Annales Mathematicae Silesianae **24** (2010), 39–60
Prace Naukowe Uniwersytetu Śląskiego nr 2874, Katowice

APPROXIMATION METHODS FOR SOLVING THE STOCHASTIC NETWORK FLOW PROBLEM WITH THE MOMENT MULTICRITERION

Marta Kostrzewska, Lesław Socha

Abstract. In this paper, the stochastic modification of the bicriterial minimum cost flow problem is presented. After the problem's formulation two approximate algorithms based on sandwich method for a convex curve approximation are presented. The obtained results are illustrated by examples.

1. Introduction

The bicriteria network cost flow problems, which describe a lot of real-life problems have been studied recently in many operation research papers. Although there exist exact computation methods for finding the analytic solution sets of bicriteria linear and quadratic cost flow problems, see e.g. [6], [8], Ruhe [5] and Zadeh [10] have shown that the determination of these sets may be very perplexing because of the possibility of exponential number of extreme nondominated objective vectors on the efficient frontier. The fact that efficient frontiers of the bicriteria linear and quadratic cost flow problems are convex curves in $\mathbb{R}^2$ allows to apply the sandwich methods for approximating convex curve in this field of optimization, see e.g. [1], [2], [7], [9]. However, some of these algorithms require a derivative information. A derivative free method was introduced in [9] by Yang and Goh, who applied it to bicriteria

Received: 18.01.2010. Revised: 30.09.2010.
(2010) Mathematics Subject Classification: 90C29.
Key words and phrases: bicriteria network problems, sandwich algorithm, stochastic costs.

quadratic minimum cost flow problems. The efficient frontiers of these problems are approximated by two piecewise linear functions, which construction requires solving one dimensional minimum cost flow problems. Also, Siem in [7] has proposed an algorithm based only on the function value information with the interval bisection partition rule and two new iterative strategies for the determination of a new input data point in each iteration.

In this paper we consider the network cost problem with random costs variables. We are interested in minimizing the expected value and the second moment of the total cost of the flow in a network, that is in solving the bicriterial minimum cost flow problem with linear and quadratic objective function, respectively. We present two methods of the approximation of the efficient frontier for this problem. In the first algorithm, based on the algorithm proposed by Siem [7], new points on the efficient frontier are computed according to the chord rule or the maximum error rule by solving proper convex quadratic network problems. In the second one, we modify the lower approximation function discussed in [9] what decreases the Hausdorff distance between upper and lower bounds. We give the proofs of the quadratic convergence property of Algorithm 2 and the linear convergence property of Algorithm 1.

The paper is organized as follows. In Section 2 we define the problem. In Section 3, we consider the case when the cost variables are mutually independent and we present two new methods of approximation the efficient frontier of our problem. Section 4 includes the information how to use described methodology from Section 3 in general case of cost variables. In Section 5, we discuss the convergence of the presented algorithms. In Section 6, we give some numerical examples to illustrate the discussed methods with the comparison of algorithms presented in [9] and [7]. Finally, Section 7 contains the conclusions and future research direction. Proofs of Lemma 1, Lemma 2 and Theorem 2 are given in Appendix.

2. Problem statement

Consider the directed network $G = (N, A)$, where N and A represent the node set and the arc set, respectively. Let $|N| = n$ and $|A| = m$. For each node $i \in N$ let the integer b_i be the supply or the demand of the node i. Let $l_{ij} \in [0, \infty)$ and $u_{ij} \in (l_{ij}, \infty)$ be the lower and upper bounds of flow through arc from the node i to the node j denoted by $(i, j) \in A$ and let $C_{ij} \colon \Omega \to [0, \infty)$ be a random variable representing the cost per unit of flow on this arc. If we assume that each variable C_{ij} has positive expected value $E[C_{ij}] = c_{ij}$, then we state the stochastic minimum cost flow problem with the moment multicriterion (SMCFP) in the following form

$$\min \left[E\Big[\sum_{(i,j)\in A} C_{ij}x_{ij}\Big], E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}\Big)^2\Big]\right]^T$$

$$\text{(1)} \qquad \text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N,$$

$$l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A,$$

where $x\colon A \to \mathbb{R}$ denotes the network flow, $x_{ij} = x(i,j)$ is the amount of flow on arc $(i,j) \in A$ and $\mathcal{X}$ is the set of all flows satisfying the above constraints.

According to the concept of Pareto optimality we define the relations $\le$ and $<$ in $\mathbb{R}^2$. Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^2$, then

$$\mathbf{a} \le \mathbf{b} \Longleftrightarrow [a_1, a_2]^T \le [b_1, b_2]^T \Longleftrightarrow a_1 \le b_1 \text{ and } a_2 \le b_2,$$
$$\mathbf{a} < \mathbf{b} \Longleftrightarrow [a_1, a_2]^T < [b_1, b_2]^T \Longleftrightarrow a_1 < b_1 \text{ and } a_2 < b_2.$$

Applying these definitions to bicriteria programming, a feasible solution $x \in \mathcal{X}$ is called the *efficient solution* of problem (1) if there does not exist a feasible solution $y \in \mathcal{X}$ such that

$$\text{(2)} \quad \left[E\Big[\sum_{(i,j)\in A} C_{ij}y_{ij}\Big], E\Big[\Big(\sum_{(i,j)\in A} C_{ij}y_{ij}\Big)^2\Big]\right]^T < \left[E\Big[\sum_{(i,j)\in A} C_{ij}x_{ij}\Big], E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}\Big)^2\Big]\right]^T.$$

The set of all efficient solutions and the image of this set under the objective functions are called the *efficient set* and the *efficient frontier*, respectively.

Lemma 1. *Efficient frontier of problem* (1) *is a convex curve in* $\mathbb{R}^2$.

We shall prove Lemma 1 in Appendix.

3. Independent cost variables

In this section we consider the case when the cost variables are mutually independent. Two sandwich algorithms for approximation the efficient frontier of problem (1) are presented.

Assume that variables C_{ij} and $C_{i'j'}$ are mutually independent for $(i,j) \neq (i',j')$ and that each C_{ij} has the finite second moment $E[C_{ij}^2] = d_{ij}$. The following problem called the stochastic independent minimum cost flow problem with the moment multicriterion (SICFP)

$$\begin{aligned}
&\min\Big[\sum_{(i,j)\in A} c_{ij}x_{ij}, \sum_{(i,j)\in A} d_{ij}{x_{ij}}^2 + \sum_{(i,j)\neq(i',j')} 2c_{ij}c_{i'j'}x_{ij}x_{i'j'}\Big]^T \\
&\text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N, \\
&l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A,
\end{aligned} \tag{3}$$

is equivalent to problem (1). Note that problem (3) is the bicriterial optimization problem with linear and quadratic objective functions, respectively.

Suppose that

$$P_k = \Big(\sum_{(i,j)\in A} c_{ij}x_{ij}^k, \sum_{(i,j)\in A} d_{ij}(x_{ij}^k)^2 + \sum_{(i,j)\neq(i',j')} 2c_{ij}c_{i'j'}x_{ij}^k x_{i'j'}^k\Big) \tag{4}$$

for $k = 1,2,\dots,r$ are given r points on the efficient frontier of problem (3) such that x^1 and x^r are the lexicographical minimum for the first and the second objective, respectively. Although in the algorithms given at the end of this section we use only three points to start Algorithm 1 and two points to start Algorithm 2, the described methodologies work for any number r of initial points, which may be obtained by solving the scalarization problems of problem (3) (SSMCFP)

$$\begin{aligned}
&\min\Big(\lambda_k \sum_{(i,j)\in A} c_{ij}x_{ij} + (1-\lambda_k)\Big(\sum_{(i,j)\in A} d_{ij}{x_{ij}}^2 + \sum_{(i,j)\neq(i',j')} 2c_{ij}c_{i'j'}x_{ij}x_{i'j'}\Big)\Big) \\
&\text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N, \\
&l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A,
\end{aligned} \tag{5}$$

where $\lambda_k = \frac{k-1}{r-1}$ for $k = 1,2,\dots,r$.

Another method is to find lexicographical minima of problem (3) and then solve $r-2$ quadratic programming problems with additional equality constraints called the first fixed coordinate SMCFP (FFCFP)

$$
\begin{aligned}
&\min\Big(\sum_{(i,j)\in A} d_{ij}{x_{ij}}^2 + \sum_{(i,j)\neq(i',j')} 2c_{ij}c_{i'j'}x_{ij}x_{i'j'}\Big)\\
(6)\qquad &\text{s.t.}\ \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i\in N,\\
&l_{ij}\le x_{ij}\le u_{ij}\quad \forall (i,j)\in A,\\
&\sum_{(i,j)\in A} c_{ij}x_{ij} = \sum_{(i,j)\in A} c_{ij}x_{ij}^1 + \frac{k}{r-1}\Big(\sum_{(i,j)\in A} c_{ij}x_{ij}^r - \sum_{(i,j)\in A} c_{ij}x_{ij}^1\Big),
\end{aligned}
$$

where $k = 1,\dots,r-2$. This method gives r points on the efficient frontier with the following property

$$
(7)\qquad \sum_{(i,j)\in A} c_{ij}x_{ij}^{k+1} - \sum_{(i,j)\in A} c_{ij}x_{ij}^{k} = \frac{1}{r-1}\Big(\sum_{(i,j)\in A} c_{ij}x_{ij}^r - \sum_{(i,j)\in A} c_{ij}x_{ij}^1\Big)
$$

for $k = 1,2,\dots,r-1$. Problems (5) and (6) can be solved using the method proposed by Goldfarb and Idnani in [3].

If the initial set $P = \{P_1,\dots,P_r\}$ of points on the efficient frontier is known and if we denote

$$
(8)\qquad f_1(x) = \sum_{(i,j)\in A} c_{ij}x_{ij}
$$

and

$$
(9)\qquad f_2(x) = \sum_{(i,j)\in A} d_{ij}{x_{ij}}^2 + \sum_{(i,j)\neq(i',j')} 2c_{ij}c_{i'j'}x_{ij}x_{i'j'},
$$

then $P_k = \left(f_1\left(x^k\right), f_2\left(x^k\right)\right)$ and $f_1(x^k) < f_1(x^{k+1})$ for $k = 1,\dots,r-1$ and we start building the approximation bounds. The upper approximation function of the frontier on the subinterval $[f_1(x^k), f_1(x^{k+1})]$ may be defined as the straight line through the points P_k and P_{k+1}, that is

$$
(10)\qquad u^k(a) = f_2(x^k) + \frac{f_2(x^{k+1}) - f_2(x^k)}{f_1(x^{k+1}) - f_1(x^k)}\big(a - f_1(x^k)\big)
$$

for $a \in \big[f_1(x^k), f_1(x^{k+1})\big]$. We will discuss two different methods of building the lower approximation function.

According to [7] the straight lines through the points P_{k-1} and P_k and the points P_{k+1} and P_{k+2} approximate the frontier from below so the lower bound l^k on the interval $[f_1(x^k), f_1(x^{k+1})]$ can be constructed in the following form

$$l^k(a) = \begin{cases} u^{k-1}(a) & \text{for } a \in \big[f_1(x^k), a_k\big], \\ u^{k+1}(a) & \text{for } a \in \big[a_k, f_1(x^{k+1})\big], \end{cases} \tag{11}$$

where $k = 2, \dots, r-2$ and a_k is the point of intersection of two linear functions u^{k-1} and u^{k+1}.

Moreover, we define the lower approximation bound on the most left and the most right interval as follows

$$l^1(a) = u^2(a) \quad \text{for } a \in \big[f_1(x^1), f_1(x^2)\big] \tag{12}$$

and

$$l^r(a) = \begin{cases} u^{r-1}(a) & \text{for } a \in \big[f_1(x^{r-1}), a_{r-1}\big], \\ f_2\,(x^r) & \text{for } a \in [a_{r-1}, f_1(x^r)], \end{cases} \tag{13}$$

where a_{r-1} is the point of intersection of function u^{r-1} and the constant function $f_2\,(x^r)$.

On the other hand, the simple modification of the definition presented in [9] yields to the following form of lower approximation bound l_*^k on the interval $[f_1(x^k), f_1(x^{k+1})]$

$$l_*^k(a) = \begin{cases} u^{k-1}(a) & \text{for } a \in [f_1(x^k), b_k], \\ f_2(y^k) + \dfrac{f_2(x^{k+1}) - f_2(x^k)}{f_1(x^{k+1}) - f_1(x^k)}(a - f_1(y^k)) & \text{for } a \in [b_k, c_k], \\ u^{k+1}(a) & \text{for } a \in [c_k, f_1(x^{k+1})], \end{cases} \tag{14}$$

where $k = 2, \dots, r-2$, constants b_k and c_k are the points of intersection of corresponding linear functions and y^k is the solution of the following convex quadratic network problem (the chord rule problem)

$$\begin{gathered} \min\Big(f_2(x) - \frac{f_2(x^k) - f_2(x^{k+1})}{f_1(x^k) - f_1(x^{k+1})} f_1(x)\Big) \\ \text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N, \\ l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A. \end{gathered} \tag{15}$$

Moreover, we define the lower approximation bound on the most left and the most right interval as follows

$$(16)\quad l_*^1(a) = \begin{cases} f_2(y^1) + \dfrac{f_2(x^2) - f_2(x^1)}{f_1(x^2) - f_1(x^1)}(a - f_1(y^1)) & \text{for } a \in [f_1(x^1), c_1], \\ u^2(a) & \text{for } a \in [c_1, f_1(x^2)], \end{cases}$$

and

$$(17)\qquad l_*^r(a) = \begin{cases} u^{r-1}(a) & \text{for } a \in [f_1(x^{r-1}, b_{r-1})], \\ f_2\left(x^r\right) & \text{for } a \in [b_{r-1}, f_1(x^r)]. \end{cases}$$

Let $\delta = \max\{\delta_1, \ldots, \delta_{r-1}\}$, where δ_k denotes the distance between the upper and lower approximation functions on interval $[f_1(x^k), f_1(x^{k+1})]$. We consider two measures the *Maximum error measure* (δ_k^M) and the *Hausdorff distance measure* (δ_k^H) defined as follows

$$(18)\qquad \delta_k^M = \max_{a\in[f_1(x^k),f_1(x^{k+1})]} \{u^k(a) - l^k(a)\}$$

and

$$(19)\qquad \delta_k^H = \max\Big\{\sup_{v\in L}\inf_{w\in U} \|v - w\|, \sup_{w\in U}\inf_{v\in L} \|v - w\|\Big\},$$

where

$$U = \big\{(a, u^k(a)) : a \in [f_1(x^k), f_1(x^{k+1})]\big\}$$

and

$$L = \big\{(a, l^k(a)) : a \in [f_1(x^k), f_1(x^{k+1})]\big\}.$$

We will also use for the comparison study in examples in Section 6 the *Uncertainty area measure* discussed in [7]

$$(20)\qquad \delta_k^U = \int_{f_1(x^k)}^{f_1(x^{k+1})} \left(u^k\left(a\right) - l^k\left(a\right)\right) da.$$

If δ does not satisfy a desired accuracy, we choose $k \in \{1, \ldots, r\}$ for which $\delta = \delta_k$ and determine new point $P^* = (f_1(x^*), f_2(x^*))$ on the efficient frontier of problem (3) such that $f_1(x^*) \in [f_1(x^k), f_1(x^{k+1})]$, then we update the set P of given points on efficient frontier due to following equality

$$P_i := \begin{cases} P_i & \text{for } i < k, \\ P^* & \text{for } i = k, \\ P_{i-1} & \text{for } i > k. \end{cases} \tag{21}$$

If the lower bound was built according to the first method we compute the new point using the *chord rule* or the *maximum error rule*, that is by solving the quadratic problem (15) or the following problem (the maximum error rule problem)

$$\begin{gathered} \min f_2(x) \\ \text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N, \\ l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A, \\ f_1(x) = a_k, \end{gathered} \tag{22}$$

where a_k is the point of intersection of linear functions u^{k-1} and u^{k+1}.

These partition rules return a new point $P^* = (f_1(x^*), f_2(x^*))$ on the efficient frontier of problem (3) such that $f_1(x^*) \in [f_1(x^k), f_1(x^{k+1})]$, although the maximum error rule can be used only under some assumptions concerning the first interval from the left side which are discussed in the next section. In [7] the new point is chosen according to the *interval bisection rule*, that is the interval with the greatest error is partitioned into two equal parts. In Section 6 we present two numerical examples which show that using the chord rule or the maximum error rule gives better results than the method used by Siem [7].

If we construct the lower approximation function due to definition (14), then the new point P^* on the efficient frontier is evaluated according to the chord rule, because quadratic problem (15) has been already solved.

After the set P of given points on the efficient frontier is updated, we determine new upper and lower bounds and repeat the procedure until we obtain an error δ smaller than the prescribed accuracy.

Algorithm 1 (The Simple Triangle Algorithm):

`Step 1.` Given an accuracy parameter $\epsilon > 0$ and an initial set of points on the efficient frontier $P = \{P_1, P_2, P_3\}$. Calculate lower and upper bounds l, u and error δ. Check if $\delta > \epsilon$, then go to `Step 2`, otherwise stop.

`Step 2.` Choose interval $\left[f_1\left(x^k\right), f_1\left(x^{k+1}\right)\right]$ for which the maximum error is achieved. Solve the quadratic problem (15) or (22) to obtain new point P^*. Update set P, lower and upper bounds l, u and error δ. Go to `Step 3`.

`Step 3.` Check if $\delta > \epsilon$, then go to `Step 2`, otherwise stop.

Algorithm 2 (The Trapezium Algorithm):

`Step 1.` Given an accuracy parameter $\epsilon > 0$ and an initial set of points on the efficient frontier $P = \{P_1, P_2\}$. Solve problem (15) and calculate lower and upper bounds l_*, u and error δ. Check if $\delta > \epsilon$, then go to `Step 2`, otherwise stop.

`Step 2.` Choose interval $\left[f_1\left(x^k\right), f_1\left(x^{k+1}\right)\right]$ for which the maximum error is achieved. New point $P^* = \left(f_1(y^k), f_2(y^k)\right)$. Update set P, solve problems (15) and calculate lower and upper bounds l_*, u and error δ. Go to `Step 3`

`Step 3.` Check if $\delta > \epsilon$, then go to `Step 2`, otherwise stop.

The geometric illustration of the Simple Triangle Algorithm and the Trapezium Algorithm is given in Fig. 1 and Fig. 2, respectively (the efficient frontier is the bold line). In Section 5 we study the convergence of these algorithms.

4. General case of the cost variables

In the general case, when the cost variables are not mutually independent, we have to compute values $d_{ij,i'j'} = E[C_{ij}C_{i'j'}]$ for $(i,j),(i',j') \in A$. If we know the covariance matrix of the cost variables, then $d_{ij,i'j'} = c_{ij}c_{i'j'} + cov[C_{ij}, C_{i'j'}]$ for $(i,j),(i',j') \in A$ and problem (1) again can be rewritten into following form

$$\begin{gathered}\min\Big[\sum_{(i,j)\in A} c_{ij}x_{ij},\ \sum_{(i,j)\in A} d_{ij,ij}{x_{ij}}^2 + \sum_{(i,j)\neq(i',j')} 2d_{ij,i'j'}x_{ij}x_{i'j'}\Big]^T \\ \text{s.t.} \sum_{\{j:(i,j)\in A\}} x_{ij} - \sum_{\{j:(j,i)\in A\}} x_{ji} = b_i \quad \forall i \in N, \\ l_{ij} \le x_{ij} \le u_{ij} \quad \forall (i,j) \in A. \end{gathered} \tag{23}$$

In the case when the covariance matrix is given, the solution of problem (23) can be obtained by the application of the methodology and both algorithms described in the previous section.

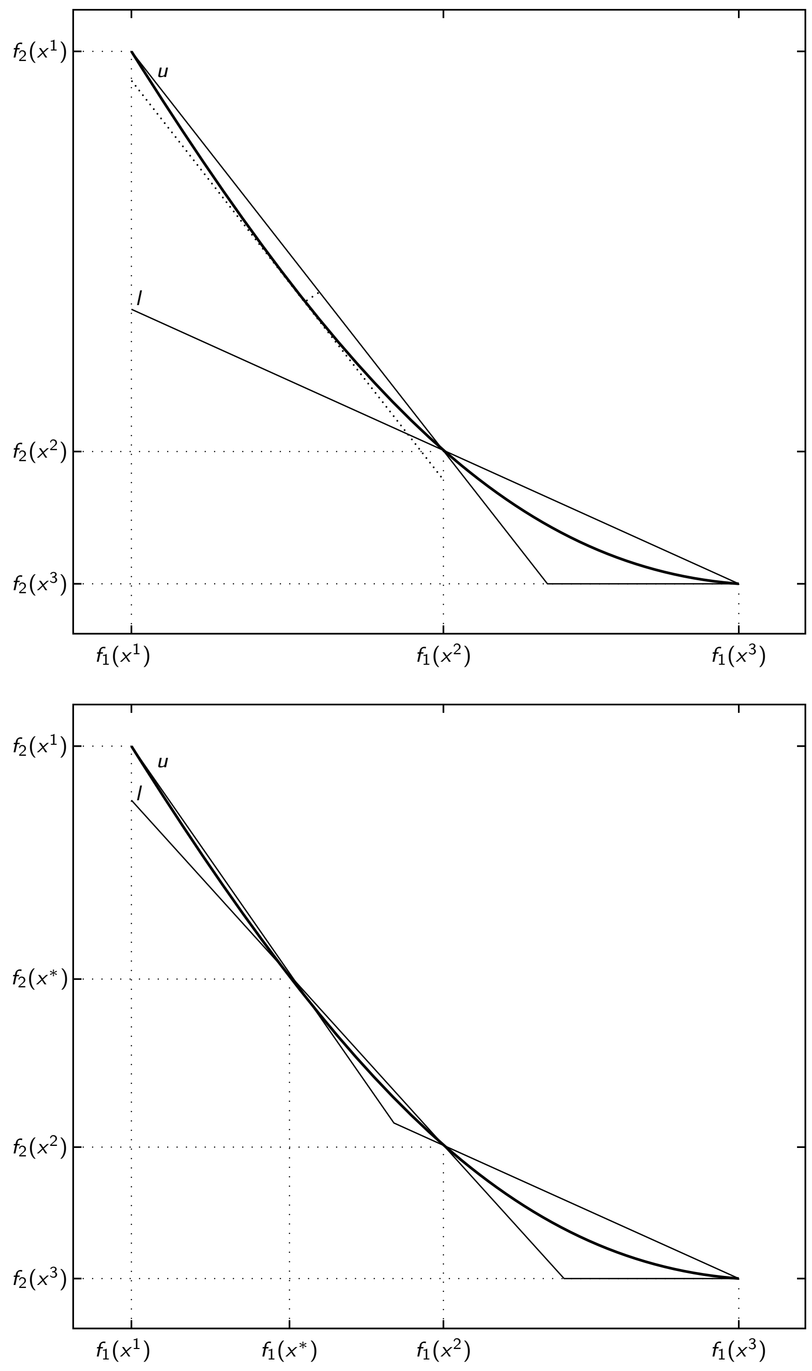

Figure 1. Lower and upper bounds built due to the Simple Triangle Algorithm with the chord rule

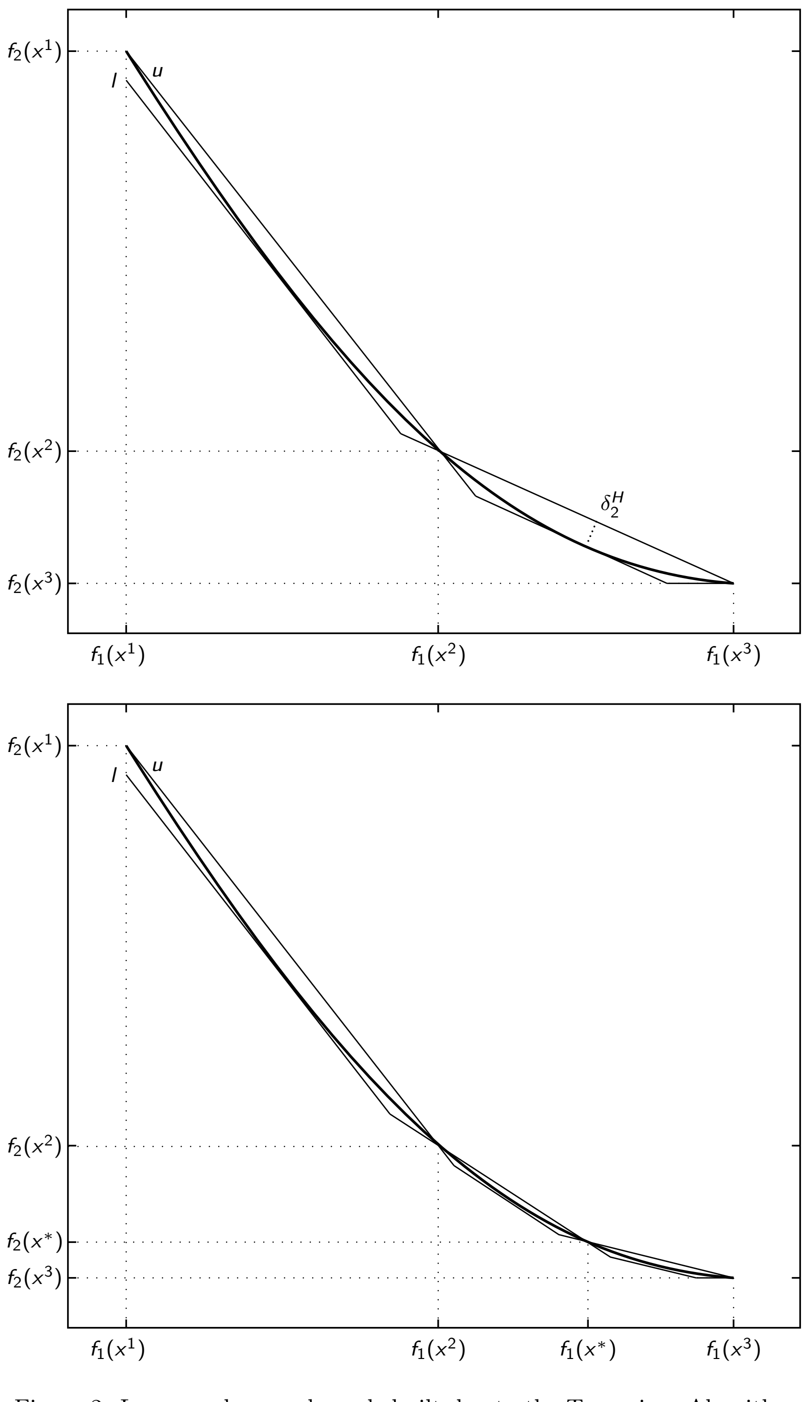

Figure 2. Lower and upper bounds built due to the Trapezium Algorithm

5. Convergence

In this section we present the convergence analysis of Algorithm 1 and Algorithm 2 based on proofs given in Rote [4] and Yang and Goh [9]. First, let us formulate the following remark, which shows the relation between two distance measures used in our algorithms.

Remark 1. For both given methods of construction of the lower and upper approximation bounds of the efficient frontier of problem (3), we have

$$\delta_k^M \leq \delta_k^H\left(1+\frac{f_2\left(x^k\right)-f_2\left(x^{k+1}\right)}{f_1\left(x^{k+1}\right)-f_1\left(x^k\right)}\right). \tag{24}$$

See Fig. 3.

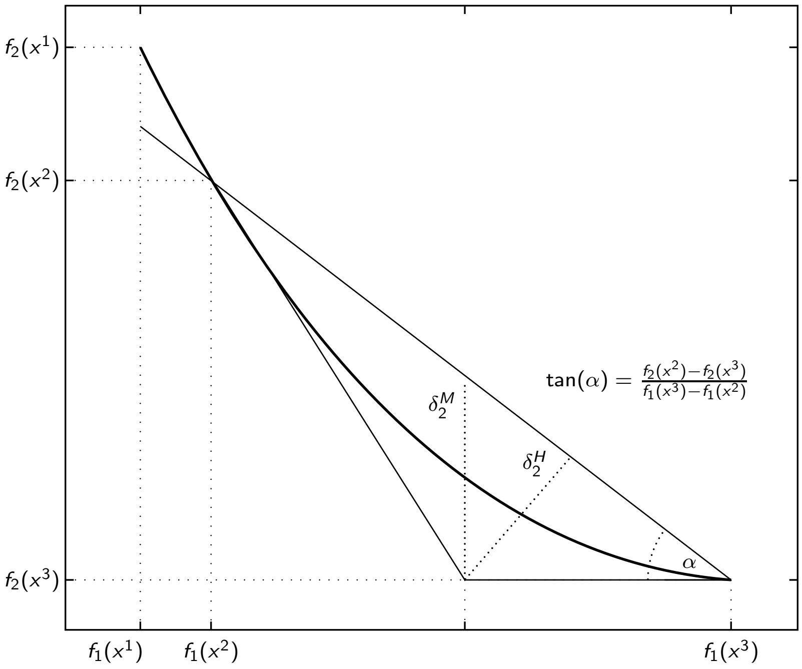

Figure 3. Illustration of error measures considered in Remark 1

Suppose that the efficient frontier of problem (3) is given as a convex function $f\colon [a,b] \to \mathbb{R}$ and the one-sided derivatives $f^+(a)$ and $f^-(b)$ have been evaluated. The next theorem based on Remark 3 and Theorem 1 from [9], Theorem 2 from [4] and Lemma 2 shows the quadratic convergence property of Algorithm 2.

THEOREM 1. *Let $L = b - a$ and let $\Delta = f^-(b) - f^+(a)$. The number M of quadratic optimization problems* (15) *which have to be solved in order to obtain the Hausdorff distance between upper and lower bound in Algorithm* 2 *smaller than or equal to ϵ satisfies the following inequality*

$$M \leq \max\left\{2\left\lceil\sqrt{\frac{L\Delta}{\epsilon}}\right\rceil - 1,\ 3\right\}. \tag{25}$$

As Yang and Goh [9] noticed the right directional derivative $f^+(a)$ may be close to $-\infty$, that is why using the fact that the Hausdorff distance is invariant under rotation it is better to consider the efficient frontier rotated by $\frac{\pi}{4}$ with the modified directional derivatives $\bar{f}^+(\bar{a})$ and $\bar{f}^-(\bar{b})$ and with $\bar{L} = \bar{b} - \bar{a}$ as the projective distance of the segment between points $(f_1(x^1), f_2(x^1))$ and $(f_1(x^3), f_2(x^3))$ onto the line $g(x) = -x$.

Also for Algorithm 1 we may find the upper bound for the number N of optimization problems (15) or (22) which have to be solved. First, let us formulate the following lemma.

LEMMA 2. *Consider the function $f\colon [a, b] \to \mathbb{R}$ and a constant $c \in (a, b)$ such that $f(a) \geq f(c) \geq f(c)$. Let $\Delta_1 = \frac{f(b)-f(c)}{b-c} - \frac{f(b)-f(a)}{c-a}$, $L_1 = c - a$, $\Delta_2 = -\frac{f(c)-f(a)}{c-a}$, $L_2 = b - c$, $\Delta = -\frac{f(b)-f(a)}{c-a}$ and $L = b - a$, then the following inequality is satisfied*

$$\Delta_1 L_1 + \Delta_2 L_2 \leq \Delta L. \tag{26}$$

An illustrative curve is shown in Fig. 4. We shall prove Lemma 2 in Appendix.

Note that Δ_1 and Δ_2 are the differences of the slopes of lower approximation functions computed according to Algorithm 1 for interval $[a, c]$ and $[c, b]$, respectively, and L_1 and L_2 are the lengths of these intervals.

The next theorem based on Lemma 5 and Theorem 2 from [4] together with Lemma 2 shows the linear convergence property of Algorithm 1.

THEOREM 2. *Let $(a, f(a))$, $(c, f(c))$ and $(b, f(b))$ are three initial points that are necessary to start Algorithm 1 and suppose that $(c, f(c))$ was chosen to satisfy the following inequality*

$$c \leq a + \epsilon \frac{b - a}{f(a) - f(b)}. \tag{27}$$

Let $L = b - a$ and let $\Delta = -f^+(a)$, then the number N of quadratic optimization problems ((15) *or* (22)*) to solve in order to make the Hausdorff distance between upper and lower bound in Algorithm* 1 *with the chord rule*

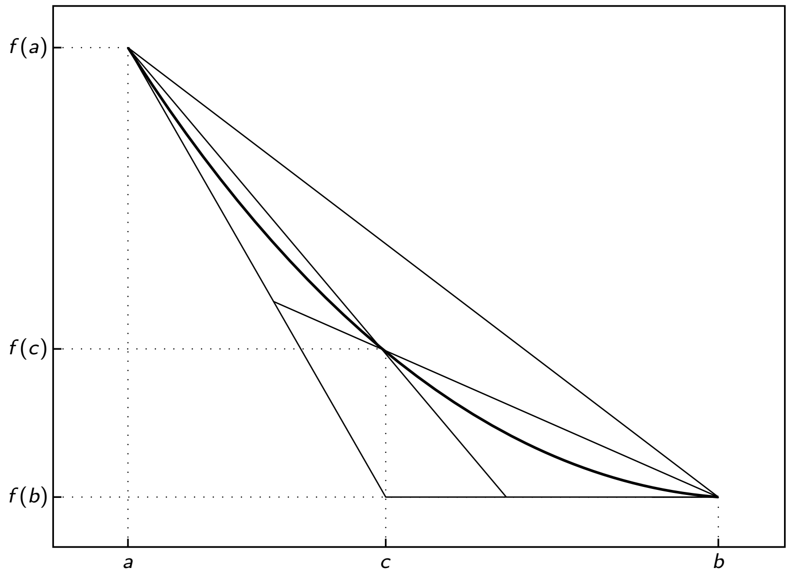

Figure 4. Illustration of functions considered in Lemma 2

or the maximum error rule smaller than or equal to ϵ satisfies the following inequality

$$N \le \max\left\{ \left\lceil \frac{L\Delta}{\epsilon} \right\rceil - 4,\ 0 \right\}. \tag{28}$$

We shall prove Theorem 2 in Appendix. Moreover, from Remark 1 it follows that $\delta^M \le \delta^H (1+\gamma)$, where

$$\gamma = \max\left\{ \frac{f(c)-f(a)}{a-c}, \frac{f(b)-f(c)}{c-b} \right\}.$$

That is why the number of additional steps of Algorithm 1 with the chord rule or the maximum error rule in order to make the maximum error between upper and lower bound smaller than or equal to ϵ satisfies the following inequality

$$N \le \max\left\{ \left\lceil \frac{L\Delta}{\epsilon} (1+\gamma) \right\rceil - 4,\ 0 \right\}. \tag{29}$$

It is clear that, if $f^+(a)$ is close to $-\infty$, then we can rotate f by $\frac{\pi}{4}$ as Yang and Goh [9] suggested and consider the modified directional derivative $\bar{f}^+(\bar{a})$ with $\bar{L} = \bar{b} - \bar{a}$.

6. Examples

In this section we give two numerical examples, which illustrate algorithms presented in Section 3. We compare the results of Algorithm 1 with the results of the method described in [7] and the results of Algorithm 2 with the results of Yang and Goh's algorithm, see [9].

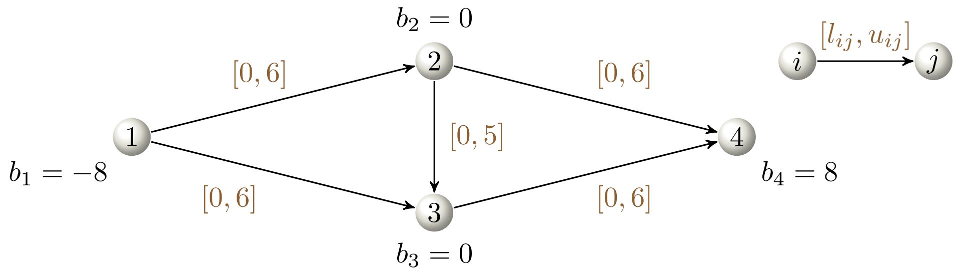

Figure 5. Network described in Example 1

EXAMPLE 1. Consider the network given in Fig. 5. Let $E[C_{12}] = 3$, $E[C_{13}] = 6$, $E[C_{23}] = 1$, $E[C_{24}] = 4$, $E[C_{34}] = 2$ and $E[C_{12}^2] = 75$, $E[C_{13}^2] = 36.11$, $E[C_{23}^2] = 20$, $E[C_{24}^2] = 16.31$, $E[C_{34}^2] = 4.5$. We are interested in solving the following problem

$$(30) \qquad \min \left[\begin{array}{c} 3x_{12} + 6x_{13} + x_{23} + 4x_{24} + 2x_{34}, \\ 75x_{12}^2 + 36.11x_{13}^2 + 20x_{23}^2 + 16.31x_{24}^2 + 4.5x_{34}^2 \\ +2(18x_{12}x_{13} + 3x_{12}x_{23} + 12x_{12}x_{24} + 6x_{12}x_{34} + 6x_{13}x_{23} \\ +24x_{13}x_{24} + 12x_{13}x_{34} + 4x_{23}x_{24} + 2x_{23}x_{34} + 8x_{24}x_{34}) \end{array} \right]$$

s.t.

$$2 \le x_{12} \le 6, \quad 2 \le x_{13} \le 6, \quad 0 \le x_{23} \le 3, \quad 2 \le x_{24} \le 6, \quad 2 \le x_{34} \le 6,$$

and

$$x_{12} + x_{13} = 8, \quad x_{12} = x_{23} + x_{24}, \quad x_{13} + x_{23} = x_{34}, \quad x_{24} + x_{34} = 8.$$

The vectors $x^1 = (6, 2, 3, 3, 5)$ and $x^3 = (2, 6, 0, 2, 6)$ are the lexicographical minima due to the first and the second objective of problem (30) and $P_1 = (f_1(x^1), f_2(x^1)) = (55, 5587.73)$ and $P_3 = (f_1(x^3), f_2(x^3)) = (62, 4131.2)$ are corresponding to these vectors points on the efficient frontier.

Let $P_2 = (57.5, 4615.349)$ be the third point necessary to start Algorithm 1. Table 1 includes the results of next evaluations of Algorithm 1, when new points are computed according to the chord rule in common with the Hausdorff

Table 1. The results of Algorithm 1 for Example 1

Step	$(f_1(x^*), f_2(x^*))$	δ^M	δ^H	δ^U
1		703.379	3.255	879.157
2	$(59.755, 4259.93)$	578.371	**1.487**	722.909
3	$(56, 4977.23)$	369.241	**1.428**	184.621
4	$(60.883, 4167.19)$	369.241	**0.679**	184.621
5	$(61.447, 4142.07)$	369.241	**0.605**	184.621
6	$(55.5, 5265.5)$	100.419	**0.567**	100.851
7	$(58.633, 4408.49)$	74.872	**0.31**	56.145
8	$(56.755, 4782.56)$	33.96	**0.309**	15.807
9	$(61.729, 4134.82)$	33.96	**0.295**	15.807
10	$(60.325, 4206.04)$	33.96	**0.183**	14.655
11	$(59.2, 4326.47)$	33.96	**0.136**	12.759
12	$(61.171, 4152.61)$	33.96	**0.13**	12.759
13	$(61.87, 4132.53)$	33.96	**0.123**	12.759
14	$(58.072, 4503.84)$	33.96	**0.089**	8.618
15	$(56.383, 4875.42)$	33.96	**0.087**	8.49

Table 2. The results of Algorithm 1 for Example 1

Step	$(f_1(x^*), f_2(x^*))$	δ^M	δ^H	δ^U
1		703.379	3.255	879.157
2	$(55.075, 5537.231)$	**347.13**	3.226	781.042
3	$(58.774, 4386.79)$	**288.905**	1.802	350.297
4	$(56.061, 4960.71)$	**142.771**	1.802	230.322
5	$(60.198, 4216.79)$	**74.042**	1.085	53.293
6	$(56.275, 4903.38)$	**68.093**	1.085	46.443
7	$(55.8462, 5096.9)$	**51.538**	1.085	46.443
8	$(60.9148, 4165.4)$	**37.874**	0.608	26.969
9	$(59.4037, 4300.42)$	**30.655**	0.607	19.52
10	$(58.051, 4507.64)$	**21.697**	0.607	**13.289**
11	$(56.951, 4736.04)$	**19.162**	0.607	**10.4**
12	$(59.813, 4253.86)$	**10.564**	0.326	**5.148**
13	$(58.423, 4442.48)$	**9.156**	0.326	**3.818**
14	$(60.571, 4187.21)$	**7.775**	0.326	**3.283**
15	$(56.667, 4804.02)$	**7.678**	0.326	**2.63**

measure. In Table 2 we present the results of Algorithm 1, when new points are computed according to the maximum error rule in common with the Maximum

Table 3. The results of Siem's algorithm for Example 1

Step	$(f_1(x^*), f_2(x^*))$	δ^M	δ^H	δ^U
1		703.379	3.255	879.157
2	$(56.25, 4909.95)$	383.169	2.446	592.06
3	$(55.625, 5224.74)$	263.138	2.446	592.06
4	$(59.75, 4260.49)$	98.683	1.43	111.018
5	$(58.625, 4409.73)$	73.284	1.275	82.445
6	$(60.875, 4167.65)$	55.232	0.683	34.52
7	$(56.875, 4753.95)$	48.212	0.683	15.86
8	$(55.313, 5416.65)$	48.121	0.683	15.86
9	$(55.938, 5046.09)$	36.516	0.683	15.86
10	$(55.469, 5319.04)$	28.195	0.683	15.86
11	$(60.313, 4207.02)$	24.671	0.604	13.877
12	$(58.063, 4505.49)$	24.556	0.604	11.895
13	$(56.094, 4951.68)$	21.146	0.604	11.895
14	$(59.188, 4328.06)$	19.578	0.603	11.013
15	$(61.438, 4142.38)$	8.479	0.313	3.774

error measure. To avoid the problem with the leftmost interval we have taken $P_4 = (55.075, 5537.231)$ as a forth point. Table 3 includes the results of the method described in [7], which uses the interval bisection method of the computing new points with the Maximum error measure. After each step of algorithms we present the new selected point and the maximum values of three error measures: the Maximum error, the Hausdorff distance and the Uncertainty area.

From the tables one can conclude that Algorithm 1 with the chord rule and the Hausdorff distance gives the smallest values of the Hausdorff measure (δ^H) in each step. Moreover, Algorithm 1 with the maximum error rule and the Maximum error gives smaller values of the Maximum error measure (δ^M) in each step than the algorithm described in Siem [7] which uses the interval bisection rule as the method of the selection of new points.

Example 2. We consider the network with 12 nodes and 17 arcs. The expected values of cost variables lie in the interval $[0, 2]$ and the second moments in the interval $[1, 3]$. Table 4 includes the comparison of Algorithm 2 with the method presented in [9]. After each step we present the values of the Hausdorff distance and the Maximum error and a new evaluated point.

As we can notice Algorithm 2 performs better in comparison with Yang and Goh's algorithm giving in each step smaller value of the Hausdorff distance between upper and lower approximation bounds.

Table 4. The results of next evaluations of Algorithm 2 and Yang and Goh's method for Example 2

Step	Algorithm 1			YG algorithm		
	$(f_1(x^*), f_2(x^*))$	δ^H	δ^M	$(f_1(x^*), f_2(x^*))$	δ^H	δ^M
1		1.507	458.259		1.507	458.259
2	$(63.410, 5259.94)$	**0.598**	117.196	$(63.410, 5259.94)$	28.567	117.196
3	$(64.194, 5176.09)$	**0.282**	117.196	$(62.824, 5551.23)$	27.968	36.287
4	$(64.733, 5153.34)$	**0.168**	117.196	$(62.534, 5782.65)$	0.598	36.287
5	$(62.824, 5551.23)$	**0.141**	29.355	$(64.194, 5176.09)$	10.113	28.735
6	$(65.013, 5147.54)$	**0.134**	29.355	$(63.112, 5379.36)$	6.998	14.123
7	$(63.584, 5227.26)$	**0.071**	29.355	$(62.968, 5458.16)$	5.596	14.123
8	$(65.1528, 5146.1)$	**0.058**	29.355	$(63.584, 5227.26)$	6.766	7.28
9	$(64.4381, 5163.53)$	**0.0578**	29.355	$(63.261, 5312.5)$	1.748	7.28
10	$(63.112, 5379.36)$	**0.032**	29.355	$(62.896, 5503.22)$	0.797	7.28

7. Conclusions

Two sandwich algorithms for approximation of the efficient frontier in the stochastic minimum cost flow problem with the moment multicriterion have been described.

The Simple Triangle Algorithm uses the lower bound proposed by Siem in [7] with two different partition rules. The presented example shows that this modification causes faster decrease of the Maximum error measure and the Hausdorff distance measure and, as a result, reduces the number of steps of algorithm in comparison to the Siem's method.

The Trapezium Algorithm performs better than all of the mentioned derivative free algorithms (Siem's method, Yang and Goh's method) giving in each step the smallest value of the Hausdorff distance between lower and upper bound. A quadratic and linear convergence property have been obtained for Algorithm 2 and Algorithm 1, respectively.

For further research we are interested in constructing an exact algorithm for solving the stochastic minimum cost flow problem with the moment multicriterion. Since the method for finding an analytic exact efficient solution set for bicriteria quadratic problems proposed by Yang and Goh [8] requires the second moment cost matrix to be positive definite, an open problem is to find an exact algorithm for the case when the second moment cost matrix is semi positive definite.

Appendix

We present the proofs of Lemma 1, Lemma 2 and Theorem 2.

PROOF OF LEMMA 1. Let $P_k = \big(\sum_{(i,j)\in A} c_{ij}x_{ij}^k,\, E\big[\big(\sum_{(i,j)\in A} C_{ij}x_{ij}^k\big)^2\big]\big)$ for $k = 1, 2$ be two given points on the efficient frontier of problem (1) and let $\lambda \in [0, 1]$. If $\sum_{(i,j)\in A} c_{ij}x_{ij}^1 = \sum_{(i,j)\in A} c_{ij}x_{ij}^2$ then for all b such that

$$E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}^1\Big)^2\Big] \geq b \geq E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}^2\Big)^2\Big]$$

the point $\big(\sum_{(i,j)\in A} c_{ij}x_{ij}^1,\, b\big)$ lies also on the efficient frontier of problem (1).

Suppose now that $\sum_{(i,j)\in A} c_{ij}x_{ij}^1 \neq \sum_{(i,j)\in A} c_{ij}x_{ij}^2$. Note that for

$$C = \Big\{x \in \mathcal{X} : \sum_{(i,j)\in A} c_{ij}x_{ij} = \lambda \sum_{(i,j)\in A} c_{ij}x_{ij}^1 + (1-\lambda) \sum_{(i,j)\in A} c_{ij}x_{ij}^2\Big\},$$

$$B_1 = \Big\{y \in \mathcal{X} : \sum_{(i,j)\in A} c_{ij}y_{ij} = \sum_{(i,j)\in A} c_{ij}x_{ij}^1\Big\},$$

and

$$B_2 = \Big\{z \in \mathcal{X} : \sum_{(i,j)\in A} c_{ij}z_{ij} = \sum_{(i,j)\in A} c_{ij}x_{ij}^2\Big\},$$

we have $\lambda B_1 + (1-\lambda)B_2 \subseteq C$, what yields

$$\min_{x\in C} E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}\Big)^2\Big] \leq \min_{y\in B_1, z\in B_2} E\Big[\Big(\lambda \sum_{(i,j)\in A} C_{ij}y_{ij} + (1-\lambda) \sum_{(i,j)\in A} C_{ij}z_{ij}\Big)^2\Big].$$

Due to the convexity of function $f(a) = a^2$ and properties of the expected value we have

$$E\Big[\Big(\lambda \sum_{(i,j)\in A} C_{ij}y_{ij} + (1-\lambda) \sum_{(i,j)\in A} C_{ij}z_{ij}\Big)^2\Big] \leq E\Big[\lambda\Big(\sum_{(i,j)\in A} C_{ij}y_{ij}\Big)^2 + (1-\lambda)\Big(\sum_{(i,j)\in A} C_{ij}z_{ij}\Big)^2\Big]$$

for $y \in B_1$, $z \in B_2$. Finally, we have

$$\begin{aligned}
&\min_{x\in C} E\Big[\Big(\sum_{(i,j)\in A} C_{ij}x_{ij}\Big)^2\Big] \\
&\qquad \le \min_{y\in B_1, z\in B_2} E\Big[\lambda\Big(\sum_{(i,j)\in A} C_{ij}y_{ij}\Big)^2 + (1-\lambda)\Big(\sum_{(i,j)\in A} C_{ij}z_{ij}\Big)^2\Big] \\
&\qquad = \lambda \min_{y\in B_1} E\Big[\Big(\sum_{(i,j)\in A} C_{ij}y_{ij}\Big)^2\Big] + (1-\lambda)\min_{z\in B_2} E\Big[\Big(\sum_{(i,j)\in A} C_{ij}z_{ij}\Big)^2\Big],
\end{aligned}$$

which shows the convexity of the efficient frontier of problem (1). □

Proof of Lemma 2. If we denote $c = \lambda a + (1-\lambda)b$, where $\lambda \in [0,1]$, then we have

$$\begin{aligned}
\Delta_1 &= \frac{f(b)-f(c)}{\lambda(b-a)} - \frac{f(b)-f(a)}{(1-\lambda)(b-a)}, \quad L_1 = (1-\lambda)(b-a),\\
\Delta_2 &= -\frac{f(c)-f(a)}{(1-\lambda)(b-a)}, \quad L_2 = \lambda(b-a),\\
\Delta &= -\frac{f(b)-f(a)}{(1-\lambda)(b-a)}, \quad \text{and} \quad L = b-a.
\end{aligned}$$

It is easy to show that inequality (26) is equivalent to

$$\left(2\lambda^2 - 2\lambda + 1\right)\left(f(c) - f(b)\right) \ge 0.$$

Using the fact that $f(c) \ge f(b)$ we prove the lemma. □

Proof of Theorem 2. Suppose that we have found point $(c, f(c))$ with property (27). Of course it is possible to find such a point by solving quadratic programming problem with additional constraint similar to problem (6). From condition (27) it follows that $\delta_1^M \le \epsilon$ and $\delta_1^H \le \epsilon$.

Now, if we consider the interval $[c,b]$, then we have $c-b \le L$ and $\frac{f(c)-(b)}{b-c} \le \Delta$. Similar to [4] we prove the theorem by induction on number $N = N\left(\frac{L\Delta}{\epsilon}\right)$.

The induction basis, $N\left(\frac{L\Delta}{\epsilon}\right) = 0$, is equivalent to Lemma 1 from [1], which also holds for lower approximation function built according to definition (11).

Suppose that $N \ge 1$. If after one step of Algorithm 1 the error $\delta \le \epsilon$, then we had only one additional evaluation and the assertion is true.

In the other case, let $d \in [c,b]$ be the new computed point and let $L_1 = d-c$ and $L_2 = b-d$ let Δ_1 and Δ_2 denote the slope differences of the linear functions building the lower bounds in the interval $[c,d]$ and $[d,b]$, respectively.

We can assume without lost of generality that the error δ exceeds ϵ in the right subinterval. Lemma 5 and Lemma 1 from [4] used for the lower approximation bounds built according to definition (12) give the following inequalities

$$\frac{L_1\Delta_1}{\epsilon} > 1 \tag{31}$$

and

$$\frac{L_2\Delta_2}{\epsilon} > 4. \tag{32}$$

Lemma 2 and inequality (31) gives

$$\frac{L_2\Delta_2}{\epsilon} < \frac{L\Delta}{\epsilon}. \tag{33}$$

Similarly, Lemma 2 and inequality (32) gives

$$\frac{L_1\Delta_1}{\epsilon} < \frac{L\Delta}{\epsilon}. \tag{34}$$

From (33) and (34) we have

$$N\Big(\frac{L_1\Delta_1}{\epsilon}\Big) < N\Big(\frac{L\Delta}{\epsilon}\Big) \quad \text{and} \quad N\Big(\frac{L_2\Delta_2}{\epsilon}\Big) < N\Big(\frac{L\Delta}{\epsilon}\Big).$$

Now the induction hypothesis can be applied for $N\big(\frac{L_1\Delta_1}{\epsilon}\big)$ and $N\big(\frac{L_2\Delta_2}{\epsilon}\big)$. If $N\big(\frac{L_1\Delta_1}{\epsilon}\big) = 0$, then the theorem's statement follows directly.

Otherwise, from Lemma 2 we have

$$\begin{aligned}
&\max_{\substack{\Delta_1L_1+\Delta_2L_2\le\Delta L,\\ L_1+L_2=L,\ L_1\Delta_1>\epsilon,\ L_2\Delta_2>4\epsilon}} \Big(1+N\Big(\frac{L_1\Delta_1}{\epsilon}\Big)+N\Big(\frac{L_2\Delta_2}{\epsilon}\Big)\Big)\\
&\quad= \max_{\substack{\Delta_1L_1+\Delta_2L_2\le\Delta L,\\ L_1+L_2=L,\ L_1\Delta_1>\epsilon,\ L_2\Delta_2>4\epsilon}} \Big(1+\Big\lceil\frac{L_1\Delta_1}{\epsilon}\Big\rceil+\Big\lceil\frac{L_2\Delta_2}{\epsilon}\Big\rceil-7\Big)\\
&\quad\quad\le \Big\lceil\frac{L\Delta}{\epsilon}\Big\rceil-6 \le \Big\lceil\frac{L\Delta}{\epsilon}\Big\rceil = N\left(\frac{L\Delta}{\epsilon}\right),
\end{aligned}$$

which completes the proof. □

References

[1] Burkard R.E., Hamacher H.W., Rote G., *Sandwich approximation of univariate convex functions with an applications to separable convex programming*, Naval Res. Logist. **38** (1991), 911–924.

[2] Fruhwirth B., Burkard R.E., Rote G., *Approximation of convex curves with application to the bi-criteria minimum cost flow problem*, European J. Oper. Res. **42** (1989), 326–338.

[3] Goldfarb D., Idnani A., *A numerically stable dual method for solving strictly convex quadratic programs*, Math. Program. **27** (1983), 1–33.

[4] Rote G., *The convergence rate of the sandwich algorithm for approximating convex functions*, Computing **48** (1992), 337–361.

[5] Ruhe G., *Complexity results for multicriterial and parametric network flows using a pathological graph of Zadeh*, Z. Oper. Res. **32** (1988), 9–27.

[6] Sedeno-Noda A., Gonzalez-Martin C., *The biobjective minimum cost flow problem*, European J. Oper. Res. **124** (2000), 591–600.

[7] Siem A.Y.D., den Hertog D., Hoffmann A.L., *A method for approximating univariate convex functions using only function value evaluations*, CentER Discussion Paper **67** (2007), 1–26.

[8] Yang X.Q., Goh C.J., *Analytic efficient solution set for multi-criteria quadratic programs*, European J. Oper. Res. **92** (1996), 166–181.

[9] Yang X.Q., Goh C.J., *A method for convex curve approximation*, European J. Oper. Res. **97** (1997), 205–212.

[10] Zadeh N., *A bad network for the simplex method and other minimum cost flow algorithms*, Math. Program. **5** (1973), 255–266.

Institute of Mathematics
Silesian University
Bankowa 14
40-007 Katowice
Poland
e-mail: marta.kostrzewska83@gmail.com

Institute of Physics
Silesian University
Uniwersytecka 4
40-007 Katowice
Poland
and
Faculty of Mathematics
and Natural Sciences
Cardinal Stefan Wyszyński
University in Warsaw
01-938 Warszawa
Poland
e-mail: leslawsocha@poczta.onet.pl