the original optimization problem. The objectives of the surrogate problem correspond to those of the original problem in the sense that they have the same interpretations, which allows the surrogate problem to be used for decision making in place of the original problem. Moreover, because the surrogate problem is computationally inexpensive, its use eliminates the waiting times in the interactive process and hence improves the applicability of interactive methods to computationally intensive multiobjective optimization.

The use of a surrogate problem for real-life decision making consists of steps that fall naturally into three separate stages. In this section, we develop the idea of a three-stage solution process as a way to solve a computationally expensive multiobjective design optimization problem. The consecutive stages can be distinguished by several features such as the computational cost, the level of involvement of the DM, or the type of optimization applied. To emphasize the role of the DM in the process, we refer to the stages by their precedence with respect to decision making as the *pre-decision making* stage, the *decision making* stage, and the *post-decision making* stage. The DM is supposed to be directly involved only in the decision making stage, and all computationally intensive tasks should, thus, be confined to the pre- and post-decision making stages so as not to interfere with the interaction with the DM.

The stages along with their main components and the dependences among them are illustrated in Figure 1. The solution process starts with the pre-decision making stage, in which an *a posteriori* method is applied to the multiobjective design optimization problem to produce a set of Pareto optimal decision vectors and the corresponding set of objective vectors is used to define a surrogate problem. In the decision making stage, the computationally expensive design optimization problem is substituted with the computationally inexpensive surrogate problem, and the DM is asked to identify

Figure 1: Schematic of multiobjective design optimization in three stages (MO – multiobjective, SO – single-objective)



the preferred objective vector with the help of an interactive method. In the post-decision making stage, the original multiobjective design optimization problem is solved in a scalarized form to project the objective vector (of the surrogate problem) preferred by the DM to the Pareto optimal set of the original design optimization problem. Thus, the post-decision making stage can be likened to the application of an *a priori* method to the original design optimization problem.

All three stages are necessarily coupled because each one depends on the output of its predecessor. At each stage, a different kind of optimization problem is solved and the result thereof is passed to the following stage. To ease discussion, we refer to the principal output of each stage as its *solution*. More specifically, a *Stage 1 solution* is a set of Pareto optimal decision vectors produced in the pre-decision making stage and the corresponding objective vectors are used to define the surrogate problem. A *Stage 2 solution* is an objective vector indicated by the DM in the interactive decision making stage as the preferred solution to the surrogate problem, and a *Stage 3 solution* is a projected decision vector obtained in the post-decision making stage by solving a scalarized version of the original design optimization problem.

3.1. Stage 1: pre-decision making

The purpose of the pre-decision making stage is to prepare a surrogate problem that enables an interactive method to be applied in the decision

making stage. Additionally, the ideal vector and the nadir vector of the surrogate problem, or their estimates, should be determined.

In principle, there are only two requirements for a surrogate problem: it must be solvable by scalarization (employed as a part of an interactive method) with a negligible computational cost and its Pareto optimal set must match that of the original problem, up to some tolerance. We assume that the surrogate problem can be defined on the basis of a set of Pareto optimal decision or objective vectors. Therefore, we define the Stage 1 solution as a set of Pareto optimal decision vectors obtained by applying an *a posteriori* method to the original design optimization problem.

The PAINT method [17] can be used to formulate a surrogate problem. It produces a piecewise linear approximation of a set of Pareto optimal objective vectors and guarantees that no two vectors in the approximation dominate one another. The surrogate problem based on the approximation is a multiobjective mixed-integer linear optimization problem.

3.2. Stage 2: decision making

The purpose of the decision making stage is, as the name suggests, to help the DM to arrive at the preferred Pareto optimal solution. Because the choice of the preferred Pareto optimal objective vector is entirely subjective, an interactive method, which allows the DM to explore and learn about the problem, is used. We assume that an interactive method employing scalarization is used (see, *e.g.*, [20, 21]). For an interactive method to be successfully applied, however, it is required that scalarizations of the underlying optimization problem can be solved relatively quickly. Therefore, the surrogate problem defined in Stage 1 is used instead of the original, computationally expensive, design optimization problem.

The surrogate problem in itself is like any other multiobjective optimization problem and thus no special provisions are required to solve it with an interactive method. Nonetheless, the DM should be made aware of the fact that a surrogate problem is being solved and that the obtained objective function values are only approximate.

3.3. Stage 3: post-decision making

The purpose of the post-decision making stage is to determine a decision vector in the Pareto optimal set of the original design optimization problem such that the objective vector corresponding to it is as close to the Stage 2 solution as possible. In principle, it suffices to minimize a scalarization that

projects the Stage 2 solution to the Pareto optimal set of the original design optimization problem, but the amount of computation required to do so can be reduced by utilizing the information acquired in Stage 1.

Because the optimization problems in Stages 1 and 3 are identical except for the scalarization, the information obtained about the problem in Stage 1 can be used to *warm start* the optimization in Stage 3. A warm start may consist of selecting a starting point, or an initial population in the case of population-based algorithms, among the decision vectors evaluated in Stage 1 based on the value of the scalarized objective function. It is also possible to warm start optimization algorithms with more complex internal states given that the algorithms employed in the two stages are similar enough so that the internal state of one can be translated into that of the other. In particular, the metamodels of the objective functions maintained internally by many global optimization algorithms can be aggregated to obtain a metamodel of the scalarized objective function.

The computational cost of Stage 3 can be reduced virtually to zero by projecting an objective vector not to the Pareto optimal set but to the set of objective vectors corresponding to the Stage 1 solution. This can be done by calculating the scalarized objective function value for all decision vectors in the Stage 1 solution and selecting the one with the least value. Although restricting the minimization to the Stage 1 solution does not, in general, minimize the scalarized objective function over the feasible set, the obtained decision vector is always Pareto optimal. Because of its low computational cost, the projection to the Stage 1 solution may be used also during the interactive decision making in Stage 2 to provide the DM with a rough estimate of the projection of a given objective vector.

3.4. Discussion

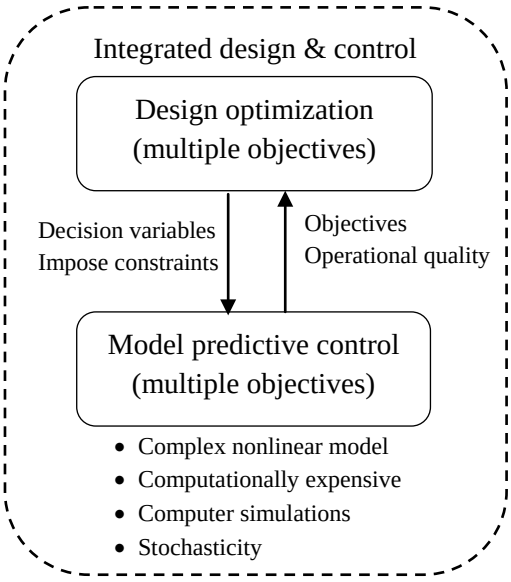
So far, we have assumed that all the optimization problems encountered in the different stages can be solved to global optimality. In practice, however, this is rarely the case. Instead, one may only be able to say about the Stage 1 solution that no decision vector in it dominates each other. In itself, the three-stage solution process does not require Pareto optimality of any of the decision vectors involved. If the decision vectors in the Stage 1 solution are not Pareto optimal, this naturally decreases the accuracy of the approximation used in Stage 2 and it may happen that the Stage 3 solution dominates one or more of them, but this does not affect the applicability of the solution process.

4. A case study: optimal design of a new paper mill concept

Our case study arises in papermaking industry and addresses a design optimization problem of a paper mill. The optimization problem to be considered has all the computational challenges discussed in the introduction. Three key ingredients: wood, water and energy are essential for papermaking which is a very complex process consisting of many steps. The first step is the preparation of chemical and/or mechanical pulp, i.e., the concentrated mixture of separate wood fibers and water. Before pulp enters the paper machine, some chemical additives are mixed in to ensure specific characteristics of paper being produced. Then the pulp mixture is pumped into a headbox, from where it is screened, drained, and passed through a moving wire where a sheet of paper called a web is formed. Next, it goes to press and drying sections where the excess of water is removed to ensure uniform thickness and smoothness. The paper can be coated to improve printing properties and/or calendered to improve the smoothness or the gloss of the paper. The final step is paper cutting into rolls or sheets. The papermaking process is normally continuous. However, sometimes web breaks occur generating broke, the discarded paper which is recycled in the process. For more details of a papermaking process, see, e.g., [24].

This case study is based on a research project and the model employed in this study was developed in [25, 26, 27, 28]. Here we discuss optimization-related aspects with the given model. Since design and control of a paper mill are interdependent, we consider them simultaneously, i.e., we optimize both design and control objectives at the same time. Optimizing the process design and control at the same time enables a designer or a DM to get a deeper insight about the dynamic behavior of a complex process. A simultaneous approach to the design and control leads to significant economic benefits and improved dynamic performance during plant control [29]. For overviews of integrated design and control, we refer, e.g., to [29, 30, 31]. In particular, the integrated design and control problem in papermaking has been studied in [25, 26] where the optimization is done by evaluating a predetermined number of decision vectors and then allowing the DM to choose the preferred decision vector from nondominated ones. In integrated design and control, the control equipment is designed at the same time as the rest of the process to determine the optimal design and control conditions of a process. The goal of simultaneously optimizing design and control of a process often requires to find a compromise between conflicting design and control objectives.

Figure 2: Problem structure



The optimization model is assumed to exhibit a hierarchical structure (see Figure 2), i.e., the upper level of the problem corresponds to the design problem, while the lower level is the operational performance problem, both involving multiple objectives. The operational performance is quantified in terms of the expected risk for tower under- and overflows and the expected uniformity of paper strength and light scattering. In order to optimize the performance of a paper mill, the control of the process must be optimized. To summarize, the goal is to decrease the investment cost and enhance the quality of paper (on the design level), and at the same time to guarantee the runnability and process stability of the production systems (on the operational level).

Section 4.1 describes a simulation model. The design problem is presented in Section 4.2. The operational performance problem is formulated in Section 4.3. For more detailed problem description we refer to [25]. Figure 2 outlines the main characteristics of our problem which we discuss in what follows.

4.1. A simulation model

The objectives of the design problem are calculated based on the outputs of a simulation model which is used to simulate a paper production process

in a paper mill. A set of decision variable values is given to the following simulation model and the corresponding output values are produced:

$$(d_1, d_2, T_1, T_2) = \phi_i(V_1, ..., V_4, W, p_0),\ i = 1, ..., N, \tag{5}$$

where d_1 corresponds to squared deviation of paper strength from its nominal value (some desirable value), d_2 is squared deviation of paper light scattering from its nominal value, T_1 is a number of time steps, i.e., successive time points at which the model states and outputs are computed, used for simulation, and T_2 is a number of time steps when paper breaks before the coating process occurs. Furthermore, $V_1, ..., V_4$, W and p_0 are decision variables at the design level, i.e., V_i, $i = 1, ..., 4$, correspond to maximum volumes of a white water tower, an uncoated paper broke tower, a coated paper broke tower and a clear water tower, respectively, W is a scalarization parameter and p_0 is a risk level parameter to reduce the risk of under- or overflow at the operational level. As in (4), also in (5) N refers to the number of replications. The simulation is terminated when the first of the following stopping criteria is satisfied: the over- or underflow, i.e., when the level in at least one of the storage towers exceeded its upper or lower bound, or 40000 time steps have been taken which corresponds to approximately nine months of simulation time of paper mill operation.

Our simulation model is computationally very expensive to evaluate and assumptions like convexity and differentiability that are desirable for optimization cannot be made. The other characteristic of the problem is stochasticity. The frequency and duration of the web breaks are modeled by a Markov chain in which the probabilities of both a break to occur and a continuing break to end depend on the strength of the paper web [26, 27]. The strength of the web is determined by the proportions of the fiber components.

At the operational level, the simulation model is connected to model predictive control (MPC). In MPC, a control action is obtained by solving online, at each time step, a finite horizon optimal control problem in which the initial state is the current state of the paper mill. The optimization yields an optimal control sequence and the first control action in this sequence is applied to the paper mill. The paper mill simulation model provides the value of paper strength at the current time m which is used to predict future values of paper strength $q_1(m + 1)$,...,$q_1(m + K_H)$ (future behavior of the paper mill) for K_H time steps and the control variables are optimized based on the predicted output. After the simulation ends, the simulation outputs

d_1, d_2, T_1, T_2 are produced. For fundamentals of MPC, see, e.g., [32]. The MPC problem formulation is discussed in Section 4.3.

The step size in the simulation model is set to be 10 minutes, and a single replication requires roughly 2175 time steps on average. During the initialization of the simulator, sufficient memory must be preallocated for the time series data to be collected, which requires the maximum number of time steps to be set. We have chosen an upper limit of 40000 time steps which, for example, in Stage 1 was reached 158 times for a total of 75 different decision vectors where, in total, more than 100000 replications were simulated for a total of more than 1000 decision vectors. Compared to the relatively high number of replications, the maximum number of time steps was reached so rarely that the bias in the runtime estimates due to truncation was considered negligible. Each time, the simulation was started from fixed initial conditions. For all storage towers, the lower bound was set to zero and the upper bound was considered as a decision variable.

4.2. Design level

In our problem, the design of a paper mill is defined by four large storage volumes called towers for white water (water containing small amounts of fiber and chemical additives, which drains through the paper machine wire, and is recycled into the paper pulp preparation system), uncoated paper broke (waste paper made during a web break before coating process), coated paper broke (waste paper to which a coating has been applied) and clear (i.e., pure) water. While smaller tower volumes are desirable from capital cost investment point of view, larger tower volumes are desired for higher paper quality and easier control, i.e., tower volumes act as a buffer and enable slower control. Slower control is attractive because fast control causes disturbances that result in deviations in paper quality. Therefore, our target is to find a trade-off between the investment cost of storage towers and the operational performance which is affected by stochastic paper breaks during a manufacturing process.

The design problem consists of five objectives: i) minimization of the expected long term variation in paper strength, ii) minimization of the expected long term variation in a light scattering coefficient which is monitored during the computer simulation, iii) minimization of the expected proportional time spent in uncoated paper breaks, iv) maximization of the expected system runtime, i.e., time till one of the towers runs empty or overflows and v) minimization of investment cost of storage towers. The mathematical model of

the design problem is formulated based on (3) as:

$$\begin{array}{ll} \min & f_1(V_1,...,V_4,W,p_0) = E\{d_1\}, \\ \min & f_2(V_1,...,V_4,W,p_0) = E\{d_2\}, \\ \min & f_3(V_1,...,V_4,W,p_0) = E\{\frac{T_2}{T_1}\}, \\ \max & f_4(V_1,...,V_4,W,p_0) = E\{T_1\}, \\ \min & f_5(V_1,...,V_4,W,p_0) = \sum_{i=1}^{4} H(V_i) \\ \text{s.t.} & 500 \leq V_i \leq 4000,\ i = 1,...,4, \\ & 0.1 \leq W \leq 1, \\ & 0.001 \leq p_0 \leq 0.01, \end{array} \tag{6}$$

where $H(V_i)$ is the investment cost of the ith tower. The expected system performance is based on the optimal operational policy, i.e., selecting an appropriate level of pulp, water and broke dosage.

In this paper, we consider a classical approach in stochastic optimization to find optimal design and control under uncertainty and address the stochasticity issue. Therefore, the expected values of the objective functions with respect to uncertainty are calculated as sample mean values (see (4)) per time step as in [25]. We would like to clarify that for the first two objectives, a sample per time step is used, while for the next two objectives, a sample per replication is used. This implies that we cannot select the number of samples for the first two objective functions as it is determined by the number of both replications and simulation time steps and the latter depends on the termination conditions. Every time we want to evaluate a decision vector, we need to call the simulation model for which a single replication is computationally demanding taking from minutes to several hours. Our case becomes computationally extremely expensive because we need multiple replications for a single decision vector. Fortunately, multiple replications can be very well carried out in parallel leading to considerable speed up by means of parallelization. When a simulation is time consuming, optimization methods devoted to computationally expensive problems should be applied.

4.3. Operational level

At the operational level the target is to dose pulp and water from the storage towers to the papermaking process. As mentioned earlier, the operational optimization problem is formulated as an MPC problem. The objective is to minimize paper quality variation that is defined by a strength parameter over a certain time horizon. The strength variation is determined as

$(q_1(m+l)-q_0)^2$, where $q_1(m+l)$ is a predicted strength at l time steps from the present time m and q_0 is the nominal value of the strength parameter (a prediction model is provided in Appendix A (A.1g)). The control vector $u(m)$ at the current time m consists of seven control variables $u(m)=(u_1(m),...,u_7(m))^T$ which define the dosage of pulp, water and broke to and from a particular storage tower. In order to prevent fast changes in the control vector $u(m)$, a second objective is introduced. The operational problem is formulated as follows:

$$\begin{aligned} &\min \sum_{l=0}^{K_H-1} \gamma(l)(q_1(m+l+1)-q_0)^2 \\ &\min \sum_{l=0}^{K_H-1} \gamma(l)(u(m+l)-u(m+l-1))^2 \\ &\text{s.t. (A.1) (see Appendix A)}, \end{aligned} \tag{7}$$

where $\gamma(l)$ is a time-wise weighting factor and K_H is the optimization horizon.

There is a rather small number of publications devoted to multiobjective MPC [33, 34]. Usually multiple objectives are transformed to a single-objective problem and solved using traditional optimization methods [35]. The operational problem is solved as a scalarized quadratic single-objective optimization problem

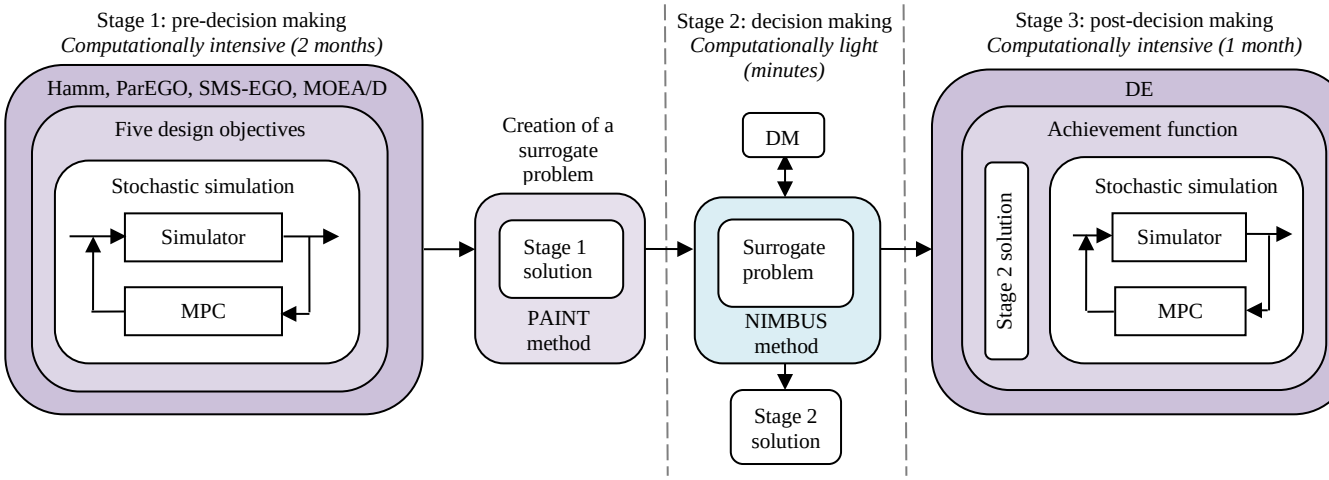
$$\begin{aligned} \min \sum_{l=0}^{K_H-1} \gamma(l)\big(&W(q_1(m+l+1)-q_0)^2+(u(m+l) \\ &-u(m+l-1))^T\alpha(u(m+l)-u(m+l-1))\big) \\ \text{s.t. } &(A.1), \end{aligned} \tag{8}$$

where α is a penalty term and W is a scalarization parameter treated as variable and bounds for its values are given in (6). Since the multiobjective optimization problem is a scalarized quadratic problem, it can be solved online without a DM, i.e., there is no need to apply any interactive approach and get any preference information from the DM at each time step solving the problem because different W values lead to different solutions, and an MPC procedure can be applied to optimize the dosage of recycled broke and water.

5. A three-stage solution process applied to the case study

After having introduced the case study, we devote this section to the application of the three-stage solution process to solve it. At the same time we describe how we have tackled the computational challenges related to this practical application. As proposed in Section 3, the solution process is divided into three stages as shown in Figure 3. These stages can also be seen in Figure 1 only on a more general level.

Figure 3: Schematic of the three-stage solution process applied to the case study



In order to solve an optimization problem, a suitable optimization method has to be applied. As mentioned earlier, the case study includes a black-box formulation which raises a requirement for global optimization. Many global optimization methods require a large number of objective function evaluations that we could not afford due to a time limit. Methods to solve computationally expensive black-box problems have been developed by exploiting knowledge acquired during the solution process [36, 37]. Knowledge of past evaluations can also be used to build an empirical model that approximates the objective function to be optimized. This approximation can then be used to predict promising new solutions at a smaller evaluation cost than that of the original problem [18, 38]. One of the state-of-art methods for costly optimization problems is EGO [37].

In Stage 1, we applied two versions of EGO for multiobjective optimization problems, namely, ParEGO [18] and SMS-EGO [19]. The experiment

performed in [19] demonstrated a significantly better performance of SMS-EGO over ParEGO on considered test problems. SMS-EGO is developed to handle noisy data, which is very desirable in our case where four design objective functions are calculated based on the stochastic simulation model output. To test multiobjective evolutionary methods, we selected the MOEA/D method which demonstrated good performance in solving unconstrained problems in the CEC'09 competition [39]. It should be noted that none of these optimization methods can guarantee global optimality as they use heuristics. It is important to mention that the three-stage solution process does not necessitate many different methods being applied in Stage 1. We used three methods as we were interested in their performance in the case study.

The surrogate problem was prepared by the PAINT method [17] and then solved in Stage 2 by the interactive NIMBUS method [40, 41] as shown in Figure 3. To project the Stage 2 solution onto the Pareto optimal set of the original problem in Stage 3, we solved problem (2) with the differential evolution (DE) method [42]. This method has been widely applied in many applications [43] as it is able to find approximate solutions to noisy, black-box optimization problems.

In this section, we present the results obtained in the following setting. The simulation model was implemented in MATLAB and subsequently compiled into a stand-alone executable so that multiple simulations could be run in parallel using a computer cluster consisting of 160 cores that did not have MATLAB installed. The quadratic optimization problem arising from the optimal control of the simulated plant by MPC was solved using the IBM ILOG CPLEX optimization software. The surrogate problem was solved on a computer with the following characteristics: Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz, 3401 Mhz, 4 Core(s), 8 Logical Processor(s).

An analysis performed before the actual solution process is described in Section 5.1. The following sections describe each stage in detail.

5.1. Preliminary analysis

Before any optimization was performed, we evaluated a sample of 256 decision vectors with 160 replications. The decision vectors were obtained from a six-dimensional Hammersley sequence, which is one of the so-called low discrepancy or quasirandom sequences that fill the n-dimensional Euclidean space more evenly than uniformly distributed random vectors [44, 45]. Because the vectors in the Hammersley sequence lie in the n-dimensional unit

interval, an affine transformation was applied that maps the unit interval onto the interval defined by the bound constraints in (6). The number of samples was selected to be a power of 2 so that it would be easy to divide the sequence into equally sized parts for parallel evaluation.

The initial sampling was done in part to help verify that the simulation model and the interfacing code work as intended. In addition, the sampling provided useful data about the optimization problem and allowed us to estimate the amount of time and computational resources required by the optimization runs. We observed that the simulation time for a single decision vector varied greatly between replications (from few minutes up to 5 hours), which resulted in an uneven load when parallelizing because a small number of runs would continue long after the others were completed. We chose to use 80 replications, a half of the number of cores in the computer cluster, to evaluate the objective functions in Stages 1 and 3. This allowed us to obtain relatively accurate estimates of the expected values while being able to carry out all the runs with the different optimization algorithms in parallel without exceeding, except momentarily, the capacity of the computer cluster. Also we performed the correlation analysis with the aim to check whether there was any strong positive correlation between the objectives in order to eliminate one of them from the optimization model. However, no strong linear correlation was found.

5.2. Stage 1: pre-decision making

This section is devoted to the results of Stage 1. Method performance varies depending on problem characteristics. While a method can demonstrate a very good efficiency for one type of problems, its performance can significantly degrade in coping with other ones. Thus, we extended Stage 1 to include several methods: ParEGO, SMS-EGO and MOEA/D. We applied three versions of SMS-EGO: one without smoothing technique (SMS-EGO-0) and two with different smoothing models (SMS-EGO-1 and SMS-EGO-2) [46]. The budget of objective function evaluations per each method was limited to 200. It should be noted that by applying different multiobjective optimization methods we aimed at generating a better approximation of the Pareto optimal set.

As the methods used are all population based, they each produced a set of decision vectors (and corresponding objective vectors). These sets were merged and vectors dominated by some others were removed. After this, the Stage 1 solution consisted of 394 decision vectors. Note that Pareto

optimality cannot be guaranteed because of the nature of the methods used. How each of the methods contributed to the Stage 1 solution can be seen in Table 1 which shows how many decision vectors (not dominated by each other) were generated by each method and how many of them remained in the Stage 1 solution. Results of Hammersley sequence sampling were included as well. For example, SMS-EGO-1 produced a smaller set than other methods consisting only of 102 decision vectors out of 200 objective function evaluations. However, it provided the largest contribution of 91 decision vectors to the Stage 1 solution. A further discussion on method performance can be found in Appendix B (as comparison is not a main scope of this paper). The minimal and maximal values of the objective functions are shown in Table 2 while different pairwise projections of 394 objective vectors corresponding to the Stage 1 solution are presented in Figures 4 and 5.

Table 1: Contribution of different methods to the Stage 1 solution

	# of non. vectors	# in the Stage 1 solution
Hamm	99 out of 256	55
SMS-EGO-0	108 out of 200	70
SMS-EGO-1	102 out of 200	91
SMS-EGO-2	107 out of 200	50
ParEGO	134 out of 200	80
MOEA/D	116 out of 200	58
Combined set	-	394

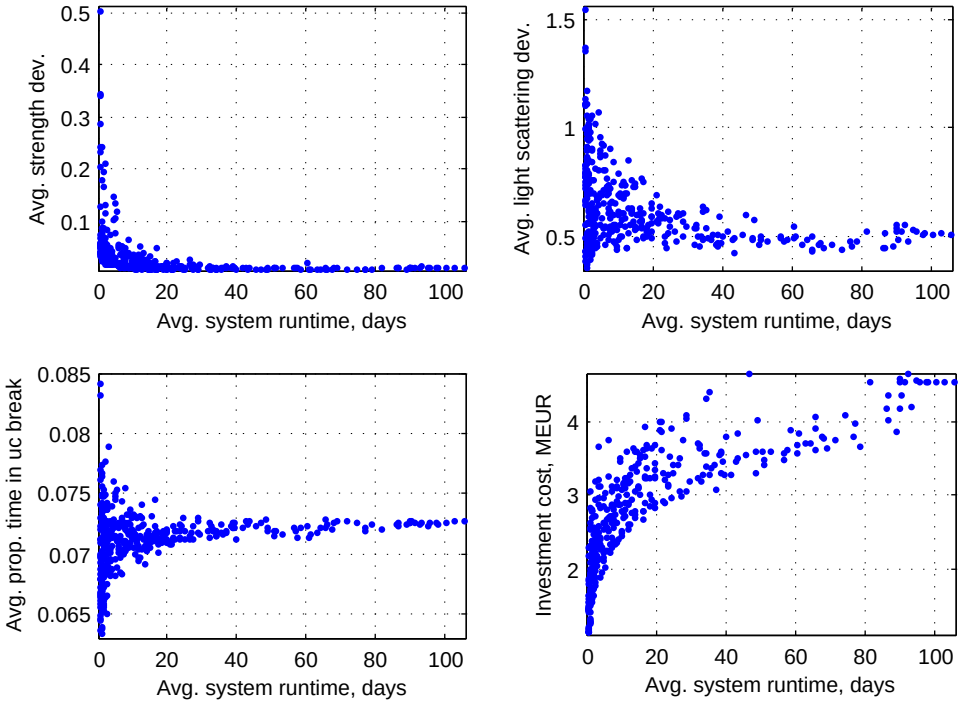
Table 2: Min and max values of the objective functions in the Stage 1 solution

	f_1	f_2	f_3	f_4	f_5
Min	0.00658	0.34785	0.06341	0.55851	1.14118
Max	0.50354	1.55069	0.08415	105.5457	4.66508

Figures 4 and 5 give a general idea of the trade-offs involved in the Stage 1 solution. They present different projections of the five-dimensional vectors in the objective space. The figures demonstrate that the strongest pairwise

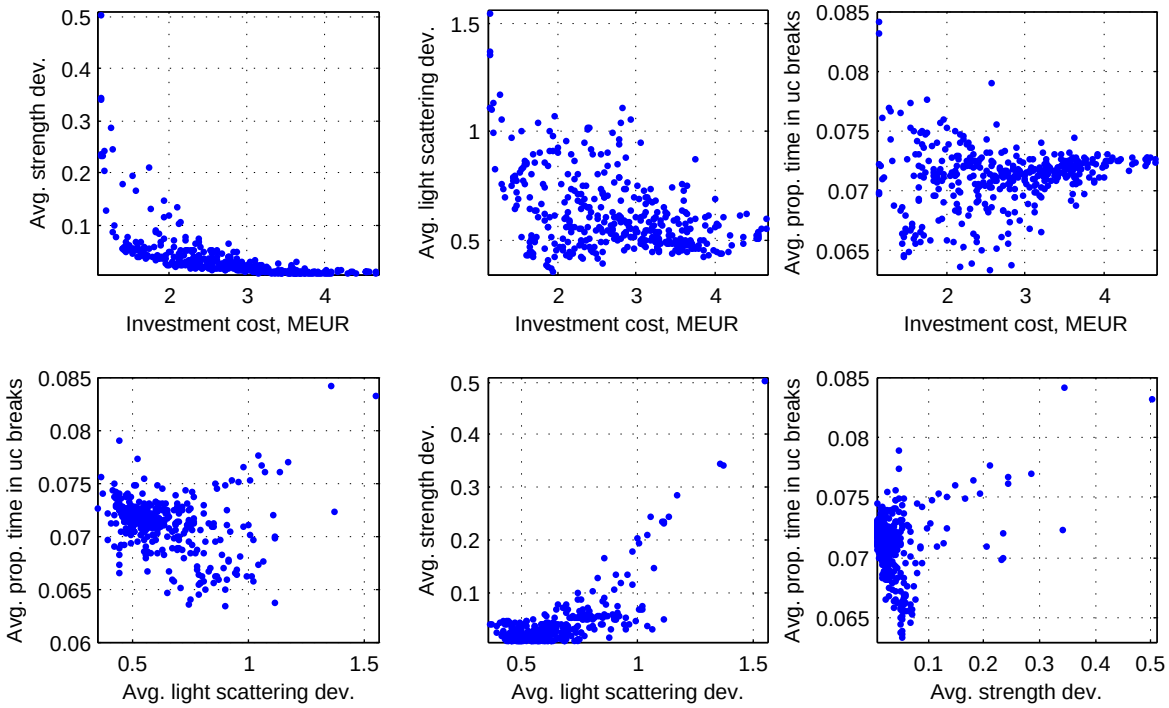
conflict between the objectives appears to be between minimizing the investment cost on the one hand and maximizing the average system runtime and minimizing the average strength deviation on the other hand. The former can be explained by the fact that greater tower volumes result in longer system runtime before an overflow occurs but also in greater investment cost. The latter results from the fact that small tower volumes necessitate aggressive broke dosage, which is the main cause of strength deviation. The pairwise conflict is weakest between minimizing the average strength deviation and maximizing the average system runtime, which is explained by the fact that strength deviations increase the probability of breaks and thus make it more likely for an overflow to occur. The rest of the pairs of objectives show weak or moderate conflict with less apparent trends.

Figure 4: The Stage 1 solution in the objective space – projections with respect to f_4



The calculation time in Stage 1 was approximately 2 months. After the Stage 1 solution was found, we used PAINT [17] to create a surrogate problem. The surrogate problem had 5 objective functions as the original problem, 4542 continuous variables, 757 binary variables, 757 inequality con-

Figure 5: The Stage 1 solution in the objective space – projections with respect to f_1, f_2 and f_5



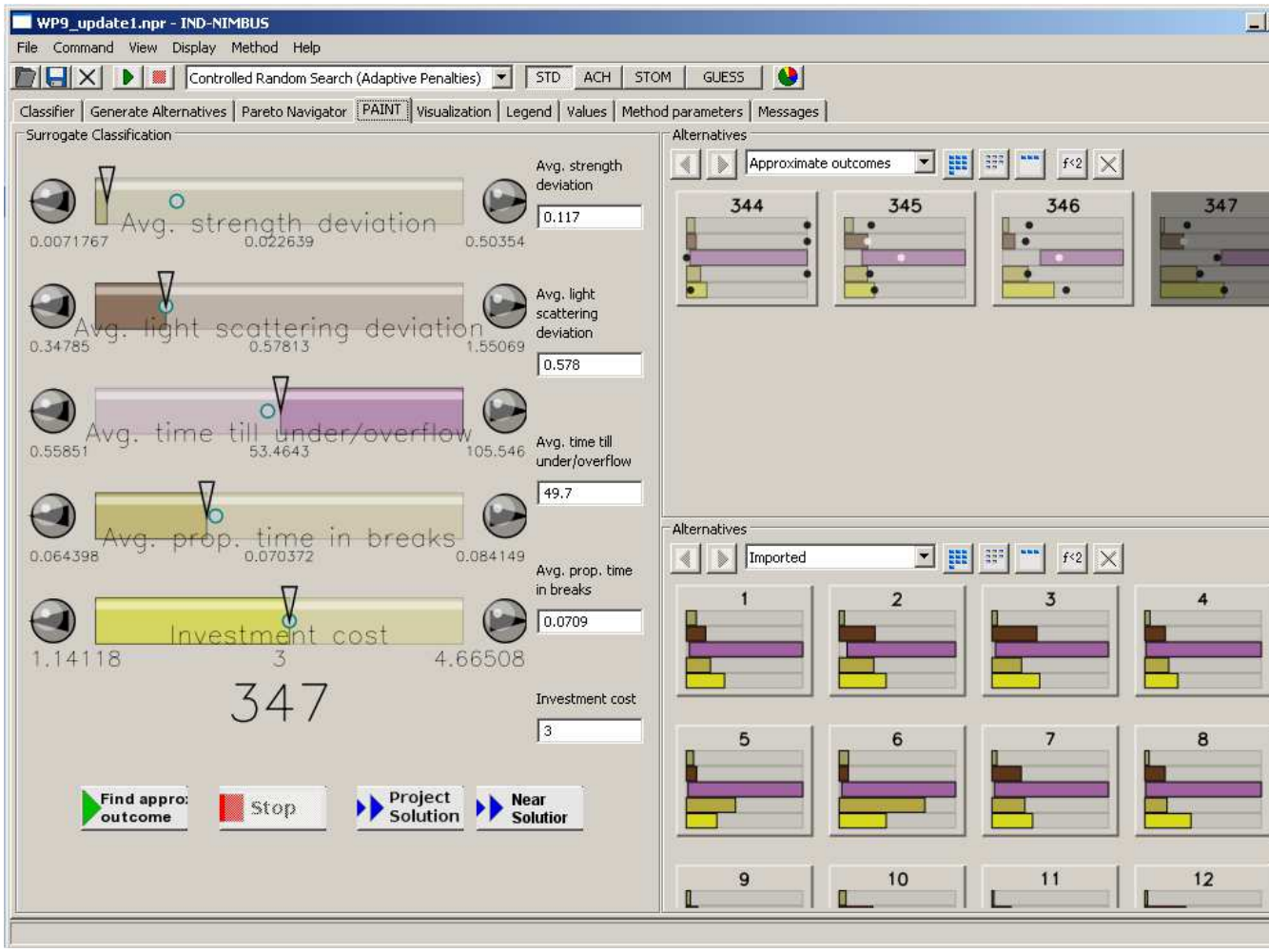
straints and 2 equality constraints. Because it was a multiobjective mixed-integer linear optimization problem, it was computationally inexpensive to solve. The calculation done in this stage took approximately 2 months. After Stage 1, Stage 2 could be started with the DM.

5.3. Stage 2: decision making

To effectively cope with the challenge of dealing with multiple conflicting objectives and to support the DM in a search of the preferred solution, we used in Stage 2 the IND-NIMBUS software [47, 48] implementing the interactive NIMBUS method [40, 41]. The DM directed the search in finding the best solution of the surrogate problem according to one's preferences with IND-NIMBUS. At each iteration, IND-NIMBUS solved the surrogate problem and showed an objective vector to the DM and asked him/her to provide preference information of how it should be improved. The DM decided whether each of the objective function values of the current solution: i) should be improved as much as possible, ii) should be improved until a given aspiration level, iii) is satisfactory at the moment, iv) is allowed to impair up to a given bound, or v) can be changed freely. Then, a new computationally inexpensive scalarized problem was formulated with the preference information and solved using CPLEX to get a Pareto optimal solution to the surrogate problem which reflected the preferences as well as possible. This process was continued until a final decision was made. Since decision making is an iterative process, we have a double arrow in Figure 3. Solving the surrogate problem created by PAINT enabled a decision making process without waiting times for the DM (generating Pareto optimal solutions for the surrogate problem took on the average 0.5 seconds).

In this case study, an initial solution randomly taken from the Stage 1 solution with objective function values of $f_1 = 0.01090$, $f_2 = 0.60110$, $f_3 = 0.07080$, $f_4 = 21.3947$ days and $f_5 = 3.875$ M€ was provided to the DM. The DM was interested in an investment cost lower than 3 M€ and an average time until the under- or overflow longer than 5 days. The other objectives were allowed to change freely. The solution obtained for the surrogate problem with this preference information had the following objective function values: $f_1 = 0.02264$, $f_2 = 0.57813$, $f_3 = 0.07037$, $f_4 = 53.4643$ days, and $f_5 = 3$ M€. It is presented in Figure 6, which also demonstrates the user interface of IND-NIMBUS. The DM was satisfied with this solution and no more iterations were needed. Thus, this is the Stage 2 solution.

Figure 6: A screen shot of the graphical user interface of the NIMBUS method in the IND-NIMBUS software



5.4. Stage 3: post-decision making

The aim of Stage 3 is to find the preferred solution of the original problem based on the DM's preferences, namely the Stage 2 solution. As mentioned in Section 3.3, one can use a computationally inexpensive way if no further computations can be afforded.

In our case study, the decision vector of the Stage 1 solution having the closest objective vector to the Stage 2 solution in the objective space was $V_1 = 2287$, $V_2 = 2357$, $V_3 = 4000$, $V_4 = 855$, $p_0 = 0.001$ and $W = 0.4935$ when measured by the Euclidean distance. The same decision vector also produced the best value of $\bar{f}_{\bar{\mathbf{z}}}$ over the Stage 1 solution (equal to 0.087979). The corresponding objective vector is given in Table 3 (best in Stage 1).

Table 3: Objective vectors

	$\min f_1$	$\min f_2$	$\min f_3$	$\max f_4$	$\min f_5$
Stage 2 solution	0.02264	0.57813	0.07037	53.4643	3.00
best in Stage 1	0.01110	0.48563	0.07186	48.9504	3.31
Stage 3 solution	0.01016	0.47363	0.07137	48.7919	3.27

To actually project the Stage 2 solution to the Pareto optimal set of the original problem, we minimized achievement function (2) in S with the DE method using different mutation factor F and crossover rate CR values recommended in [42]. In order to reduce the overall computational cost and to benefit from calculations done in Stage 1, DE was also run with a warm start, i.e., the decision vectors producing the least values of the achievement function were selected as initial population members. The results obtained clearly demonstrate that better achievement function values were obtained by a warm start. Moreover, none of the runs without a warm start reached the starting level of runs with a warm start.

The Stage 3 solution was found by the DE run with parameters $F = 0.5$ and $CR = 0.1$ using 200 objective function evaluations. The decision vector had the values: $V_1 = 2287$, $V_2 = 2217$, $V_3 = 4000$, $V_4 = 855$, $p_0 = 0.001$ and $W = 0.7179$ (and the value of $\bar{f}_{\bar{\mathbf{z}}}$ being 0.077936) (for the corresponding objective vector, see Table 3). One can say that the Stage 3 solution matched better the DM's preferences than the closest objective vector based on the Stage 1 solution since it gave a smaller value of the achievement function $\bar{f}_{\bar{\mathbf{z}}}$.

The Stage 3 solution is the final solution of the original design optimization problem. Stage 3 was computationally demanding and took around 1 month as it solved the original problem.

Because the number of solution generated at Stage 1 is limited, one may wonder whether the information is accurate enough for Stage 2 to produce a good approximation. It is important to note here that thanks to the interactive method used, it is not critical to cover the whole Pareto optimal set equally accurately as long as the part of the set that is interesting to the decision maker is properly covered. Based on the closeness of the projected solution in Stage 3 we can see that the accuracy in this case was sufficient in the interesting area. (Otherwise, more solutions would have been needed in Stage 1.)

6. Conclusions

In this paper, we have highlighted some of the computational challenges of solving complex simulation-based design optimization problems involving multiple conflicting objectives and have proposed a three-stage solution process to tackle them. Applying this solution process was demonstrated with a case study where the optimal design of a paper mill was considered. Our case study is very challenging due to the following characteristics: i) high computational cost, ii) multiple conflicting criteria, iii) black-box problem, and iv) stochasticity. Despite these computational difficulties, the problem was successfully solved following the proposed solution process.

In Stage 1, different multiobjective optimization methods tailored for computationally expensive problems were applied to solve the original problem. Although a number of different methods were used to approximate the Pareto optimal set, it is not necessarily recommended that one should always use as many of them. Then based on the Stage 1 solution, a computationally inexpensive surrogate problem was created by the approximation method PAINT. The creation of the surrogate problem took time but solving the surrogate problem in Stage 2 of the solution process was not computationally expensive. The surrogate problem was solved by the interactive multiobjective optimization method NIMBUS without creating waiting times for the DM and enabling an efficient decision making process. In this stage, the DM was involved and asked to express one's preferences in order to direct the search towards the preferred approximated solution. In Stage 3, a single-objective optimization problem was solved to get a solution best matching

the Stage 2 solution by projecting it to the original problem. In order to reduce computational cost, the information obtained in Stage 1 was employed in Stage 3 as a warm start. The three-stage solution process proposed is not application-specific but can be applied in various design optimization problems facing computational challenges discussed.

Acknowledgements

This research was partly financially supported by the EffNet Research Program WP9 of FIBIC (former Forestcluster) Ltd., Tekes and COMAS (Jyväskylä Graduate School in Computing and Mathematical Sciences). The authors wish to thank Markus Hartikainen and Vesa Ojalehto from the Industrial Optimization Group at the University of Jyväskylä for valuable help, Tobias Wagner for the SMS-EGO implementation and the project partners in WP9 for providing the model.

Appendix A. Constraints of operational optimization

Constraints of the operational level problem (7) [25, 26, 27, 28]:

$$A_1 \sum_{l'=0}^{K_H-1} u(m+l') \leq V_{\max,1} - V_1(m) - lC_1 - B_1 F^{-1}_{Z_{b(m)}(l)}(1-p_1^{(up)})^l \tag{A.1a}$$

$$-A_1 \sum_{l'=0}^{K_H-1} u(m+l') \leq V_1(m) + lC_1 + B_1 F^{-1}_{Z_{b(m)}(l)}(1-p_1^{(low)})^l - V_{\min,1} \tag{A.1b}$$

$$A_i \sum_{l'=0}^{K_H-1} u(m+l') \leq V_{\max,i} - V_i(m) - lC_i - B_i F^{-1}_{Z_{b(m)}(l)}(1-(1-p_i^{(up)})^l), i=2,3 \tag{A.1c}$$

$$-A_i \sum_{l'=0}^{K_H-1} u(m+l') \leq V_i(m) + lC_i + B_i F^{-1}_{Z_{b(m)}(l)}(1-(1 - p_i^{(low)})^l) - V_{\min,i}, i=2,3 \tag{A.1d}$$

$$A_4 \sum_{l'=0}^{K_H-1} u(m+l') \leq V_{\max,4} - V_4(m) - lC_4 \tag{A.1e}$$

$$-A_i \sum_{l'=0}^{K_H-1} u(m+l') \leq V_4(m) + lC_4 - V_{\min,4} \tag{A.1f}$$

$$q_1(m+l) = q_1(m) + \sum_{l'=1}^{K_m} c(l')(u(m+l-l') - u(m-l')) \tag{A.1g}$$

$$u_{j,\min} \leq u_j \leq u_{j,\max}, \tag{A.1h}$$

where $V_{\min,i}$ and $V_{\max,i}$ stand for the minimum and maximum amounts of water or pulp in the towers, respectively, $1-\left(1-p_i^{(up/low)}\right)^k$ is a function for the accepted risk $p_i^{(up/low)}$ that the ith tower runs empty or overflows ($p_{1,4}^{up}=0$, $p_{2,3}^{up}=p_0$; $p_1^{low}=p_0$, $p_{2,3,4}^{low}=0$, where p_0 is a decision variable), and $F^{-1}_{Z_{b(m)}(l)}$ is the cumulative distribution of the number of breaks, A, B and C are matrices of break model parameters, $c(l')$ is a matrix of coefficients

(impulse response) obtained through step response tests by changing one control variable at a time and K_m is the number of steps in "the history" that is contributing to the dynamics of the model. The constraints (A.1a)-(A.1f) prevent the towers from running empty or overflowing; (A.1a)-(A.1b), (A.1c)-(A.1d), and (A.1e)-(A.1f) correspond to white water tower, uncoated and coated broke towers and clear water tower, respectively. A prediction model of paper strength is defined by (A.1g). The last constraint (A.1h) determines the minimum and maximum dosage of pulp, water and broke.

Appendix B. Performance of different methods in Stage 1

All the methods were run only once due to a high computational cost. The minimal and maximal values of objective functions in the nondominated sets produced by different methods are given in Table B.4. Here, the nondominated set refers to the subset of vectors which are not dominated by any other vector in the set considered. The bold face stands for the highest and lowest values of each objective function. This table demonstrates that SMS-EGO-0 found decision vectors with the highest values of f_1, f_2 and f_3 as well as the lowest value of f_5. SMS-EGO-2 obtained a decision vector with the lowest value of f_4, however it was dominated by one found by SMS-EGO-0. The ParEGO method provided decision vectors with the lowest value of f_2 and highest values of f_4 and f_5 while MOEA/D obtained decision vectors with lowest values of f_1 and f_3. The obtained nondominated sets (objective vectors) are shown in Figures B.7 and B.8 as various projections. They confirm that the ParEGO method better explored the ranges of f_4, i.e., it found decision vectors where the average system runtime was longer than 89.27 days and reached even 105 days, and it also provided decision vectors where f_2 had smaller values than 0.39.

It is difficult to analyze method performance based only on visual information which might not well represent reality (and based on only one run per method). To obtain a trend-setting estimate of method performance, we calculated hypervolume (HV) [49] and inverted generational distance (IGD) [50] metrics of the nondominated sets obtained by each method (see Table B.5). In order to calculate an IGD metric, one must know the true Pareto optimal set, which is usually unavailable in practical applications. Thus, we calculated IGD metric values with respect to the combined set (that is why the IGD value of the combined set is equal to 0). The experimental settings did not allow us to perform statistical analysis, as we could not afford running

Table B.4: Min and max values of the design objectives in the nondominated sets obtained by different methods

		f_1	f_2	f_3	f_4	f_5
Hamm	Min	0.00742	0.39052	0.06510	1.08941	1.89092
	Max	0.05279	0.96061	0.07894	55.1299	4.66252
SMS-EGO-0	Min	0.00834	0.42451	0.06587	0.55851	**1.14118**
	Max	**0.50354**	**1.55069**	**0.08415**	61.3110	3.89702
SMS-EGO-1	Min	0.00786	0.41991	0.06460	0.68481	**1.14118**
	Max	0.34199	1.36831	0.07760	89.2707	3.97326
SMS-EGO-2	Min	0.00793	0.40955	0.06587	**0.49254**	**1.14118**
	Max	0.39700	1.47611	0.07934	86.1966	4.19094
ParEGO	Min	0.00718	**0.34785**	0.06440	0.70503	1.14123
	Max	0.24286	1.13433	0.07732	**105.5457**	**4.66508**
MOEA/D	Min	**0.00658**	0.43716	**0.06341**	0.92014	2.07562
	Max	0.12823	1.12786	0.07282	47.7186	4.40904

Figure B.7: Nondominated sets of different methods in the objective space – projections with respect to f_4

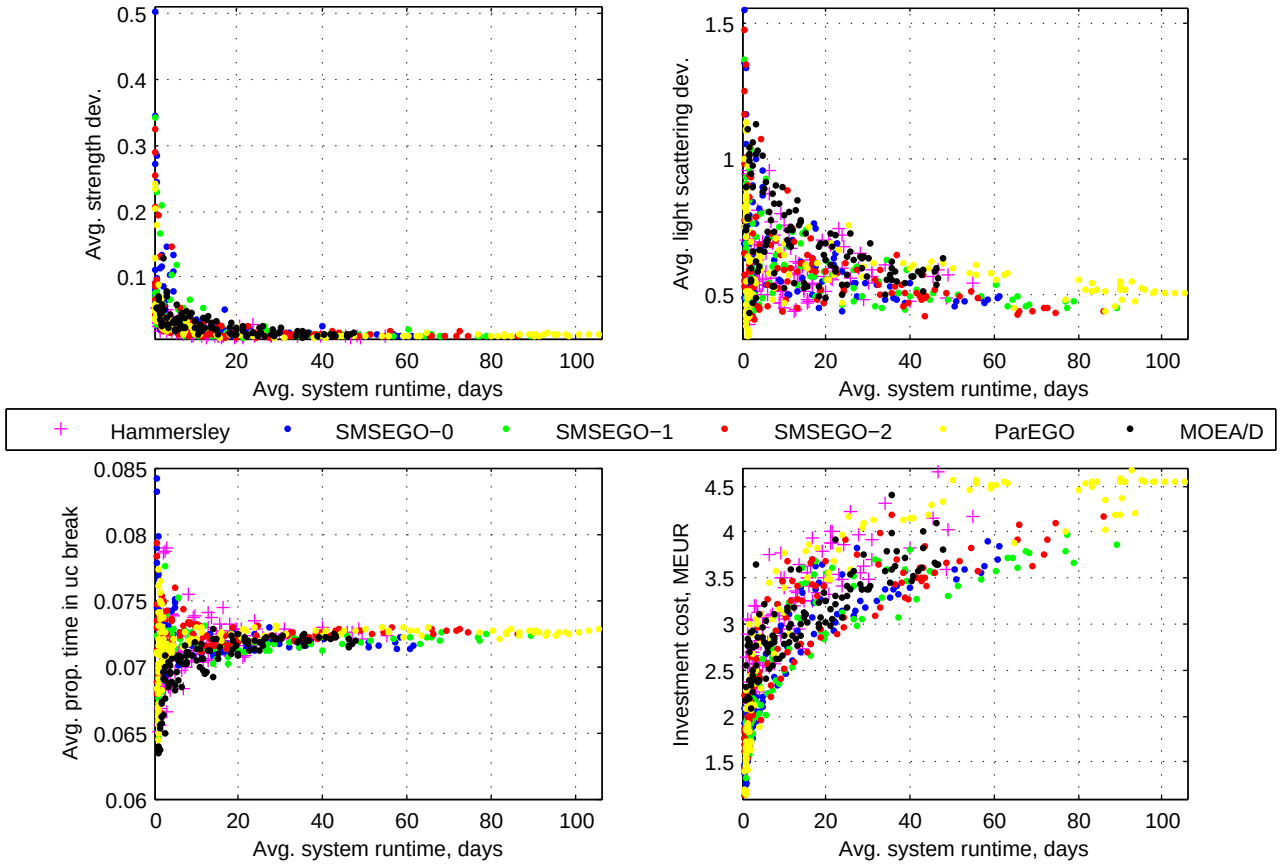
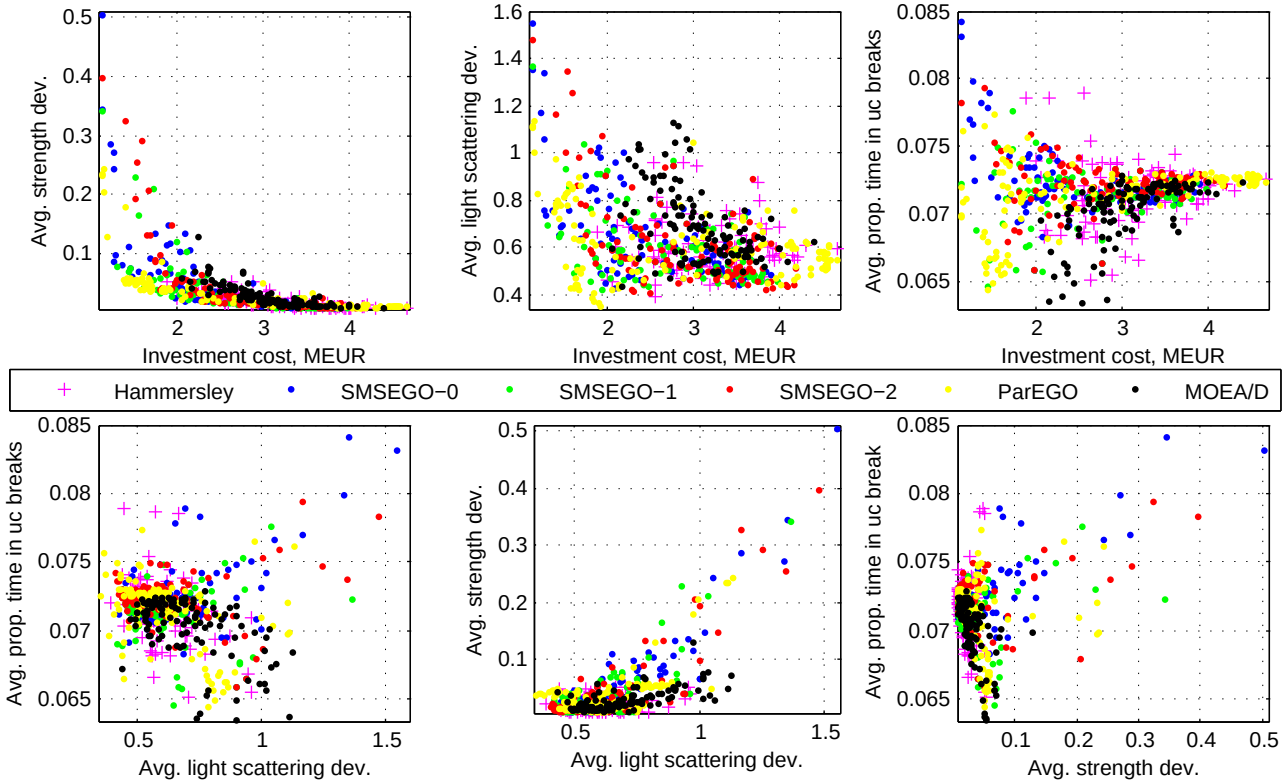


Figure B.8: Nondominated sets of different methods in the objective space – projections with respect to f_1, f_2 and f_5



methods more than once. Thus, the results obtained must be interpreted cautiously and have only an advisory value. According to these metrics, SMS-EGO-1 outperformed others because the HV metric value is the greatest (a greater HV value is better) and the IGD value is the lowest (a lower IGD value is better). We expected that all applied optimization methods would perform better than the Hammersley sequence sampling technique. HV values support this, while the largest IGD value obtained by MOEA/D showed its poor performance with respect to IGD values. Poor MOEA/D performance in our case can be explained by the fact that we had a very small budget of objective function evaluations and were forced to use a small population for MOEA/D while the recommended size is much higher.

Table B.5: Performance measures of different methods

	HV	IGD	Time, s
Hamm	0.5050	2.8405	**0.0391**
SMS-EGO-0	0.6682	2.1374	1506.73
SMS-EGO-1	**0.8919**	**0.5379**	1975.28
SMS-EGO-2	0.7648	0.7660	1874.28
ParEGO	0.5943	0.5656	1231.81
MOEA/D	0.5145	3.5249	11.125
Combined set	0.9597	0.0000	-

Table B.5 also provides computing time needed for each method calculations without objective function evaluations (i.e., excluding simulation time), which shows that the SMS-EGO method required more time than the other methods. The fastest among optimization methods was MOEA/D (at the cost of poor performance). It was clear at the beginning that Hammersley sampling takes the shortest time, because it only generates a set of decision vectors and does not perform other calculations as the optimization methods do.

References

[1] P. Y. Papalambros, The optimization paradigm in engineering design: Promises and challenges, Comput Aided Design 34 (2002) 939–951.

[2] R. Roy, S. Hinduja, R. Teti, Recent advances in engineering design optimisation: Challenges and future trends, CIRP Ann-Manuf Techn 57 (2008) 697–715.

[3] T. W. Simpson, V. V. Toropov, V. Balabanov, F. A. Viana, Design and analysis of computer experiments in multidisciplinary design optimization: A review of how far we have come - or not, in: Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, British Columbia, Canada, 2008, pp. 1–22.

[4] R. T. Marler, J. S. Arora, Survey of multi-objective optimization methods for engineering, Struct Multidisc Optim 26 (2004) 369–395.

[5] G. G. Wang, S. Shan, Review of metamodeling techniques in support of engineering design optimization, J Mech Design 129 (4) (2007) 370–380.

[6] A. I. Forrester, A. J. Keane, Recent advances in surrogate-based optimization, Prog Aerosp Sci 45 (1–3) (2009) 50–79.

[7] A. Kurpati, S. Azarm, Immune network simulation with multiobjective genetic algorithms for multidisciplinary design optimization, Eng Optimiz 33 (2) (2000) 245–260.

[8] R. Farmani, D. A. Savic, G. A. Walters, Evolutionary multi-objective optimization in water distribution network design, Eng Optimiz 37 (2) (2005) 167–183.

[9] M. Li, G. Li, S. Azarm, A Kriging metamodel assisted multi-objective genetic algorithm for design optimization, J Mech Design 130 (2008) 031401–1 – 031401–10.

[10] H. Liu, S. Maghsoodloo, Simulation optimization based on Taylor Kriging and evolutionary algorithm, Appl Soft Comput 11 (4) (2011) 3451–3462.

[11] R. Tappeta, J. Renaud, J. Rodríguezc, An interactive multiobjective optimization design strategy for decision based multidisciplinary design, Eng Optimiz 34 (5) (2002) 523–544.

[12] J. Hämäläinen, K. Miettinen, P. Tarvainen, J. Toivanen, Interactive solution approach to a multiobjective optimization problem in paper

machine headbox design, J Optimiz Theory App 116 (2) (2003) 265–281.

[13] E. Heikkola, K. Miettinen, P. Nieminen, Multiobjective optimization of an ultrasonic transducer using NIMBUS, Ultrasonics 44 (4) (2006) 368–380.

[14] K. Miettinen, J. Hakanen, Why use interactive multi-objective optimization in chemical process design?, in: G. Rangaiah (Ed.), Multi-Objective Optimization: Techniques and Applications in Chemical Engineering, World Scientific, 2009, pp. 153–188.

[15] J. Hakanen, K. Miettinen, K. Sahlstedt, Wastewater treatment: New insight provided by interactive multiobjective optimization, Decic Support Syst 51 (2011) 328–337.

[16] S. Taras, A. Woinaroschy, An interactive multi-objective optimization framework for sustainable design of bioprocesses, Comput Chem Eng 43 (2012) 10–22.

[17] M. Hartikainen, K. Miettinen, M. M. Wiecek, PAINT: Pareto front interpolation for nonlinear multiobjective optimization, Comput Optim Appl 52 (2012) 845–867.

[18] J. Knowles, ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems, IEEE T Evolut Comput 10 (1) (2006) 50–66.

[19] W. Ponweiser, T. Wagner, D. Biermann, M. Vincze, Multiobjective optimization on a limited budget of evaluations using model-assisted S-metric selection, in: G. Rudolph, T. Jansen, S. Lucas, C. Poloni, N. Beume (Eds.), Parallel Problem Solving from Nature - PPSN X Proceedings, Springer, 2008, pp. 784–794.

[20] K. M. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers, Boston, 1999.

[21] K. Miettinen, F. Ruiz, A. P. Wierzbicki, Introduction to multiobjective optimization: interactive approaches, in: J. Branke, K. Deb, K. Miettinen, R. Słowiński (Eds.), Multiobjective Optimization: Interactive and Evolutionary Approaches, Springer, 2008, pp. 27–57.

[22] A. P. Wierzbicki, On the completeness and constructiveness of parametric characterizations to vector optimization problems, OR Spektrum 8 (2).

[23] A. P. Wierzbicki, A methodological approach to comparing parametric characterizations of efficient solutions, in: G. Fandel, M. Grauer, A. Kurzhanski, A. P. Wierzbicki (Eds.), Large-Scale Modelling and Interactive Decision Analysis, Vol. 273 of Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, 1986, pp. 27–45.

[24] C. J. Biermann, Handbook of Pulping and Papermaking, 2nd Edition, Academic Press, 1996.

[25] A. Ropponen, M. Rajala, R. Ritala, Multiobjective optimization of the pulp/water storage towers in design of paper production systems, Comp Aid Ch 29 (2011) 612–616.

[26] A. Ropponen, R. Ritala, E. Pistikopoulos, Broke management optimization in design of paper production systems, Comp Aid Ch 28 (2010) 865–870.

[27] A. Ropponen, R. Ritala, E. Pistikopoulos, Optimization issues of the broke management system in papermaking, Comput Chem Eng 35 (11) (2011) 2510–2520.

[28] A. Ropponen, R. Ritala, Operational optimization of paper flow management in papermaking, in: Proceedings of PaperCon 2012 Conference, New Orleans, Louisiana, USA, 2012, pp. 142–184.

[29] P. Seferlis, M. Georgiadis (Eds.), The Integration of Process Design and Control, Elsevier, 2004.

[30] V. Sakizlis, J. Perkins, E. Pistikopoulos, Recent advances in optimization-based simultaneous process and control design, Comput Chem Eng 28 (10) (2004) 2069–2086.

[31] L. Ricardez-Sandoval, H. Budman, P. Douglas, Integration of design and control for chemical processes: A review of the literature and some recent results, Annu Rev Control 33 (2) (2009) 158–171.

[32] J. M. Maciejowski, Predictive Control with Constraints, Prentice Hall, 2002.

[33] A. Bemporad, D. M. de la Pena, Multiobjective model predictive control, Automatica 45 (2009) 2823–2830.

[34] F. Bouani, K. Laabidi, M. Ksouri, Conventional and non-conventional methods for nonlinear multi objective predictive control, CEAI 8 (1) (2006) 59–69.

[35] W. Wojsznis, A. Mehta, P. Wojsznis, D. Thiele, T. Blevins, Multi-objective optimization for model predictive control, ISA Trans 46 (3) (2007) 351–61.

[36] A. Žilinskas, Axiomatic characterization of a global optimization algorithm and investigation of its search strategies, Oper Res Lett 4 (1985) 35–39.

[37] D. Jones, M. Schonlau, W. Welch, Efficient global optimization of expensive black-box functions, J Global Optim 13 (4) (1998) 455–492.

[38] L. Santana-Quintero, A. Montano, C. C. Coello, A review of techniques for handling expensive functions in evolutionary multi-objective optimization, in: Y. Tenne, C.-K. Goh (Eds.), Computational Intelligence in Expensive Optimization Problems, Vol. 2, Springer, 2010, pp. 29–59.

[39] Q. Zhang, P. N. Suganthan, Final report on CEC09 MOEA competition, Tech. rep., School of Computer Science and Electrical Engineering, University of Essex (2009).

[40] K. M. Miettinen, M. Mäkelä, Interactive bundle-based method for non-differentiable multiobjective optimization: NIMBUS, Optimization 34 (1995) 231–246.

[41] K. Miettinen, M. Mäkelä, Synchronous approach in interactive multiobjective optimization, Eur J Oper Res 170 (3) (2006) 909–922.

[42] R. Storn, K. Price, Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces, J Global Optim 11 (1997) 341–359.

[43] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, 2005.

[44] J. M. Hammersley, Monte Carlo methods for solving multivariable problems, Annals of the New York Academy of Sciences 86 (3) (1960) 844–874.

[45] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, S. Tarantola, Global Sensitivity Analysis. The Primer, John Wiley & Sons, 2008.

[46] A. I. J. Forrester, A. J. Keane, N. W. Bressloff, Design and analysis of 'noisy' computer experiments, AIAA J 44 (10) (2006) 2331–2339.

[47] K. Miettinen, IND-NIMBUS for demanding interactive multiobjective optimization, in: T. Trzaskalik (Ed.), Multiple Criteria Decision Making '05, The Karol Adamiecki University of Economics in Katowice, Katowice, 2006, pp. 137–150.

[48] IND-NIMBUS, accessed on 25 February, 2013.
URL `http://ind-nimbus.it.jyu.fi/`

[49] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, IEEE T Evolut Comput 3 (4) (1999) 257–271.

[50] H. Sato, H. Aguirre, K. Tanaka, Local dominance using polar coordinates to enhance multiobjective evolutionary algorithms, in: Congress on Evolutionary Computation, 2004. CEC2004, 2004, pp. 188–195.