A dynamic programming approach to segmented isotonic regression

Víctor Bucarey, Martine Labbé, Juan M Morales, Salvador Pineda

A dynamic programming approach to segmented isotonic regression

Víctor Bucarey[a], Martine Labbé[b,e], Juan M. Morales[c,*], Salvador Pineda[d]

[a]*Data Analytics Laboratory, Vrije Universiteit Brussel, Brussels, Belgium*
[b]*Département d'Informatique, Université Libre de Bruxelles, Brussels, Belgium*
[c]*Department of Applied Mathematics, University of Malaga, Malaga, Spain*
[d]*Department of Electrical Engineering, University of Malaga, Malaga, Spain*
[e]*INRIA Lille Nord-Europe, France*

Abstract

This paper proposes a polynomial-time algorithm to construct the monotone stepwise curve that minimizes the sum of squared errors with respect to a given cloud of data points. The fitted curve is also constrained on the maximum number of steps it can be composed of and on the minimum step length. Our algorithm relies on dynamic programming and is built on the basis that said curve-fitting task can be tackled as a shortest-path type of problem. To ease the computational burden of the proposed algorithm, we develop various strategies to efficiently calculate upper and lower bounds that substantially reduce the number of paths to be explored. These bounds are obtained by combining relaxations of the original problem, clustering techniques and the well-known and well understood isotonic regression fit. Numerical results on synthetic and realistic data sets reveal that our algorithm is able to provide the globally optimal monotone stepwise curve fit for samples with thousands of data points in less than a few hours. Furthermore, the algorithm gives a certificate on the optimality gap of any incumbent solution it generates. From a practical standpoint, this piece of research is motivated by the roll-out of smart grids and the increasing role played by the small flexible consumption of electricity in the large-scale integration of renewable energy sources into current power systems. Within this context, our algorithm constitutes an

[*]Corresponding author

Email addresses: `vbucarey@vub.be` (Víctor Bucarey), `mlabbe@ulb.ac.be` (Martine Labbé), `juan.morales@uma.es` (Juan M. Morales), `spinedamorente@gmail.com` (Salvador Pineda)

useful tool to generate *bidding* curves for a pool of small flexible consumers to partake in wholesale electricity markets.

Keywords: Cardinality-constrained shortest path problem, isotonic regression, segmented regression, inverse optimization, data clustering, consumers' price-response

1. Introduction

In this paper, we deal with the problem of how to fit a curve to a given cloud of data points under the conditions that the fitted curve must be non-increasing (or non-decreasing) and piecewise constant (or, equivalently, *step-wise*), with a predefined limited number of pieces (also referred to as *steps* or *blocks* in what follows). This problem is inspired by the bidding rules that large consumers or a pool of small consumers must comply with when participating in an electricity market. Their bids for purchasing electricity in these markets must be often submitted in the form of a non-increasing stepwise price-consumption curve, for which the maximum number of bid blocks is also constrained. These curves reflect how consumers value electricity and therefore, their sensitivity to its price (which is referred to as consumers' elasticity), see, for instance, Su and Kirschen [23]. With the advent of Information and Communications Technologies and the roll-out of the so-called *smart grids*, small consumers of electricity are being provided with the means to actively adjust their consumption in response to the electricity price. However, their consumption patterns are still uncertain, dynamic and affected by other factors different from the electricity price. The result is that estimating a bidding curve that properly reflects consumers' sensitivity to the electricity price is a statistical challenge. This paper provides an algorithm to efficiently compute that curve from a set of price-consumption observations.

Beyond the practical context that inspires this piece of research, our work is closely related to various thrusts of research or thematic areas that also motivate it, namely:

Statistical regression. We desire to fit a monotonically decreasing curve to a given cloud of data points, while satisfying the following two extra conditions: i) The fitted curve must be piecewise constant and ii) there is a maximum number of pieces the fitted curve can be comprised of.

While the literature review includes a wealth of research papers analyzing related concepts and tools such as *isotonic* regression (see, e.g., Mair et al. [15], Tibshirani et al. [25], Guyader et al. [11] and references therein), *segmented* regression (Muggeo [16]), and the popular *multivariate adaptive regression spline* (Friedman [9]), these regression techniques produce fitted curves that fail to satisfy at least one of the conditions mentioned above. Furthermore, they are frequently based on iterative, greedy or heuristic algorithms. Indeed, the fitted response of the isotonic regression is a monotone piecewise constant function (although efficient algorithms to produce smooth continuous functions are also available, see, e.g., Sysoev and Burdakov [24]), but is not limited in the number of pieces it may be comprised of. For its part, segmented regression leads to curve fits that are not necessarily monotone. Against this background, we propose an exact shortest-path algorithm that is capable of delivering, in polynomial time, the monotone stepwise curve (with a maximum of K steps) that constitutes the globally optimal data fit according to the least-squares criterion.

We remark that, as pinpointed in Lerman [14], the stepwise shape of the target curve releases the fitting process from the continuity condition at the breakpoints that is typically enforced in segmented regression, thus making it computationally easier. On the other hand, we additionally impose that the fitted curve be non-increasing, which adds an extra layer of complexity to the regression problem at hand. Actually, to our knowledge, the works that are the closest to ours are those of Hawkins [12] and Dahl and Realfsen [5]. In the former, they describe a dynamic programming approach to perform segmented regression over a sequence of observations with at most K segments and no continuity requirement at the transition points. Dahl and Realfsen [5] offer an interesting computational perspective on this same problem, which they pose as a cardinality-constrained shortest path problem and for which they propose several solution algorithms. Neither Hawkins [12], nor Dahl and Realfsen [5] consider, however, any monotonicity constraint, which is, though, critical to our problem (seen as an extension or generalization of isotonic regression) and to the practical application that motivates it.

Finally, we mention that Rote [20] uses dynamic programming for isotonic regression, but, again, with no constraint on the number of pieces the fitted curve can be made up of.

Inverse optimization. Recently, inverse optimization has emerged as a promising mathematical framework to infer the input parameters to an optimization problem that have given rise to a series of optimal (or quasi-optimal) solutions (Ahuja and Orlin [1], Esfahani et al. [6], Chan et al. [4]). In the last few years, inverse optimization has been widely used to infer consumers' utility from a certain product (Keshavarz et al. [13], Aswani et al. [2]), in particular, electricity (Saez-Gallego et al. [22], Saez-Gallego and Morales [21]). Essentially, it is often assumed that the market behavior of a pool of (rational) electricity consumers is driven by the following maximization problem

$$\underset{x \geq 0}{\text{Maximize}} \quad \int_0^x b(s)ds - px$$

where px is the payment the pool of consumers has to make for purchasing x units of electricity in the market at price p, and $b(\cdot)$ is the so-called *bidding curve* expressing the response of the consumers to the electricity price. Many electricity markets around the world require that this bidding curve be non-increasing and stepwise, with a maximum number of steps. Dealing with this problem by way of inverse optimization involves estimating the step function values of this curve and the breakpoints from a series of observed pairs $\{(\hat{p}_i, \hat{x}_i)\}_{i=1}^{I}$. As highlighted in Aswani et al. [2], however, the available estimation approaches based on inverse optimization may result in statistically *inconsistent* estimators or require the reformulation of the problem as a bilevel (NP-hard) problem. In this regard, Aswani et al. [2] propose a statistically consistent polynomial-time semiparametric algorithm to tackle a certain class of inverse optimization problems. Nevertheless, the regression problem we address here, when seen from the lens of inverse optimization, does not comply with the conditions that ensure the statistical and polynomial-time performance of their algorithm, because some of the parameters to be estimated, specifically, the breakpoints, appear in the constraints defining the feasible region of the forward problem. In contrast, we propose an algorithm that directly solves the statistically consistent formulation of the problem to optimality in polynomial time.

Unsupervised learning. The problem we address in this paper can be also interpreted as a clustering problem through which a series of observed pairs $\{(\hat{p}_i, \hat{x}_i)\}_{i=1}^{I}$ are grouped in such a way that:

1. There is a maximum number of clusters K into which the data points can be grouped into.
2. The resulting clusters must satisfy some connectivity constraints. In our particular case, these connectivity constraints impose that only clusters with adjacent prices $\hat{p}_i$ can be merged together (see, e.g., Guo [10]).
3. If (p_m^*, x_m^*) and (p_n^*, x_n^*) are the centroids of clusters m and n, respectively, then $x_m^* \geq x_n^* \iff p_m^* \leq p_n^*$ in order to guarantee a non-increasing curve.

The technical literature includes some works in which structured clustering is used in power system applications. For instance, Pineda and Morales [19] propose a hierarchical clustering methodology to approximate time series that are used to determine the optimal expansion planning of the European electricity network. Due to the usual NP-hard nature of clustering methods, the clusters are often obtained through computationally efficient greedy algorithms. However, to the best of our knowledge, the technical literature does not report any clustering methodology that simultaneously satisfies the three conditions specified above. Therefore, our work also contributes to the realm of structured data clustering.

The rest of this paper is organized as follows. In Section 2, we formulate the curve-fitting problem that we aim to solve. Section 3 introduces the solution algorithm we propose to that end, which is based on dynamic programming and, more specifically, on the cardinality-constrained shortest path problem. Section 4 provides various strategies to accelerate said algorithm, whose performance is subsequently tested in Section 5 using synthetic data sets and a data set coming from a real-life practical application. Lastly, conclusions are duly drawn in Section 6.

2. Problem formulation

Consider a given set of pairs of points on the real plane $\{(\hat{p}_i, \hat{x}_i)\}_{i=1}^{I}$. Without loss of generality, we assume that $\hat{p}_1 < \hat{p}_2 < \ldots < \hat{p}_I$. Let $\mathcal{F}$ be the class of real functions $f : [\hat{p}_1, \hat{p}_I] \rightarrow \mathbb{R}$ that are non-increasing and piecewise constants, with at most K blocks or steps, $K \in \mathbb{Z}_+$. We seek to solve the following least-square minimization problem, hereinafter referred to as *LSP*:

$$\text{(LSP)} \qquad \min_{f \in \mathcal{F}} \sum_{i=1}^{I} \left(\hat{x}_i - f(\hat{p}_i)\right)^2 \tag{1}$$

A function f member of the class $\mathcal{F}$ can be expressed as

$$f(p) = \sum_{k=1}^{K} u_k \mathbb{I}_{[p_k, p_{k+1})}(p) \tag{2}$$

where $\mathbb{I}_{[p_k, p_{k+1})}(p)$ is the indicator function equal to 1 if $p_k \leq p < p_{k+1}$, and 0 otherwise. Again without loss of generality, we set $p_1 = \hat{p}_1$ and use $p_{K+1} = \hat{p}_{I+1} > \hat{p}_I$ as a dummy p-coordinate to guarantee that all $\hat{p}_i$ are covered by the solution. Besides, $u_1 \geqslant u_2 \geqslant \ldots \geqslant u_K \geqslant 0$ represent the *step values* of the blocks. We remark that functions $f \in \mathcal{F}$ with less than K blocks can be also represented in this way, since two consecutive blocks are allowed to have the same function value.

Using this characterization of the class of functions $\mathcal{F}$ and taking $p_1 = \hat{p}_1$ and $p_{K+1} > \hat{p}_I$ a dummy price coordinate as mentioned above, problem (1) can be recast as follows.

$$\min_{\mathbf{u},\mathbf{p}} \sum_{i=1}^{I} \left(\hat{x}_i - \sum_{k=1}^{K} u_k \mathbb{I}_{[p_k, p_{k+1})}(\hat{p}_i)\right)^2 \tag{3a}$$

$$\text{s.t. } u_k \geqslant u_{k+1}, \ \forall k \leqslant K-1 \tag{3b}$$

$$p_{k+1} \geqslant p_k, \ \forall k \leqslant K \tag{3c}$$

Determining the breakpoints $\{p_k\}_{k=2}^{K}$, which are needed to compute the indicator functions appearing in the objective function (3a), constitutes the major source of complexity in problem (3). Constraint (3b), which enforces the non-increasing character of the fitted curve, also adds another layer of difficulty to the selection of those breakpoints. The easiest task in problem (3) is to compute the values u_k that minimize the squared error (3a) for a given set of intervals $[p_k, p_{k+1})$. In the following section, we introduce a shortest path algorithm through which we can solve problem (3) in polynomial time. This algorithm starts from the evidence that the optimal breakpoints are within the p-coordinates of the cloud of points $\{(\hat{p}_i, \hat{x}_i)\}_{i=1}^{I}$.

3. Cardinality-constrained shortest-path algorithm

For a function $f \in \mathcal{F}$, the objective function value of problem (1) can be rewritten as:

$$\sum_{k=1}^{K} \sum_{i:p_k \leq \hat{p}_i < p_{k+1}} (\hat{x}_i - u_k)^2 \tag{4}$$

First, let $\hat{p}_i$ be the smallest p-coordinate of a data point larger than or equal to p_k. Replacing p_k by any value $p_{k'}$ with $p_k \leq p_{k'} \leq \hat{p}_i$ does not change the value of the objective function. Hence, we can restrict our search for the breakpoints p_k to the set $\{\hat{p}_i : 1 \leq i \leq I+1\}$ of the p-coordinates of the data points.

Second, consider an optimal solution of problem (1) represented by breakpoints $\{p_k^*\}_{k=1}^K$ and step values $\{u_k^*\}_{k=1}^K$. Each time that two consecutive blocks, say k' and $k'+1$, have the same function value, i.e., $u_{k'}^* = u_{k'+1}^*$, we can merge them and reduce the number of blocks. Consequently, this optimal solution can be described by K' blocks, with $K' \leq K$, such that $u_k^* < u_{k+1}^*$, for $k = 1, \ldots, K'-1$. Given that this solution is optimal, it must be locally optimal, i.e., u_k^* must minimize the contribution of block k to (4). In other words,

$$\begin{aligned} u_k^* \in \arg\min_{u_k}\{ &\sum_{i:p_k^* \leq \hat{p}_i^* < p_{k+1}^*} (\hat{x}_i - u_k)^2 : u_{k-1}^* \leq u_k \leq u_{k+1}^*\} \\ &= \{AV(p_k^*, p_{k+1}^*), u_{k-1}^*, u_{k+1}^*\}, \end{aligned} \tag{5}$$

where $AV(p_k^*, p_{k+1}^*)$ represents the average value of the coordinates $\hat{x}_i$ of data points such that $p_k^* \leq \hat{p}_i < p_{k+1}^*$. Given that the step values are all different, it follows that $u_k^* = AV(p_k^*, p_{k+1}^*)$. Further, the contribution of block k to the total error is equal to

$$ER(p_k^*, p_{k+1}^*) = \sum_{i:p_k^* \leq \hat{p}_i < p_{k+1}^*} (\hat{x}_i - (AV(p_k^*, p_{k+1}^*))^2 \tag{6}$$

We remark that a similar reasoning applies if we consider the least absolute error (that is, the minimization of the sum of absolute values of errors), instead of the least squares. In that case, it suffices to replace the average value of the $\hat{x}_i$-coordinates of the data points such that $p_k^* \leq \hat{p}_i < p_{k+1}^*$ with their *median.*

The above two properties allow to translate problem (1) into a shortest path problem with at most K arcs on a particular directed graph $G = (V, A)$ defined as follows. Its vertex set is $V := \{v_{ij} : 2 \leq i < j \leq I\} \cup \{s, t\}$. Vertices s and t constitute the source and the sink between which a shortest path must be found and correspond to the first and the last block of the solution, respectively. Each intermediate vertex v_{ij} represents a block that is neither the first one nor the last one, limited by $\hat{p}_i$ and $\hat{p}_j$ and whose step value can be explicitly computed as $AV(\hat{p}_i, \hat{p}_j) = (\sum_{i \leq h < j} \hat{x}_h)/(j - i)$.

The arc set A contains four different types of arcs:

- Arcs from the source to intermediate vertices: $(s, v_{ij}) \in A$ for all $v_{ij} \in V$ for which $AV(\hat{p}_1, \hat{p}_i) > AV(\hat{p}_i, \hat{p}_j)$. They represent the possible first block of a solution $[\hat{p}_1, \hat{p}_i)$ and their cost is given by $ER(\hat{p}_1, \hat{p}_i)$.

- Arcs between intermediate vertices: $(v_{ij}, v_{jh}) \in A$ for all vertices $v_{ij}, v_{jh} \in V$ such that $1 \leq i < j < h \leq I$ and $AV(\hat{p}_i, \hat{p}_j) > AV(\hat{p}_j, \hat{p}_h)$. An arc (v_{ij}, v_{jh}) translates the existence of the two consecutive blocks $[\hat{p}_i, \hat{p}_j)$ and $[\hat{p}_j, \hat{p}_h)$ in the solution and its cost is given by the error contribution of the first of the two blocks, i.e., $c(v_{ij}, v_{jh}) = ER(\hat{p}_i, \hat{p}_j)$.

- Arcs from intermediate vertices to the sink: $(v_{ij}, t) \in A$ for all $v_{ij} \in V$ for which $AV(\hat{p}_i, \hat{p}_j) > AV(\hat{p}_j, \hat{p}_{I+1})$ with cost $c(v_{ij}, s) = ER(\hat{p}_j, \hat{p}_{I+1})$. They represent the last block of a solution that necessarily contains the last data point.

- Arc from the source to the sink: $(s, t) \in A$ with an associated cost $c(s, t) = ER(\hat{p}_1, \hat{p}_{I+1})$ corresponds to the solution with one single block $[\hat{p}_1, \hat{p}_{I+1})$.

Our least square minimization problem is equivalent to finding a path in the graph $G = (V, A)$, from s to t with at most K arcs and with minimum total cost. Given that all costs are non-negative and the graph is *acyclic*, an optimal path can be found in $\mathcal{O}(K|A|)$, see, e.g., Wolsey [26, ch. 5, pp. 68]. Further, since $K \leq I$ and $|A| = \mathcal{O}(I^3)$, it follows that LSP can be solved in polynomial time.

Alternatively, LSP can be equivalently posed as a shortest path problem with resource constraints in a different, but substantially smaller graph $G'(V', A')$ with vertex set $V' = \{v_i : 1 \leq i \leq I + 1\}$ and edge set $A' = \{(v_i, v_j) : 1 \leq i < j \leq I + 1\}$. Each arc (v_i, v_j) corresponds to a block

$[\hat{p}_i, \hat{p}_j)$ with step value $AV(\hat{p}_i, \hat{p}_j)$ and cost $c(v_i, v_j) = ER(\hat{p}_i, \hat{p}_j)$. There is a one-to-one correspondence between the paths in G' from v_1 to v_{I+1} and the set of stepwise functions with breakpoints in $\{\hat{p}_i : 1 \leq i \leq I+1\}$. To obtain feasible solutions to LSP, we must impose two resource constraints. The first one consists in setting an upper bound K on the numbers of arcs of a path and the second one excludes the presence of consecutive arcs with increasing step values in a path.

The standard approach for solving such a problem consists in using dymamic programming to construct the path from v_1 to v_{I+1} progressively, see e.g. Feillet et al. [7]. The procedure contains I iterations and at iteration i, partial paths ending in vertex $v_i \in V'$ are extended by adding one arc (v_i, v_j).

Further, a label is associated to each feasible partial path π from v_1 to $v_i \in V'$ specifying the consumption of the resources. Here the label is a triplet $(c(\pi), k(\pi), st(\pi))$ where $c(\pi)$ denotes the total error of the partial path, $k(\pi)$ its number of arcs and $st(\pi)$ the step value of the block corresponding to the last arc of the partial path. On the one hand, the label allows to check whether extending a path π ending in v_i by an arc (v_i, v_j) is feasible since we need that $k(\pi) \leq K$ and $st(\pi) > AV(\hat{p}_i, \hat{p}_j)$. On the other hand, *dominance* between partial paths ending in a same vertex can be exploited. If π and π' are two partial paths ending in v_i, such that $c(\pi) \leq c(\pi'), k(\pi) \leq k(\pi')$ and $st(\pi) \geq st(\pi')$, then clearly path π' cannot be part of a feasible path from v_1 to v_{I+1} that has a strictly better total error. In consequence, all along the execution of the algorithm, we only need to consider the partial paths with different and non-dominated labels.

The number of different non-dominated partial paths ending in vertex v_i is in $\mathcal{O}(KI)$ (bear in mind that vertex v_i can be reached in at most K arcs and that the last of these arcs can be associated with at most $i-1$ different step values, namely, $AV(\hat{p}_j, \hat{p}_i)$, with $j = 1, 2, \ldots, i-1$). Besides, the number of arcs with v_i as the origin vertex is in $\mathcal{O}(I)$. Consequently, the number of new candidate partial paths generated at iteration i is in $\mathcal{O}(KI^2)$ and the overall complexity of this algorithm is thus the same as the previous one, i.e, $\mathcal{O}(KI^3)$.

Conveniently, we may also take advantage of upper and lower bounds to accelerate the search for the optimal path. Indeed, let INC be the value of a feasible solution to LSP obtained either in some previous iteration of the algorithm or by some other means. Consider a partial path π ending in v_i and let $LB(v_i, k(\pi), st(\pi))$ be a lower bound on the cost of a partial path from v_i to v_{I+1} with at most $K - k(\pi)$ arcs, non-increasing step values and smaller than $st(\pi)$. If $c(\pi) + LB(v_i, k(\pi), st(\pi)) \geq INC$, then the partial path π can be directly discarded.

A quick valid lower bound can be obtained by relaxing either the condition on the maximum number of arcs or the constraint on the monotonicity of the step values. In the former case, the problem boils down to an isotonic regression problem on the data points $\{(\hat{p}_j, \hat{x}_j)\}_{j=i}^{I}$. In the latter, it is a lighter shortest path problem from v_i to v_{I+1} in G' with at most $K - k(\pi)$ arcs.

The whole procedure is described in Algorithm 1 in which N_i represents the set of labels in the form $\ell(\pi) = (c(\pi), k(\pi), st(\pi))$ of different and non-dominated partial paths ending in v_i. Further, $PRED(\pi)$ is used to store the "predecessor" of π, which is the partial path, say π', that has been extended by one arc to obtain π. Once the algorithm terminates, this information allows to reconstruct the optimal path backward, starting with $PRED(OPTIMAL - PATH)$.

4. Acceleration strategies

Despite the fact that LSP can be solved in polynomial time, computing the optimal solution can be expensive for realistic instances. The overall solution time relies heavily on how tight the upper bound INC and the lower bounds $LB(\cdot)$ are. In this section we discuss strategies to find good bounds that are easy to compute.

4.1. Computing an upper bound: Combining isotonic regression with adjacency-constrained data clustering

Feasible solutions provide us with an upper bound on the optimal error that can help us reduce the computational burden of the shortest path problem presented in Section 3. One efficient procedure to compute a tight upper bound runs as follows:

1. We use isotonic regression to fit a monotone stepwise function to the original data set. However, one should expect the number of blocks of this fit to be higher than K.
2. We reduce the number of blocks of the output of the isotonic regression to K by grouping the consumption values of the isotonic fit into K clusters. For this purpose, we use the fast greedy algorithm proposed in Pineda and Morales [19] for adjacency-constrained hierarchical clustering.

Algorithm 1 Shortest path algorithm for LSP

```
1: Initialization: $N_1 = \{(0, 0, \max_{i \in I} \hat{x}_i + 1)\}$, $INC = +\infty$
2: for $i \in \{1, \ldots, I\}$ do
3:   while $N_i \neq \emptyset$ do
4:     Select $\pi^* \in \arg\min_{\ell(\pi) \in N_i}\{c(\pi)\}$ and remove $\ell(\pi^*)$ from $N_i$
5:     if $c(\pi^*) > INC$ then
6:       $N_i = \emptyset$
7:     else if $c(\pi^*) + c(v_i, v_{I+1}) < INC$ and $AV(p_i, p_{I+1}) < st(\pi^*)$ then
8:       $PRED(OPTIMAL - PATH) = \pi^*$, $INC = c(\pi^*) + c(v_i, v_{I+1})$
9:     end if
10:    if $k(\pi^*) < K - 1$ then
11:      for $h \in i + 1, \ldots, I$ do
12:        if $AV(p_i, p_h) < st(\pi^*)$ and $c(\pi^*) + c(v_i, v_h) + LB(v_h, k(\pi^*) + 1, AV(p_i, p_h)) < INC$ then
13:          $new = (c(\pi^*) + c(v_i, v_h), k(\pi^*) + 1, AV(p_i, p_h))$, $PRED(new) = \pi^*$
14:          if $new \notin N_h$ then
15:            Add label $new$ to $N_h$ if it is not dominated.
16:            Delete all dominated labels.
17:          end if
18:        end if
19:      end for
20:    end if
21:  end while
22: end for
```

3. The step value is computed as the average consumption of the isotonic fit values within each of the K clusters obtained in the previous point.

The procedure above yields a monotone stepwise function with K pieces that is a feasible solution to LSP. This methodology is depicted in Figure 1, with each subfigure representing one of the actions described above, from left to the right. We implement the calculation of the so-obtained upper bound on Python, using the isotonic regression and the agglomerative clustering functions of package *Scikit-learn*, see Pedregosa et al. [18].

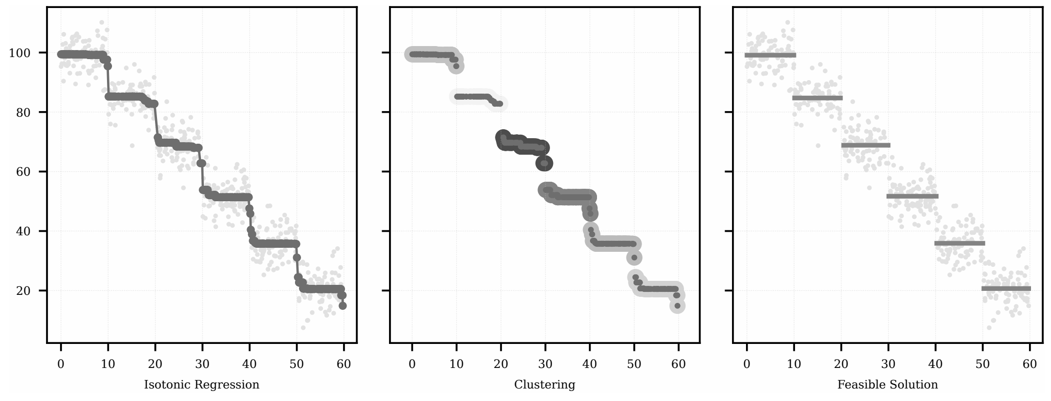


Figure 1: Algorithm to compute a feasible solution and an upper bound to the minimum error.

4.2. Computing a lower bound

Lower bounds are useful in several ways. First, as we discussed in Section 3, they prevent the shortest-path algorithm from creating sub-optimal labels. To do so, it is necessary to compute lower bounds for each partial path π. Depending on the method, this can be computationally expensive. Second, lower bounds give a guarantee of how far any feasible solution is from the optimal one. We obtain these lower bounds by relaxing either the constraint on the number of blocks/arcs or the monotonicity constraint of the fitted curve.

Relaxing the constraint on the number of arcs in the path: The isotonic fitted curve. When the number of blocks is not limited, problem LSP is equivalent to the well-known isotonic fit, (Fielding [8]). Isotonic regression can be solved in linear time (Best and Chakravarti [3]). Given the efficiency of this method, we generate lower bounds for any partial path π by computing one lower bound for each vertex v_i. In other words, we calculate $LB(v_i, k(\pi), st(\pi))$ as $LB(v_i)$ for each partial path π. The total time to compute this lower bound for all v_i is in $O(I^2)$. We use the isotonic regression function implemented in the Python package *Scikit-learn* to this end (Pedregosa et al. [18]).

Relaxing the monotonicity constraint. As mentioned in Section 3, another lower bound can be obtained by relaxing the monotonicity constraint on the step values. Then, for a given partial path π, a lower bound can be computed by solving a shortest path problem with at most $K - k(\pi)$ arcs from v_i to v_{I+1} in G'. One can determine the fitting error associated with the shortest paths from all vertices v_i to the sink v_{I+1} and containing at most k arcs, for all $k = 1, \ldots, K$, with dynamic programming. The corresponding total computing time is in $O(K|A|) = O(KI^2)$ (essentially, if we disregard the monotonicity constraint, our problem translates into a standard cardinality-constrained shortest problem whose computational complexity is known to be in $\mathcal{O}(K|A|)$, with $|A| = \mathcal{O}(I^2)$ in our case).

In terms of implementation, to relax the monotonicity constraint is equivalent to suppressing the third component st of each label and the corresponding monotonicity conditions in line 7 and 12 in Algorithm 1. In particular, for the path that corresponds to the initial label $(0, 0)$ and contains the single vertex v_1, this shortest path problem returns a lower bound on the minimum fitting error. In that case, besides, if the resulting function turns out to be non-increasing, then it must be optimal to LSP.

We end this subsection with a remark on the so modified algorithm: The minimum cost computed over all the partial paths reaching a layer i in the modified algorithm is a lower bound on that very same cost in the original Algorithm 1. Consequently, we can get an even tighter lower bound by running Algorithm 1 with the monotonicity constraint dropped and with $LB(v_h, k(\pi^*)+1, AV(p_i, p_h))$ in line 12 of the pseudocode given by the isotonic fit. The total cost at termination does not necessary correspond to the fitting error of the optimal, possibly non-monotone, stepwise curve, but it is a still valid lower bound on the cumulative error of LSP. This is indeed the lower bound (obtained from dropping the monotonicity constraint on the optimal fitted curve) that we will consider in the numerical experiments below.

4.3. Imposing constraints on the length of steps

In some cases, it may be interesting and practically useful to impose constraints on the length of the function blocks, i.e., to restrict the set of functions $\mathcal{F}$ to decreasing stepwise functions with a step length bigger than `step_min`. This is equivalent to adding the following set of constraints to model (3):

$$\texttt{step\_min} \leq p_{k+1} - p_k, \quad \forall k \leq K \tag{7}$$

Moreover, Algorithm 1 can still be used after removing from the arc set A all (v_i, v_j) for which the corresponding p-coordinates violate condition (7). As a result, the number of operations to compute the optimal solution to LSP decreases. We show the impact of imposing this type of constraint experimentally in the next section.

We remark that the upper bound described in Section 4.1 may no longer be feasible after enforcing the constraint on the minimum step length. Nevertheless, if that is the case, we can always gradually decrease K in the algorithm outlined in that section until a valid upper bound is eventually recovered.

5. Numerical experiments

Next we run a series of numerical experiments to test the effectiveness and performance of the proposed algorithm under different settings. To this end, we first use synthetic data sets to assess the sensitivity of the algorithm performance to the maximum number of steps, the noise level in the input data and the sample size. Subsequently, we consider a realistic data set consisting of price-power measurements at the main substation of a distribution power grid that includes distributed energy resources. This data can be download from [17].

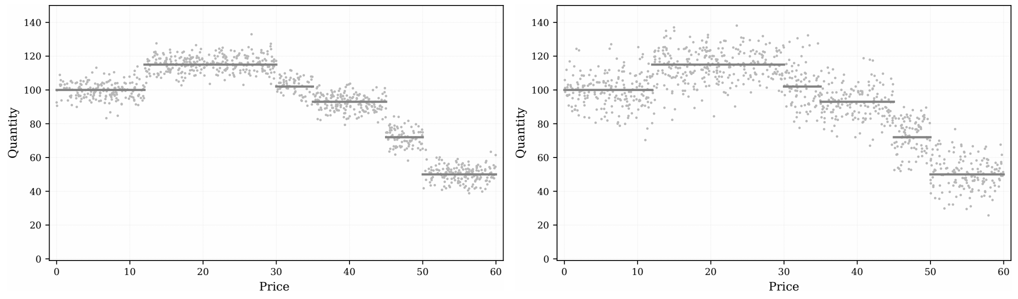
5.1. Synthetic data sets

We first test our algorithm and the effectiveness of the acceleration strategies described above on a controlled experiment, where we know the true data-generating distribution. More specifically, the response variable x is given by

$$x = f^*(p) + \varepsilon \tag{8}$$

k-th block	$[p_k^*, p_{k+1}^*)$	u_k^*
1	[0,12)	100
2	[12,30)	115
3	[30,35)	102
4	[35,45)	93
5	[45,50)	72
6	[50,60]	50

Table 1: Stepwise characterization of the *true* relationship between the response x and the covariate p.



(a) $\sigma = 5$ (b) $\sigma = 10$

Figure 2: Synthetic data for different noise levels.

where ε is a Gaussian noise of zero mean and standard deviation σ and f^* is the stepwise function depicted in Figure 2, that is,

$$f^*(p) = \sum_{k=1}^{6} u_k^* \mathbb{I}_{[p_k^*, p_{k+1}^*)}(p) \tag{9}$$

with u_k^* and $[p_k^*, p_{k+1}^*)$, $k = 1, \ldots, 6$, provided in Table 1. Notice that f^* is a stepwise function made up of six blocks of different sizes. Furthermore, f^* is neither increasing, nor decreasing in its entire domain. For illustration purposes, Figures 2a and 2b plot 1000 data points $\{(\hat{p}_i, \hat{x}_i)\}_{i=1}^{I}$ randomly generated for two different noise levels, namely, $\sigma = 5$ and $\sigma = 10$, respectively. Both figures also include the true function (9) to compute the response variable x.

Next, we run our shortest path algorithm using datasets of 1000 points that are randomly generated from (8) for a noise level σ taking the values of the natural numbers between zero and ten. Besides, the number of arcs K

is set to six, which is the true number of blocks of the function that relates the response variable x to the covariate p. Results for all these cases are collated in Table 2 and include the aggregated square error (Error) and three different computational times (with a maximum value of 12 hours):

- $\mathrm{T}^{\mathrm{ISO}}$: Computational time of the proposed shortest path algorithm including the upper bound discussed in Section 4.1 and the lower bound per layer provided by the isotonic fit.
- $\mathrm{T}^{\mathrm{RLX}}$: This computational time is obtained as follows. Let $\mathrm{T}^{\mathrm{W/O}}$ denote the time needed to run the proposed shortest path algorithm *without the monotonicity constraint*, but including the upper bound discussed in Section 4.1 and the lower bounds (one per layer) provided by the isotonic regression fits. As mentioned in Section 4.2, the cost of this shortest path constitutes a valid lower bound on the cumulative fitting error associated with the optimal monotone curve. Actually, if this shortest path leads to a nonincreasing curve, then this is the optimal one. In this case, we set $\mathrm{T}^{\mathrm{RLX}} = \mathrm{T}^{\mathrm{W/O}}$ (because the optimal fit has been found). Otherwise, we need to rerun our algorithm *with the monotonicity constraint* back in force, and therefore, we set $\mathrm{T}^{\mathrm{RLX}} = \mathrm{T}^{\mathrm{W/O}} + \mathrm{T}^{\mathrm{ISO}}$.
- $\mathrm{T}^{\mathrm{NOB}}$: Computational time of the proposed shortest path algorithm if no acceleration strategies are employed.

By comparing the computational times $\mathrm{T}^{\mathrm{NOB}}$ and $\mathrm{T}^{\mathrm{ISO}}$ of Table 2, we can conclude that the use of the proposed upper and lower bounds has a tremendous impact on the ability of the algorithm to quickly identify the globally optimal curve to be fitted. Besides, these results also reveal that our algorithm is robust to the level of noise, since the computational time $\mathrm{T}^{\mathrm{ISO}}$ is relatively stable as noise increases. Finally, the lower bound provided by the relaxation of the monotonicity constraint is, nevertheless, of little value for this instance, in which $\mathrm{T}^{\mathrm{RLX}}$ is higher than $\mathrm{T}^{\mathrm{ISO}}$ for most noise levels. We will see, however, that this lower bound can be useful when the data features a sufficiently marked monotonic layout.

In order to better understand the intuition behind the acceleration strategies described in Section 3 and their impact on the computational time of the shortest path problem proposed in Section 4, Figure 3 displays the following:

I	σ	K	Error	$\mathrm{T}^{\mathrm{ISO}}$(s)	$\mathrm{T}^{\mathrm{RLX}}$(s)	$\mathrm{T}^{\mathrm{NOB}}$(s)
1000	0	6	28081	110	164	13053
1000	1	6	27491	647	837	>43200
1000	2	6	30716	1296	1569	39205
1000	3	6	35320	441	644	33675
1000	4	6	41852	1258	1501	41128
1000	5	6	52919	1038	1177	37186
1000	6	6	64416	1400	1547	>43200
1000	7	6	67985	1222	1447	>43200
1000	8	6	102860	1998	2346	>43200
1000	9	6	113847	1570	1903	42156
1000	10	6	123856	1162	1143	32689

Table 2: Impact of data noise

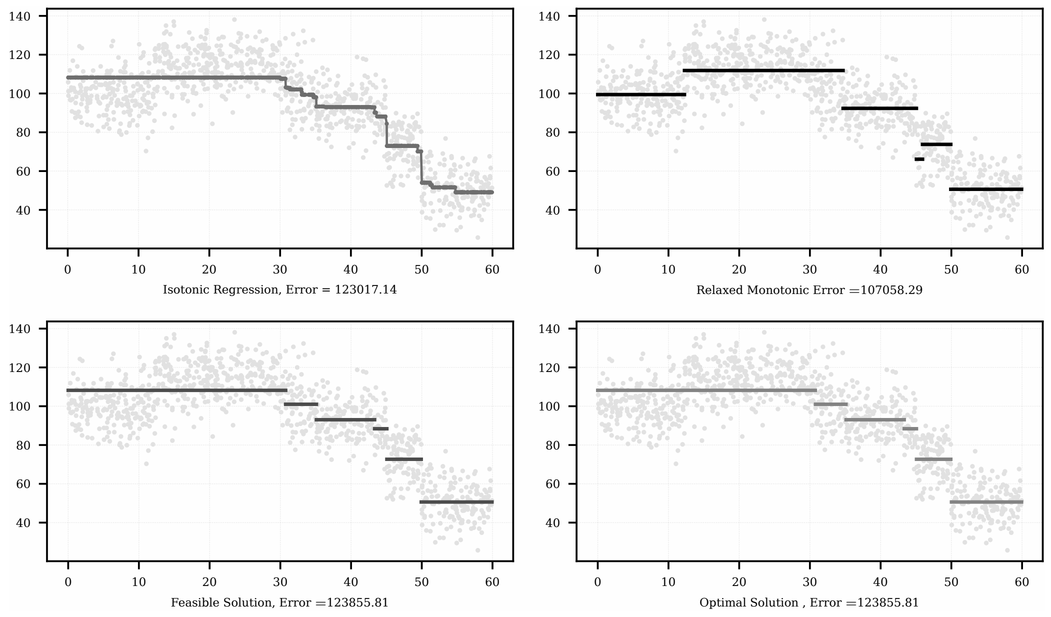


Figure 3: Illustration of the various bounds and the optimal solution for the synthetic data with noise $\sigma = 10$ and $K = 6$

- Top left plot: The curve provided by the isotonic regression. As observed, the isotonic fit is non-increasing, but the number of steps is higher than $K = 6$. Therefore, the aggregated squared error associated with this curve can be used to lower-bound the optimal solution.

- Bottom left plot: The curve obtained through the adjacency-constrained hierarchical clustering technique using the isotonic fit as input. The number of blocks is equal to six and the monotonic condition is also satisfied. Therefore, this curve represents a feasible solution in LSP and its accrued squared error is a valid upper bound of the optimal objective function value.

- Top right plot: The curve computed by the relaxed shortest path algorithm without the monotonicity constraint. The number of blocks is also equal to six, but the curve is not monotone. Therefore, the corresponding aggregate squared error can also be used as a lower bound.

- Bottom right plot: The global optimal solution obtained by the proposed shortest path algorithm.

Interestingly, the optimal solution of this particular instance coincides with that resulting from the combination of the isotonic regression and the structured hierarchical clustering. Furthermore, the lower bound provided by the isotonic fit is notoriously tight, which is, most likely, the reason behind the good performance exhibited by our algorithm. Notice that the lower bound achieved by relaxing the monotonicity constraint is significantly less tight than the one given by the isotonic regression fit.

To further illustrate how the proposed bounds can accelerate the solution of the shortest path algorithm, Figure 4 shows the time spent per iteration by our algorithm for noise levels $\sigma = 0$, 5 and 10. In this figure, the dashed plots refer to the raw implementation of the algorithm, i.e., with no bounds; the dotted lines correspond to the version of the algorithm where only the proposed upper bound is used; finally, the solid plots provide the time our algorithm spends per iteration when both the lower and the upper bounds are exploited. It is apparent that using bounds in the proposed methodology has a remarkable beneficial effect on the algorithm performance, to such an extent that the joint use of both bounds manages to immunize the algorithm against the noise. Indeed, our upper and lower bounds noticeably reduce the number of labels that Algorithm 1 generates in the intermediate layers

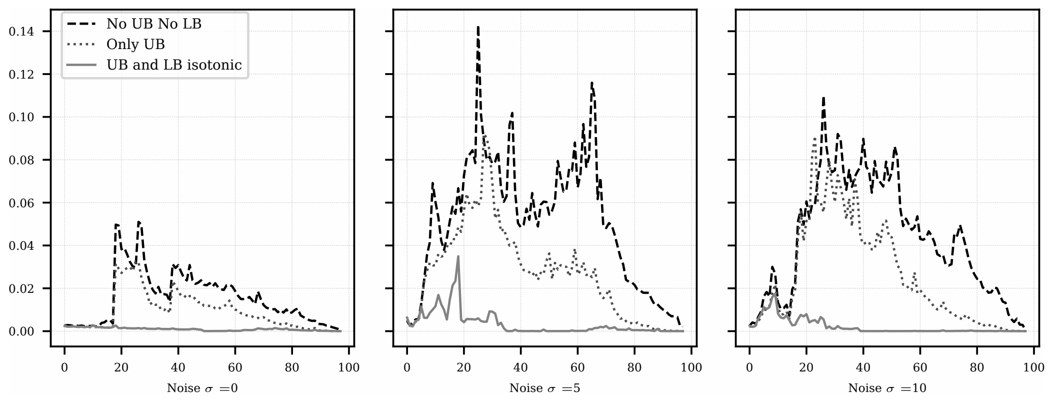


Figure 4: Effect of bounds on computational burden per iteration. Example with 100 data points and $K = 6$

of the graph. In the limiting case where there is no noise, both the upper and the lower bounds coincide with the optimal fit and no intermediate label is generated at all, thus taking a marginal amount of time per iteration. As the noise level is increased, more and more intermediate labels are to be handled, which essentially tells us that the optimization underlying the regression problem becomes harder and harder to perform. Furthermore, as can be inferred from the plots of Figure 4, the inclusion of the upper bound only is not enough to keep the computational burden of our algorithm per iteration low, because of the high amount of labels that are produced in the first layers of the graph. It is the synergistic effect of the lower and upper bounds which prevents the number of labels in the early stages of our algorithm from exploding.

We note that, given the notorious superiority of our algorithm with bounds to its raw version, computational time $\mathrm{T}^{\mathrm{NOB}}$ is no longer included in the remaining analysis performed in this section.

Now we fix the noise level σ to five and change the sample size instead. Still we have $K = 6$. The comparison results of $\mathrm{T}^{\mathrm{ISO}}$ and $\mathrm{T}^{\mathrm{RLX}}$ for this new experiment are collated in Table 3. Naturally, the aggregate squared error and the solution times increase with the sample size I. However, our algorithm appears to scale relatively well, given its theoretical complexity.

Finally, we fix the sample size to 1000 and the noise level of the data to five, and change the maximum number of arcs K our algorithm may use to reduce the error. The so obtained results are compiled in Table 4. As expected, by increasing the maximum number of arcs K (also referred

I	σ	K	Error	$\mathrm{T}^{\mathrm{ISO}}$(s)	$\mathrm{T}^{\mathrm{RLX}}$(s)
100	5	6	5074	0	0
200	5	6	11176	2	3
500	5	6	28810	73	99
1000	5	6	52919	1038	1177
2000	5	6	112410	9092	10726

Table 3: Impact of sample size

I	σ	K	Error	$\mathrm{T}^{\mathrm{ISO}}$(s)	$\mathrm{T}^{\mathrm{RLX}}$(s)
1000	5	2	118510	0	0
1000	5	3	81549	42	26
1000	5	4	55242	585	52
1000	5	5	53275	744	884
1000	5	10	52528	1799	1821
1000	5	12	52493	1622	1798
1000	5	14	52479	1817	2058
1000	5	16	52471	1951	2361
1000	5	18	52466	2234	2791
1000	5	20	52465	2734	3637

Table 4: Impact of the maximum number of arcs K

to as *number of blocks*), we enrich the family $\mathcal{F}$ of non-increasing stepwise functions we consider and thus, the error of the data fitting is reduced. If the number of blocks K is lower than or equal to four, the solution obtained by the relaxed shortest path without the monotonicity constraint happens to be non-incresing and thus, optimal. This explains why $\mathrm{T}^{\mathrm{RLX}}$ is significantly lower than $\mathrm{T}^{\mathrm{ISO}}$ if $K \leq 4$. On the contrary, if K is higher or equal to five, relaxing the monotonicity constraint leads to non-monotone solutions in order to adapt as much as possible to the original function, which is also non-monotone. In such cases, the time required to compute the lower bound through the relaxed shortest path problem is significantly higher than the time savings originated by such lower bound and consequently, $\mathrm{T}^{\mathrm{ISO}}$ is lower than $\mathrm{T}^{\mathrm{RLX}}$ for $K \geq 5$.

5.2. Realistic application: Estimating the bidding curve of a pool of flexible consumers

Here we consider the problem of estimating the price-response of a cluster of flexible consumers of electricity, that is, how much energy the cluster consumes as a function of the electricity price. Similar instances of this problem has been considered, for example, in Aswani et al. [2], Saez-Gallego and Morales [21], Saez-Gallego et al. [22]. In our particular case, these consumers are located within a distribution grid that interacts with the transmission system and the wholesale energy market. The distribution grid receives a nodal price at the main substation, to which the consumers react according to their energy needs, generation assets, and sensitivity to the electricity cost. The aggregate amount of energy demanded by the pool, paired with the nodal price (at the main substation) that induced such a demand, constitutes an observation and form a data point on the plane. The collection of the 2400 observations at our disposal are plotted in Figure 5. Since power consumption is typically affected by other factors besides the electricity price, similar price signals may yield quite different demand levels.

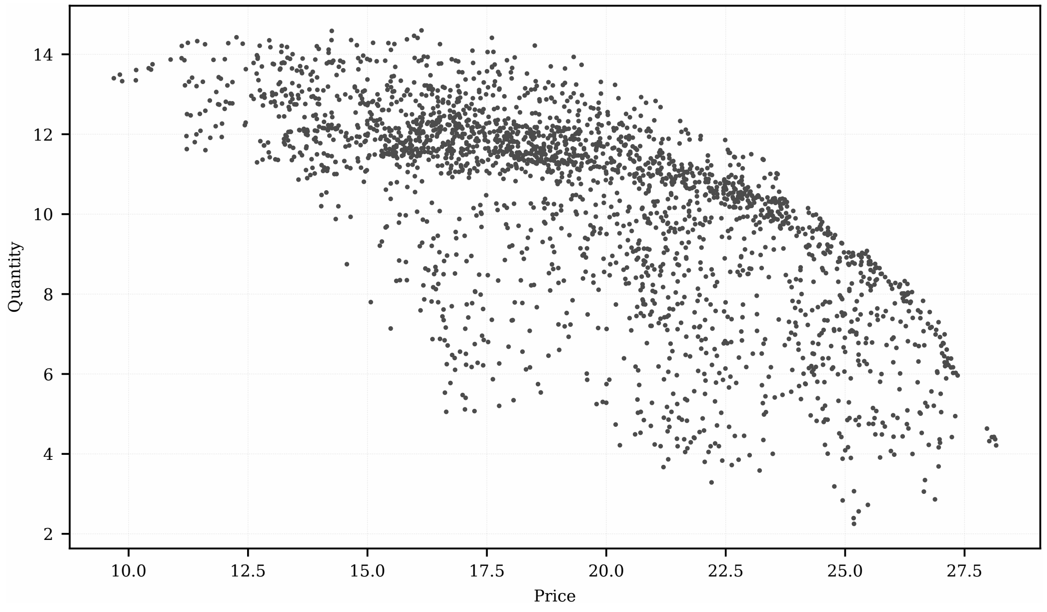


Figure 5: Price-consumption data from realistic application

The cumulative squared error of the curve fit provided by our algorithm for a different number of arcs (or steps) is compiled in Table 5. This table also shows the solution times $\mathrm{T}^{\mathrm{ISO}}$ and $\mathrm{T}^{\mathrm{RLX}}$ defined in the previous example, the initial upper bound and the lower bound obtained by relaxing the monotonicity constraint from which our algorithm starts to iterate. From these bounds, we can compute the optimality gap $\mathrm{GAP}_0 = \frac{UB-LB}{LB}100\%$ at the beginning of the algorithm, which we include in such a table too. We omit time $\mathrm{T}^{\mathrm{NOB}}$, as the raw algorithm is unable to deliver the optimal solution within a day in most cases, which proves the computational efficiency of the proposed acceleration strategies for our shortest path algorithm. As a matter of fact, the initial optimality gap that our algorithm needs to close is always below 0.25%, which reveals that the heuristic procedures we have devised to construct a (feasible) upper bound and a tight lower bound are remarkably good.

For those cases in which Algorithm 1 reaches the maximum time limit and thus, is terminated without having certified that the optimal curve fit has been found, we include, within parentheses, the optimality gap at termination. This optimality gap is calculated from the best upper and lower bounds on the optimal solution that are available after the time limit has expired. In the case of the experiment associated with time $\mathrm{T}^{\mathrm{ISO}}$, which only considers the lower bounds given by the isotonic fit, the best lower bound at termination is computed as follows:

1. Let i' be the layer being processed by the loop *for* in line 2 in the pseudocode of Algorithm 1 at termination. Consider each feasible path reaching layer $i'-1$ and the cost accrued by this path until that layer. Increase this cost by the error of the isotonic fit from layer $i'-1$ to the last one $I+1$. Denote the result as the *extended cost* of a feasible path at layer $i'-1$.
2. Compute the minimum extended cost for each layer $i \in \{1, 2, \ldots, i'-1\}$.
3. The best lower bound is then given by the maximum over layers $\{1, 2, \ldots, i'-1\}$ of their associated minimum extended cost.

Once again, the optimality gaps provided within parentheses confirm that the feasible solution we construct at the beginning of the algorithm, by modifying the isotonic fit through adjacency-constrained data clustering, is nearly optimal and that the lower bound given by relaxing the monotonicity constraint is hard to beat for this data set.

Results in Table 5 also show that the accrued fitting error decreases with the number of blocks, since the family $\mathcal{F}$ of non-increasing stepwise functions becomes larger as K is augmented. Nonetheless, the reduction in the fitting

K	Error	$\mathrm{T}^{\mathrm{ISO}}$(s)	$\mathrm{T}^{\mathrm{RLX}}$(s)	LB	UB	GAP_0(%)
1	14901	1.1	1.1	14901	14901	0
2	9201	2.5	2.2	9201	9201	0
3	8213	731.4	581.3	8213	8222	0.11
4	7726	30602	1077.2	7726	7732	0.08
5	7596	65995	1881.3	7596	7602	0.08
6	7502	>86400 (2.71%)	2477.3	7502	7519	0.23
7	7442	>86400 (1.88%)	2895.8	7442	7448	0.08
8	-	>86400 (1.31%)	>86400 (0.14%)	7392	7402	0.14

Table 5: Realistic application: Cumulative squared error, solution times, bounds and optimality gaps

error we get by increasing K rapidly plateaus after $K > 4$. On the contrary, the solutions times $\mathrm{T}^{\mathrm{ISO}}$ and $\mathrm{T}^{\mathrm{RLX}}$ feature a steady increase as K grows. This is consistent with the computational complexity of our algorithm, which depends linearly on K. Interestingly, $\mathrm{T}^{\mathrm{RLX}}$ is substantially smaller than $\mathrm{T}^{\mathrm{ISO}}$ for $K < 8$. The reason for this is that the lower bound we compute by relaxing the monotonicity constraint of the fitted curve naturally produces, however, a fit that is non-increasing and thus, globally optimal. In contrast, when $K \geq 8$, such a lower bound does no longer coincide with the globally optimal solution and as a result, $\mathrm{T}^{\mathrm{RLX}}$ end up surpassing the time limit set to one day (which is also exceeded by $\mathrm{T}^{\mathrm{ISO}}$). To support this argument, Figure 6 shows the fitted curves provided by the monotonicity-relaxed lower bound for $K = 7$ and $K = 8$. Notice that, if $K = 7$, the fitted curve associated with this lower bound is non-increasing, which allows our algorithm to certificate that this curve is, in fact, the optimal one in around 2900 seconds (with essentially all that time devoted to computing such a lower bound, logically). In contrast, when $K = 8$, the curve delivered by the monotonicity-relaxed lower bound features a tiny step that destroys its otherwise non-increasing appearance. It is clear that this tiny step can only be attributed to the random nature of the data and not to the price-sensitivity of the pool of flexible consumers. Indeed, it is not reasonable to expect that a price variation lower than €0.1/MWh has such an impact on the consumption of the pool. In order to discard these implausible non-monotone stepwise functions from the family $\mathcal{F}$, our algorithm also includes the possibility to enforce a minimum arc length, that is, a minimum step size. Very conveniently, besides, this constraint helps reduce the solution time of our algorithm by pruning some paths in the graph that become thus infeasible and by increasing the chances that the monotonicity-relaxed lower bound corresponds to the optimal curve fitting. To illustrate the impact of the constraint on the minimum arc length on the computational performance of the proposed algorithm, we provide Figure 7, which shows the solution time for various step sizes and number K of arcs. It can be seen that, while the case $K = 8$ cannot be solved to optimality within a day time, if no constraint on the minimum arc length is enforced, the solution time is drastically reduced below 3000 seconds when a (very small) minimum block size of €0.5/MWh is imposed.

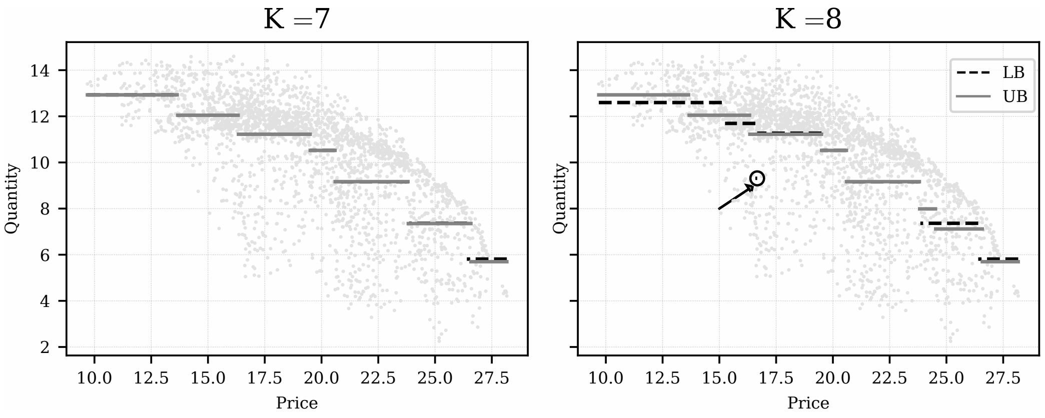


Figure 6: Lower and upper bound solutions for $K = 7$ and $K = 8$. Notice the tiny step that appears in the fit provided by the lower bound for $K = 8$

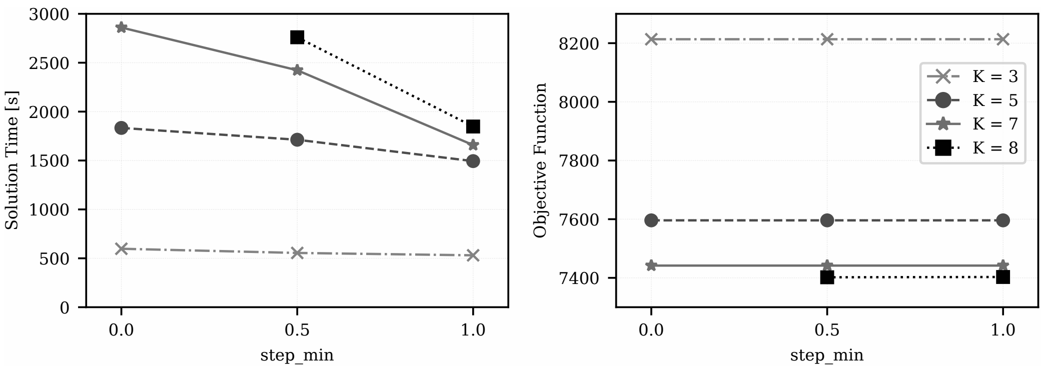


Figure 7: Objective function and solution times for different values of K and `step_min`.

6. Conclusions

In this paper, we have developed an algorithm to compute the curve that best fits to a certain cloud of data points in the sense of the least square error, under the conditions that said curve must be monotone and stepwise with a maximum number of steps. The proposed algorithm has been shown to run in polynomial time and is based on the finding that the curve-fitting problem can be addressed as a shortest-path type of problem. We have also proposed several strategies to cut down the execution time of the algorithm, all of which are based on computing upper and lower bounds that reduce the number of paths that the algorithm needs to explore. More specifically, the upper bound is given by a feasible solution that is swiftly built by combining the isotonic fit with clustering. The relaxation of either the constraint on the maximum number of steps or the monotonicity condition provides two different lower bounds, with the former being computationally much cheaper than the latter and not always necessarily looser. Our algorithm also allows for setting a minimum step length. This constraint notoriously speeds up the algorithm by pruning infeasible paths, while avoiding curve fits with implausible spurious tiny steps.

Through a series of numerical experiments built on both synthetic and realistic data sets, we have demonstrated that our algorithm, in conjunction with the proposed acceleration strategies, is robust to the level of noise in the data and able to certificate the globally optimal curve in less than a few hours for sample sizes in the order of the thousands of data points. Furthermore, through a data set comprising power-price measurements at the main substation of a distribution power grid, we have shown that our algorithm serves as an useful tool to estimate the bidding curve whereby the distributed energy sources in the grid can trade in wholesale energy markets. The extension of our algorithm to a multivariate setup is clearly an avenue of potentially fruitful research.

Acknowledgments

This research has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 755705). This work was also supported in part by the Spanish Ministry of Economy, Industry and Competitiveness and the European Regional Development Fund (ERDF) through project

ENE2017-83775-P. Martine Labbé has been partially supported by the Fonds de la Recherche Scientifique - FNRS under Grant(s) no PDR T0098.18.

CRediT author statement

Víctor Bucarey: Conceptualization, Methodology, Formal Analysis, Validation, Writing - Original Draft, Software. **Martine Labbé**: Conceptualization, Methodology, Formal Analysis, Writing - Original Draft. **Juan Miguel Morales**: Conceptualization, Methodology, Formal Analysis, Validation, Writing - Original Draft, Supervision. **Salvador Pineda**: Methodology, Formal Analysis, Validation, Software, Data curation.

[1] Ahuja, R. K. and Orlin, J. B. (2001). Inverse optimization. *Operations Research*, 49(5):771–783.

[2] Aswani, A., Shen, Z.-J., and Siddiq, A. (2018). Inverse optimization with noisy data. *Operations Research*, 66(3):870–892.

[3] Best, M. J. and Chakravarti, N. (1990). Active set algorithms for isotonic regression; a unifying framework. *Mathematical Programming*, 47(1-3):425–439.

[4] Chan, T. C., Lee, T., and Terekhov, D. (2019). Inverse optimization: Closed-form solutions, geometry, and goodness of fit. *Management Science*, 65(3):1115–1135.

[5] Dahl, G. and Realfsen, B. (2000). The cardinality-constrained shortest path problem in 2-graphs. *Networks: An International Journal*, 36(1):1–8.

[6] Esfahani, P. M., Shafieezadeh-Abadeh, S., Hanasusanto, G. A., and Kuhn, D. (2018). Data-driven inverse optimization with imperfect information. *Mathematical Programming*, 167(1):191–234.

[7] Feillet, D., Dejax, P., Gendreau, M., and Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks*, 44(3):216–229.

[8] Fielding, A. (1974). Statistical inference under order restrictions. the theory and application of isotonic regression. *Journal of the Royal Statistical Society: Series A (General)*, 137(1):92–93.

[9] Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67.

[10] Guo, D. (2009). Greedy optimization for contiguity-constrained hierarchical clustering. In *2009 IEEE International Conference on Data Mining Workshops*, pages 591–596. IEEE.

[11] Guyader, A., Jégou, N., Németh, A. B., and Németh, S. Z. (2014). A geometrical approach to iterative isotone regression. *Applied Mathematics and Computation*, 227:359–369.

[12] Hawkins, D. M. (1976). Point estimation of the parameters of piecewise regression models. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 25(1):51–57.

[13] Keshavarz, A., Wang, Y., and Boyd, S. (2011). Imputing a convex objective function. In *2011 IEEE International Symposium on Intelligent Control*, pages 613–619. IEEE.

[14] Lerman, P. (1980). Fitting segmented regression models by grid search. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 29(1):77–84.

[15] Mair, P., Hornik, K., and de Leeuw, J. (2009). Isotone optimization in r: pool-adjacent-violators algorithm (PAVA) and active set methods. *Journal of statistical software*, 32(5):1–24.

[16] Muggeo, V. M. (2003). Estimating regression models with unknown break-points. *Statistics in medicine*, 22(19):3055–3071.

[17] OASYS (2020). https://github.com/groupoasys/segisoreg. *GitHub repository.*

[18] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[19] Pineda, S. and Morales, J. M. (2018). Chronological time-period clustering for optimal capacity expansion planning with storage. *IEEE Transactions on Power Systems*, 33(6):7162–7170.

[20] Rote, G. (2018). Isotonic regression by dynamic programming. In *2nd Symposium on Simplicity in Algorithms (SOSA 2019)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[21] Saez-Gallego, J. and Morales, J. M. (2017). Short-term forecasting of price-responsive loads using inverse optimization. *IEEE Transactions on Smart Grid*, 9(5):4805–4814.

[22] Saez-Gallego, J., Morales, J. M., Zugno, M., and Madsen, H. (2016). A data-driven bidding model for a cluster of price-responsive consumers of electricity. *IEEE Transactions on Power Systems*, 31(6):5001–5011.

[23] Su, C.-L. and Kirschen, D. (2009). Quantifying the effect of demand response on electricity markets. *IEEE Transactions on Power Systems*, 24(3):1199–1207.

[24] Sysoev, O. and Burdakov, O. (2019). A smoothed monotonic regression via l2 regularization. *Knowledge and Information Systems*, 59(1):197–218.

[25] Tibshirani, R. J., Hoefling, H., and Tibshirani, R. (2011). Nearly-isotonic regression. *Technometrics*, 53(1):54–61.

[26] Wolsey, L. A. (1998). *Integer programming*, volume 52. John Wiley & Sons.