Box search for the data mining of the key parameters of an industrial process

Q. Louveaux, A. Mathei, S. Mathieu*
Quartier Polytech 1, Allée de la Découverte 10, B-4000 Liège, Belgium
q.louveaux@ulg.ac.be, axel.mathei@gmail.com, sebastien.mathieu@ulg.ac.be

December 15, 2015

Abstract

To increase their competitiveness, many industrial companies monitor their production process, collecting large amount of measurements. This paper describes a technique using this data to improve the performance of a monitored process. In particular we wish to find a set of rules, i.e. intervals on a reduced number of parameters, for which an output value is maximized. The model-free optimization problem to solve is to find a box, restricted on a limited amount of dimensions, with the maximum mean value of the included points. This article compares a machine learning-based heuristic to the solution computed by a mixed-integer linear program on real-life databases from steel and glass manufacturing. Computational results show that the heuristic obtains comparable solutions to the mixed integer linear approach. However, the exact approach is computationally too expensive to tackle real life databases. Results show that the restriction of five process parameters, on these databases, may improve the quality of the process by 50%.

Keywords: Data mining, Integer programming, Heuristics, Industrial process, Quality management

*Corresponding author. sebastien.mathieu@ulg.ac.be, Phone: +32 4 366 2631

1 Introduction

Competitiveness is a key element to ensure the prosperity of an industry. Numerous factors may influence the success of a product. To identify these factors, industrial companies invest more and more in sensors to retrieve information on their processes. The measurements are commonly stored in large databases containing millions of tuples with hundreds parameters. Once in possession of these measurements, possibly cleaned to remove the most obvious errors, there is still the need of efficient and robust techniques to extract useful information from this big data. In this paper, our aim is to use the data in order to handle one of the following questions. Can we analyze the data in order to find a setting that improves the quality or the production costs of the industrial process, is it possible to identify on a batch of flawed products, the parameters that were badly tuned or on the contrary given a batch of exceptionally good products, was there a set of key parameters that were well tuned?

These data analysis questions can be mathematically formulated as follows. We assume that for each produced item i, a series of measurements are retrieved in a D-dimensional vector $\mathbf{x}^i$ of input parameters. These coordinates of the vector may be either continuous or categorical. Examples of continuous coordinates are temperature, pressure, flow of gas, etc. Categorical coordinates may represent types of raw material, gas composition, etc. The output value of the item i is given by $c^i \in \mathbb{R}$ and might be either positive or negative. c^i may represents the production cost of product i, its quality, etc. The question addressed in this paper is to find a box, i.e. for each continuous dimension t, an interval $[l_t, u_t]$ and for each categorical dimension a list of categories, which maximizes the mean values of the points included in the box. The purpose of a box is its simplicity and robustness. Robustness is further increased by imposing a minimal number of points in the box and a minimum interval for each dimension. Finally, we restrict ourselves to constrain the box on only N dimensions. As the databases may include many coordinates, the control of all parameters is difficult to apply in practice. This restriction also provides the key parameters that condition the output value and therefore provide a more intuitive solution. The challenge is to find, from the data analysis, a reduced set among a large set of parameters which, once controlled, leads to the least amount of downgraded products.

The algorithm proposed in this paper is tested a set of real databases from steel and glass manufacturing provided by industrial partners. Galvanization is the covering up of a piece of metal by zinc or a zinc alloy in order to protect the piece from corrosion. Galvanized steel is highly used in the automotive industry. Succeeding a good galvanization requires that the final galvanized metal sheets meet technical and aesthetic quality criteria. Among these criteria, one can name the yield stress, the thickness of the zinc layer, etc. For instance, if this thickness is below client's requirement, the product is downgraded to be sold ten to twenty percent cheaper. Other flaws are also monitored for instance by camera or manual inspection. These monitorings allow the manufacturer to quantify numerically the quality of its final products. This quality depends on the parameters of the process. In the databases at our disposal, the settings of up to 272 parameters are provided, such as the decreasing quality of the metal sheets, pollution in the zinc bath, cleanliness of some mechanical parts, temperatures of the furnaces, cooling temperature, etc.

The production of glass panes is also a process which can benefit from the optimization of its parameters. Basics of glass production can be summarized in a few steps. Silica sand is first melted and stirred with various ingredients such as sodium carbonate and calcium oxide to ease the process. Additives are added to the mixture to control special properties from the glass like the heat resistance or the taint. Once the mixture is turned into a homogeneous liquid with as few bubbles as possible, the molten glass is brought onto a surface of molten tin and cooled down progressively. The next step is the annealing phase which aims to heat-treat the glass to strengthen it. The whole process is very sensitive to the temperature set points and to external pollutants. The databases analyzed in this article includes settings of 40 to 120 parameters depending on the type of glass pane produced. Among the quality criterion, one can cite the number of bubbles in the final product, the number and size of scratches and spot defects, the tightness, etc.

The outline is as follows. Section 3 states the problem and defines the notations used along the paper. The problem is modeled as a mixed-integer linear program in Section 4. A heuristic method to address the problem is presented in Section 5. Both methods are compared in Section 6 on real industrial databases from steel and glass manufacturing. Finally, Section 7 concludes.

2 Related work

Data mining has been extensively applied to this kind of problem. Köksal, Batmaz, and Testik present a large review of data mining methods to describe, predict, classify or optimize the quality of a manufacturing processes [1]. Among these methods, the model-free method PRIM (Patient Rule Induction Method) [2] appeared to be well tailored to address robustly this kind of problem. Application to the improvement of steel making processes using PRIM is investigated in [3]. Chong, Albin, and Jun compare the application of the classic PRIM algorithm to a database with a reduced number of parameters, taking some linear combination of them. They apply PRIM on the modified database and projecting back the resulting box on the initial set of parameters [3]. Clustering is a problem close to the one considered in the paper. Some clustering algorithm provides multiple boxes, dividing the data into clusters [4]. The problem tackled in the paper is different as we aim to find only a single box optimizing the process. Providing multiple boxes would increase the complexity of application of the result to the industrial process.

In this article, we compare the box obtained by a mixed integer linear program (MILP) to the solution given by a heuristic algorithm using PRIM as a main component. Combinatorial approaches have been proposed to obtain a solution with a guarantee of optimality for the logical analysis of data [5], [6]. In these articles, the authors present a method to obtain the box which maximizes the number of points of a given set while excluding every point from another set of points. In [5], the authors provide a mixed-integer linear formulation for the problem and use it on medical databases. Eckstein, Hammer, Liu, *et al.* propose an alternative branch and bound algorithm to solve this problem to optimality [6]. A combinatorial algorithm is proposed in [7] for a generalized version of the maximum box problem which maximizes the weighted sum over a set of points with positive and negative objective values. We extend the best formulation of [7] to account for the three following criteria: (i) categorical values, (ii) restriction of the box on a limited number of dimensions and (iii) missing values.

An important issue when considering automatically generated data is to tackle missing values. Various approaches have been proposed to handle missing data in the field of statistics [8]. Some techniques simply ignore records with missing values. Others replace the missing values by a random value or the mean value of the record. A more sophisticated technique is to complete the data sets by multiple imputation [9]. The method consists in creating multiple data sets by filling in alternative values for the missing data which may require computationally intensive and complex algorithms. In this paper, we adopt a conservative approach detailed in Section 3 and adapt the algorithms in consequence.

3 Problem statement

We consider that there is a set $\mathcal{D}$ of D input parameters, which may influence the output. Each produced item i is represented by a D-dimensional vector $\mathbf{x}^i$ of the input parameters. Missing values are represented by the symbol $\emptyset$. The output value of the item i is given by $c^i \in \mathbb{R}$ and might be either positive or negative. The set of all M produced items is denoted $\mathcal{M}$. The D dimensions are split into two sets: the set of continuous dimensions $\mathcal{L}$ and the set of categorical dimensions $\mathcal{G}$. Accepting two values α and γ in a continuous dimension implies that each values in the range $[\alpha, \gamma]$ are also accepted. Assume a categorical dimension which includes the values α, β and γ. The final box may reject the category β while accepting categories α and γ. The set of categories in dimension t is denoted $\mathbb{G}_t$. The domain of all categorical dimensions is given by $\mathbb{G} = \prod_{t \in \mathcal{G}} \mathbb{G}_t$.

The problem addressed in this paper is to find a box restricted on only N dimensions which maximizes the sum of the values c^i of each point i included in the box. A box which defines the limits on the input parameters of a setup is easy to understand and to implement in an industrial process. Depending on the input parameter type, we now define a box.

Definition 1. *A box is a set* $\mathbf{b}$ *such that*

$$\mathbf{b} = \left\{ \mathbf{x} \in \mathbb{R}^L \times \mathbb{G} : \begin{cases} x_t \in [l_t(\mathbf{b}), u_t(\mathbf{b})], & \forall t \in \mathcal{L} \\ x_t \in \mathcal{S}_t(\mathbf{b}), & \forall t \in \mathcal{G} \end{cases} \right\} \tag{1}$$

where $\mathcal{S}_t(\mathbf{b}) \subseteq \mathbb{G}_t$ *is the set of selected categories in dimension* $t \in \mathcal{G}$ *and*

$[l_t(\mathbf{b}), u_t(\mathbf{b})]$ *is the projection of the box in dimension* $t \in \mathcal{L}$.

The box $\mathbf{b}$ is restricted only on N dimensions if there are N dimensions such that $\mathcal{S}_t \neq \mathbb{G}_t$ if $t \in \mathcal{G}$, or $l_t(\mathbf{b}) \neq -\infty$ or $u_t(\mathbf{b}) \neq +\infty$ if $t \in \mathcal{L}$.

The objective is to maximize the mean of the points included in the box $\mathbf{b}$,

$$f^{\mu}(\mathbf{b}) = \frac{\sum_{i \in \mathcal{M}: \mathbf{x}^i \in \mathbf{b}} c^i}{|\mathbf{b}|} \tag{2}$$

where $|\mathbf{b}|$ is the number of points in $\mathcal{M}$ included in the box $\mathbf{b}$. The optimization of problems with nonlinear objective function often leads to computationally hard problems. State of the art nonlinear solvers are only able to solve problems of limited size. To make our problem easier to solve, we propose to optimize on a new database whose output values are shifted by the mean output $\bar{c}$, i.e. $c^i \leftarrow c^i - \bar{c}$ for every setting i. On this new database, one could maximize the sum of the output values of the points in the candidate box. This provides us with a linear objective function and allows us to use powerful MILP solvers able to handle problems with thousands of variables. A comparison of the two modeling approaches is provided in Section 6. The corresponding value of the box with this sum version objective function is given by

$$f^{\Sigma}(\mathbf{b}) = \sum_{i \in \mathcal{M}: \mathbf{x}^i \in \mathbf{b}} \left(c^i - \bar{c}\right) \tag{3}$$

where $\bar{c}$ is the mean output value over every point of the database. Note that $f^{\Sigma}(\mathbf{b})$ and $f^{\mu}(\mathbf{b})$ take values of different orders of magnitudes. They should be seen as distinct modeling approaches with the same goal: to model the problem of optimizing an industrial process.

A point is inside a box if it is included in all dimensions. A special attention is required for missing values. Consider a missing value $x_t^i = \emptyset$. Directly excluding the point i in dimension t is equivalent to removing it from the database and therefore losing the information provided by this point. Therefore, we choose to always consider point i to be included in dimension t. This is motivated by the following argument. The point still needs to be included in all other dimensions to be in the box. If $c^i > 0$, the algorithm tries to include the point by including it in all other dimensions. If $c^i < 0$, this conservative approach gives incentives to the algorithms to try to exclude the point. By default, all the missing values are considered in the box. To be sure to exclude the point, it has to be excluded in a dimension for which the value is known. Figure 1 provides an example of a database with one point $\mathbf{x}^{\emptyset}$ which data $x_2^{\emptyset}$ is missing and with a very negative output value. The other points are considered positive. If $\mathbf{x}^{\emptyset}$ is removed from the database, the optimal box would be given by box 1 including all points. Assume that the actual coordinates of $\mathbf{x}^{\emptyset}$ are given by the non-filled circle. Box 1 is worse than box 2, including only the five points on the left hand side. Box 2 is the optimal solution which is obtained if we consider that $\mathbf{x}^{\emptyset}$ is included in dimension 2 as suggested in this paper.

To statistically make sense, we impose that a box includes at least βM points. A further restriction is imposed on the length of the intervals of the box.

Definition 2. *The length of a box* $\mathbf{b}$ *in dimension* t *is given by:*

$$d_t(\mathbf{b}) = \begin{cases} u_t(\mathbf{b}) - l_t(\mathbf{b}), & \textit{if } t \in \mathcal{L} \\ |\mathcal{S}_t(\mathbf{b})|, & \textit{if } t \in \mathcal{G} \end{cases} \tag{4}$$

For a continuous dimension $t \in \mathcal{L}$, we impose the length on the box on this dimension to be greater than a user-defined parameter Δ_t. For a categorical dimension $t \in \mathcal{G}$, Δ_t define the minimum number of categories covered by the box.

4 Integer programming model

We explore a D-dimensional space with a set of M points, $\mathcal{M}$. The coordinates of a point i are denoted $\mathbf{x}^i$ with $\mathbf{x}^i \in [0,1]^D$. The value of a point i is $c^i \in \mathbb{R}$. Those points are partitioned into two sets: the set of positive points $\mathcal{P} = \{i \in \mathcal{M} | c^i > 0\}$ and the set of negative points $\mathcal{N} = \{i \in \mathcal{M} | c^i < 0\}$. Points such that $c^i = 0$ can be ignored.

We apply the following transformation to the coordinates of the points. The coordinates of the points are sorted along each dimension $t \in \mathcal{L}$. As some points have the same coordinate in dimension t, the sorted list of coordinates has $K_t \leq M$ distinct values. For categorical dimensions $t \in \mathcal{L}$, we define $K_t = |\mathbb{G}_t|$. In the following formulation, each point i is encoded by r_t^i, its index in the sorted list corresponding to dimension t. If $x_t^i = \emptyset$, we define $r_t^i = \emptyset$.

The inclusion of a point i in the candidate box is modeled by a binary variable z^i. The point i is included in the box if $z^i = 1$ and 0 otherwise. We introduce for each dimension t, K_t binary variables y_t^k for $k \in \{1, \dots, K_t\}$ where k corresponds to all distinct values in dimension t. The additional binary variable u_t is equal to one if dimension t is constrained, else $u_t = 0$.

To impose the constraint on the minimum length of dimension t of the box we define, for each $k \in \{1, ..., K_t\}$, a parameter a_t^k equal to the length of the interval associated to value k on dimension $t \in \mathcal{L}$. This length is defined as half the distance between the previous and the next point in the ordered list. If $t \in \mathcal{G}$, $a_t^k = 1$.

We restrict ourselves to the sum version of the objective function (3). Adapting the formulation to the mean version of the objective function (2) is trivial but gives rise to a mixed integer non-linear program which would be too computationally intensive to solve. The MILP formulation of the problem is the following:

$$\max \sum_{i \in \mathcal{M}} \left(c^i - \bar{c}\right) z^i \tag{5}$$

subject to:

$$z^i \leq y_t^{r_t^i} \qquad \forall i \in \mathcal{M}, t \in \mathcal{D} : r_t^i \neq \emptyset \tag{6}$$

$$z^i + \sum_{t \in \mathcal{D}: r_t^i \neq \emptyset} (1 - y_t^{r_t^i}) \geq 1 \qquad \forall i \in \mathcal{N} \tag{7}$$

$$y_t^k = p_t^k - q_t^k \qquad \forall t \in \mathcal{L}, k \in \{1, ..., K_t\} \tag{8}$$

$$p_t^k \leq p_t^{k+1} \qquad \forall t \in \mathcal{L}, k \in \{1, ..., K_t - 1\} \tag{9}$$

$$q_t^k \leq q_t^{k+1} \qquad \forall t \in \mathcal{L}, k \in \{1, ..., K_t - 1\} \tag{10}$$

$$\sum_{t \in \mathcal{D}} u_t \leq N \tag{11}$$

$$y_t^k \geq (1 - u_t) \qquad \forall t \in \mathcal{D}, k \in \{1, ..., K_t\} \tag{12}$$

$$\sum_{i \in \mathcal{M}} z^i \geq \lceil \beta |\mathcal{M}| \rceil \tag{13}$$

$$d_t = \sum_{k \in \{1, ..., K_t\}} a_t^k y_t^k \geq \Delta_t \qquad \forall t \in \mathcal{D} \tag{14}$$

Inequality (6) expresses that a point i is excluded if one of its coordinate does not lie in the projection of the box. Constraint (7) imposes that a negative point is included if all of its coordinates lie within the box boundaries. Observe that for positive points, this constraint is enforced by the objective function. For a continuous dimension $t \in \mathcal{L}$, the values that are included must represent a continuous interval and therefore be consecutive. This consecutivity property is modeled by (8)-(10) with the auxiliary variables p_t^k and q_t^k. The variable $p_t^k = 1$ if there is at least one $y_\tau^k = 1$ for $\tau \leq t$. Similarly, $q_t^k = 1$ if there is at least one transition $y_\tau^k = 1$ for $\tau \leq t$. Fig. 2 illustrates the values taken by these auxiliary variables.

The total number of constrained dimensions is imposed to be less than N by (11). If the dimension is unconstrained, $y_t^k = 1$ for all k, which is enforced by (12). Constraints (13) and (14) impose respectively the minimal number of points in the box and the minimal box length in each dimension.

This mixed integer linear program includes $(3D+1)M+D$ binary variables. This number can be used to obtain an order of magnitude of the worst cast complexity of the algorithm. In the worst case scenario, the branch and bound algorithm, solving the discrete part of the problem, enumerates the whole set of solutions, leading to a maximum complexity of $\mathcal{O}(2^{(3D+1)M+D})$. However, the practical complexity is far lower.

5 Heuristic method

This section proposes a heuristic method to find a good candidate box $\mathbf{b}$ with a complexity of $\mathcal{O}(M^2D)$. This heuristic is decomposed in two steps. First, a rule induction method provides a list $\mathcal{B}$ of good boxes constrained on every dimension. This rule induction method is decomposed in a peeling and a pasting phase described later. This is an adaptation of the original PRIM algorithm [2] which considers only a single box instead of a list of boxes. Second, the heuristic performs a deconstraint operation which relaxes the box until it is constrained on only N dimensions. The whole heuristic is summarized in Algorithm 1.

Algorithm 1 Heuristic algorithm

```
1: $\mathcal{B} \leftarrow$ peeling phase
2: for all $\mathbf{b} \in \mathcal{B}$ do
3:     $\mathbf{b} \leftarrow$ pasting phase($\mathbf{b}$)
4:     $\mathbf{b} \leftarrow$ deconstraint($\mathbf{b}$)
5: end for
6: return $\arg\max_{\mathbf{b}\in\mathcal{B}} f(\mathbf{b})$
```

The first phase of the algorithm is a slightly modified version of PRIM [2].

This phase consists in two steps: a peeling step, shrinking an initial box, followed by a pasting step, expanding a given box. The algorithm is independent of the form of the objective function and can therefore be applied to the mean version (2) and to the sum version (3) objective function.

The peeling step starts from a list $\mathcal{I}$ of all points of the database. The algorithm removes at each iteration a fraction α of the current points in the dimension that improves the most the score, until it remains a fraction β of points from the original set. Each iteration generates one box added to the list of boxes $\mathcal{B}$. In the case in which we restrict ourselves to a single box, we choose the last one generated as in the original PRIM algorithm [2]. Each of these boxes is built using the procedure $\texttt{mkbox}(\mathcal{I})$ defined in the following.

Definition 3. *The limits of a box* $\mathbf{b} = \texttt{mkbox}(\mathcal{I})$*, built from a set of included points* $\mathcal{I}$*, are given by* $\mathcal{S}_t(\mathbf{b}) = \{c \in \mathbb{G}_t : \exists i \in \mathcal{I} : x_t^i = c\}$ *in dimensions* $t \in \mathcal{G}$*;* $l_t(\mathbf{b}) = \min_{i \in \mathcal{I}} x_t^i$ *and* $u_t(\mathbf{b}) = \max_{i \in \mathcal{I}} x_t^i$ *in dimensions* $t \in \mathcal{L}$*.*

Note that $\texttt{mkbox}(\mathcal{I})$ can include more than $|\mathcal{I}|$ points of $\mathcal{M}$.

The method depends on a parameter α which defines the amount of points removed at each iteration of the peeling step. To be effective, α must be set to a small value, typically $5\% \leq \alpha \leq 10\%$. Such small values allow each peeling to be less important and gives opportunity to the algorithm to mitigate an early bad peeling. These small peelings are the reason why this strategy is called “patient” compared to other algorithms (e.g. decision trees like CART [10]). The peeling step is summarized in Algorithm 2. In instruction lines 1 to 3, the algorithm starts by considering that all points in the corresponding box. Lines 6 to 15 are dedicated to building the list of sets of candidate included points $\mathcal{P}$. Starting from the list of included point in the last candidate box, $\mathcal{I}$, the algorithm removes at each iteration $\alpha|\mathcal{I}|$ points in each continuous dimension and adds the resulting sets of points to $\mathcal{P}$ a list of sets of candidate included points. If the dimension is categorical, the algorithm removes one category and adds the resulting set of included points to $\mathcal{P}$. This list is filtered in instruction 16 to remove set of points which are not large enough or leading to too small boxes. Finally, instruction 17 selects the list of points leading to the best score and stores it in $\mathcal{I}$. Finally, the candidate box is added to the list of candidate boxes $\mathcal{B}$ in instruction 18. The worst-case complexity of this step is $\mathcal{O}(M^2D)$. Note that in the original PRIM algorithm [2], the missing values are considered to belong to a special category. Our implementation is simplified as we consider that missing values always belong to the box.

Algorithm 2 Rule induction method : Peeling step

```
1: $\mathcal{I} \leftarrow \mathcal{M}$, the set of included points in the last peeling
2: $\mathcal{B} \leftarrow \emptyset$, the list of peeled boxes
3: $\mathcal{P} \leftarrow \{\mathcal{I}\}$, list of sets of candidate included points at each iteration.
4: while $\mathcal{P} \neq \emptyset$ do
5:   $\mathcal{P} \leftarrow \emptyset$
6:   for all $t \in \mathcal{D}$ do
7:     if $t \in \mathcal{L}$ then
8:       $\mathcal{P} \leftarrow \mathcal{P} \cup \big\{\mathcal{I} \setminus \{\text{at most } \alpha|\mathcal{I}| \text{ points} \in \mathcal{I} \text{ with the smallest } x_t^i \neq \emptyset\}\big\}$
9:       $\mathcal{P} \leftarrow \mathcal{P} \cup \big\{\mathcal{I} \setminus \{\text{at most } \alpha|\mathcal{I}| \text{ points} \in \mathcal{I} \text{ with the largest } x_t^i \neq \emptyset\}\big\}$
10:    else if $t \in \mathcal{G}$ then
11:      for all $c \in x_t$ do
12:        $\mathcal{P} \leftarrow \mathcal{P} \cup \{\{i \in \mathcal{I} : x_t \neq c\}\}$
13:      end for
14:    end if
15:  end for
16:  $\mathcal{P} \leftarrow \{\mathcal{I}' \in \mathcal{P} : |\mathcal{I}'| \geq \beta|\mathcal{M}| \text{ and } \forall t \in \mathcal{D}, d_t(\texttt{mkbox}(\mathcal{I}')) \geq \Delta_t\}$
17:  $\mathcal{I} \leftarrow \arg\max_{\mathcal{I}' \in \mathcal{P}} f(\texttt{mkbox}(\mathcal{I}'))$
18:  $\mathcal{B} \leftarrow \mathcal{B} \cup \{\texttt{mkbox}(\mathcal{I})\}$
19: end while
20: return $\mathcal{B}$
```

Each box $\mathbf{b} \in \mathcal{B}$ is expanded in the pasting step. This step expands iteratively an initial box $\mathbf{b}$ on one dimension per iteration until the score cannot be improved on any dimension. The pasting step is detailed in Algorithm 3. In each iteration, the algorithm starts from the list of initially included points $\mathcal{I}$ and creates for each dimension t a new list of included points $\mathcal{I}_t$ if the box is expanded in dimension t. Continuous dimensions are processed in instructions 4-7 and categorical dimensions in instructions 8-12. The best expansion is selected in instruction 15 and accepted in instructions 17-18 if this expansion improves the score, else the algorithm terminates with instruction 20. The worst-case complexity of the pasting step is $\mathcal{O}(M^2D)$. Note that one may sort the database in each dimension at the beginning of the algorithm and exploit this preprocessing to implement the algorithm efficiently. One may also compute the score and the inclusion/exclusion of a point in each dimension iteratively.

Algorithm 3 Rule induction method: Pasting step of an initial box $\mathbf{b}$

```
1: $\mathcal{I} \leftarrow \mathcal{M} \cap \mathbf{b}$, the set of included points in the last pasting step
2: while True do
3:   for all $t \in \mathcal{D}$ do
4:     if $t \in \mathcal{L}$ then
5:       $\mathcal{I}^t_- \leftarrow \mathcal{I} \cup \{$at most $\alpha|\mathcal{I}|$ points $\in \mathcal{M} \setminus \mathcal{I}$ with the largest $x^i_t : x^i_t \leq l_t(\mathbf{b})$ and $x^i_\tau \in [\mathbf{b}.l_\tau, \mathbf{b}.u_\tau]\ \forall \tau \in \mathcal{L} \setminus \{t\}\}$
6:       $\mathcal{I}^t_+ \leftarrow \mathcal{I} \cup \{$at most $\alpha|\mathcal{I}|$ points $\in \mathcal{M} \setminus \mathcal{I}$ with the smallest $x^i_t : x^i_t \geq u_t(\mathbf{b})$ and $x^i_\tau \in [\mathbf{b}.l_\tau, \mathbf{b}.u_\tau]\ \forall \tau \in \mathcal{L} \setminus \{t\}\}$
7:       $\mathcal{I}_t \leftarrow \arg\max_{\mathcal{I}' \in \{I^t_-, I^t_+\}} f(\texttt{mkbox}(\mathcal{I})')$
8:     else if $t \in \mathcal{G}$ then
9:       for all $c \in \mathcal{G}_t \setminus x_t$ do
10:        $\mathcal{I}^t_c \leftarrow \{i \in \mathcal{M} \setminus \mathcal{I} : x_t = c$ and $\forall \tau \in \mathcal{D} \setminus \{t\}, x_\tau \in \mathbf{b}.\mathcal{S}_\tau\}$
11:      end for
12:      $\mathcal{I}_t \leftarrow \arg\max_{\mathcal{I}' \in \{\mathcal{I}^c_t : c \in \mathcal{G}_t\}} f(\texttt{mkbox}(\mathcal{I}'))$
13:    end if
14:  end for
15:  $\mathcal{I}^c \leftarrow \arg\max_{\mathcal{I}' \in \{\mathcal{I}_t : t \in \mathcal{D}\}} f(\texttt{mkbox}(\mathcal{I}'))$
16:  if $f(\texttt{mkbox}(\mathcal{I}^c)) < f(\texttt{mkbox}(\mathcal{I}))$ then
17:    $\mathcal{I} \leftarrow \mathcal{I}^c$
18:    $\mathbf{b} \leftarrow \texttt{mkbox}(\mathcal{I})$
19:  else
20:    return $\mathbf{b}$
21:  end if
22: end while
```

The last step simplifies the box definition and selects the N most important dimensions. This final step is given in Algorithm 4. In each iteration, the algorithm expands the candidate box $\mathbf{b}$ in each dimension in instructions 3 to

10. The dimension which decreases the less $f(\mathbf{b})$ is selected in instruction 11 and this dimension is removed from the list of dimensions to relax $\mathcal{Q}$ with instruction 12. This expansion is carried out on the remaining dimensions until the box is constrained on only N dimensions.

Algorithm 4 Rule induction method: Deconstraint on an initial box $\mathbf{b}$

```
1:  $\mathcal{Q} \leftarrow \mathcal{D}$
2:  while $|\mathcal{Q}| > N$ do
3:      for all $t \in \mathcal{Q}$ do
4:          $\mathbf{b}^t \leftarrow \mathbf{b}$
5:          if $t \in \mathcal{L}$ then
6:              $\mathbf{b}^t.l_t \leftarrow -\infty, \mathbf{b}^t.u_t \leftarrow +\infty$
7:          else if $t \in \mathcal{G}$ then
8:              $\mathbf{b}^t.\mathcal{S}^t \leftarrow \mathbb{G}_t$
9:          end if
10:     end for
11:     $\mathbf{b} \leftarrow \arg\max_{\mathbf{b}' \in \{\mathbf{b}^t : t \in \mathcal{D}\}} f(\mathbf{b}')$
12:     $\mathcal{Q} \leftarrow \mathcal{Q} \setminus \{t\}$
13: end while
14: return b
```

Note that the parameter N is only used in the deconstraint phase. Each iteration of the main loop of Algorithm 4 provides a box with one dimension less constrained. Running this algorithm with $N = 1$ can provide in a single run the evolution of the quality of the process in function of the number of constrained dimensions.

6 Results

In this section, the previous algorithms are applied to randomly generated databases and industrial databases. We compare the *MILP* model with four versions of the heuristic presented in Section 5. We use the notation *OBS* (Operational Box Search) to refer to the heuristic presented in Section 5 applied to a single box with the sum score (3) as objective function. The variant with the mean version objective function (2) is denoted by *OBS Mean*. The heuristic applied to a list of boxes is denoted by *OBS - Multiple Boxes* and *OBS Mean - Multiple Boxes*. If not said otherwise, the tests are conducted with the arbitrarily chosen parameters: $N = 5$, $\alpha = 5\%$ and $\beta = 20\%$. The influence of the α and β parameters have been investigated in the original PRIM article [2]. We choose to restrict only five dimensions since it seems a reasonable number of controlled parameters for an industrial process. We impose that the width of the box is larger than 10% of the initial parameter range for continuous dimensions and to select at least one category for categorical dimensions.

Independently of the score function, every algorithm outputs a valid box on the original database (i.e. non centered). The two scores associated with the box, $f^{\mu}(\mathbf{b})$ and $f^{\Sigma}(\mathbf{b})$ are computed using respectively (2) and (3). In the following results, we compare the improvement with respect to the unconstrained box including every point of the database. This score improvement provides an order of magnitude of the expected improvement of the process that can be achieved using the restriction of the parameters given by the candidate box. This score improvement is given by

$$MSI(\mathbf{b}) = \frac{f^{\mu}(\mathbf{b}) - f^{\mu}(\mathbf{b}^0)}{f^{\mu}(\mathbf{b}^0)} \tag{15}$$

where $\mathbf{b}$ is the box provided as the solution by the algorithm which is evaluated and $\mathbf{b}^0$ is the box including all points in the database.

The results are organized as follows. A typical run on one industrial database is detailed in Section 6.1. The algorithms are compared on randomly generated databases in Section 6.2 and on industrial databases in Section 6.3. All results have been obtained on two *Intel Core i7* with a clock rate of 3.47 *GHz* and 24 *GB* of RAM. The mixed-integer linear programs are solved using *CPLEX 12.6* with default parameters (including presolve, cutting planes, etc.) using up to eight threads. The mean values provided in these results are geometric means, shifted if required.

6.1 Typical run and influence of the number of unconstrained dimensions

Hereafter are discussed results obtained on one of the galvanized steel manufacturing database with 101 points, 6 categorical dimensions and 262 numerical dimensions. Results provided by the five algorithms are given in Table 1. Solving the problems took 19 seconds with the MILP while the heuristics delivers their solutions in less than a second. As expected, each algorithm performs better on the score function it is optimizing. Since the MILP terminates, it obtains the optimal solution with respect to the sum score objective function. One can compare the results given by the heuristic using the sum objective function (3) to see that it provides a 30% suboptimal solution. Using multiple boxes reduces the suboptimality to 26%.

The MSI shows that restricting the parameters of the process may increase the quality between 82.75% using the box provided by the PRIM-based heuristic and 99.55% using the MILP solution. The heuristic using the sum score function provides lower improvements but a larger box which might be easier to implement in practice. Note that the MILP obtains a better solution from the MSI point of view even though the objective function of the MILP is not to optimize the mean.

Figs. 3, 4, 5 and 6 show the influence of the number of constrained dimensions on the solution. Restricting only a few dimensions has a large impact on the solution as highlighted by Figs. 3 and 5. The MILP naturally dominates the other algorithms on the sum score. The MILP requires only four dimensions to provide a solution with 100% MSI. Following Figs. 3 and 5, the OBS algorithms should constrain 15 parameters to provide good results for this database. The

MILP formulation finds a better solution only constraining 4 parameters.

Fig. 4 shows that the sum score for OBS decreases as the number of constrained dimensions, N, increases. This behavior is caused by the peeling step of Algorithm 2. This step imposes to start from a box with only βM points. The pasting step of Algorithm 3 increases the box size using a greedy method. This step is known to have little effect on the solution [2]. Consequently, the OBS algorithm misses easily boxes of intermediate sizes. As the number of constrained dimension decreases, the deconstraint phase provides larger boxes with more positive points and better scores which correct the over-shrinking effect of the peeling steps. This effect is less important for OBS Multiple Boxes as the pasting and deconstraint phase may be performed on boxes with more than βM points.

In Figure 6, the MILP dominates the other algorithms on the MSI as long as $N \leq 8$. For $8 < N \geq 157$, mean OBS obtains better results and for $N > 157$, the basic OBS with the sum score function find an even better box. The mean OBS algorithm may miss this solution since it is a greedy algorithm and the problem, formulated with the mean version, has many local optima which trap the greedy algorithm.

6.2 Randomly generated databases

The aim of this section is to present results on randomly generated databases which are tractable to solve to optimality by the MILP formulation. The following tests are conducted on 100 randomly generated databases with 100 points on 2 categorical and 8 continuous dimensions. The value of the points are generated following a uniform distribution between 10 and 110. Table 2 provides a comparison of the algorithms.

As expected, each algorithm performs better on the score function it is optimizing. Comparing the sum score, the heuristic using the sum objective function (3) provides a 16% suboptimal solution. Using multiple boxes reduces the suboptimality to 8%. The best mean score is obtained using the OBS algorithm with the mean score objective function. One can note from the results that using the sum score objective functions, using MILP or OBS, provides only 10% less MSI but doubles the size of the box. The box obtained has therefore wider ranges in the constrained dimension which might be, in function of the process, easier to implement in practice.

6.3 Industrial databases

The following tests are performed on 5 glass manufacturing databases and 25 galvanized-steel manufacturing databases provided by industrial partners. Table 3 presents the results with a time limit of two hours. The complete table of results can be found in the appendix, in Table 5.

On a total of 30 tests, only one test with the OBS heuristic required the two hours. Most of the tests are completed by the heuristics in less than three minutes. The MILP algorithm obtains a feasible solution for 28 problems. Among these tests, 19 have a guarantee of optimality on the sum score. The mean gap for the 9 remaining tests is 36.44%. On these tests, OBS multiple boxes obtains the best sum score since the MILP cannot reach optimality. Surprisingly, the best MSI is obtained by the heuristic using the sum score objective function. This motivates the approach of the sum score which in this case performs even better than using the mean objective function f^{μ}. The sum score provides boxes twice larger than with the mean score objective function. The results also confirm that using the multiple box variant provides substantial improvement of the quality while requiring in most cases less than twice the computation time. The ratio of the time taken by the single box version and the multiple box version of the OBS Mean algorithm can be observed on Figure 7. As the number of dimensions increases, the time taken to test multiple boxes decreases down to 25% of additional time with respect to the obtention of a single box. However, increasing the number of points increases the time to test multiple boxes.

Table 4 provides the summary results for $N = 15$. Changing $N = 5$ to $N = 15$ has a larger effect on the MILP algorithm which MSI changes from 29.81% to 40.85%. With this setting, 22 problems where solved by the MILP algorithm to optimality. The best MSI is given now by the OBS algorithm using the mean score objective function with an improvement of 5.34% with respect to the case where $N = 5$. In practice, we advice to run the heuristic simultaneously with the sum version and the mean version.

7 Conclusion

This paper describes a technique to optimize the parameters of an industrial process. An industrial process may be dependent on a substantial amount of parameters and selecting few of them, impacting the most the process, is a challenging data analysis task. In particular we wish to find a set of rules, i.e. intervals on a reduced number of parameters, for which an output value is maximized. The model-free optimization problem to solve is to find a box, restricted on a limited amount of dimensions, with the maximum mean value of the included points. We compare a machine learning-based heuristic to the optimal solution computed by a mixed-integer linear program on real-life databases from steel and glass manufacturing. Computational results show that the heuristic achieves comparable solutions with respect to the exact approach. However, the exact approach is computationally too expensive to tackle real life databases. In such cases, the heuristic delivers better results. Results show that the restriction of five process parameters, on these databases, may improve the quality by

50%.

The work presented in this paper could be extended along several lines. Advanced preprocessing may improve the computation times. For example, a preprocessing algorithm could consider skewness and kurtosis to scale or discard some dimensions of the initial database. Other algorithms could be implemented to tackle this box search problem. For instance, topographic map, a two-dimensional, nonlinear approximation of a potentially high-dimensional data manifold, is known to help with the visualizion and exploration of high-dimensional data [11]. One could use this technique to identify interesting regions of the multidimensional space using self-organized maps and translating the regions into a box. Another idea would be to learn a decision tree from the data and to combine interesting zone identified by the tree model to build a box.

Acknowledgments

This research is supported by the public service of Wallonia within the framework of the VIRTUOSO project. The authors thank the companies Segal and AGC for providing the industrial databases used in this paper.

References

[1] G. Köksal, İnci Batmaz, and M. C. Testik, "A review of data mining applications for quality improvement in manufacturing industry," *Expert Systems with Applications*, vol. 38, no. 10, pp. 13448 –13 467, 2011, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2011.04.063.

[2] J. H. Friedman and N. I. Fisher, "Bump hunting in high-dimensional data," *Statistics and Computing*, vol. 9, no. 2, pp. 123–143, Apr. 1999, ISSN: 0960-3174.

[3] I.-G. Chong, S. L. Albin, and C.-H. Jun, "A data mining approach to process optimization without an explicit quality function," *IIE Transactions*, vol. 39, no. 8, pp. 795–804, 2007. DOI: `10.1080/07408170601142668`.

[4] B. Mirkin, *Mathematical classification and clustering: From how to what and why*. Springer, 1998.

[5] T. Bonates, P. L. Hammer, and A. Kogan, "Maximum patterns in datasets," *Discrete Applied Mathematics*, vol. 156, no. 6, pp. 846 –861, 2008, ISSN: 0166-218X. DOI: `10.1016/j.dam.2007.06.004`.

[6] J. Eckstein, P. L. Hammer, Y. Liu, M. Nediak, and B. Simeone, "The maximum box problem and its application to data analysis," *Computational Optimization and Applications*, vol. 23, no. 3, pp. 285–298, 2002.

[7] Q. Louveaux and S. Mathieu, "A combinatorial branch-and-bound algorithm for box search," *Discrete Optimization*, vol. 13, no. 0, pp. 36 –48, 2014, ISSN: 1572-5286. DOI: `10.1016/j.disopt.2014.05.001`.

[8] A Feelders, H Daniels, and M Holsheimer, "Methodological and practical aspects of data mining," *Information and Management*, vol. 37, no. 5, pp. 271 –281, 2000, ISSN: 0378-7206. DOI: `10.1016/S0378-7206(99)00051-8`.

[9] D.-S. Kwak and K.-J. Kim, "A data mining approach considering missing values for the optimization of semiconductor-manufacturing processes," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2590 –2596, 2012, ISSN: 0957-4174. DOI: `10.1016/j.eswa.2011.08.114`.

[10] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and regression trees*. Chapman & Hall/CRC, 1984, p. 358, ISBN: 0-412-04841-8.

[11] M. M. Van Hulle, "Self-organizing maps," in *Handbook of Natural Computing*, Springer, 2012, pp. 585–622.

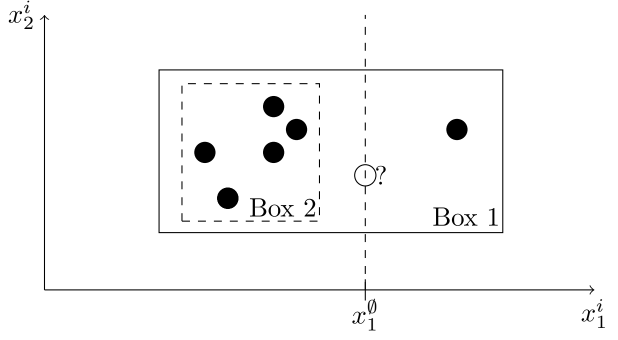

Figure 1: Example of database with one very negative point $\mathbf{x}^\emptyset$ given by the non-filled circle with its missing coordinate. Other points are positives. If $\mathbf{x}^\emptyset$ is removed from the database, the candidate box is box 1. The optimal box is box 2 which is obtained if we consider $\mathbf{x}^\emptyset$ included in dimension 2.

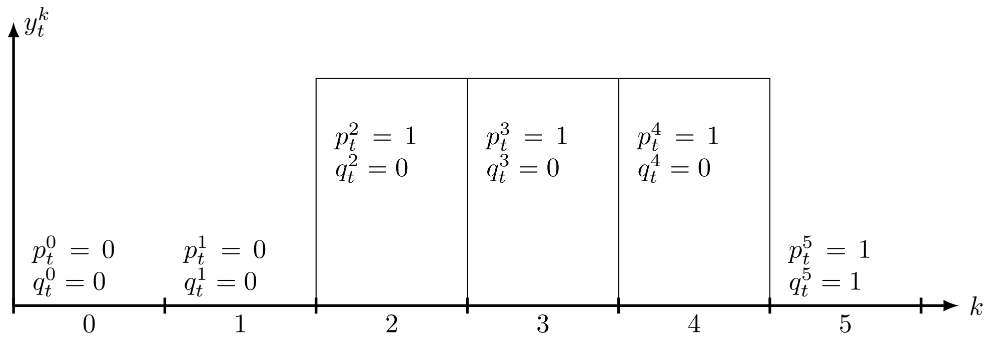

Figure 2: Example of values of the variable y_t^k, q_t^k and p_t^k for a box in a dimension t including the third to the fifth point in this dimension.

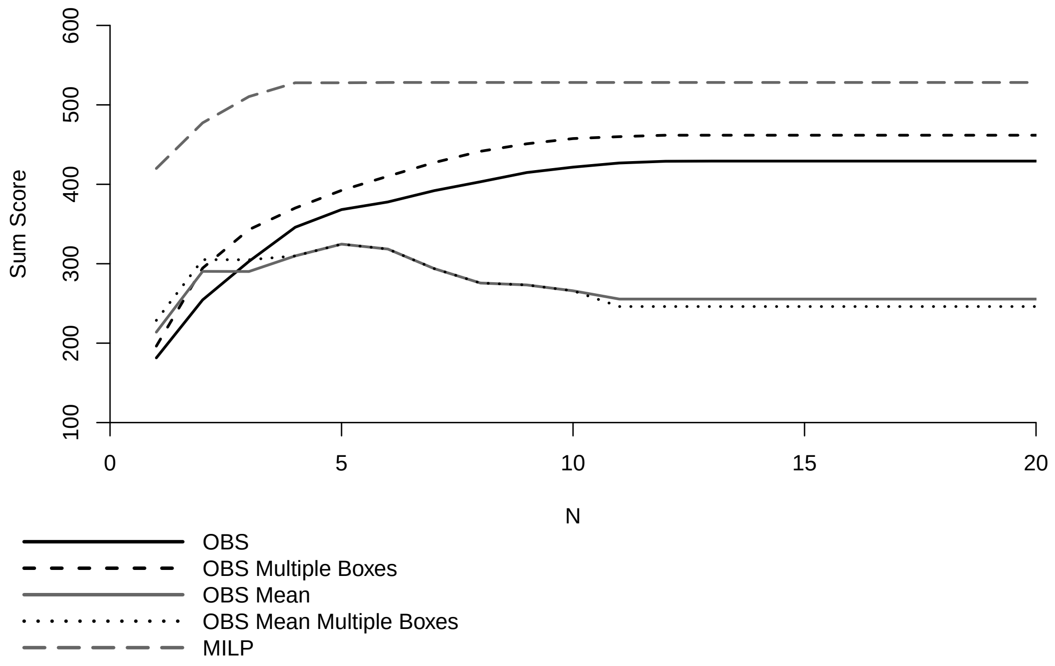

Figure 3: Influence of the number of dimension constrained (1 to 20) on the sum score for one galvanized steel database.

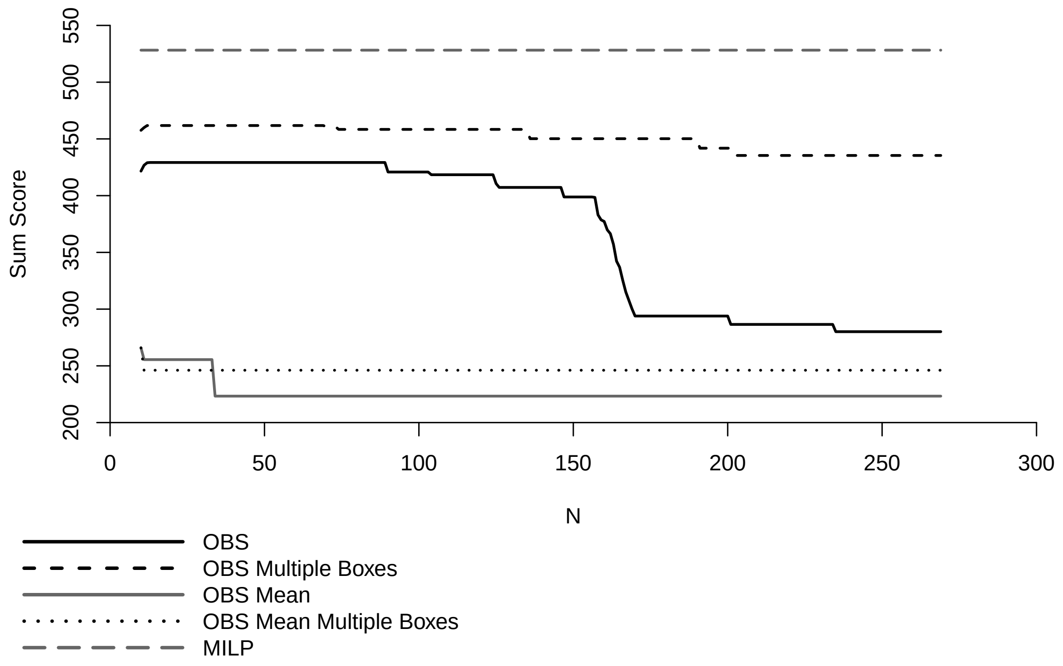

Figure 4: Influence of the number of dimension constrained (10 to 268) on the sum score for one galvanized steel database.

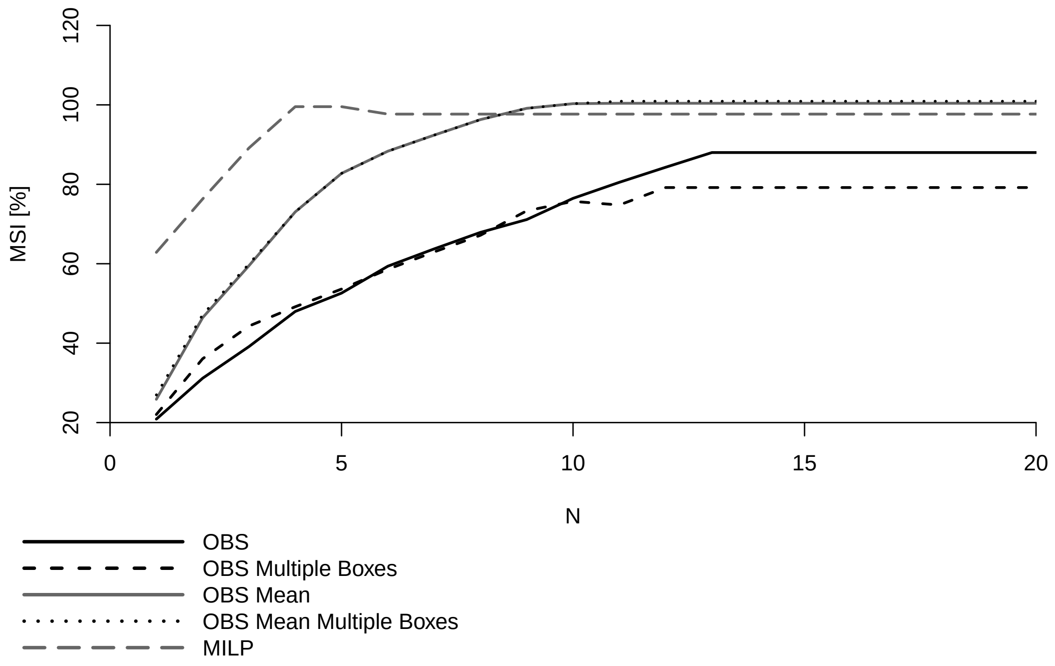

Figure 5: Influence of the number of dimension constrained (1 to 20) on the MSI for one galvanized steel database.

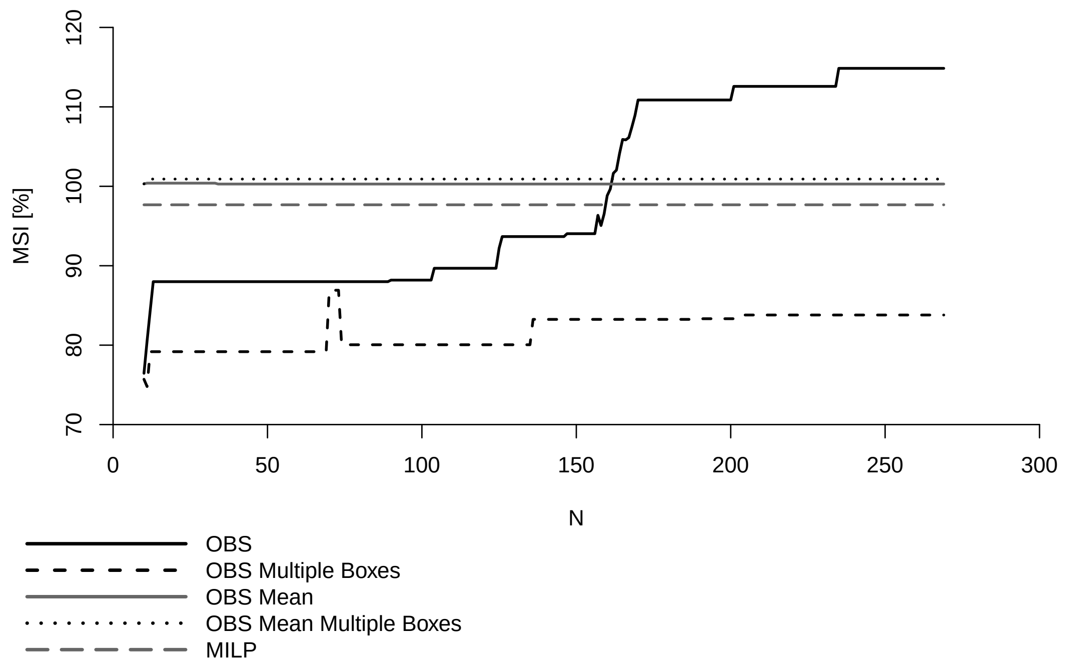

Figure 6: Influence of the number of dimension constrained (10 to 268) on the MSI for one galvanized steel database.

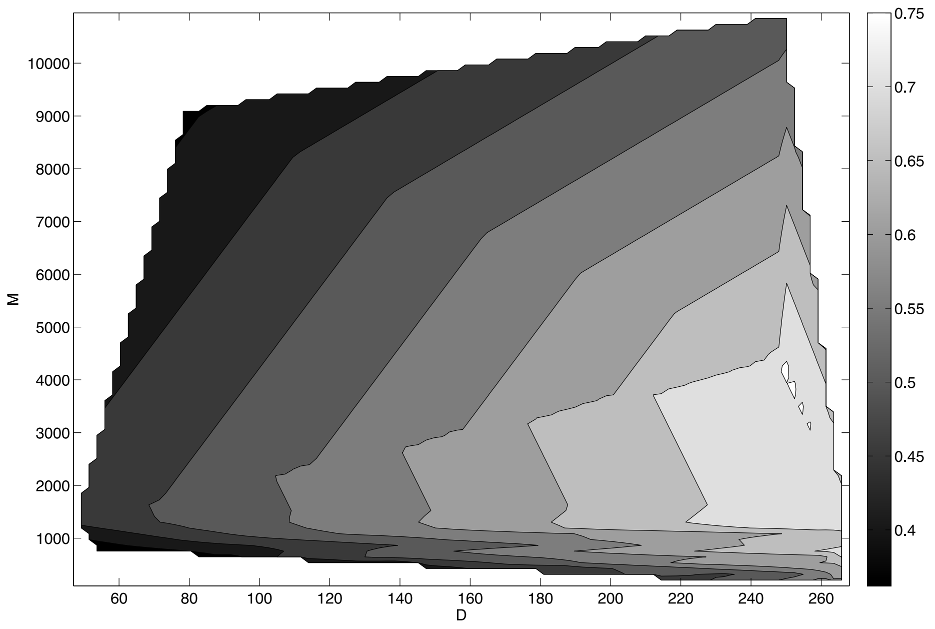

Figure 7: Heatmap of the ratio of the time taken by the single box version and the multiple box version of the OBS Mean algorithm.

	OBS	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
Time [*s*]	0.24	0.48	**0.17**	0.34	19.00
Sum Score	368.14	392.33	324.65	324.65	**527.80**
Mean Score	5.58	5.69	8.77	8.77	**10.56**
MSI [%]	52.60	53.62	82.75	82.75	**99.55**
Box size [%]	65.35	68.32	36.63	36.63	49.50

Table 1: Comparison of the five algorithms for one galvanized steel database.

	OBS	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
Time $[s]$	0.0034	0.008	**0.0005**	0.0037	28.77
Sum Score	620.5	676.37	480.13	482.13	**736.32**
Mean Score	17.41	15.89	22.15	**22.24**	18.03
MSI [%]	29.92	27.38	37.67	**37.8**	31.05
Box Size [%]	35.23	42.19	21.41	21.42	39.91

Table 2: Comparison of the algorithms for random databases ($M = 100$, $C = 2$, $L = 8$).

	OBS	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
Mean time [s]	12.46	17.69	**2.21**	3.13	669.02
Mean Sum Score	1009.53	**1236.25**	608.37	623.06	758.55
Mean MSI [%]	36.81	**44.95**	37.90	39.89	29.81
Mean box size [%]	44.27	41.69	29.40	28.59	40.58

Table 3: Results obtained on the industrial databases (N = 5).

	OBS	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
Mean time [s]	12.51	17.72	**2.21**	3.11	465.32
Mean Sum Score	1118.60	**1274.93**	641.22	643.95	1069.97
Mean MSI [%]	47.95	48.43	50.14	**50.29**	40.85
Mean box size [%]	37.25	41.02	23.13	23.12	45.10

Table 4: Results obtained on the industrial databases (N = 15).

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
Glass 1	0.93	1.53	**0.56**	1.04	7213.00
	70.48	71.63	61.75	62.47	**75.78**
109	114.95	113.03	142.79	**143.50**	105.99
1100	38.55	40.36	27.55	27.73	45.36
					62.9264
Glass 2	0.06	0.14	**0.05**	0.14	7201.00
	92.51	**92.51**	90.31	90.31	12.63
53	41.42	41.42	**52.26**	**52.26**	2.47
738	41.33	41.33	31.98	31.98	94.72
					901.358
Glass 3	**0.25**	0.49	0.61	1.56	7208.00
	−1360.74	−1360.74	−1058.11	−1058.11	–
78	−32.07	−32.07	**140.56**	**140.56**	–
9137	20.04	20.04	20.09	20.09	–
					–
Glass 4	0.06	0.13	**0.04**	0.09	7201.00
	46.73	**46.73**	38.79	38.79	44.26
80	47.65	47.65	**108.84**	**108.84**	43.84
712	58.01	58.01	20.08	20.08	57.02
					133.132
Glass 5	0.09	0.18	**0.08**	0.17	7201.00
	52439.50	**52439.50**	17944.03	17944.03	35913.80
47	**173.63**	**173.63**	144.37	144.37	20.47
1373	21.05	21.05	20.10	20.10	45.45
					53.4726
Steel 1	4.85	8.53	**0.77**	1.28	7240.00
	771.04	814.96	456.14	474.46	**1443.01**

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
265	7.09	**20.09**	8.30	11.07	16.77
394	72.84	27.16	36.80	28.68	57.61
					7.81987
Steel 2	17.78	28.02	**0.95**	1.43	1419.00
	660.10	874.16	422.43	422.43	**1159.33**
266	8.24	**15.96**	7.45	7.45	15.14
488	60.25	41.19	42.62	42.62	57.58
					0.00127451
Steel 3	0.45	0.86	**0.28**	0.55	6767.00
	395.82	468.34	415.74	421.98	**538.98**
244	25.83	29.86	29.99	30.85	**39.40**
154	54.55	55.84	49.35	48.70	48.70
					0.0
Steel 4	7162.41	7213.08	**730.62**	1385.27	7211.00
	20227.80	**20265.50**	10912.81	10912.81	–
250	18.65	18.72	**23.43**	**23.43**	–
10954	46.71	46.63	20.07	20.07	–
					–
Steel 5	82.07	164.72	**2.95**	4.53	5404.00
	765.74	1482.14	512.78	512.78	**2020.40**
260	3.31	10.46	8.08	8.08	**10.58**
1066	74.95	45.87	20.54	20.54	61.82
					0.00153221
Steel 6	2035.33	3013.35	**57.93**	76.93	7204.00
	5621.40	6914.04	2740.40	2740.40	**10296.50**
250	**40.90**	31.49	19.96	19.96	29.07
4287	20.04	32.00	20.01	20.01	51.64

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
					0.0971005
Steel 7	10.34	18.66	**1.47**	2.35	7201.00
	539.25	**572.76**	321.77	353.57	136.57
259	**20.51**	14.88	7.68	7.85	1.54
681	20.56	30.10	32.75	35.24	69.46
					700.167
Steel 8	2592.65	3502.61	**36.18**	51.19	2136.00
	4855.74	5812.27	2642.59	2642.59	**8137.03**
249	**60.03**	43.77	32.63	32.63	41.00
4418	20.01	32.84	20.03	20.03	49.09
					0.00614475
Steel 9	3.10	5.27	**0.81**	1.31	107.00
	1147.86	1310.38	850.96	834.72	**1777.79**
266	15.12	20.74	16.39	18.32	**24.04**
325	47.69	39.69	32.62	28.62	46.46
					0.0
Steel 10	1.13	2.12	**0.48**	0.87	11.00
	474.50	614.87	417.78	463.30	**742.90**
247	40.61	**80.78**	45.85	45.98	77.86
218	50.00	32.57	38.99	43.12	40.83
					0.0
Steel 11	71.38	102.92	**3.99**	5.41	217.00
	3285.19	4517.21	2659.27	2659.27	**5644.89**
249	18.53	43.81	**45.18**	**45.18**	37.94
1247	60.38	35.12	20.05	20.05	50.68
					0.0
Steel 12	0.78	1.57	**0.27**	0.46	7201.00

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
	998.45	**1076.91**	701.21	813.21	361.56
264	1.67	**1.83**	1.57	1.82	1.69
371	65.50	64.69	49.06	49.06	23.45
					372.441
Steel 13	235.58	322.02	**6.14**	8.20	1102.00
	3030.93	3375.06	1455.60	1455.60	**4727.62**
266	**46.52**	35.41	22.28	22.28	30.99
1684	20.01	29.28	20.07	20.07	46.85
					0.00705913
Steel 14	0.24	0.48	**0.17**	0.34	19.00
	368.14	392.33	324.65	324.65	**527.80**
245	52.60	53.62	82.75	82.75	**99.55**
101	65.35	68.32	36.63	36.63	49.50
					0.0
Steel 15	1.48	2.57	**0.55**	0.95	12.00
	130.87	143.23	115.22	118.22	**231.81**
266	6.48	6.68	6.42	6.59	**9.89**
138	58.70	62.32	52.17	52.17	68.12
					0.0
Steel 16	48.47	71.70	**3.58**	4.99	50.00
	1796.76	2586.74	1096.97	1315.33	**3191.03**
266	16.62	**35.34**	15.04	18.65	33.42
695	64.17	43.45	43.31	41.87	56.69
					0.0
Steel 17	75.52	117.52	**3.06**	4.60	156.00
	1993.82	2626.25	818.90	836.64	**3799.56**
248	**49.30**	33.59	13.24	13.94	33.03

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
1121	20.07	38.80	30.69	29.79	57.09
					0.0
Steel 18	113.81	162.80	**6.03**	8.54	179.00
	4479.98	6390.55	3199.88	3199.88	**7979.68**
267	**108.08**	74.80	76.86	76.86	72.27
1144	20.02	41.26	20.10	20.10	53.32
					0.0
Steel 19	3.12	5.53	**0.74**	1.31	7201.00
	1133.31	1272.21	865.81	900.88	**1546.78**
257	273.54	**460.60**	289.76	448.47	380.25
319	51.72	34.48	37.30	25.08	50.78
					1.41818
Steel 20	83.01	122.57	**6.55**	9.58	135.00
	2862.39	3693.37	1860.70	1860.70	**4904.25**
268	20.19	41.87	**42.84**	**42.84**	40.13
964	65.35	40.66	20.02	20.02	56.33
					0.0
Steel 21	1.24	2.33	**0.40**	0.72	16.00
	331.61	451.44	394.65	394.65	**593.13**
264	14.23	27.12	17.92	17.92	**30.08**
138	65.94	47.10	62.32	62.32	55.80
					0.0
Steel 22	0.58	1.09	**0.21**	0.43	8.00
	409.14	476.14	391.70	391.70	**630.40**
248	854.93	994.93	1548.49	1548.49	**1676.50**
98	71.43	71.43	37.76	37.76	56.12
					0.0

Name	**OBS**	**OBS Multiple Boxes**	**OBS Mean**	**OBS Mean Multiple Boxes**	**MILP**
	Time [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]	*Time* [*s*]
D	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*	*Sum Score*
M	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]	*MSI* [%]
	Box size [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]	*Box size* [%]
					Gap [%]
Steel 23	2.73	4.73	**0.32**	0.52	345.00
	548.72	1002.92	397.44	380.52	**1103.50**
257	360.33	**1033.10**	453.90	547.69	1017.05
228	70.18	44.74	40.35	32.02	50.00
					0.0
Steel 24	84.96	131.16	**4.25**	6.02	135.00
	6303.90	7692.69	2990.58	2990.58	**8971.56**
267	43.49	67.93	**72.70**	**72.70**	68.44
826	70.82	55.33	20.10	20.10	64.04
					0.0
Steel 25	3.92	7.12	**1.23**	2.45	266.00
	1022.33	1157.13	491.85	531.46	**1688.35**
246	50.67	44.75	71.89	72.75	**81.34**
287	60.63	77.70	20.56	21.95	62.37
					0.0
Mean time [s]	12.46	17.69	**2.21**	3.13	669.02
Mean Sum Score	1009.53	**1236.25**	608.37	623.06	758.55
Mean MSI [%]	36.81	**44.95**	37.90	39.89	29.81
Mean box size [%]	44.27	41.69	29.40	28.59	40.58

Table 5: Results obtained on the industrial databases.