DCA-based algorithms for DC Fitting

Vinh Thanh Ho, Hoai An Le Thi, Tao Pham Dinh

Version of Record: https://www.sciencedirect.com/science/article/pii/S0377042720306440
Manuscript_6f7bd57cbfdf4d6b7c18f884e2ef34dc

DCA-based algorithms for DC Fitting

Vinh Thanh Ho[a,b], Hoai An Le Thi[c,*], Tao Pham Dinh[d]

[a] *Department for Management of Science and Technology Development, Ton Duc Thang University, Ho Chi Minh City, Vietnam*
[b] *Faculty of Mathematics and Statistics, Ton Duc Thang University, Ho Chi Minh City, Vietnam*
[c] *Université de Lorraine, LGIPM, F-57000 Metz, France*
[d] *Laboratory of Mathematics, INSA - Rouen, University of Normandie, 76801 Saint-Etienne-du-Rouvray Cedex, France*

Abstract

We investigate a nonconvex, nonsmooth optimization approach based on DC (Difference of Convex functions) programming and DCA (DC Algorithm) for the so-called DC fitting problem, which aims to fit a given set of data points by a DC function. The problem is tackled as minimizing the squared Euclidean norm fitting error. It is formulated as a DC program for which a standard DCA scheme is developed. Furthermore, a modified DCA scheme with successive DC decomposition is proposed. These standard/modified versions of DCA are applied for solving the continuous piecewise-linear fitting problem. Numerical experiments on many synthetic and real datasets with small-to-large sizes show the efficiency of our DCA-based approach in comparison with the existing approaches for constructing continuous piecewise-linear models.

Keywords: DC programming, DCA, DCA with successive DC decomposition, DC fitting, Piecewise-linear fitting

*Corresponding author
Email addresses: hovinhthanh@tdt.edu.vn (Vinh Thanh Ho), hoai-an.le-thi@univ-lorraine.fr (Hoai An Le Thi), pham@insa-rouen.fr (Tao Pham Dinh)

1. Introduction

The fundamental problem of fitting a certain mathematical function to a given set of data points has been studied extensively. The fitting problem often involves, e.g., smoothing to remove noise from datasets but preserve important patterns, or interpolation/extrapolation to construct new data points within/beyond the range of the given datasets. The obtained fitting models can be used for data visualization, for estimating the relationships among variables, and for predicting values of a function at unavailable data points.

The related works considered many fitting functions: for example, linear/polynomial functions, convex/nonconvex continuous piecewise-linear functions, exponential functions, logarithms, etc. In this paper, we take into account a DC (Difference of Convex functions) fitting function that covers most of all existing fitting functions. Here, a function F is called a DC function if it takes the form $F = G - H$ (with G, H being convex functions); and $G - H$ is called a DC decomposition of F, while G and H are DC components of F (see, e.g., [1, 2, 3]).

Let us introduce the so-called DC fitting problem that fits a given set of m points $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \mathbb{R}$ by a DC function $\phi : \mathbb{R}^p \to \mathbb{R}$ with the form

$$\phi(\boldsymbol{x}) := f(\boldsymbol{x}, \boldsymbol{\alpha}),$$

where, for two variables $\boldsymbol{x} \in \mathbb{R}^p$ and $\boldsymbol{\alpha} \in \mathbb{R}^n$, the function $f : \mathbb{R}^p \times \mathbb{R}^n \to \mathbb{R}$ is DC in each variable when the other variable is fixed. The set of such functions ϕ is denoted by $\mathcal{F}$.

With the least-squares fitting criterion, the objective of DC fitting problem is to find the parameter vector $\boldsymbol{\alpha} \in \mathbb{R}^n$ such that the mean-square error, defined by

$$\frac{1}{m}\sum_{i=1}^{m} \left[\phi(\boldsymbol{x}_i) - y_i\right]^2 = \frac{1}{m}\sum_{i=1}^{m} \left[f(\boldsymbol{x}_i, \boldsymbol{\alpha}) - y_i\right]^2, \tag{1}$$

is as small as possible. It can be expressed as the following squared ℓ_2-norm (or Euclidean norm) optimization formulation

$$\min\left\{ F(\boldsymbol{\alpha}) := \sum_{i=1}^{m} \left[f(\boldsymbol{x}_i, \boldsymbol{\alpha}) - y_i\right]^2 : \boldsymbol{\alpha} \in \mathbb{R}^n \right\}. \tag{2}$$

Note that the optimization problem (2) is, in general, nonconvex and nonsmooth. Thus, globally solving such a problem is very difficult in large-scale setting.

The backbone of our approach for solving the problem (2) is DC programming and DCA (DC Algorithm) (see, e.g., [1, 4, 2, 3, 5, 6, 7] and the references in [6]) which are well-known as powerful nonsmooth, nonconvex optimization tools. DCA aims to solve a standard DC program that consists in minimizing a DC function $F = G - H$ over a convex set or on the whole space. The idea of standard DCA is, at each iteration k, approximating the second DC component H by its affine minorant H_k and then solving the resulting convex subproblem. In other words, one approximates the DC function F by the convex majorization $F^k := G - H_k$. It is clear that the closer to F the function F^k is, the better DCA could be. The recent work [6] has introduced a modified version of the standard DCA scheme, named DCA with *successive DC decomposition*. In this version, the DC decomposition of F is successively updated during DCA iterations in order to better approximate the DC function F i.e. $F := G^k - H^k$ at each iteration k and its corresponding convex majorization is $F^k := G^k - H^k_k$. The "successive DC decomposition" idea has been used in the recent works [8, 9]. Le Thi et al. [8, 9] have investigated the standard/modified DCA scheme for minimizing the sum of squared convex (piecewise-linear) functions, which is a particular case of this work.

Contributions: We investigate DC programming and DCA for solving the general DC fitting problem (2). Particularly, the problem (2) is formulated as a DC program for which a standard DCA scheme is developed. A modified version of DCA with successive DC decomposition for the problem (2) is proposed to better approximate the DC objective function. Next, we apply the proposed DCA schemes for solving a class of *continuous piecewise-linear* fitting problems. We provide numerical experiments of our DCA-based algorithms on both synthetic and real datasets with small-to-large size of data points in comparison to the existing approaches for constructing the piecewise-linear models.

The rest of the paper is organized as follows. Section 2 shows how to apply DC programming and DCA for solving the squared ℓ_2-norm optimization formulation of DC fitting problem. In Section 3, we develop two DCA-based algorithms for solving a continuous piecewise-linear fitting problem. Section 4 reports the numerical results on several test problems. Finally, Section 5 concludes the paper.

2. Solution methods based on DC programming and DCA

DC programming and DCA were introduced by Pham Dinh Tao in a preliminary form in 1985 as a natural and logical extension of his previous works on convex maximization since 1974 (see e.g. [10, 11]), and have been extensively developed by Le Thi Hoai An and Pham Dinh Tao since 1994. DCA is well-known as an efficient approach in the nonconvex programming framework. In recent years, numerous DCA-based algorithms have successfully solved large-scale nonsmooth/nonconvex problems arising in several application areas (see, e.g., [5, 6, 8, 12, 13, 14, 15, 16, 17] and the list of references in [6]). For a comprehensive survey on thirty years of development of DCA, the reader is referred to the recent paper [6].

The original key idea of DCA relies on the DC structure of the objective function F. The standard DCA scheme is described as follows.

Standard DCA scheme

Initialization: Let $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$ be a best guess. Set $k = 0$.

repeat

1. Calculate $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$.
2. Calculate $\boldsymbol{\alpha}^{k+1} \in \operatorname{argmin}\{G(\boldsymbol{\alpha}) - H(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle : \boldsymbol{\alpha} \in \mathbb{R}^n\}$.
3. $k = k + 1$.

until convergence of $\{\boldsymbol{\alpha}^k\}_k$.

Convergence properties of the standard DCA are described completely in [5, 2, 3].

2.1. Standard DCA for solving the problem (2)

The squared ℓ_2-norm formulation (2) of DC fitting problem can be rewritten as

$$\min \left\{ F(\boldsymbol{\alpha}) = \sum_{i=1}^{m} [p_i(\boldsymbol{\alpha})]^2 : \boldsymbol{\alpha} \in \mathbb{R}^n \right\} \tag{3}$$

where for $i = 1, \ldots, m$, the function $p_i : \mathbb{R}^n \to \mathbb{R}$ is defined as

$$p_i(\boldsymbol{\alpha}) := f(\boldsymbol{x}_i, \boldsymbol{\alpha}) - y_i.$$

From the definition of $\mathcal{F}$, we assume that $f(\boldsymbol{x}_i, \boldsymbol{\alpha})$ is a DC function, for $i = 1, \ldots, m$, with DC decomposition

$$f(\boldsymbol{x}_i, \boldsymbol{\alpha}) = g_i(\boldsymbol{\alpha}) - h_i(\boldsymbol{\alpha}),$$

where g_i and h_i are continuous convex functions. Thus, p_i is also a DC function with DC components

$$p_{i1}(\boldsymbol{\alpha}) = g_i(\boldsymbol{\alpha}) - y_i \text{ and } p_{i2}(\boldsymbol{\alpha}) = h_i(\boldsymbol{\alpha}).$$

Knowing that (see, e.g., [2]) if the function p_i is DC with a nonnegative DC decomposition then $[p_i]^2$ is DC too, we will exploit the convexity of p_{ij} ($j \in \{1, 2\}$) to highlight a nonnegative DC decomposition of p_i. In particular, we define the affine minorization of p_{i1} and p_{i2} ($i = 1, \ldots, m$) at an arbitrary point $\overline{\boldsymbol{\alpha}} \in \mathbb{R}^n$ as follows:

$$l_{ij}(\boldsymbol{\alpha}) := p_{ij}(\overline{\boldsymbol{\alpha}}) + \langle \boldsymbol{\alpha} - \overline{\boldsymbol{\alpha}}, \overline{\boldsymbol{\gamma}}_{ij} \rangle \text{ with } \overline{\boldsymbol{\gamma}}_{ij} \in \partial p_{ij}(\overline{\boldsymbol{\alpha}}),\ j \in \{1, 2\}. \tag{4}$$

Let us define

$$l_i := \max\{-l_{i1}, -l_{i2}\}.$$

Then the functions $p_{i1} + l_i$ and $p_{i2} + l_i$ are nonnegative and convex on $\mathbb{R}^n$, and so are $[p_{i1} + l_i]^2$ and $[p_{i2} + l_i]^2$.

We consider the following nonnegative DC decomposition of p_i:

$$p_i = [p_{i1} + l_i] - [p_{i2} + l_i].$$

Applying the equality $(a - b)^2 = 2(a^2 + b^2) - (a + b)^2$ for $a = p_{i1} + l_i$ and $p_{i2} + l_i$, we get a DC decomposition of $[p_i]^2$, that is,

$$[p_i]^2 = 2\left\{[p_{i1} + l_i]^2 + [p_{i2} + l_i]^2\right\} - [p_{i1} + p_{i2} + 2l_i]^2.$$

As a result, F is a DC function. We thus derive the DC formulation of (3) as follows:

$$\min\{F(\boldsymbol{\alpha}) := G(\boldsymbol{\alpha}) - H(\boldsymbol{\alpha}) : \boldsymbol{\alpha} \in \mathbb{R}^n\} \tag{5}$$

where

$$G = \sum_{i=1}^{m} 2\left\{[p_{i1} + l_i]^2 + [p_{i2} + l_i]^2\right\},\ H = \sum_{i=1}^{m} [p_{i1} + p_{i2} + 2l_i]^2.$$

Applying the standard DCA scheme to (5) leads us, at the iteration k, to compute a subgradient $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$ and find an optimal solution $\boldsymbol{\alpha}^{k+1}$ to the following convex subproblem

$$\min\left\{\sum_{i=1}^{m} 2\left\{[p_{i1}(\boldsymbol{\alpha}) + l_i(\boldsymbol{\alpha})]^2 + [p_{i2}(\boldsymbol{\alpha}) + l_i(\boldsymbol{\alpha})]^2\right\} - \langle \boldsymbol{\beta}^k, \boldsymbol{\alpha} \rangle : \boldsymbol{\alpha} \in \mathbb{R}^n\right\}. \tag{6}$$

Standard DCA for solving (5) **(DCA)**

Initialization: Let $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$. Set $k = 0$.

repeat

1. Compute $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$.
2. Solve the convex program (6) to obtain $\boldsymbol{\alpha}^{k+1}$.
3. $k = k + 1$.

until Stopping criteria are satisfied.

Finally, standard DCA applied to (5) is summarized below.

As direct consequences of the convergence properties of standard DC programs (see, e.g., [5, 2]), we have the convergence properties of DCA in Theorem 1.

Theorem 1. *Convergence properties of* DCA
i) DCA *generates the sequence* $\{\boldsymbol{\alpha}^k\}$ *such that the sequence* $\{F(\boldsymbol{\alpha}^k)\}$ *is non-increasing.*
ii) If the sequences $\{\boldsymbol{\alpha}^k\}$ *and* $\{\boldsymbol{\beta}^k\}$ *are bounded, then every limit point* $\boldsymbol{\alpha}^*$ *of the sequence* $\{\boldsymbol{\alpha}^k\}$ *is a critical point of* $G - H$ *i.e.* $\partial G(\boldsymbol{\alpha}^*) \cap \partial H(\boldsymbol{\alpha}^*) \neq \emptyset$.

2.2. DCA with successive DC decomposition for solving the problem (2)

We see from (4) that for $i = 1, \ldots, m$, $j \in \{1, 2\}$, the affine minorization l_{ij} of p_{ij} can be computed easily with any point $\overline{\boldsymbol{\alpha}}$. From the fact that the closer to p_{ij} the function l_{ij} is, the better DCA could be, we suggest a modified DCA with successive DC decomposition for the problem (2), i.e. l_{ij} is updated during DCA iterations by choosing $\overline{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^k$. Particularly, at the iteration k, we set

$$\boldsymbol{\gamma}_{ij}^k \in \partial p_{ij}(\boldsymbol{\alpha}^k), \quad l_{ij}^k(\boldsymbol{\alpha}) := p_{ij}(\boldsymbol{\alpha}^k) + \langle \boldsymbol{\alpha} - \boldsymbol{\alpha}^k, \boldsymbol{\gamma}_{ij}^k \rangle.$$

Let us define $l_i^k := \max\{-l_{i1}^k, -l_{i2}^k\}$. The resulting DC formulation of (2) takes the form

$$\min\{F(\boldsymbol{\alpha}) := G^k(\boldsymbol{\alpha}) - H^k(\boldsymbol{\alpha}) : \boldsymbol{\alpha} \in \mathbb{R}^n\} \tag{7}$$

where

$$G^k = \sum_{i=1}^{m} 2\left\{ \left[p_{i1} + l_i^k\right]^2 + \left[p_{i2} + l_i^k\right]^2 \right\}, \; H^k = \sum_{i=1}^{m} \left[p_{i1} + p_{i2} + 2l_i^k\right]^2.$$

Similarly, DCA with successive DC decomposition applied to (7), named DCA^k, consists in computing two sequences $\{\boldsymbol{\beta}^k\}$ and $\{\boldsymbol{\alpha}^k\}$ such that $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ and $\boldsymbol{\alpha}^{k+1}$ is an optimal solution to the following convex subproblem

$$\min\left\{\sum_{i=1}^{m} 2\left\{\left[p_{i1}(\boldsymbol{\alpha})+l_i^k(\boldsymbol{\alpha})\right]^2+\left[p_{i2}(\boldsymbol{\alpha})+l_i^k(\boldsymbol{\alpha})\right]^2\right\}-\langle\boldsymbol{\beta}^k,\boldsymbol{\alpha}\rangle : \boldsymbol{\alpha}\in\mathbb{R}^n\right\}. \tag{8}$$

DCA with successive DC decomposition for solving (7) (DCA^k)

Initialization: Let $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$. Set $k = 0$.

repeat

1. Compute $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$.
2. Solve the convex program (8) to obtain $\boldsymbol{\alpha}^{k+1}$.
3. $k = k + 1$.

until Stopping criteria are satisfied.

Especially, the convergence properties of DCA is still valid for DCA^k as stated in Theorem 2.

Theorem 2. *Convergence properties of* DCA^k
i) DCA^k *generates the sequence* $\{\boldsymbol{\alpha}^k\}$ *such that the sequence* $\{F(\boldsymbol{\alpha}^k)\}$ *is non-increasing.*
ii) If the sequences $\{\boldsymbol{\alpha}^k\}$ *and* $\{\boldsymbol{\beta}^k\}$ *are bounded, then every limit point* $\boldsymbol{\alpha}^*$ *of the sequence* $\{\boldsymbol{\alpha}^k\}$ *is a critical point of* $F = G^\infty - H^\infty$ *where*

$$G^\infty = \sum_{i=1}^{m} 2\left\{[p_{i1}+l_i^\infty]^2+[p_{i2}+l_i^\infty]^2\right\},$$
$$H^\infty = \sum_{i=1}^{m} [p_{i1}+p_{i2}+2l_i^\infty]^2,$$

and for $i = 1,\ldots,m$, $j \in \{1,2\}$, l_{ij}^∞ *is the affine minorization of* p_{ij} *at* $\boldsymbol{\alpha}^*$, $l_i^\infty := \max\{-l_{i1}^\infty, -l_{i2}^\infty\}$.

Proof. i) DCA^k consists in constructing two sequences $\{\boldsymbol{\alpha}^k\}$ and $\{\boldsymbol{\beta}^k\}$ such that $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ and $\boldsymbol{\alpha}^{k+1} \in \operatorname{argmin}\{G^k(\boldsymbol{\alpha}) - \langle\boldsymbol{\alpha},\boldsymbol{\beta}^k\rangle\}$, which is equivalent to $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ and $\boldsymbol{\alpha}^{k+1} \in \partial(G^k)^*(\boldsymbol{\beta}^k)$, where the conjugate of G, denoted G^*, is defined by $G^*(\boldsymbol{w}) = \sup\{\langle\boldsymbol{\alpha},\boldsymbol{w}\rangle - G(\boldsymbol{\alpha}) : \boldsymbol{\alpha}\in\mathbb{R}^n\}$.

The first inclusion $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ implies that

$$H^k(\boldsymbol{\alpha}^{k+1}) \geq H^k(\boldsymbol{\alpha}^k) + \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle.$$

Consequently,

$$G^k(\boldsymbol{\alpha}^{k+1}) - H^k(\boldsymbol{\alpha}^{k+1}) \leq G^k(\boldsymbol{\alpha}^{k+1}) - H^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle. \tag{9}$$

Similarly, we derive from the second inclusion $\boldsymbol{\alpha}^{k+1} \in \partial (G^k)^*(\boldsymbol{\beta}^k)$ that

$$G^k(\boldsymbol{\alpha}^{k+1}) - H^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle \leq (G^k - H^k)(\boldsymbol{\alpha}^k). \tag{10}$$

As a result, we have $F(\boldsymbol{\alpha}^{k+1}) = (G^k - H^k)(\boldsymbol{\alpha}^{k+1}) \leq (G^k - H^k)(\boldsymbol{\alpha}^k) = F(\boldsymbol{\alpha}^k)$. That is, the sequence $\left\{F(\boldsymbol{\alpha}^k)\right\}$ is non-increasing.

ii) For the sake of simplicity, we write $\lim_{k \to +\infty} \boldsymbol{\alpha}^k = \boldsymbol{\alpha}^*$. Since the sequence $\{\boldsymbol{\beta}^k\}$ is bounded, we imply that for $i = 1, \ldots, m$ and $j = 1, 2$, the sequence $\{\boldsymbol{\gamma}_{ij}^k\}$ is bounded too. We can suppose (by extracting a subsequence if necessary) that $\{\boldsymbol{\beta}^k\}_k$ (resp. $\{\boldsymbol{\gamma}_{ij}^k\}$) converges to $\boldsymbol{\beta}^*$ (resp. $\boldsymbol{\gamma}_{ij}^*$).

From Theorem 24.4 in [18] and $\boldsymbol{\gamma}_{ij}^k \in \partial p_{ij}(\boldsymbol{\alpha}^k)$, we get $\boldsymbol{\gamma}_{ij}^* \in \partial p_{ij}(\boldsymbol{\alpha}^*)$. Let us define the affine minorization l_{ij}^∞ of p_{ij} at $\boldsymbol{\alpha}^*$ as

$$l_{ij}^\infty(\boldsymbol{\alpha}) = p_{ij}(\boldsymbol{\alpha}^*) + \langle \boldsymbol{\alpha} - \boldsymbol{\alpha}^*, \boldsymbol{\gamma}_{ij}^* \rangle.$$

From the definition of l_i^k ($k \geq 1$, $i = 1, \ldots, m$) and the continuity of p_{ij} ($i = 1, \ldots, m$, $j = 1, 2$), we derive that

$$\lim_{k \to +\infty} l_i^k(\boldsymbol{\alpha}^k) = \lim_{k \to +\infty} p_{ij}(\boldsymbol{\alpha}^k) = p_{ij}(\boldsymbol{\alpha}^*) = l_i^\infty(\boldsymbol{\alpha}^*),$$

and for each $\boldsymbol{\alpha}$,

$$\lim_{k \to +\infty} l_i^k(\boldsymbol{\alpha}) = \lim_{k \to +\infty} \max_{j=1,2}(-l_{ij}^k(\boldsymbol{\alpha})) = \max_{j=1,2}(-l_{ij}^\infty(\boldsymbol{\alpha})) = l_i^\infty(\boldsymbol{\alpha}).$$

Similarly, we also have:

$$\lim_{k \to +\infty} G^k(\boldsymbol{\alpha}^k) = G^\infty(\boldsymbol{\alpha}^*), \lim_{k \to +\infty} H^k(\boldsymbol{\alpha}^k) = H^\infty(\boldsymbol{\alpha}^*).$$

From the definition of both convex functions G^∞ and H^∞, we see that $F = G^\infty - H^\infty$ i.e. $G^\infty - H^\infty$ is a DC decomposition of F.

We derive from Lemma 2 in [2] and the convergence properties of a series of conjugate convex functions (see e.g. [19, 20]) that

$$\lim_{k \to +\infty} (G^k)^*(\boldsymbol{\beta}^k) = (G^\infty)^*(\boldsymbol{\beta}^*), \lim_{k \to +\infty} (H^k)^*(\boldsymbol{\beta}^k) = (H^\infty)^*(\boldsymbol{\beta}^*).$$

Now, we will show that $\boldsymbol{\alpha}^*$ is a critical point of $G^\infty - H^\infty$. From the inclusions $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ and $\boldsymbol{\alpha}^{k+1} \in \partial (G^k)^*(\boldsymbol{\beta}^k)$, we get

$$\langle \boldsymbol{\alpha}^{k+1}, \boldsymbol{\beta}^k \rangle = G^k(\boldsymbol{\alpha}^{k+1}) + (G^k)^*(\boldsymbol{\beta}^k) \text{ and } \langle \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle = H^k(\boldsymbol{\alpha}^k) + (H^k)^*(\boldsymbol{\beta}^k). \tag{11}$$

By taking the limit of the second equality of (11), we get

$$\langle \boldsymbol{\alpha}^*, \boldsymbol{\beta}^* \rangle = H^\infty(\boldsymbol{\alpha}^*) + (H^\infty)^*(\boldsymbol{\beta}^*).$$

Therefore, we yield $\boldsymbol{\beta}^* \in \partial H^\infty(\boldsymbol{\alpha}^*)$.

To indicate the inclusion $\boldsymbol{\beta}^* \in \partial G^\infty(\boldsymbol{\alpha}^*)$, it suffices to show that

$$\lim_{k \to +\infty} \{(G^k)^*(\boldsymbol{\beta}^k) + G^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle\} = 0. \tag{12}$$

It results from *i)* and $F(\boldsymbol{\alpha}) \geq 0$ for all $\boldsymbol{\alpha} \in \mathbb{R}^n$ that the sequence $\{(G^k - H^k)(\boldsymbol{\alpha}^k)\}$ is non-increasing and converges to the limit $\delta \geq 0$.

We imply from (11) that

$$(H^k)^*(\boldsymbol{\beta}^k) - (G^k)^*(\boldsymbol{\beta}^k) = G^k(\boldsymbol{\alpha}^{k+1}) - H^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle. \tag{13}$$

Combining (13) with (9) and (10), we see that the sequence $\{[(H^k)^* - (G^k)^*](\boldsymbol{\beta}^k)\}$ is non-increasing and also converges to the limit $\delta \geq 0$.

Taking (13) and the same limit of two above sequences into account, we have

$$\lim_{k \to +\infty} \{G^k(\boldsymbol{\alpha}^{k+1}) - H^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle\} = \lim_{k \to +\infty} \{(G^k)(\boldsymbol{\alpha}^k) - (H^k)(\boldsymbol{\alpha}^k)\}.$$

Thus,

$$\lim_{k \to +\infty} \{G^k(\boldsymbol{\alpha}^{k+1}) - G^k(\boldsymbol{\alpha}^k) - \langle \boldsymbol{\alpha}^{k+1} - \boldsymbol{\alpha}^k, \boldsymbol{\beta}^k \rangle\} = 0.$$

From (11), we obtain (12). The proof of *ii)* is complete. □

3. DCA and DCA^k for continuous piecewise-linear fitting problem

The piecewise-linear fitting problem is finding a piecewise-linear function that furnishes the smallest mean-square fitting error (1) to a given set of data points. The importance of using a piecewise-linear function has been emphasized in many real-world applications due to a number of its merits: they are compact, easy to compute, visualize, interpret, and able to approximate even highly complex nonlinear functions (e.g., concave/convex function) [21].

Its application areas include from mathematical modeling/optimization (see, e.g., [21, 22, 23, 24, 25, 26, 27, 28]) to econometrics, transportation, prediction and forecasting (see, e.g., [29, 30, 31, 32, 33]), and to statistics, machine learning and data mining (see, e.g., [34, 21, 35, 36, 37, 38] and references therein), etc. In machine learning, developing regression techniques using the piecewise-linear function plays an important role for practical problems such as energy storage optimization with a solar source, beer brewery optimization, customer demand forecasting (see, e.g., [34, 35, 37] for more practical problems). Previous and relevant works for piecewise-linear fitting functions are given completely in several works (see, e.g., [34, 21, 24, 26, 27, 35, 37]).

Piecewise-linear fitting problems can be classified into two categories: discontinuous and continuous ones. As for the discontinuous category, the problem focuses on constructing the general piecewise-linear model whose each affine submodel tries to fit the data points on the corresponding continuous subdomain. In related works (see, e.g., [21, 26]), the problem can be divided into two sequential subproblems. The first is a clustering problem for partitioning the data points and building the affine fitting submodels while the second is a classification problem for partitioning the domain with respect to the obtained submodels and the corresponding point partition. The recent work [21] considered the problem of minimizing the mean-absolute error with a piecewise-linearly separable domain partition. The authors proposed a mix-integer linear program formulation with big-M for small-scale instances and then developed a four-step heuristic algorithm for solving it in medium-to-large-scale datasets.

Works in the class of continuous piecewise-linear fitting problems can be found in, e.g., [27, 39, 40]. The work [40] proposed the well-known Multivariate Adaptive Regression Splines (MARS) method which consists in constructing the continuous fitting model in the form of an expansion in a set of basis functions (see, e.g., [34, 40, 41, 42]). There are three forms of basis functions: a constant, a hinge function, and the multiplication of a number of hinge functions. Jekabsons [42] developed an open source Matlab/Octave toolbox for building piecewise-linear and piecewise-cubic regression models using the MARS method. Another approach for solving this problem is based on discrete optimization and mixed-integer programming. The authors in [27] introduced mixed-binary models under various conditions with the additional convexity-enforcing constraints. A drawback of this approach is the scalability of the obtained models. In addition, the work [39] considered the continuous piecewise-linear fitting problem with the MSE criterion over the

class of maxima of minima of linear functions. In general, this problem is nonsmooth, nonconvex. Exploiting the semismooth, quasidifferentiable and piecewise partially separable properties of the objective function, the authors described a technique to approximate its subdifferential and then applied the Discrete Gradient Method (DGM) for directly solving the minimization problem. This method is impractical for large datasets. In the same direction but with the use of another method based on DC optimization, Bagirov et al. [43] indicated that the objective function is, in general, a DC function without highlighting a DC decomposition. The authors designed a descent algorithm with a line search to find critical points. It computes search directions by calculating the distance between two polytopes approximating subdifferentials of DC components (without knowing the explicit form of these DC components). However, the computation of these subdifferentials stated in Proposition 2 of [43] is not correct. A counterexample is given in Appendix A.

In this section, we consider the class of continuous piecewise-linear fitting problems. We show that these fitting problems can be formulated as the DC fitting problem (2) for which two proposed algorithms DCA and DCA^k in Section 2 are applied.

3.1. Problem formulation

It is well-known from [44] that any continuous piecewise-linear function can be represented as a DC function:

$$\phi(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{\alpha}) = \max_{j=1,\ldots,K}\{\langle \boldsymbol{x}, \boldsymbol{a}_j\rangle + b_j\} - \max_{q=1,\ldots,M}\{\langle \boldsymbol{x}, \boldsymbol{a}_{K+q}\rangle + b_{K+q}\},$$

where the parameters $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{K+M} \in \mathbb{R}^p$, $b_1, \ldots, b_{K+M} \in \mathbb{R}$, $n = (K + M)(p + 1)$, and $\boldsymbol{\alpha} = (\boldsymbol{a}_1, b_1, \ldots, \boldsymbol{a}_K, b_K, \boldsymbol{a}_{K+1}, b_{K+1}, \ldots, \boldsymbol{a}_{K+M}, b_{K+M}) \in \mathbb{R}^n$. Obviously, $\phi \in \mathcal{F}$. Thus, the continuous piecewise-linear fitting problem can be rewritten as a DC fitting problem (2)

$$\min\left\{F(\boldsymbol{\alpha}) = \sum_{i=1}^{m}\left[(g_i(\boldsymbol{\alpha}) - y_i) - h_i(\boldsymbol{\alpha})\right]^2 : \boldsymbol{\alpha} \in \mathbb{R}^n\right\}, \tag{14}$$

where

$$g_i(\boldsymbol{\alpha}) := \max_{j=1,\ldots,K}\langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)}\rangle,\ h_i(\boldsymbol{\alpha}) := \max_{q=1,\ldots,M}\langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)}\rangle,$$

and for $i = 1, \ldots, m$, $j = 1, \ldots, K$, $q = 1, \ldots, M$, the vectors

$$\boldsymbol{u}^{(i,j)} = (\boldsymbol{u}_1^{(i,j)}, \ldots, \boldsymbol{u}_K^{(i,j)}, \boldsymbol{u}_{K+1}^{(i,j)}, \ldots, \boldsymbol{u}_{K+M}^{(i,j)}) \in \mathbb{R}^n,$$
$$\boldsymbol{v}^{(i,q)} = (\boldsymbol{v}_1^{(i,q)}, \ldots, \boldsymbol{v}_K^{(i,q)}, \boldsymbol{v}_{K+1}^{(i,q)}, \ldots, \boldsymbol{v}_{K+M}^{(i,q)}) \in \mathbb{R}^n,$$

with $\boldsymbol{u}_r^{(i,j)} = (\boldsymbol{x}_i, 1)$ if $r = j$, $(\boldsymbol{0}, 0)$ if $r \neq j$; $\boldsymbol{v}_r^{(i,q)} = (\boldsymbol{x}_i, 1)$ if $r = K + q$, $(\boldsymbol{0}, 0)$ if $r \neq K + q$.

Next, we apply DCA and DCA^k for solving the continuous piecewise-linear fitting problem (14). In particular, we first show how to compute a subgradient, then solve the convex subproblem, and finally derive the resulting algorithms, named DCA-PL and DCA^k-PL respectively.

3.2. DCA for solving the problem (14)

Compute a subgradient $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$:

By the definition of H in (5), the subdifferential of H is defined as

$$\partial H = \sum_{i=1}^{m} \partial [p_{i1} + p_{i2} + 2l_i]^2.$$

Applying the inclusion $2r_i(\boldsymbol{\alpha})\partial r_i(\boldsymbol{\alpha}) \subset \partial [r_i(\boldsymbol{\alpha})]^2$ for $r_i = p_{i1} + p_{i2} + 2l_i$, and summing this inclusion over $i = 1, ..., m$, we have

$$\sum_{i=1}^{m} 2[p_{i1}(\boldsymbol{\alpha}) + p_{i2}(\boldsymbol{\alpha}) + 2l_i(\boldsymbol{\alpha})][\partial p_{i1}(\boldsymbol{\alpha}) + \partial p_{i2}(\boldsymbol{\alpha}) + \partial(2l_i)(\boldsymbol{\alpha})] \subset \partial H(\boldsymbol{\alpha}).$$

Thus, we can choose a subgradient $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$ as follows:

$$\boldsymbol{\beta}^k \in \sum_{i=1}^{m} 2[p_{i1}(\boldsymbol{\alpha}^k) + p_{i2}(\boldsymbol{\alpha}^k) + 2l_i(\boldsymbol{\alpha}^k)][\partial p_{i1}(\boldsymbol{\alpha}^k) + \partial p_{i2}(\boldsymbol{\alpha}^k) + \partial(2l_i)(\boldsymbol{\alpha}^k)].$$

Now, we need to compute subgradients $\overline{\boldsymbol{\gamma}}_{i1} \in \partial p_{i1}(\overline{\boldsymbol{\alpha}})$, $\overline{\boldsymbol{\gamma}}_{i2} \in \partial p_{i2}(\overline{\boldsymbol{\alpha}})$, $\boldsymbol{\gamma}_{i1}^k \in \partial p_{i1}(\boldsymbol{\alpha}^k)$, $\boldsymbol{\gamma}_{i2}^k \in \partial p_{i2}(\boldsymbol{\alpha}^k)$. In particular, by the definition of p_{i1}, p_{i2}, l_i and according to the rule of computing the subdifferential of a function

being the maximum of a finite family of convex functions [45], we have

$$\begin{aligned}
\partial l_i(\boldsymbol{\alpha}) &= \{-\overline{\boldsymbol{\gamma}}_{i1}\} \text{ if } l_{i1}(\boldsymbol{\alpha}) < l_{i2}(\boldsymbol{\alpha}), \{-\overline{\boldsymbol{\gamma}}_{i2}\} \text{ if } l_{i1}(\boldsymbol{\alpha}) > l_{i2}(\boldsymbol{\alpha}), \\
&\quad [-\overline{\boldsymbol{\gamma}}_{i1}, -\overline{\boldsymbol{\gamma}}_{i2}] \text{ otherwise}, \\
\partial p_{i1}(\boldsymbol{\alpha}) &= \partial \left[\max_{j=1,\ldots,K} \left\{ \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} \rangle - y_i \right\} \right] = \text{co}\{\boldsymbol{u}^{(i,j_i)} : j_i \in I_{i1}(\boldsymbol{\alpha})\}, \\
\partial p_{i2}(\boldsymbol{\alpha}) &= \partial \left[\max_{q=1,\ldots,M} \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} \rangle \right] = \text{co}\{\boldsymbol{v}^{(i,q_i)} : q_i \in I_{i2}(\boldsymbol{\alpha})\}, \text{ and} \\
I_{i1}(\boldsymbol{\alpha}) &:= \text{argmax}_{j=1,\ldots,K} \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} \rangle, \; I_{i2}(\boldsymbol{\alpha}) := \text{argmax}_{q=1,\ldots,M} \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} \rangle.
\end{aligned}$$

Here $[\boldsymbol{a}, \boldsymbol{b}]$ is a line segment between $\boldsymbol{a}$ and $\boldsymbol{b}$, co denotes the convex hull of a set of points. Thus, we can take $\overline{\boldsymbol{\gamma}}_{i1} = \boldsymbol{u}^{(i,\overline{j}_i)}, \overline{\boldsymbol{\gamma}}_{i2} = \boldsymbol{v}^{(i,\overline{q}_i)}, \boldsymbol{\gamma}^k_{i1} = \boldsymbol{u}^{(i,j^k_i)}, \boldsymbol{\gamma}^k_{i2} = \boldsymbol{v}^{(i,q^k_i)}$ where $\overline{j}_i \in I_{i1}(\overline{\boldsymbol{\alpha}}), \overline{q}_i \in I_{i2}(\overline{\boldsymbol{\alpha}}), j^k_i \in I_{i1}(\boldsymbol{\alpha}^k), q^k_i \in I_{i2}(\boldsymbol{\alpha}^k), i = 1, \ldots, m$.

Finally, the subgradient $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$ is computed as

$$\begin{aligned}
\boldsymbol{\beta}^k = 2 \sum_{i=1}^{m} &[p_{i1}(\boldsymbol{\alpha}^k) + p_{i2}(\boldsymbol{\alpha}^k) + 2l_i(\boldsymbol{\alpha}^k)] \times \\
&[\boldsymbol{\gamma}^k_{i1} + \boldsymbol{\gamma}^k_{i2} - 2\overline{\boldsymbol{\gamma}}_{i1} - 2(\overline{\boldsymbol{\gamma}}_{i2} - \overline{\boldsymbol{\gamma}}_{i1}) \mathbb{1}_{\{l_{i1}(\boldsymbol{\alpha}) > l_{i2}(\boldsymbol{\alpha})\}}(\boldsymbol{\alpha}^k)]. \qquad (15)
\end{aligned}$$

Here the function $\mathbb{1}_{\mathcal{A}}(\boldsymbol{\alpha}) = 1$ if $\boldsymbol{\alpha} \in \mathcal{A}$, 0 otherwise.

Solve the convex subproblem (6):

The convex subproblem (6) can be reformulated as

$$\min \left\{ \sum_{i=1}^{m} 2(t_i)^2 + \sum_{i=1}^{m} 2(\tau_i)^2 - \langle \boldsymbol{\beta}^k, \boldsymbol{\alpha} \rangle : \begin{array}{l} t_i \geq -l_{i1}(\boldsymbol{\alpha}) + p_{i1}(\boldsymbol{\alpha}), i = 1, \ldots, m, \\ t_i \geq -l_{i2}(\boldsymbol{\alpha}) + p_{i1}(\boldsymbol{\alpha}), i = 1, \ldots, m, \\ \tau_i \geq -l_{i1}(\boldsymbol{\alpha}) + p_{i2}(\boldsymbol{\alpha}), i = 1, \ldots, m, \\ \tau_i \geq -l_{i2}(\boldsymbol{\alpha}) + p_{i2}(\boldsymbol{\alpha}), i = 1, \ldots, m \end{array} \right\}$$

which is in fact a convex quadratic program of the form

$$\left\{ \begin{array}{l} \min_{\boldsymbol{\alpha}, \boldsymbol{t}, \boldsymbol{\tau}} \sum_{i=1}^{m} 2(t_i)^2 + \sum_{i=1}^{m} 2(\tau_i)^2 - \langle \boldsymbol{\beta}^k, \boldsymbol{\alpha} \rangle, \\ \text{s.t. } t_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} - \boldsymbol{u}^{(i,\overline{j}_i)} \rangle, \; i = 1, \ldots, m, j = 1, \ldots, K, \\ \quad t_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} - \boldsymbol{v}^{(i,\overline{q}_i)} \rangle - y_i, \; i = 1, \ldots, m, j = 1, \ldots, K, \\ \quad \tau_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} - \boldsymbol{u}^{(i,\overline{j}_i)} \rangle + y_i, \; i = 1, \ldots, m, q = 1, \ldots, M, \\ \quad \tau_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} - \boldsymbol{v}^{(i,\overline{q}_i)} \rangle, \; i = 1, \ldots, m, q = 1, \ldots, M. \end{array} \right. \qquad (16)$$

Algorithm 1 DCA for the problem (14) (DCA-PL)

Initialization: Let $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$. Set $k = 0$.

repeat

1. Compute $\boldsymbol{\beta}^k \in \partial H(\boldsymbol{\alpha}^k)$ using (15).
2. Solve the convex quadratic program (16) to obtain $(\boldsymbol{\alpha}^{k+1}, \boldsymbol{t}^{k+1}, \boldsymbol{\tau}^{k+1})$.
3. $k = k + 1$.

until Stopping criteria are satisfied.

3.3. DCA^k for solving the problem (14)

At the iteration k in DCA^k, we choose $\overline{\boldsymbol{\alpha}} = \boldsymbol{\alpha}^k$. In this case, $\overline{j}_i = j_i^k$, $\overline{q}_i = q_i^k$, and $l_{ij}(\boldsymbol{\alpha}^k) = p_{ij}(\boldsymbol{\alpha}^k)$, $\overline{\boldsymbol{\gamma}}_{ij} = \boldsymbol{\gamma}_{ij}^k$. The computation of $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ becomes simpler than (15) as follows:

$$\boldsymbol{\beta}^k = 2\sum_{i=1}^{m} p_i(\boldsymbol{\alpha}^k)[\boldsymbol{u}^{(i,j_i^k)} - \boldsymbol{v}^{(i,q_i^k)}],\ j_i^k \in I_{i1}(\boldsymbol{\alpha}^k), q_i^k \in I_{i2}(\boldsymbol{\alpha}^k). \tag{17}$$

And the convex program in DCA^k is the same as the convex quadratic program (16) with the indexes $\overline{j}_i$ (resp. $\overline{q}_i$) replaced by j_i^k (resp. q_i^k):

$$\begin{cases} \min_{\boldsymbol{\alpha},\boldsymbol{t},\boldsymbol{\tau}} \sum_{i=1}^{m} 2(t_i)^2 + \sum_{i=1}^{m} 2(\tau_i)^2 - \langle \boldsymbol{\beta}^k, \boldsymbol{\alpha} \rangle, \\ \text{s.t. } t_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} - \boldsymbol{u}^{(i,j_i^k)} \rangle,\ i = 1,\ldots,m, j = 1,\ldots,K, \\ \quad t_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{u}^{(i,j)} - \boldsymbol{v}^{(i,q_i^k)} \rangle - y_i,\ i = 1,\ldots,m, j = 1,\ldots,K, \\ \quad \tau_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} - \boldsymbol{u}^{(i,j_i^k)} \rangle + y_i,\ i = 1,\ldots,m, q = 1,\ldots,M, \\ \quad \tau_i \geq \langle \boldsymbol{\alpha}, \boldsymbol{v}^{(i,q)} - \boldsymbol{v}^{(i,q_i^k)} \rangle,\ i = 1,\ldots,m, q = 1,\ldots,M. \end{cases} \tag{18}$$

Hence, DCA^k differs from DCA by computing the subgradient $\boldsymbol{\beta}^k$ and solving the convex quadratic program in the step 1 and step 2 respectively. The algorithm, named DCA^k-PL, can be described as follows.

3.4. Starting point for the proposed DCA algorithms

In this section, we suggest a deterministic strategy to compute the starting point $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$ for the proposed DCA algorithms. From the fact that

$$f(\boldsymbol{x}_i, \boldsymbol{\alpha}) - y_i = (g(\boldsymbol{x}_i, \boldsymbol{\alpha}) - y_i/2) - (h(\boldsymbol{x}_i, \boldsymbol{\alpha}) + y_i/2),$$

the idea is to fit the convex functions $g(\boldsymbol{x}, \boldsymbol{\alpha})$ (resp. $h(\boldsymbol{x}, \boldsymbol{\alpha})$) to the data points $\{(\boldsymbol{x}_i, y_i/2)\}_{i=1,\ldots,m}$ (resp. $\{(\boldsymbol{x}_i, -y_i/2)\}_{i=1,\ldots,m}$), by the following strategy which is quite similar to the random version in [24]:

Algorithm 2 DCA^k for the problem (14) (DCA^k-PL)

Initialization: Let $\boldsymbol{\alpha}^0 \in \mathbb{R}^n$. Set $k = 0$.

repeat

1. Compute $\boldsymbol{\beta}^k \in \partial H^k(\boldsymbol{\alpha}^k)$ using (17).
2. Solve the convex quadratic program (18) to obtain $(\boldsymbol{\alpha}^{k+1}, \boldsymbol{t}^{k+1}, \boldsymbol{\tau}^{k+1})$.
3. $k = k + 1$.

until Stopping criteria are satisfied.

Step 1: generate the set $\mathcal{X}_1$ of K points in $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1,\ldots,m}$ using the KKZ method [46] with the first point $\overline{\boldsymbol{x}} = \frac{1}{m}\sum_{i=1}^m \boldsymbol{x}_i$; then, similarly generate the set $\mathcal{X}_2$ of M points in the set $\mathcal{X} \setminus \mathcal{X}_1$ with the same first point $\overline{\boldsymbol{x}}$ in the KKZ method.
Step 2: find an index $j_i^1 \in \{1, \ldots, K\}$ (resp. $j_i^2 \in \{1, \ldots, M\}$) $(i = 1, \ldots, m)$ such that the j_i^1-th point in $\mathcal{X}_1$ (resp. the j_i^2-th point in $\mathcal{X}_2$) is closest to the point $\boldsymbol{x}_i$ (see Voronoi partitions in [24] for more details).
Step 3: compute the starting point

$$\boldsymbol{\alpha}^0 = (\boldsymbol{a}_1^0, b_1^0, \ldots, \boldsymbol{a}_K^0, b_K^0, \boldsymbol{a}_{K+1}^0, b_{K+1}^0, \ldots, \boldsymbol{a}_{K+M}^0, b_{K+M}^0)$$

where for $j = 1, \ldots, K$, $(\boldsymbol{a}_j^0, b_j^0)$ is an optimal solution to the linear least-square problem

$$\min_{(\boldsymbol{a}_j, b_j)} \sum_{i \in \mathcal{J}_1(j)} \left(\langle \boldsymbol{a}_j, \boldsymbol{x}_i \rangle + b_j - y_i/2\right)^2,$$

with $\mathcal{J}_1(j) := \{i \in \{1, \ldots, m\} : j_i^1 = j\}$; and for $j = 1, \ldots, M$, $(\boldsymbol{a}_{K+j}^0, b_{K+j}^0)$ is an optimal solution to the linear least-square problem

$$\min_{(\boldsymbol{a}_{K+j}, b_{K+j})} \sum_{i \in \mathcal{J}_2(j)} \left(\langle \boldsymbol{a}_{K+j}, \boldsymbol{x}_i \rangle + b_{K+j} + y_i/2\right)^2,$$

with $\mathcal{J}_2(j) := \{i \in \{1, \ldots, m\} : j_i^2 = j\}$.

4. Numerical experiments

In this section, our experiments aim to show the efficiency of our DCA-based approach for solving the continuous piecewise-linear fitting problem (14). In particular, we make a comparison between our algorithms DCA-PL and DCA^k-PL with two methods mentioned in Section 3: Discrete Gradient

Method (DGM) [39] for directly solving the problem (14) and Multivariate Adaptive Regression Splines (MARS) [40] for constructing a piecewise-linear model. We test the four algorithms on two synthetic datasets D1-D2 and seven real datasets D3-D9, which are summarized in Table 1. We generate a synthetic dataset of the points $\{(\boldsymbol{x}_i, y_i = \varphi(\boldsymbol{x}_i))\}_{i=1,\ldots,m}$ where $\boldsymbol{x}_i \in \mathcal{X} = \{-v, -v+1, \ldots, v-1, v\}^3$ and the function $\varphi : \mathbb{R}^3 \to \mathbb{R}$ is defined as

$$\varphi(\boldsymbol{x}) = \varphi(x_1, x_2, x_3) = \ln(\exp(x_1) + 2\exp(x_2)) - \ln(\exp(x_2) + \exp(x_3)).$$

Two synthetic datasets, named log-exp6, log-exp7, corresponds to the value of $v \in \{6, 7\}$, respectively. Their numbers of data points, m, are 3375, 9261, respectively. The real datasets in many areas are taken from UCI machine learning repository[1] and LIBSVM website[2].

Table 1: Descriptions of datasets: p is the dimension of the space; m is the number of data points.

Data	Name	m	p
D1	log-exp6	3375	3
D2	log-exp7	9261	3
D3	Yacht hydrodynamics	308	6
D4	housing	506	13
D5	abalone	4177	8
D6	White wine quality	4898	11
D7	Combined cycle power plant	9568	4
D8	cadata	20640	8
D9	Physicochemical properties of protein tertiary structure	45730	9

Comparison criteria: We are interested in the following criteria to evaluate the effectiveness of DCA-PL and DCA^k-PL when comparing with

[1] `http://www.ics.uci.edu/~mlearn/MLRepository.html`
[2] `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

DGM and MARS: the mean-square error (MSE) defined as

$$\text{MSE}(\boldsymbol{\alpha}) := \frac{1}{m}\sum_{i=1}^{m}[p_i(\boldsymbol{\alpha})]^2$$

and the CPU time (in seconds).

Set up experiments: All experiments were performed on a PC Intel(R) Xeon(R) CPU E5-2630 v2, @2.60GHz of 32GB RAM and implemented in Matlab R2016b. The software CPLEX 12.8 was used for solving convex quadratic programs in DCA-PL, DCA^k-PL, and DGM. The parameters of DGM are set the same as in [39]. The MARS method is implemented in a Matlab toolbox ARESLab [42] and its parameters are set by default for building piecewise-linear models. The stopping criterion at the iteration k for DCA-PL and DCA^k-PL is

$$|\text{MSE}(\boldsymbol{\alpha}^k) - \text{MSE}(\boldsymbol{\alpha}^{k-1})| \leq \varepsilon(1 + \text{MSE}(\boldsymbol{\alpha}^{k-1}))$$

or the CPU time exceeds the maximum time. The default tolerance ε is set to 10^{-4}. The maximum time for all algorithms is set to 3600 seconds for D9 and 1800 seconds for the others. As for the synthetic datasets D1-D2, the pair (K, M) belongs to the set $\big\{(2,4),(2,6),(3,2),(3,3),(4,3),(4,5),(6,5),(6,7),(8,5),(8,7)\big\}$ while it is set to $(3,2)$ for the real datasets D3-D9 due to the best trade-off between rapidity and quality of solutions on synthetic datasets. The same starting point of three algorithms DCA-PL, DCA^k-PL, and DGM is described in Section 3.4. The point $\overline{\boldsymbol{\alpha}}$ in DCA-PL is set to $\boldsymbol{\alpha}^0$. We perform a validation procedure on real datasets to evaluate the piecewise-linear model obtained by the algorithm. In particular, each dataset is divided into two subsets: training set containing 80% of dataset, and test set containing 20% of dataset. First, we learn a piecewise-linear model on the training set. Then, we use that piecewise-linear model on the test set to compute the MSE.

Descriptions of results' tables: The comparative results of all four algorithms on two synthetic datasets in terms of the MSE and the CPU time with different pairs of (K, M) are reported in Table 2. As for seven real datasets, we report the MSE and the CPU time of our notable algorithm DCA^k-PL and two existing algorithms DGM, MARS in Table 3.

Table 2: Comparative results of four algorithms DCA-PL, DCA^k-PL, DGM and MARS on two synthetic datasets D1 and D2 in terms of Mean-Square Error (MSE) and CPU time (in seconds). Bold values indicate the best results for each pair (K,M).

			Mean-Square Error MSE			CPU (in seconds)		
Data	K	M	DCA-PL	DCA^k-PL	DGM	DCA-PL	DCA^k-PL	DGM
D1	2	4	2.19e-02	**1.33e-02**	2.36e-02	2.94e+02	**4.78e+01**	1.80e+03
	2	6	1.92e-02	**1.29e-02**	2.64e-02	3.28e+02	**5.81e+01**	1.80e+03
	3	2	**1.89e-02**	1.93e-02	4.32e-02	1.44e+02	**3.51e+01**	1.80e+03
	3	3	**6.92e-03**	7.71e-03	2.37e-02	2.37e+02	**3.73e+01**	1.80e+03
	4	3	8.18e-03	**2.75e-03**	2.42e-02	2.49e+02	**6.81e+01**	1.80e+03
	4	5	8.13e-03	**4.31e-03**	2.40e-02	3.00e+02	**7.60e+01**	1.80e+03
	6	5	7.73e-03	**1.48e-03**	2.25e-02	6.16e+02	**1.05e+02**	1.80e+03
	6	7	1.68e-02	**2.03e-03**	1.22e-02	8.25e+02	**1.40e+02**	1.80e+03
	8	5	3.74e-03	**5.63e-04**	2.37e-02	4.06e+02	**9.32e+01**	1.80e+03
	8	7	6.03e-03	**1.05e-03**	2.33e-02	3.46e+02	**1.20e+02**	1.80e+03
	MARS		MSE = 4.26e+00			CPU = 2.43e-01		
D2	2	4	2.38e-02	**1.30e-02**	2.56e-02	1.18e+03	**1.70e+02**	1.80e+03
	2	6	1.81e-02	**1.30e-02**	4.27e+00	1.16e+03	**2.21e+02**	1.80e+03
	3	2	**1.89e-02**	1.92e-02	2.57e-02	6.34e+02	**1.13e+02**	1.80e+03
	3	3	1.82e-02	**5.94e-03**	2.53e-02	7.27e+02	**1.27e+02**	1.80e+03
	4	3	5.72e-03	**2.47e-03**	2.56e-02	9.59e+02	**2.38e+02**	1.80e+03
	4	5	7.59e-03	**4.12e-03**	3.19e-02	1.16e+03	**2.84e+02**	1.80e+03
	6	5	1.15e-02	**1.40e-03**	2.39e-02	1.81e+03	**3.32e+02**	1.80e+03
	6	7	5.78e-02	**2.04e-03**	4.80e-02	1.80e+03	**4.12e+02**	1.80e+03
	8	5	6.53e-03	**4.76e-04**	2.82e-02	1.80e+03	**2.95e+02**	1.80e+03
	8	7	9.43e-03	**9.88e-04**	2.36e-02	1.83e+03	**5.23e+02**	1.80e+03
	MARS		MSE = 8.97e+00			CPU = 5.13e-01		

Table 3: Comparative results of our algorithm DCA^k-PL, MARS and DGM on seven real datasets in terms of the mean-square error (MSE) and CPU time (in seconds). Bold values indicate the best results.

			MSE			CPU (in seconds)		
Data	m	p	DCA^k-PL	MARS	DGM	DCA^k-PL	MARS	DGM
D3	308	6	**2.03e+00**	3.04e+00	7.11e+01	3.45e+01	**1.91e-01**	2.94e+02
D4	506	13	**1.04e+01**	1.68e+01	1.11e+01	1.48e+02	**1.18e+00**	1.80e+03
D5	4177	8	**4.13e+00**	4.34e+00	4.15e+00	5.31e+01	**8.92e+00**	1.80e+03
D6	4898	11	**5.58e-01**	6.35e-01	6.29e-01	5.76e+01	**6.26e+00**	1.80e+03
D7	9568	4	**1.68e+01**	1.81e+01	2.25e+01	7.71e+01	**1.98e+01**	1.80e+03
D8	20640	8	**4.10e+09**	4.32e+09	5.23e+09	9.55e+02	**7.31e+01**	1.80e+03
D9	45730	9	**4.10e+00**	4.30e+00	8.21e+00	3.31e+03	**5.14e+02**	3.60e+03

Comments on computational results:

a) On two synthetic datasets D1-D2 (see Table 2): our DCA-based approach is more efficient than the others on both datasets with different pairs of (K, M). Indeed, the MSE of DCA-PL is better than that of DGM and MARS in most cases of D1-D2 – the ratio of gain of DCA-PL versus DGM and MARS in D1 (resp. D2) varies from 7.21% to 84.1% in 9/10 cases (resp. from 6.98% to 99.5% in 9/10 cases) and from 99.4% to 99.9% (resp. from 99.3% to 99.9%) in all cases, respectively. Meanwhile, DCA^k-PL usually obtains the best fitting error with the ratio of gain versus DCA-PL, DGM, and MARS in D1 (resp. D2) from 32.7% to 87.9% in 8/10 cases (resp. from 28.2% to 96.4% in 9/10 cases), from 43.6% to 97.6% (resp. from 25.2% to 99.7%) in all cases, and from 99.5% to 99.9% (resp. from 99.7% to 99.9%) in all cases, respectively. On these two datasets, the MSE of DGM is smaller than that of MARS in all cases; MARS furnishes the worst MSE. In terms of CPU times, MARS runs the fastest while DCA^k-PL is the best among three algorithms DCA^k-PL, DCA-PL, and DGM for solving the same problem (14). The ratio of gain of DCA^k-PL versus DCA-PL and DGM on the dataset D1 (resp. D2) varies from 2.88 times to 6.36 times (resp. from 3.50 times to 6.94 times) and from 12.8 times to 51.3 times (resp. from 3.44 times to 15.9 times), respectively. DCA-PL is faster than DGM in all cases of D1 (resp. in 6/10 cases of D2) with the ratio from 2.18 times to 12.5 times (resp. from 1.52 times to 2.84 times). Note that DGM always exceeds the limited time (1800 seconds) on both datasets.

b) On seven real datasets D3-D9 (see Table 3): concerning the MSE, DCA^k-PL is more efficient than DGM and MARS on all seven datasets, specially large-scale datasets D7-D9. The ratio of gain of DCA^k-PL versus DGM and MARS in terms of MSE varies from 0.59% to 97.1% and from 4.65% to 37.8%, respectively. Meanwhile, MARS is better than DGM on 4/7 datasets and runs the fastest. Regarding the CPU time for solving the problem (14), DGM consumes more time than DCA^k-PL on all datasets – the ratio of gain of DCA^k-PL versus DGM varies from 1.09 times to 33.9 times.

5. Conclusions

We have investigated DC programming and DCA for solving a wide class of DC fitting problems. Two DCA schemes have been developed: a standard DCA and its modified version DCA^k with successive DC decomposition. The

DCA^k differs from the DCA in that the DC decompositions are updated during DCA iterations in order to better approximate the DC objective function. These two DCA-based algorithms have been deployed for solving the continuous piecewise-linear fitting problem. The numerical results on nine synthetic and real datasets with small-to-large sizes have turned out that, on both quality and rapidity, the proposed DCAs outperform the related existing approaches and the DCA^k is better than the standard DCA. We plan in future works to develop our DC programming and DCA-based approach for fitting applications.

Acknowledgment

This research is funded by Foundation for Science and Technology Development of Ton Duc Thang University (FOSTECT), website: http://fostect.tdtu.edu.vn, under Grant FOSTECT.2017.BR.10.

Appendix A.

In this appendix, we indicate a falsehood of Proposition 2 from the article [43]. Before doing that, we briefly introduce the maxima of minima of linear fitting problem presented in [43].

Given the set of data $A = \{(\boldsymbol{a}^i, b_i) \in \mathbb{R}^n \times \mathbb{R} : i \in I := \{1, \ldots, m\}\}$, the optimization problem can take of the form

$$\min \left\{ \frac{1}{m} \sum_{i=1}^{m} (\varphi(\boldsymbol{a}^i) - b_i)^2 : \varphi \in \mathcal{F} \right\},$$

where the set of continuous piecewise linear functions, denoted $\mathcal{F}$, is defined as: for $K \in \mathbb{N}$, $M_1, \ldots, M_K \in \mathbb{N}$,

$$\mathcal{F} = \left\{ \varphi : \mathbb{R}^n \to \mathbb{R} : \begin{array}{l} \varphi(\boldsymbol{a}) = \max_{k=1,\ldots,K} \min_{j=1,\ldots,M_k} (\langle \boldsymbol{x}^{kj}, \boldsymbol{a} \rangle + y_{kj}), \\ \boldsymbol{a} \in \mathbb{R}^n, \text{ for some } \boldsymbol{x}^{kj} \in \mathbb{R}^n, y_{kj} \in \mathbb{R} \end{array} \right\}.$$

It can be reformulated as

$$\begin{aligned} \min \quad F(\boldsymbol{x}, \boldsymbol{y}) &:= \frac{1}{m} \sum_{i=1}^{m} \left(\max_{k=1,\ldots,K} \min_{j=1,\ldots,M_k} (\langle \boldsymbol{x}^{kj}, \boldsymbol{a}^i \rangle + y_{kj}) - b_i \right)^2 \\ &= \frac{1}{m} \sum_{i=1}^{m} (\varphi_i(\boldsymbol{x}, \boldsymbol{y}) - b_i)^2, \end{aligned} \tag{A.1}$$

where

$$\begin{aligned}
\boldsymbol{x} &= \left(x_1^{11}, \ldots, x_n^{11}, \ldots, x_1^{KM_K}, \ldots, x_n^{KM_K}\right) \in \mathbb{R}^{nq},\\
\boldsymbol{y} &= (y_{11}, \ldots, y_{KM_K}) \in \mathbb{R}^q,\\
q &= \sum_{k=1}^{K} M_k,\\
\varphi_i(\boldsymbol{x}, \boldsymbol{y}) &= \max_{k=1,\ldots,K} \psi_{ik}(\boldsymbol{x}, \boldsymbol{y}),\\
\psi_{ik}(\boldsymbol{x}, \boldsymbol{y}) &= \min_{j=1,\ldots,M_k} \{\langle \boldsymbol{x}^{kj}, \boldsymbol{a}^i \rangle + y_{kj}\},\ i = 1, \ldots, m, k = 1, \ldots, K.
\end{aligned}$$

Let us define the functions

$$\begin{aligned}
\psi_{ik}^1(\boldsymbol{x}, \boldsymbol{y}) &= \sum_{j=1}^{M_k} (\langle \boldsymbol{x}^{kj}, \boldsymbol{a}^i \rangle + y_{kj}),\ i = 1, \ldots, m, k = 1, \ldots, K,\\
\psi_{ik}^2(\boldsymbol{x}, \boldsymbol{y}) &= \max_{j=1,\ldots,M_k} \sum_{t=1, t\neq j}^{M_k} (\langle \boldsymbol{x}^{kt}, \boldsymbol{a}^i \rangle + y_{kt}),\ i = 1, \ldots, m, k = 1, \ldots, K,\\
\varphi_{i1}(\boldsymbol{x}, \boldsymbol{y}) &= \max_{k=1,\ldots,K} \left(\psi_{ik}^1(\boldsymbol{x}, \boldsymbol{y}) + \sum_{j=1, j\neq k}^{K} \psi_{ij}^2(\boldsymbol{x}, \boldsymbol{y}) \right),\ i = 1, \ldots, m,\\
\varphi_{i2}(\boldsymbol{x}, \boldsymbol{y}) &= \sum_{j=1}^{K} \psi_{ij}^2(\boldsymbol{x}, \boldsymbol{y}),\ i = 1, \ldots, m.
\end{aligned}$$

As stated in Proposition 1 in [43], the function F is a DC function without its explicit DC representation. The functions ψ_{ik} and φ_i are DC, and their DC decompositions are

$$\begin{aligned}
\psi_{ik}(\boldsymbol{x}, \boldsymbol{y}) &= \psi_{ik}^1(\boldsymbol{x}, \boldsymbol{y}) - \psi_{ik}^2(\boldsymbol{x}, \boldsymbol{y}),\\
\varphi_i(\boldsymbol{x}, \boldsymbol{y}) &= \varphi_{i1}(\boldsymbol{x}, \boldsymbol{y}) - \varphi_{i2}(\boldsymbol{x}, \boldsymbol{y}).
\end{aligned}$$

Proposition 2 in [43] was stated as follows.

Let F_1 and F_2 be DC components of the function F: $F(\boldsymbol{x}, \boldsymbol{y}) = F_1(\boldsymbol{x}, \boldsymbol{y}) - F_2(\boldsymbol{x}, \boldsymbol{y})$, $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}^{nq} \times \mathbb{R}^q$. Then subdifferentials of the functions F_1 and

F_2 at $(\boldsymbol{x}, \boldsymbol{y})$ can be expressed as:

$$\partial F_1(\boldsymbol{x}, \boldsymbol{y}) = 2 \left[\sum_{i \in I_+(\boldsymbol{x}, \boldsymbol{y})} (\varphi_i(\boldsymbol{x}, \boldsymbol{y}) - b_i) \partial \varphi_{i1}(\boldsymbol{x}, \boldsymbol{y}) - \sum_{i \in I_-(\boldsymbol{x}, \boldsymbol{y})} (\varphi_i(\boldsymbol{x}, \boldsymbol{y}) - b_i) \partial \varphi_{i2}(\boldsymbol{x}, \boldsymbol{y}) \right], \quad \text{(A.2)}$$

$$\partial F_2(\boldsymbol{x}, \boldsymbol{y}) = 2 \left[\sum_{i \in I_+(\boldsymbol{x}, \boldsymbol{y})} (\varphi_i(\boldsymbol{x}, \boldsymbol{y}) - b_i) \partial \varphi_{i2}(\boldsymbol{x}, \boldsymbol{y}) - \sum_{i \in I_-(\boldsymbol{x}, \boldsymbol{y})} (\varphi_i(\boldsymbol{x}, \boldsymbol{y}) - b_i) \partial \varphi_{i1}(\boldsymbol{x}, \boldsymbol{y}) \right]. \quad \text{(A.3)}$$

Here $I_+(\boldsymbol{x}, \boldsymbol{y}) = \{i \in I : \varphi_i(\boldsymbol{x}, \boldsymbol{y}) > b_i\}$, $I_-(\boldsymbol{x}, \boldsymbol{y}) = \{i \in I : \varphi_i(\boldsymbol{x}, \boldsymbol{y}) < b_i\}$.

We see that this proposition is not true for any DC components F_1 and F_2 of F, and the following is a counterexample. We consider the linear regression problem, a special case of the considered problem when $K = 1$ and $M_K = 1$. The problem (A.1) becomes

$$\min \quad F(\boldsymbol{x}, y) := \frac{1}{m} \sum_{i=1}^{m} \left[(\langle \boldsymbol{x}, \boldsymbol{a}^i \rangle + y) - b_i \right]^2,$$

Moreover, we have that $\varphi_i(\boldsymbol{x}, y) = \psi_{i1}(\boldsymbol{x}, y) = \langle \boldsymbol{x}, \boldsymbol{a}^i \rangle + y$, $\psi^1_{i1}(\boldsymbol{x}, y) = \psi_{i1}(\boldsymbol{x}, y)$, $\psi^2_{i1}(\boldsymbol{x}, y) = 0$, $\varphi_{i1}(\boldsymbol{x}, y) = \varphi_i(\boldsymbol{x}, y)$, and $\varphi_{i2}(\boldsymbol{x}, y) = 0$.

It is obvious that F is a convex function and thus, we have trivial DC components of F:

$$F_1 = F \text{ and } F_2 = 0. \quad \text{(A.4)}$$

Consequently, the subdifferential $\partial F_2(\boldsymbol{x}, y) = \{\mathbf{0}\}$ for all $(\boldsymbol{x}, y)$, which contradicts the fact from (A.3) that $\partial F_2(\boldsymbol{x}, y) = -\sum_{i \in I_-(\boldsymbol{x}, y)} (\varphi_i(\boldsymbol{x}, y) - b_i) \partial \varphi_i(\boldsymbol{x}, y)$. It concludes that the subdifferentials (A.2) and (A.3) do not hold for any DC components of F – that is to say, Proposition 2 is false.

References

[1] T. Pham Dinh, S. El Bernoussi, Algorithms for solving a class of nonconvex optimization problems. methods of subgradients, in: J.-B. Hiriart-Urruty (Ed.), Fermat Days 85: Mathematics for Optimization, Vol. 129

of North-Holland Mathematics Studies, North-Holland, 1986, pp. 249–271.

[2] T. Pham Dinh, H. A. Le Thi, Convex analysis approach to DC programming: theory, algorithms and applications, Acta Mathematica Vietnamica 22 (1) (1997) 289–355.

[3] T. Pham Dinh, H. A. Le Thi, DC optimization algorithms for solving the trust region subproblem, SIAM Journal of Optimization 8 (2) (1998) 476–505.

[4] H. A. Le Thi, Analyse numérique des algorithmes de l'optimisation DC. approches locale et globale. codes et simulations numériques en grande dimension. applications., Ph.D. thesis, University of Rouen, France (1994).

[5] H. A. Le Thi, T. Pham Dinh, The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems, Annals of Operation Research 133 (1–4) (2005) 23–46.

[6] H. A. Le Thi, T. Pham Dinh, DC programming and DCA: thirty years of developments, Mathematical programming, Special Issue: DC Programming - Theory, Algorithms and Applications 169 (1) (2018) 5–68.

[7] T. Pham Dinh, H. A. Le Thi, Recent advances in DC programming and DCA, in: N. T. Nguyen, H. A. Le Thi (Eds.), Transactions on Computational Intelligence XIII, Vol. 8342, Springer Berlin Heidelberg, 2014, pp. 1–37.

[8] H. A. Le Thi, V. T. Ho, T. Pham Dinh, A unified DC programming framework and efficient DCA based approaches for large scale batch reinforcement learning, Journal of Global Optimization 73 (2) (2019) 279–310.

[9] V. T. Ho, H. A. Le Thi, T. Pham Dinh, DCA with successive DC decomposition for convex piecewise-linear fitting, in: H. A. Le Thi, H. M. Le, T. Pham Dinh, N. T. Nguyen (Eds.), Advanced Computational Methods for Knowledge Engineering, Springer International Publishing, 2020, pp. 39–51.

[10] T. Pham Dinh, Elements homoduaux dune matrice a relatifs un couple de normes (ϕ, ψ). applications au calcul de $s_{(\phi,\psi)}(a)$, Séminaire dAnalyse Numérique, Grenoble (1975).

[11] T. Pham Dinh, Contribution la théorie de normes et ses applications l'analyse numérique, Thèse de doctorat d'etat es science, Université Joseph Fourier, Grenoble (1981).

[12] D. N. Phan, H. A. Le Thi, Group variable selection via $\ell_{p,0}$ regularization and application to optimal scoring, Neural Networks 118 (2019) 220–234.

[13] D. N. Phan, H. M. Le, H. A. Le Thi, Accelerated difference of convex functions algorithm and its application to sparse binary logistic regression, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 1369–1375.

[14] H. A. Le Thi, H. M. Le, D. N. Phan, B. Tran, Stochastic DCA for the large-sum of non-convex functions problem and its application to group variable selection in classification, in: Proceedings of the 34th International Conference on Machine Learning - Volume 70, JMLR.org, 2017, pp. 3394–3403.

[15] H. A. Le Thi, D. N. Phan, DC programming and DCA for sparse Fisher linear discriminant analysis, Neural Computing and Applications 28 (9) (2017) 2809–2822.

[16] H. A. Le Thi, M. C. Nguyen, DCA based algorithms for feature selection in multi-class support vector machine, Annals of Operations Research 249 (1) (2017) 273–300.

[17] D. N. Phan, H. A. Le Thi, T. Pham Dinh, Sparse covariance matrix estimation by DCA-based algorithms, Neural Computation 29 (11) (2017) 3040–3077.

[18] R. T. Rockafellar, Convex Analysis, Princeton University Press, 1970.

[19] S. Koshi, Convergence of convex functions and duality, Hokkaido Mathematical Journal 14 (3) (1985) 399–414.

[20] G. Salinetti, R. J. Wets, On the relations between two types of convergence for convex functions, Journal of Mathematical Analysis and Applications 60 (1) (1977) 211–226.

[21] E. Amaldi, S. Coniglio, L. Taccari, Discrete optimization methods to fit piecewise affine models to data points, Computers & Operations Research 75 (2016) 214–230.

[22] G. Balázs, A. György, C. Szepesvári, Max-affine estimators for convex stochastic programming, ArXiv e-prints (2016).
URL \url{https://arxiv.org/abs/1609.06331}

[23] S. Boyd, S.-J. Kim, L. Vandenberghe, A. Hassibi, A tutorial on geometric programming, Optimization and Engineering 8 (1) (2007) 67–127.

[24] A. Magnani, S. P. Boyd, Convex piecewise-linear fitting, Optimization and Engineering 10 (1) (2009) 1–17.

[25] T. Pham Dinh, V. T. Ho, H. A. Le Thi, DC programming and DCA for solving Brugnano-Casulli piecewise linear systems, Computers & Operations Research 87 (2017) 196–204.

[26] L. Taccari, Optimization methods for piecewise affine model fitting, Ph.D. thesis, Politecnico di Milano (2010).

[27] A. Toriello, J. P. Vielma, Fitting piecewise linear continuous functions, European Journal of Operational Research 219 (1) (2012) 86 – 95.

[28] Y. Wang, S. Boyd, Fast model predictive control using online optimization, IEEE Transactions on Control Systems Technology 18 (2) (2010) 267–278.

[29] P. Apps, N. V. Long, R. Rees, Optimal piecewise linear income taxation, Journal of Public Economic Theory 16 (4) (2011) 523–545.

[30] P. Campra, M. Morales, Trend analysis by a piecewise linear regression model applied to surface air temperatures in southeastern spain (1973–2014), Nonlinear Processes in Geophysics Discussions (2016) 1–25.

[31] J. Y. Choi, R. M. Kil, C.-H. Choi, Piecewise linear regression networks and its application to time series prediction, in: Proceedings of International Conference on Neural Networks, Vol. 2, 1993, pp. 1349–1352.

[32] P. C. Reiss, M. W. White, Household electricity demand, revisited, Review of Economic Studies 72 (3) (2005) 853–883.

[33] A. Toriello, G. Nemhauser, M. Savelsbergh, Decomposing inventory routing problems with approximate value functions, Naval Research Logistics 57 (8) (2010) 718–727.

[34] L. Yang, S. Liu, S. Tsoka, L. G. Papageorgiou, Mathematical programming for piecewise linear regression analysis, Expert Systems with Applications 44 (2016) 156 – 167.

[35] G. Balázs, Convex regression: Theory, practice, and applications, Ph.D. thesis, University of Alberta (2016).

[36] S. Goel, V. Kanade, A. Klivans, J. Thaler, Reliably learning the relu in polynomial time, in: S. Kale, O. Shamir (Eds.), Proceedings of the 2017 Conference on Learning Theory, Vol. 65 of Proceedings of Machine Learning Research, PMLR, Amsterdam, Netherlands, 2017, pp. 1004–1042.

[37] L. A. Hannah, D. B. Dunson, Multivariate convex regression with adaptive partitioning, Journal of Machine Learning Research 14 (2013) 3261–3294.

[38] N. Martinez, H. Anahideh, J. M. Rosenberger, D. Martinez, V. C. P. Chen, B. P. Wang, Global optimization of non-convex piecewise linear regression splines, Journal of Global Optimization 68 (3) (2017) 563–586.

[39] A. Bagirov, C. Clausen, M. Kohler, An algorithm for the estimation of a regression function by continuous piecewise linear functions, Computational Optimization and Applications 45 (1) (2010) 159–179.

[40] J. H. Friedman, Multivariate adaptive regression splines, The Annals of Statistics 19 (1) (1991) 1–67.

[41] J. D. Andrés, P. Lorca, F. J. d. C. Juez, F. Sánchez-Lasheras, Bankruptcy forecasting: A hybrid approach using fuzzy c-means clustering and multivariate adaptive regression splines (mars), Expert Systems with Applications 38 (3) (2011) 1866 – 1875.

[42] G. Jekabsons, Areslab: Adaptive regression splines toolbox for matlab/octave, available at `http://www.cs.rtu.lv/jekabsons/` (2016).

[43] A. Bagirov, S. Taheri, S. Asadi, A difference of convex optimization algorithm for piecewise linear regression, Journal of Industrial & Management Optimization 15 (2) (2019) 909–932.

[44] V. V. Gorokhovik, O. I. Zorko, G. Birkhoff, Piecewise affine functions and polyhedral sets, Optimization 31 (3) (1994) 209–221.

[45] M. Valadier, Sous-différentiels dune borne supérieure et dune somme continue de fonctions convexes, CR Acad. Sci. Paris Sér. AB 268 (1969) A39–A42.

[46] I. Katsavounidis, C. C. J. Kuo, Z. Zhang, A new initialization technique for generalized lloyd iteration, IEEE Signal Processing Letters 1 (10) (1994) 144–146.