Bundle methods for Nonsmooth DC Optimization

Adil Bagirov

Federation University Australia, Ballarat, Victoria, Australia

WOMBAT, RMIT, Melbourne, November 24, 2016

Joint work with K. Joki, N. Karmitsa, M. Makela
(Turku University, Finland)

Outline

- Introduction
- Nonsmooth DC optimization
- Cutting plane model for DC functions
- Minimization algorithm
- Convergence
- Computational results
- Conclusions

Introduction

We design an algorithm for solving a minimization problem of the form:

$$\text{minimize} \quad f(x) \quad x \in \mathbb{R}^n, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a difference of convex (DC) function:

$$f(x) = f_1(x) - f_2(x)$$

and $f_1, f_2 : \mathbb{R}^n \to \mathbb{R}$ are convex functions.

We assume that

$$\text{dom} f_i = \{x \in \mathbb{R}^n : \ |f_i(x)| < +\infty\} = \mathbb{R}^n, \ i = 1, 2.$$

Introduction

The class of DC functions is very broad. Any twice continuously differentiable function can be represented as a DC function. Moreover, any continuous function can be approximated by the sequence of DC functions. Many optimization problems of potential interest can be expressed into the form of a DC program such as production-transportation planning, location planning, engineering design, cluster analysis, multilevel programming and multi-objective programming.

Introduction

DC programming problems have been mainly considered in global optimization and some algorithms have been designed to find global solutions to such problems (L.T.H. An and P.D. Tao, Annals of Operations Research, 2005; R. Horst and N.V. Thoai, JOTA, 1999; R. Horst and H. Tuy: Global Optimization: Deterministic Approaches, 1990; H. Tuy: Convex Analysis and Global Optimization, 1998). However, so far a little attention has been paid to the development of local solution methods for specifically DC programming problems. Such methods are often needed as a part of global optimization solvers, since they typically utilize local methods.

Introduction

Here, our aim is to design a version of the bundle method to locally solve the unconstrained DC programming problem. The bundle method and its variations are among the most efficient methods in nonsmooth optimization. To date these methods have been designed based only on the convex model of a function, also in the nonconvex case. Versions of the bundle methods based on the explicitly known structures of a problem have not been studied in depth before.

Introduction

The subdifferential (or generalized gradient) of a convex function f at a point $x \in \mathbb{R}^n$ is the set:

$$\partial_c f(x) = \{\xi \in \mathbb{R}^n : f(y) - f(x) \geq f(x) + \xi^T(y - x), \forall y \in \mathbb{R}^n\}.$$

The ε-subdifferential of a convex function f at a point $x \in \mathbb{R}^n$ is given by:

$$\partial_\varepsilon f(x) = \{\xi \in \mathbb{R}^n : f(y) - f(x) \geq f(x) + \xi^T(y - x) - \varepsilon, \forall y \in \mathbb{R}^n\}.$$

Introduction

A function $f : \mathbb{R}^n \to \mathbb{R}$ is called a locally Lipschitz on $\mathbb{R}^n$ if for any bounded subset $X \subset \mathbb{R}^n$ there exists $L > 0$ such that

$$|f(x) - f(y)| \leq L\|x - y\| \ \forall x, y \in X.$$

The generalized derivative of a locally Lipschitz function f at a point x with respect to a direction $u \in \mathbb{R}^n$ is defined as:

$$f^0(x, u) = \limsup_{\alpha \downarrow 0, y \to x} \frac{f(y + \alpha u) - f(y)}{\alpha}.$$

Introduction

The subdifferential $\partial f(x)$ of the function f at x is:

$$\partial f(x) = \left\{\xi \in \mathbb{R}^n : \ f^0(x,u) \geq \langle \xi, u \rangle \ \forall u \in \mathbb{R}^n\right\}.$$

According to Rademacher's theorem any locally Lipschitz function defined on $\mathbb{R}^n$ is differentiable almost everywhere and its subdifferential can also be defined as:

$$\partial f(x) = \text{conv}\left\{\lim_{i\to\infty} \nabla f(x_i) : x_i \to x \text{ and } \nabla f(x_i) \text{ exists}\right\}.$$

Here "conv" denotes the convex hull of a set. Each vector $\xi \in \partial f(x)$ is called a subgradient. For convex function $f : \mathbb{R}^n \to \mathbb{R}$ one has $\partial f(x) = \partial_c f(x), x \in \mathbb{R}^n$. From now on we will use the notation ∂f for subdifferentials of convex functions.

Introduction

A DC function f is locally Lipschitz continuous on $\mathbb{R}^n$.

Generally a point $x \in \mathbb{R}^n$ is a local minimizer of the problem (1) if $f(x^*) = f_1(x^*) - f_2(x^*)$ is finite and there exists an $\varepsilon > 0$ such that $f_1(x^*) - f_2(x^*) \leq f_1(x) - f_2(x)$ for all $x \in B(x^*; \varepsilon)$.

Theorem 1 *Let f_1 and f_2 be convex functions. If $x^* \in \mathbb{R}^n$ is a local minimizer of f, then*

$$\partial f_2(x^*) \subset f_1(x^*). \tag{2}$$

Furthermore, this condition guarantees local optimality if f_2 is a polyhedral convex function.

Introduction

f_2 is a polyhedral convex function if

$$f_2(x) = \max_{i \in I} \left\{ a_i^T x + b_i \right\}, \quad a_i \in \mathbb{R}^n, b_i \in \mathbb{R}, I \in I.$$

A point x^* satisfied the condition (2) is called an inf-stationary point.

A point x^* is called a Clarke stationary point of the problem (1) if $0_n \in \partial f(x^*)$.

These optimality conditions are usually hard to be verified and for that reason it can be relaxed to the form:

$$\partial f_1(x^*) \bigcap \partial f_2(x^*) \neq \emptyset.$$

Introduction

Let $\varepsilon > 0$, a point $x^* \in \mathbb{R}^n$ is said to be an ε-critical point of Problem (1) if

$$\partial_\varepsilon f_1(x^*) \bigcap \partial_\varepsilon f_2(x^*) \neq \emptyset.$$

For a DC function f it is also possible to formulate the following global optimality condition. Unfortunately, this condition is rather difficult to utilize in solution methods for DC functions.

Theorem 2 *Let f_1 and f_2 be convex functions. A DC function $f = f_1 - f_2$ attains its global minimum at a point $x^* \in \mathbb{R}^n$, if and only if*

$$\partial_\varepsilon f_2(x^*) \subset \partial_\varepsilon f_1(x^*) \quad \textit{for all} \quad \varepsilon > 0.$$

Cutting plane model for DC functions

x_k is the current iteration point (or stability center) which corresponds to the best estimate of the minimum found so far by the algorithm. Furthermore, we have at our disposal two collections of auxiliary points y_j from previous iterations. One collection is for the DC component f_1 together with subgradients $\xi_{1,j} \in \partial f_1(y_j)$ for $j \in J_1^k$ and, similarly, the other is for the DC component f_2 together with subgradients $\xi_{2,j} \in f_2(y_j)$ for $j \in J_2^k$. Here J_1^k and J_2^k are nonempty sets of indices for the DC components f_1 and f_2, respectively, and these index sets need not to be similar:

$$B_i^k = \left\{(y_j, f_i(y_j), \xi_{i,j}), j \in J_i^k\right\}.$$

Cutting plane model for DC functions

In what follows, we assume that for a point $x_0 \in \mathbb{R}^n$ the set

$$\mathcal{F}_0 = \{x \in \mathbb{R}^n |\ f(x) \leq f(x_0)\}$$

is compact, where x_0 is the starting point used in our algorithm. In addition, (overestimated) Lipschitz constants of the convex DC components f_1 and f_2 are assumed to be known on the set $\mathcal{F}_\varepsilon = \{x \in \mathbb{R}^n |\ d(x, F_0) \leq \varepsilon\}, \varepsilon > 0$. We denote these constants by $L_1 > 0$ and $L_2 > 0$, respectively. It is easy to show that in this case the original function f is Lipschitz continuous on the set F_ε with a constant $L_1 + L_2$.

Cutting plane model for DC functions

The idea is to utilize the DC decomposition of the objective function f in the model construction. Since DC components f_i for $i = 1, 2$ are convex, we can construct a convex piecewise linear approximation of the DC component f_i by

$$\hat{f}_i^k(x) = \max_{j \in J_i^k} \left\{ f_i(y_j) + \xi_{i,j}^T (x - y_j) \right\}, \ x \in \mathbb{R}^n.$$

This can be rewritten in an equivalent form:

$$\hat{f}_i^k(x) = \max_{j \in J_i^k} \left\{ f_i(x_k) + \xi_{i,j}^T (x - x_k) - \alpha_{i,j}^k \right\}, \ x \in \mathbb{R}^n.$$

when we use the current iteration point x_k and the linearization error:

$$\alpha_{i,j}^k = f_i(x_k) - f_i(y_j) - (\xi_{i,j})^T (x_k - y_j), \ \forall j \in J_i^k.$$

Cutting plane model for DC functions

The linearization error is always nonnegative, that is $\alpha_{i,j}^k \geq 0$.

This property follows from the convexity of f_i and also implies that

$$\hat{f}_i^k(x) \leq f_i(x), \ \forall x \in \mathbb{R}^n, i = 1, 2.$$

The linearization error can be updated by using the formula

$$\alpha_{i,j}^{k+1} = \alpha_{i,j}^k - f_i(x_{k+1}) - f_i(x_k) - (\xi_{i,j})^T(x_{k+1} - x_k).$$

We do not need to keep track of the auxiliary points y_j together with the function values $f_i(y_j)$:

$$B_i^k = \left\{(\xi_{i,j}, \alpha_{i,j}^k) | j \in J_i^k\right\}, i = 1, 2.$$

Cutting plane model for DC functions

To approximate the original objective function f we substitute the functions f_i with their cutting plane models fi. Thus, the piecewise linear nonconvex cutting plane model of f is defined by

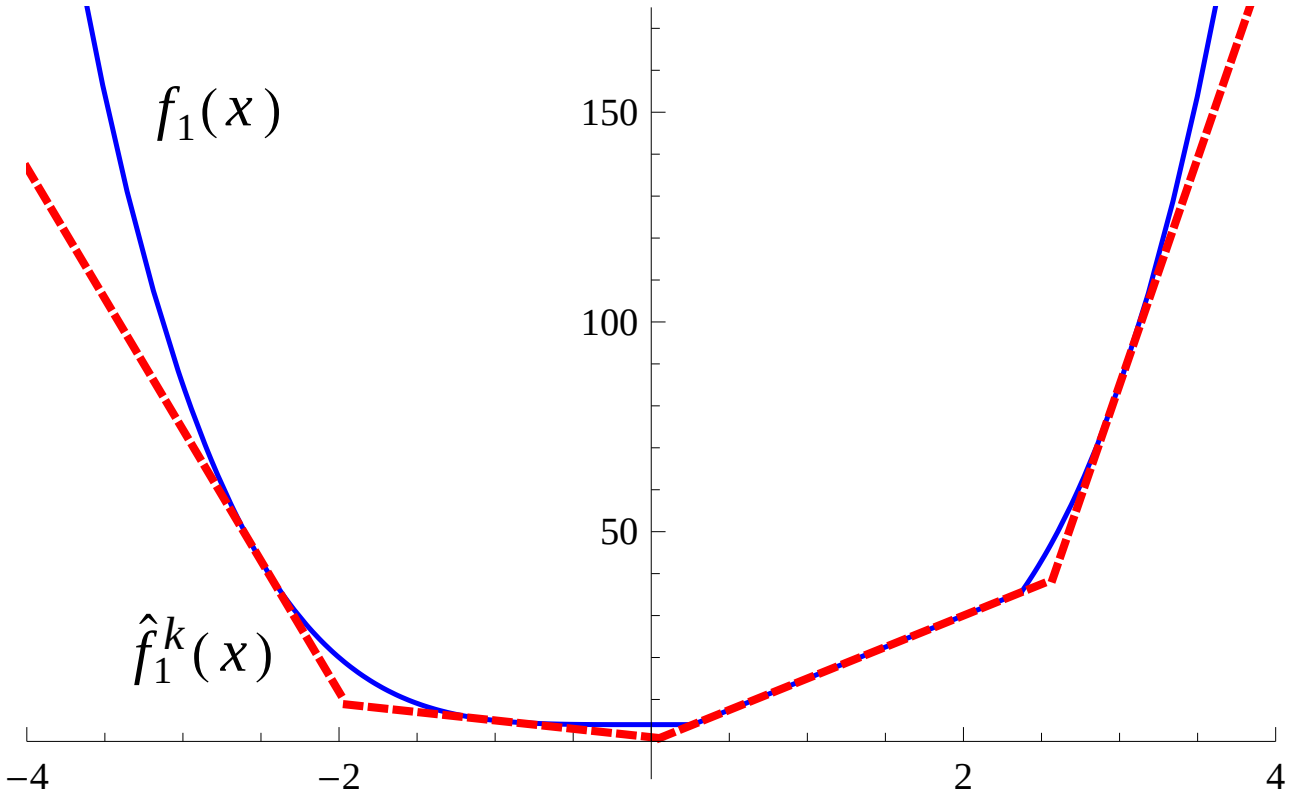
$$\hat{f}^k(x) = \hat{f}_1^k(x) - \hat{f}_2^k(x)$$

and it can be rewritten in an equivalent form

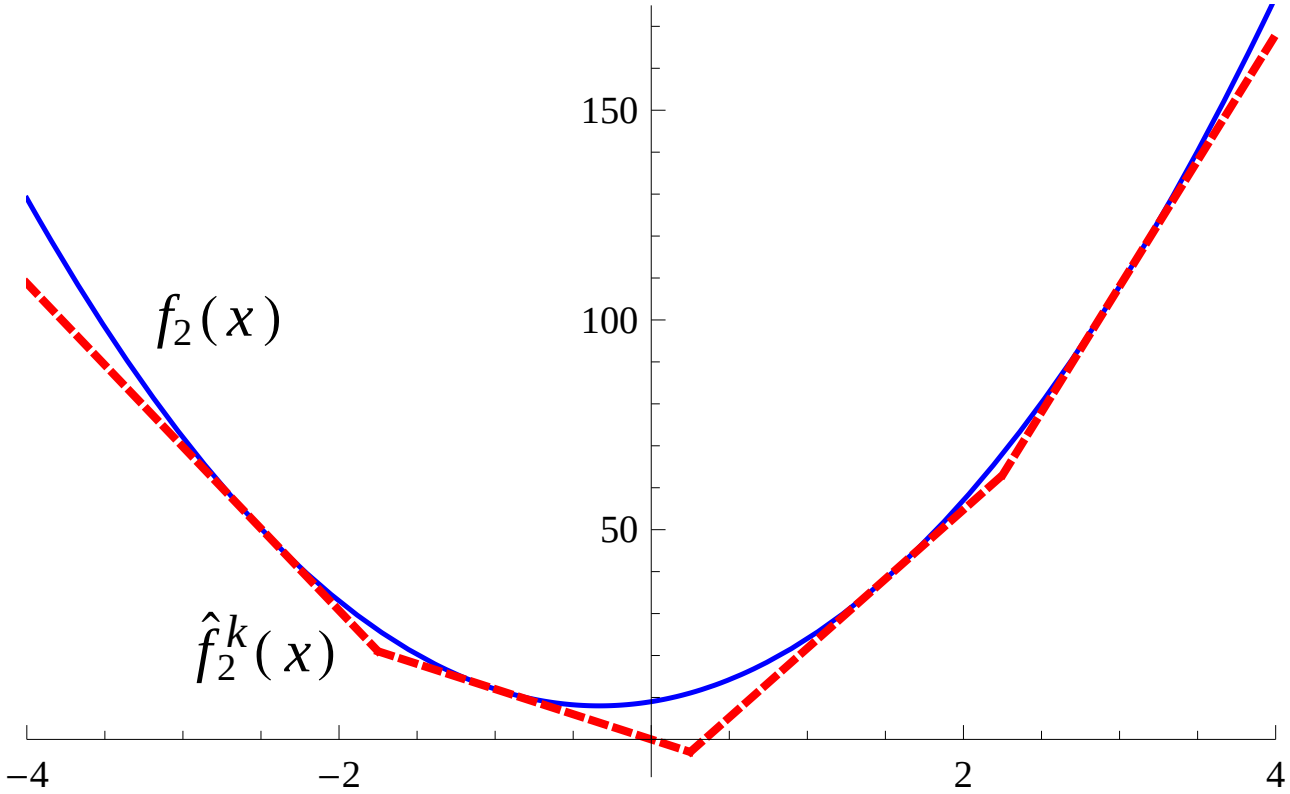
$$\hat{f}^k(x_k + d) = f(x_k) + \max_{j \in J_1^k} \left\{ (\xi_{1,j})^T d - \alpha_{1,j}^k \right\} + \min_{j \in J_2^k} \left\{ -(\xi_{2,j})^T d + \alpha_{2,j}^k \right\}$$

where the new variable $d = x - x_k$ is the search direction ("displacement") at the current iteration point x_k.

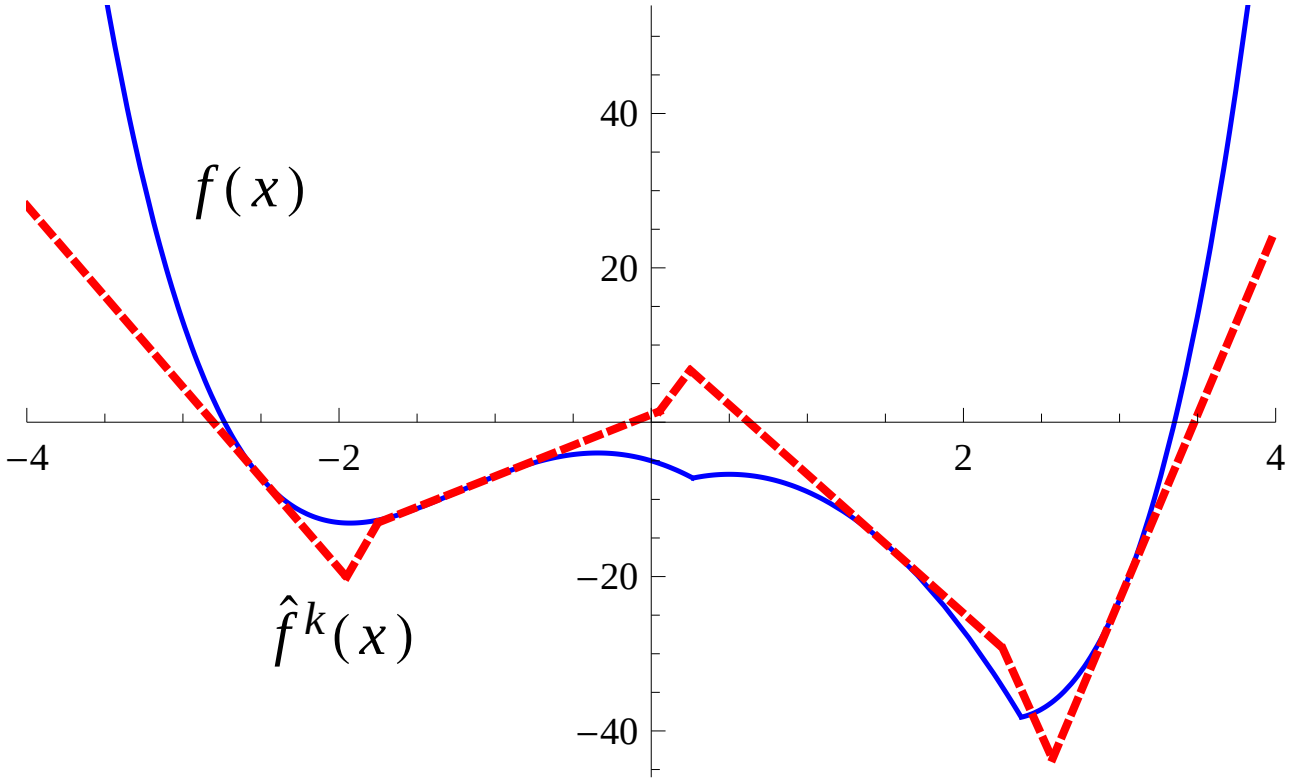
Cutting plane model for DC functions



Cutting plane model for DC functions



Cutting plane model for DC functions



Cutting plane model for DC functions

Denote

$$\Delta_1^k(d) = \max_{j \in J_1^k} \left\{ (\xi_{1,j})^T d - \alpha_{1,j}^k \right\},$$

$$\Delta_2^k(d) = \min_{j \in J_2^k} \left\{ -(\xi_{2,j})^T d + \alpha_{2,j}^k \right\}.$$

The search direction d_t^k can be computed by solving the problem:

$$\text{minimize} \quad P^k(d) = \Delta_1^k(d) + \Delta_2^k(d) + \frac{1}{2t}\|d\|^2 \tag{3}$$

subject to $d \in \mathbb{R}^n$.

Here $t > 0$ is the proximity parameter used in most bundle methods.

Cutting plane model for DC functions

The objective function $P^k(d)$ can be rewritten as:

$$P^k(d) = \min_{j \in J_2^k} \left\{ P_j^k(d) = \Delta_1^k(d) - (\xi_{2,j})^T d + \alpha_{2,j}^k + \frac{1}{2t}\|d\|^2 \right\}$$

Hence

$$\min_{d \in \mathbb{R}^n} \min_{j \in J_2^k} P_j^k(d) = \min_{j \in J_2^k} \min_{d \in \mathbb{R}^n} P_j^k(d)$$

Cutting plane model for DC functions

Due to this we can change the order of minimizations and solve first for each $j \in J_2^k$ the subproblem

$$\text{minimize} \quad P_j^k(d) = \Delta_1^k(d) - (\xi_{2,j})^T d + \alpha_{2,j}^k + \frac{1}{2t}\|d\|^2 \quad \text{s.t. } d \in \mathbb{R}^n. \tag{4}$$

Denote by $d_t^k(j)$ the solution to this subproblem. Then the search direction d_t^k is given by

$$d_t^k = d_t^k(j^*), \quad j^* = \text{argmin}_{j \in J_2^k} \left\{ P_j^k(d_t^k(j)) \right\}.$$

Cutting plane model for DC functions

The subproblem (4) can be reformulated as a smooth quadratic subproblem

$$\text{minimize} \quad v + \frac{1}{2t}\|d\|^2 \tag{5}$$

$$\text{subject to} \quad (\xi_{1,j} - \xi_{2,i})^T d - (\alpha^k_{1,j} - \alpha^k_{2,i}) \leq v, \ \forall \ j \in J^k_1.$$

$$v \in \mathbb{R}, \ d \in \mathbb{R}^n.$$

Cutting plane model for DC functions

By duality it is equivalent to the quadratic subproblem:

$$\text{minimize} \quad \frac{1}{2}t \left\| \sum_{j \in J_1^k} \lambda_j \xi_{1,j} - \xi_{2,i} \right\|^2 + \sum_{j \in J_1^k} \lambda_j \alpha_{1,j}^k - \alpha_{2,i}^k \tag{6}$$

$$\text{subject to} \quad \sum_{j \in J_1^k} \lambda_j = 1, \lambda_j \geq 0, \ \forall \ j \in J_1^k.$$

Cutting plane model for DC functions

Theorem 3 *For each $i \in J_2^k$, the problems* (5) *and* (6) *are equivalent, and they have unique solutions $(v_t^k(i), d_t^k(i))$ and $\lambda_{t,j}^k(i)$ for $j \in J_1^k$, respectively, such that*

$$d_t^k(i) = -t\left(\sum_{j\in J_1^k} \lambda_{t,j}^k(i)\xi_{1,j} - \xi_{2,i}\right)$$

$$v_t^k(i) = -\frac{1}{t}\left\|d_t^k(i)\right\|^2 - \sum_{j\in J_1^k} \lambda_{t,j}^k(i)\alpha_{1,j}^k - \alpha_{2,i}^k.$$

Cutting plane model for DC functions

$\Delta_1^k(d) + \Delta_2^k(d)$ can be seen as an approximation to the function:

$$h_k(d) = f(x_k + d) - f(x_k).$$

Lemma 1 *The following properties hold:*

(i) $\Delta_1^k(d) \leq f_1(x_k + d) - f_1(x_k).$

(ii) $\Delta_2^k(d) \geq -f_2(x_k + d) - (-f_2(x_k)).$

(iii) $\Delta_1^k(d_t^k) + \Delta_2^k(d_t^k) \leq -\frac{1}{2t}\|d_t^k\|^2 \leq 0.$

Minimization algorithm

Now we describe a new proximal bundle method (PBDC) for nonsmooth DC optimization. The core of PBDC is the "main iteration", which consists of a sequence of steps where the current iteration point remains unchanged.

The PBDC method starts with the initialization of the algorithm. In this step we need to choose a starting point $x_0 \in \mathbb{R}^n$ and set the following global parameters:

- the criticality tolerance $\delta > 0$ and the proximity measure $\varepsilon > 0$;
- the descent parameter $m \in (0, 1)$;
- the decrease parameter $r \in (0, 1)$ and the increase parameter $R > 1$.

Minimization algorithm

In addition, the following local parameters are set each time the "main iteration" is entered:

- the safeguard parameters t_{min} and t_{max}, $0 < t_{min} < t_{max}$;
- the local proximity measure $\theta > 0$.

Due to the importance of the "main iteration" we first describe in detail its algorithm and only after that we present the overall PBDC algorithm.

Minimization algorithm

Algorithm 1 Main iteration

Step 0. *(Stopping condition)* If $\|\xi_1(x_k) - \xi_2(x_k)\| < \delta$ then STOP (criticality achieved)

Step 1. *(Parameter initialization)* Calculate $j^* = \text{argmax}_{j \in J_2} \|\xi_{2,j}\|$ and set

$$\xi_{2,max} = \xi_{2,j^*}, \varepsilon_1 = \frac{\varepsilon}{2max\{L_1, L_2, 1/2\}},$$

$$t_{min} = r \frac{\varepsilon_1}{2(\|\xi_1(x_k)\| + \|\xi_{2,max}\|)}$$

$$t_{max} = Rt_{min}, \ \ \theta = rt_{min}\delta.$$

Minimization algorithm

Main iteration (continued)

Step 2. *(Search direction)* Solve the problem

$$\min_{d\in\mathbb{R}^n}\left\{\Delta_1(d)+\Delta_2(d)+\frac{1}{2t}\|d\|^2\right\}$$

and from the solution d_t compute the predicted changes

$$\Delta_1(d_t)=\max_{j\in J_1}\left\{(\xi_{1,j})^T d_t-\alpha_{1,j}\right\}$$

$$\Delta_2(d_t)=\min_{j\in J_2}\left\{-(\xi_{2,j})^T d_t+\alpha_{2,j}\right\}$$

If $d_t<\theta$ then go to Step 3 else go to Step 4.

Minimization algorithm

Main iteration (continued)

Step 3. *(Approximate stopping condition)* Set

$$J_1 = J_1 \setminus \{j \in J_1 | \alpha_{1,j} > \varepsilon\}, \;\; J_2 = J_2 \setminus \{j \in J_2 | \alpha_{2,j} > \varepsilon\}$$

and calculate values ξ_1^* and ξ_2^* such that

$$\|\xi_1^* - \xi_2^*\| = \min \left\{ \|\xi_1 - \xi_2\| : \;\; \xi_1 \in \text{conv} \left\{ \xi_{1,j} | j \in J_1 \right\}, \;\; \xi_2 \in \text{conv} \left\{ \xi_{2,j} | j \in J_2 \right\} \right\}$$

If $\|\xi_1^* - \xi_2^*\| < \delta$, then STOP ($\varepsilon$-criticality achieved) else set

$$t_{max} = t_{max} - r(t_{max} - t_{min}),$$

select the value $t \in [t_{min}, t_{max}]$ and go back to Step 2.

Minimization algorithm

Main iteration (continued)

Step 4. *(Descent test)* Set $y = x_k + d_t$. If

$$f(y) - f(x_k) \leq m\left(\Delta_1(d_t) + \Delta_2(d_t)\right) \tag{7}$$

then put $x_{k+1} = y$ and EXIT from the "main iteration".

Minimization algorithm

Main iteration (continued)

Step 5. *(Bundle update)* Compute $\xi_1 \in \partial f_1(y), \xi_2 \in \partial f_2(y)$ and set

$$\alpha_1 = f_1(x_k) - f_1(y) + \xi_1^T d_t, \quad \alpha_2 = f_2(x_k) - f_2(y) + \xi_2^T d_t$$

(a) If $f(y) - f(x_0) > 0$ and $\|d_t\| > \varepsilon_1$ then set $t = t - r(t - t_{min})$ and go back to Step 2.

(b) Otherwise insert the element:

(ξ_1, α_1) into B_1 for a suitable value of the index $\hat{j} \in J_1$

and, if $\Delta_2(d_t) \geq 0$, then insert also the element: (ξ_2, α_2) into B_2 for a suitable value of the index $\hat{j} \in J_2$.

Minimization algorithm

Main iteration (continued)

Step 6. *(Parameter update)* If $\|\xi_2\| \geq \|\xi_{2,max}\|$ then update

$$\xi_{2,max} = \xi_2,$$

$$t_{min} = r\frac{\varepsilon_1}{2(\|\xi_1(x_k)\| + \|\xi_{2,max}\|)}$$

and

$$\theta = rt_{min}\delta.$$

Go back to Step 2.

Minimization algorithm

Algorithm 2 Proximal bundle method for DC functions (PBDC)

Step 0. *(Initialization)* Select a starting point $x_0 \in \mathbb{R}^n$ and compute the value of DC components $f_1(x_0)$ and $f_2(x_0)$. Set $y_1 = x_0$ and initialize the iteration counter $k = 0$. Then calculate subgradients $\xi_{1,1} \in \partial f_1(y_1)$ and $\xi_{2,1} \in \partial f_2(y_1)$ and set $\alpha_{1,1}^k = \alpha_{2,1}^k = 0$. Initialize the bundles by setting

$$B_1^k = \left\{(\xi_{1,1}, \alpha_{1,1}^k)\right\}, \quad B_2^k = \left\{(\xi_{2,1}, \alpha_{2,1}^k)\right\}$$

and

$$J_1^k = J_2^k = \{1\}.$$

Minimization algorithm

Proximal bundle method for DC functions (PBDC) (continued)

Step 1. *(Main iteration)* Execute "main iteration" Algorithm 1. This either yields the new iteration point $x_{k+1} = x_k + d_t^k$ or indicates that the current solution x_k is ε-critical. In the case of ε-criticality we STOP the algorithm with x_k as the final solution.

Minimization algorithm

Proximal bundle method for DC functions (PBDC) (continued)

Step 2. *(Bundle update)* Compute the new DC component values and subgradients $f_i(x_{k+1})$ and $\xi(x_{k+1}) \in \partial f_i(x_{k+1})$ for $i = 1, 2$. Choose the bundles $B_1^{k+1} \subset B_1^k$ and $B_2^{k+1} \subset B_2^k$ for the next round and update the linearization errors using the formula

$$\alpha_{i,j}^{k+1} = \alpha_{i,j}^k + f_i(x_{k+1}) - f_i(x_k) - (\xi_{i,j})Td_t^k \text{ for all } i = 1, 2 \text{ and} j \in J_i^{k+1}.$$

Insert also the element
$(\xi_1(x_{k+1}), 0)$ into B_1^{k+1} for a suitable value of the index $\hat{j} \in J_1^{k+1}$ and
$(\xi_2(x_{k+1}), 0)$ into B_2^{k+1} for a suitable value of the index $\hat{j} \in J_2^{k+1}$.

Finally, update the iteration counter $k = k + 1$ and go back to Step 1.

Convergence

We require the following assumptions:

A1: The set $F_0 = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ is compact;

A2: Overestimates of the Lipschitz constants of f_1 and f_2, denoted by $L_1 > 0$ and $L_2 > 0$, respectively, are known on the set $F_\varepsilon = \{x \in \mathbb{R}^n | d(x, F_0) \leq \varepsilon\}$, where $\varepsilon > 0$ is the proximity measure.

A3: The objective function f is finite on $\mathbb{R}^n$.

Convergence

The bundle insertion rule at Step 5 of "main iteration" Algorithm 1 guarantees that all the points corresponding to the bundle elements in the bundles B_1 and B_2 are on the set F_ε. Together with the assumption **A2** this implies that we always have:

$$\|\xi_{1,j}\| \leq L_1 \text{ for points } y_j \text{ on } B_1 \text{ and,}$$

$$\|\xi_{2,j}\| \leq L_2 \text{ for points } y_j \text{ on } B_2.$$

This in turn indicates that the parameters t_{min} and θ are bounded away from zero at all iterations of the algorithm, because now $t_{min} \leq \bar{t} = r\varepsilon/(2L_1 + 2L_2) > 0$ and $\theta \geq \bar{\theta} = r\bar{t}\delta > 0$.

Convergence

Next we show that the solution d_t of the problem (3) is bounded in norm during the execution of the "main iteration".

Lemma 2 *For any proximity parameter $t > 0$ it holds that*

$$\|d_t\| \leq 2t \left(\|\xi_1(x_k)\| + \|\xi_{2,max}\| \right) \leq 2t(L_1 + L_2),$$

where x_k is the current iteration point, $\xi_1(x_k) \in \partial f_1(x_k)$ is the corresponding subgradient and $\|\xi_{2,max}\| = \max_{j \in J_2} \|\xi_{2,j}\|$.

Convergence

The following lemma is needed to prove the termination of the "main iteration".

Lemma 3 *If the condition* (7) *at Step 4 of Algorithm 1 is not satisfied, then:*

$$\xi_1^T d_t - \alpha_1 > m\Delta_1(d_t) + (m-1)\Delta_2(d_t)$$

where $\xi_1 \in \partial f_1(y)$ *is a subgradient calculated at the new auxiliary point* $y = x_k + d_t$ *and* $\alpha_1 = f_1(x_k) - f_1(y) + \xi_1^T d_t$ *is the corresponding linearization error.*

Convergence

Now we are ready to consider separately two possible cases which can occur in "main iteration" Algorithm 1.

Lemma 4 *Algorithm 1 cannot pass infinitely many times through Step 3.*

Lemma 5 *Algorithm 1 cannot pass infinitely many times through the sequence of steps from 4 to 6.*

Convergence

From Lemmas 4 and 5 we immediately get the next result.

Theorem 4 *The "main iteration" terminates after a finite number of steps.*

Theorem 5 *For any parameter $\delta > 0$ and $\varepsilon > 0$, the execution of the PBDC algorithm 2 stops after a finite number of "main iterations" at a point x^*, satisfying the approximate ε-criticality condition*

$$\|\xi_1^* - \xi_2^*\| \leq \delta \text{ with } \xi_1^* \in \partial_\varepsilon f_1(x^*),\ \xi_2^* \in \partial_\varepsilon f_2(x^*).$$

Computational results

Table 2: Summary of the numerical results

		PBDC					MPBNGC			NonsmoothDCA						TCM			
Prob.	n	n_f	n_{ξ_1}	n_{ξ_2}	*time*	f	n_f, n_ξ	*time*	f	n_{f_1}	n_{f_2}	n_{ξ_1}	n_{ξ_2}	*time*	f	n_f	n_ξ	*time*	f
1	2	22	17	16	0.00	2.0000000198	17	0.00	2.0000000000637	116	4	80	2	0.00	3.163836310813^*	333	135	0.01	2.00000000016
2	2	21	15	15	0.00	$1.10977 \cdot 10^{-12}$	39	0.00	$2.30926 \cdot 10^{-14}$	139	4	105	2	0.00	1.000000006012^*	526	129	0.01	$8.44528 \cdot 10^{-9}$
3	4	25	15	11	0.00	$2.26888 \cdot 10^{-12}$	112	0.00	$2.22091 \cdot 10^{-11}$	439	5	358	3	0.00	$3.87083 \cdot 10^{-9}$	517	218	0.01	$3.71039 \cdot 10^{-9}$
4	2	6	3	3	0.00	$8.43769 \cdot 10^{-15}$	7	0.00	$4.440892 \cdot 10^{-16}$	100	4	53	2	0.01	$2.43370 \cdot 10^{-10}$	125	55	0.00	$4.05524 \cdot 10^{-11}$
4	5	13	6	5	0.00	0.000000000000	30	0.00	$3.55271 \cdot 10^{-15}$	325	4	246	2	0.00	$2.85320 \cdot 10^{-10}$	399	180	0.00	$3.66982 \cdot 10^{-10}$
4	10	16	11	9	0.00	$5.68434 \cdot 10^{-14}$	61	0.00	0.000000000000	717	4	616	2	0.01	$9.15171 \cdot 10^{-10}$	844	388	0.01	$1.06388 \cdot 10^{-9}$
4	50	52	51	12	0.13	$9.09494 \cdot 10^{-13}$	561	0.29	$1.12621 \cdot 10^{-12}$	5823	6	5609	4	5.32	$1.26078 \cdot 10^{-9}$	6891	3368	3.56	$2.80851 \cdot 10^{-9}$
4	100	102	102	30	1.99	$9.09494 \cdot 10^{-13}$	1489	3.67	$1.17310 \cdot 10^{-11}$	12717	7	12459	5	150.2	$2.71484 \cdot 10^{-9}$	23299	11479	114.3	$1.66346 \cdot 10^{-9}$
4	150	198	198	70	15.00	$1.67092 \cdot 10^{-5}$	1592	10.17	$-1.00897 \cdot 10^{-11}$	974363	647	972524	645	15919	38.55251^*	21474	10389	75.11	1156.55819^*
4	200	341	341	132	64.15	$3.63797 \cdot 10^{-12}$	3528	36.08	$9.66338 \cdot 10^{-13}$	956640	634	954261	632	12676	570.20237^*	26645	12735	51.23	2515.04507^*
4	250	502	502	187	210.8	$7.27595 \cdot 10^{-12}$	3619	80.28	$1.04108 \cdot 10^{-10}$	938663	620	935851	618	15295	704.72555^*	55145	26833	167.4	4608.73643^*
4	350	705	705	331	743.7	$7.27595 \cdot 10^{-12}$	9594	1191.3	$2.66339 \cdot 10^{-10}$	827404	535	821439	533	13052	2091.28680^*	72241	34944	207.9	14168.40099^*
4	500	1357	1357	677	4120.3	$5.38420 \cdot 10^{-10}$	100000	16367	$-3.54219 \cdot 10^{-10}$	795446	511	788690	509	12504	5472.92134^*	61177	29345	157.2	40663.80100^*
4	750	2258	2258	1489	22919	$3.55066 \cdot 10^{-9}$	49043	57396	$7.42923 \cdot 10^{-10}$	369108	212	349011	210	4249	20872.37875^*	138040	66836	421.8	104683.25419^*
5	2	10	4	4	0.00	0.000000000000	5	0.00	$8.88178 \cdot 10^{-16}$	70	4	45	2	0.00	$1.67526 \cdot 10^{-9}$	149	66	0.00	$1.99221 \cdot 10^{-10}$
5	5	15	7	6	0.00	$6.12843 \cdot 10^{-14}$	25	0.00	$3.60822 \cdot 10^{-15}$	943	5	290	3	0.01	$1.53050 \cdot 10^{-10}$	1176	411	0.01	$7.12619 \cdot 10^{-10}$
5	10	21	14	11	0.00	$3.54161 \cdot 10^{-13}$	131	0.01	$8.08532 \cdot 10^{-11}$	104114	573	102099	571	2.01	$1.52056 \cdot 10^{-11}$	1316	479	0.01	$2.56584 \cdot 10^{-5}$
5	50	103	94	69	0.27	$8.10106 \cdot 10^{-13}$	51	0.02	$8.19851 \cdot 10^{-11}$	494993	614	491865	612	55.89	$8.45877 \cdot 10^{-4}$	1683	707	0.04	$1.73820 \cdot 10^{-5}$
5	100	47	28	23	0.36	$8.56592 \cdot 10^{-13}$	45	0.03	$1.00426 \cdot 10^{-10}$	941771	624	939068	622	153.2	$3.05781 \cdot 10^{-3}$	1785	754	0.08	$1.58279 \cdot 10^{-5}$
5	150	80	53	43	0.32	$2.58052 \cdot 10^{-12}$	43	0.05	$7.72262 \cdot 10^{-11}$	923022	611	919835	609	248.7	$7.47693 \cdot 10^{-4}$	1800	778	0.13	$3.56810 \cdot 10^{-5}$
5	200	107	56	47	0.40	$4.63365 \cdot 10^{-12}$	100000	209.8	$3.85558 \cdot 10^{-2}$	925026	614	921923	612	178.6	$3.11222 \cdot 10^{-4}$	1633	685	0.12	$2.95805 \cdot 10^{-6}$
5	250	55	49	40	0.43	$3.05058 \cdot 10^{-11}$	58	0.11	$7.69885 \cdot 10^{-12}$	958656	637	956191	635	184.6	$4.12652 \cdot 10^{-3}$	1744	745	0.19	$4.17524 \cdot 10^{-5}$
5	300	36	27	24	0.48	$1.84993 \cdot 10^{-11}$	105	0.29	$6.85744 \cdot 10^{-11}$	933889	619	931010	617	216.5	$1.93779 \cdot 10^{-3}$	1827	779	0.21	$7.26738 \cdot 10^{-5}$
5	350	43	34	27	0.39	$1.27391 \cdot 10^{-10}$	117	0.55	$9.87137 \cdot 10^{-11}$	948611	629	946077	627	248.6	$1.76198 \cdot 10^{-3}$	1599	678	0.21	$3.30601 \cdot 10^{-4}$
5	400	44	36	24	0.91	$7.88742 \cdot 10^{-11}$	95	0.36	$8.43775 \cdot 10^{-11}$	938748	623	935999	621	277.3	$2.09469 \cdot 10^{-4}$	2274	954	0.40	$8.77218 \cdot 10^{-5}$
5	500	29	22	18	1.46	$2.10487 \cdot 10^{-10}$	52	0.25	$1.84398 \cdot 10^{-12}$	935634	621	932878	619	371.5	$1.62630 \cdot 10^{-4}$	1493	637	0.30	$3.66659 \cdot 10^{-5}$
5	1000	32	26	22	0.62	$4.74784 \cdot 10^{-10}$	42	0.42	$8.81669 \cdot 10^{-11}$	959271	639	957081	637	715.2	$3.34356 \cdot 10^{-2}$	968	414	0.43	$5.39214 \cdot 10^{-4}$
5	1500	25	20	16	0.73	$6.26813 \cdot 10^{-10}$	41	0.74	$4.89645 \cdot 10^{-11}$	950142	631	947652	629	1065.0	$8.81968 \cdot 10^{-3}$	1380	562	0.93	$1.33563 \cdot 10^{-3}$
5	3000	29	24	16	2.36	$1.94150 \cdot 10^{-9}$	33	1.42	$3.54419 \cdot 10^{-11}$					fail					fail
5	10000	22	19	11	4.92	$5.93449 \cdot 10^{-9}$	33	5.53	$5.29412 \cdot 10^{-11}$					fail					fail
5	15000	178	54	48	51.83	$2.65698 \cdot 10^{-8}$	29	7.43	$2.93902 \cdot 10^{-11}$					fail					fail
5	20000	152	43	39	50.16	$1.41162 \cdot 10^{-3}$			fail					fail					fail
5	50000	99	26	26	74.97	$5.71421 \cdot 10^{-4}$			fail					fail					fail
6	2	27	19	15	0.00	-2.49999973	53	0.00	-2.4999999999998	78	4	64	2	0.00	-2.4999999999999973	164	71	0.00	-2.4999999999999942
7	2	72	63	40	0.00	0.5000000035	27	0.00	1.00000000000129^*	6671	64	4879	62	0.02	1.0000000013337100^*	482	166	0.01	0.500000057692649
8	3	84	64	40	0.00	3.5000001579	23	0.00	3.772727272785^*	114	4	82	2	0.00	3.7500000000000004^*	337	148	0.00	3.5000000000000000
9	4	90	78	41	0.00	1.8333334317	4	0.00	9.1999999999999995^*	149	6	147	4	0.00	9.1999999999999957^*	307	135	0.01	9.1999999999999886^*
10	2	21	13	7	0.00	-0.499999982	18	0.00	-0.499999999977	120	4	74	2	0.00	-0.500000000000000000	270	116	0.00	-0.499999999999999967
10	4	23	16	9	0.00	-2.499999788	11	0.00	-2.499999999991	201	5	136	3	0.00	-2.4999999999999991	342	148	0.00	-2.4999999999999996
10	5	22	13	10	0.00	-2.499999876	18	0.00	-2.499999999985	108	4	72	2	0.00	-0.49999999999999800^*	341	156	0.01	-2.4999999999999978
10	10	59	45	23	0.01	-8.499999618	27	0.00	-6.499999999994^*	504	8	355	6	0.00	-4.4999999999999982^*	408	185	0.01	-8.5000000000000000
10	20	60	44	20	0.01	-18.49998801	88	0.00	-16.499999999998^*	112	4	78	2	0.00	-0.50000003220404700^*	524	237	0.01	-18.500000000000004
10	50	140	121	54	0.12	-48.49989423	5	0.00	-0.500000000000^*	114717	661	121080	659	1.65	-4.5000000000000009^*	801	350	0.01	-48.500000000000028
10	100	352	343	92	1.50	-98.49997663	141	0.02	-90.49999999996^*	123	4	75	2	0.00	-0.50000002475810812^*	1166	520	0.62	-98.499999999999986
10	150	447	446	104	4.47	-126.489324^*	154	0.04	-134.499999999994^*	141	4	88	2	0.01	-0.49999999999999656^*	810	332	0.01	-145.54188264339533^*
10	200	321	320	86	4.34	-90.4958107^*	8	0.00	-82.500000000000^*	342628	661	347655	659	28.85	-4.4999999999999991^*	1388	587	0.13	-192.500000000000011^*

* the obtained value of the objective function f is not optimal

Conclusions and Acknowledgements

Australian Research Council (Project number: DP140103213).

THANK YOU!