www.dergipark.gov.tr
ISSN:2148-3736
El-Cezerî Fen ve Mühendislik Dergisi
Cilt: 6, No: 3, 2019 (466-481)

El-Cezerî Journal of Science and Engineering
Vol: 6, No: 3, 2019 (466-481)
DOI:10.31202/ecjse.551487

ECJSE

Research Paper / Makale

A Generalisation of Fuzzy Soft Max-Min Decision-Making Method and Its Application to a Performance-Based Value Assignment in Image Denoising

Serdar ENGINOĞLU[1,*], Samet MEMIŞ[1], Naim ÇAĞMAN[2]

[1]Department of Mathematics, Faculty of Arts and Sciences, Çanakkale Onsekiz Mart University, Çanakkale/Turkey
[2]Department of Mathematics, Faculty of Arts and Sciences, Tokat Gaziosmanpaşa University, Tokat/Turkey
serdarenginoglu@gmail.com

Received/Geliş: 09.04.2019 **Accepted/Kabul:** 12.06.2019

Abstract: Latterly, the fuzzy soft max-min decision-making method denoted by FSMmDM and provided in [Çağman, N., Enginoğlu, S., Fuzzy soft matrix theory and its application in decision making, Iranian Journal of Fuzzy Systems, 2012, 9(1), 109-119] has been configured via fuzzy parameterized fuzzy soft matrices ($fpfs$-matrices) by Enginoğlu and Memiş [A configuration of some soft decision-making algorithms via $fpfs$-matrices, Cumhuriyet Science Journal, 2018, 39(4), 871-881], faithfully to the original. Although this configured method denoted by CE12 and constructed by and-product/or-product (CE12a/CE12o) is useful in decision-making, the method should be made more attractive in terms of time and complexity in the event that a large amount of data is processed. In this paper, we propose two algorithms denoted by EMC19a and EMC19o and being new generalisations of FSMmDM. Moreover, we prove that EMC19a accept CE12a as a special case in the event that the first rows of the $fpfs$-matrices are binary. Afterwards, we compare the running times of these algorithms. The results show that EMC19a and EMC19o outperform CE12a and CE12o, respectively, in any number of data. We then apply EMC19o to a decision-making problem in image denoising. Finally, we discuss the need for further research.

Keywords: Fuzzy sets, soft sets, soft decision-making, soft matrices, $fpfs$-matrices

Bulanık Esnek Maks-Min Karar Verme Metodunun Bir Genelleştirmesi ve Gürültü Kaldırmada Performans Temelli Değer Atamaya Uygulaması

Öz: Son zamanlarda, FSMmDM ile gösterilen ve [Çağman, N., Enginoğlu, S., Fuzzy soft matrix theory and its application in decision making, Iranian Journal of Fuzzy Systems, 2012, 9(1), 109-119] çalışmasında verilen bulanık esnek maks-min karar verme metodu, Enginoğlu ve Memiş [A configuration of some soft decision-making algorithms via $fpfs$-matrices, Cumhuriyet Science Journal, 2018, 39(4), 871-881] tarafından bulanık parametreli bulanık esnek matrisler ($fpfs$-matrisler) yoluyla orijinaline sadık kalacak biçimde yapılandırıldı. CE12 ile gösterilen ve ve-çarpım/veya-çarpım (CE12a/CE12o) yoluyla inşa edilen bu yapılandırılmış metot karar vermede kullanışlı olmasına rağmen, yüksek sayıda veri işlenirken zaman ve karmaşıklık bakımından daha cazip hale getirilmesi gerekmektedir. Bu çalışmada, FSMmDM'nin genelleştirmeleri olan ve EMC19a ve EMC19o ile gösterilen iki algoritma öneriyoruz. Ayrıca, EMC19a'nın $fpfs$-matrislerin ilk satırlarındaki bileşenlerin 0 ya da 1 olduğunda CE12a'yı özel bir durum olarak kabul ettiğini gösteriyoruz. Ardından, bu algoritmaların çalışma sürelerini karşılaştırıyoruz. Sonuçlar herhangi bir veri sayısında EMC19a ve EMC19o'nun sırasıyla CE12a ve CE12o'dan daha iyi bir performans sergilediğini göstermektedir. Daha sonra, EMC19o'yu gürültü kaldırmada bir karar verme problemine uyguluyoruz. Son olarak, sonraki çalışmalar hakkında bir tartışmaya yer veriyoruz.

Anahtar Kelimeler: Bulanık kümeler, esnek kümeler, esnek karar verme, esnek matrisler, $fpfs$-matrisler

How to cite this article
Enginoğlu, S., Memiş, S., Çağman, N., "*A generalisation of fuzzy soft max-min decision-making method and its application to a performance-based value assignment in image denoising*", El-Cezerî Journal of Science and Engineering, 2019, 6(3), 466-481.

Bu makaleye atıf yapmak için
Enginoğlu, S., Memiş, S., Çağman, N., "*Bulanık esnek maks-min karar verme metodunun bir genellemesi ve gürültü kaldırmada performans temelli değer atamaya uygulaması*", El-Cezerî Fen ve Mühendislik Dergisi 2019, 6(3), 466-481.

1. Introduction

Soft sets [1] are designed to cope with uncertainties, and so far, many applied and theoretical studies have been conducted on that [2–33]. Recently, some soft decision-making algorithms have been configured via fuzzy parameterized fuzzy soft matrices ($fpfs$-matrices) by Enginoğlu and Memiş [34], faithfully to the original. Moreover, the authors have remarked that studies on the simplifications and different configurations of these methods therein are worth doing. In the recent time, several soft decision-making algorithms given in [34] have been configured and simplified [35–39] to apply them to a decision-making problem in computer science such as image denoising and machine learning.

The soft max-min decision-making method SMmDM and the fuzzy soft max-min decision-making method FSMmDM provided in [5,12] and configured in [34] have a similar disadvantage considered in [35–39]. Therefore, in this paper, we have focused on improving two new methods being a different generalisation of them and free of the disadvantages mentioned above.

In Section 2, we present the concept of $fpfs$-matrices [13,40] and give CE12 constructed by and-product/or-product (CE12a/CE12o) [5,12,34]. In Section 3, we propound two new methods, namely EMC19a and EMC19o, and prove that EMC19a equivalent to CE12a under the condition that first rows of the $fpfs$-matrices are binary. In Section 4, we compare the running times of these algorithms. In Section 5, we apply EMC19o to a decision-making problem in which the noise removal/image denoising methods can be ordered in terms of performance. Finally, we discuss the need for further research.

2. Preliminaries

In this section, firstly, the concept of $fpfs$-matrices [13,40] and some of its basic definitions have been presented. Throughout this paper, let E be a parameter set, $F(E)$ be the set of all fuzzy sets over E, and $\mu \in F(E)$. Here, a fuzzy set is denoted by $\left\{{}^{\mu(x)}x : x \in E\right\}$ instead of $\left\{\left(x, \mu(x)\right) : x \in E\right\}$.

Definition 2.1. [7,13] *Let U be a universal set, $\mu \in F(E)$, and α be a function from μ to $F(U)$. Then, the set $\{({}^{\mu(x)}x, \alpha({}^{\mu(x)}x)) : x \in E\}$ being the graphic of α is called a fuzzy parameterized fuzzy soft set ($fpfs$-set) parameterized via E over U (or briefly over U).*

In the present paper, the set of all $fpfs$-sets over U is denoted by $FPFS_E(U)$. In $FPFS_E(U)$, since the $graph(\alpha)$ and α generated each other uniquely, the notations are interchangeable. Therefore, as long as it does not cause any confusion, we denote an $fpfs$-set $graph(\alpha)$ by α.

Example 2.1. *Let $E = \{x_1, x_2, x_3, x_4\}$ and $U = \{u_1, u_2, u_3, u_4, u_5\}$. Then,*

$$\alpha = \left\{\left({}^{0}x_1, \left\{{}^{0.3}u_1, {}^{0.7}u_4\right\}\right), \left({}^{0.5}x_2, \left\{{}^{0.4}u_2, {}^{0.8}u_3, {}^{1}u_5\right\}\right), \left({}^{0.7}x_3, \left\{{}^{0.4}u_1, {}^{0.7}u_3, {}^{0.8}u_4\right\}\right), \left({}^{1}x_4, \left\{{}^{0.9}u_3, {}^{0.6}u_5\right\}\right)\right\}$$

is an $fpfs$-set over U.

Definition 2.2. [13,40] *Let $\alpha \in FPFS_E(U)$. Then, $[a_{ij}]$ is called the matrix representation of α (or briefly $fpfs$-matrix of α) and is defined by*

$$[a_{ij}] \coloneqq \begin{bmatrix} a_{01} & a_{02} & a_{03} & \cdots & a_{0n} & \cdots \\ a_{11} & a_{12} & a_{13} & \cdots & a_{1n} & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \cdots & a_{mn} & \cdots \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \end{bmatrix} \quad \textit{for } i \in \{0,1,2,\cdots\} \textit{ and } j \in \{1,2,\cdots\}$$

such that

$$a_{ij} \coloneqq \begin{cases} \mu(x_j), & i = 0 \\ \alpha({}^{\mu(x_j)}x_j)(u_i), & i \neq 0 \end{cases}$$

Here, if $|U| = m - 1$ and $|E| = n$, then $[a_{ij}]$ has order $m \times n$.

From now on, the set of all $fpfs$-matrices parameterized via E over U is denoted by $FPFS_E[U]$.

Example 2.2. *Let's consider the $fpfs$-set α provided in Example 2.1. Then, the $fpfs$-matrix of α is as follows:*

$$[a_{ij}] = \begin{bmatrix} 0 & 0.5 & 0.7 & 1 \\ 0.3 & 0 & 0.4 & 0 \\ 0 & 0.4 & 0 & 0 \\ 0 & 0.8 & 0.7 & 0.9 \\ 0.7 & 0 & 0.8 & 0 \\ 0 & 1 & 0 & 0.6 \end{bmatrix}$$

Definition 2.3. [13,40] *Let $[a_{ij}] \in FPFS_E[U]$. For all i and j, if $a_{ij} = \lambda$, then $[a_{ij}]$ is called λ-$fpfs$-matrix and is denoted by $[\lambda]$. Here, $[0]$ is called empty $fpfs$-matrix and $[1]$ is called universal $fpfs$-matrix.*

Definition 2.4. [13,40] *Let $[a_{ij}], [b_{ij}], [c_{ij}] \in FPFS_E[U]$, $I_E \coloneqq \{ j : x_j \in E\}$, and $R \subseteq I_E$. If*

$$c_{ij} \coloneqq \begin{cases} a_{ij}, & j \in R \\ b_{ij}, & j \in I_E \setminus R \end{cases}$$

then $[c_{ij}]$ is called Rb-restriction of $[a_{ij}]$ and is denoted by $[(a_{Rb})_{ij}]$. Briefly, if $[b_{ij}] = [0]$, then $[(a_R)_{ij}]$ can be used instead of $[(a_{R0})_{ij}]$. It is clear that

$$(a_R)_{ij} = \begin{cases} a_{ij}, & j \in R \\ 0, & j \in I_E \setminus R \end{cases}$$

Definition 2.5. [13,40] *Let $[a_{ij}], [b_{ij}] \in FPFS_E[U]$. For all i and j,*

If $a_{ij} \leq b_{ij}$, then $[a_{ij}]$ is called a submatrix of $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\subseteq} [b_{ij}]$,

If $a_{ij} = b_{ij}$, then $[a_{ij}]$ and $[b_{ij}]$ are called equal $fpfs$-matrices and is denoted by $[a_{ij}] = [b_{ij}]$.

Definition 2.6. [13,40] *Let $[a_{ij}], [b_{ij}], [c_{ij}] \in FPFS_E[U]$. For all i and j,*

If $c_{ij} \coloneqq max\{a_{ij}, b_{ij}\}$, then $[c_{ij}]$ is called union of $[a_{ij}]$ and $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\cup} [b_{ij}]$,

If $c_{ij} \coloneqq min\{a_{ij}, b_{ij}\}$, then $[c_{ij}]$ is called intersection of $[a_{ij}]$ and $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\cap} [b_{ij}]$,

If $c_{ij} \coloneqq max\{0, a_{ij} - b_{ij}\}$, then $[c_{ij}]$ is called difference between $[a_{ij}]$ and $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\setminus} [b_{ij}]$,

If $c_{ij} \coloneqq |a_{ij} - b_{ij}|$, then $[c_{ij}]$ is called symmetric difference between $[a_{ij}]$ and $[b_{ij}]$ and is denoted by $[a_{ij}] \tilde{\Delta} [b_{ij}]$.

Definition 2.7. [13,40] *Let $[a_{ij}], [b_{ij}] \in FPFS_E[U]$. For all i and j, if $b_{ij} \coloneqq 1 - a_{ij}$, then $[b_{ij}]$ is complement of $[a_{ij}]$ and is denoted by $[a_{ij}]^{\tilde{c}}$ or $[a_{ij}^{\tilde{c}}]$.*

Definition 2.8. [13,40] *Let* $[a_{ij}], [b_{ij}] \in FPFS_E[U]$. *If* $[a_{ij}] \,\tilde{\cap}\, [b_{ij}] = [0]$, *then* $[a_{ij}]$ *and* $[b_{ij}]$ *are called disjoint.*

Definition 2.9. [40] *Let* $[a_{ij}]_{m\times n_1} \in FPFS_{E_1}[U]$, $[b_{ik}]_{m\times n_2} \in FPFS_{E_2}[U]$, *and* $[c_{ip}]_{m\times n_1 n_2} \in FPFS_{E_1\times E_2}[U]$ *such that* $p = n_2(j-1)+k$. *For all* i *and* p,

If $c_{ip} \coloneqq min\{a_{ij}, b_{ik}\}$, *then* $[c_{ip}]$ *is called and-product of* $[a_{ij}]$ *and* $[b_{ik}]$ *and is denoted by* $[a_{ij}] \wedge [b_{ik}]$,

If $c_{ip} \coloneqq max\{a_{ij}, b_{ik}\}$, *then* $[c_{ip}]$ *is called or-product of* $[a_{ij}]$ *and* $[b_{ik}]$ *and is denoted by* $[a_{ij}] \vee [b_{ik}]$,

If $c_{ip} \coloneqq min\{a_{ij}, 1-b_{ik}\}$, *then* $[c_{ip}]$ *is called andnot-product of* $[a_{ij}]$ *and* $[b_{ik}]$ *and is denoted by* $[a_{ij}] \,\overline{\wedge}\, [b_{ik}]$,

If $c_{ip} \coloneqq max\{a_{ij}, 1-b_{ik}\}$, *then* $[c_{ip}]$ *is called ornot-product of* $[a_{ij}]$ *and* $[b_{ik}]$ *and is denoted by* $[a_{ij}] \,\underline{\vee}\, [b_{ik}]$.

Secondly, we present the algorithm CE12 [5,12,34].

Step 1. Construct two $fpfs$-matrices $[a_{ij}]$ and $[b_{ik}]$

Step 2. Find and-product/or-product $fpfs$-matrix $[c_{ip}]$ of $[a_{ij}]$ and $[b_{ik}]$

Step 3. Obtain $[s_{i1}]$ defined by

$$s_{i1} \coloneqq \max_k \begin{cases} \min\limits_{p\in I_k}\{c_{0p}c_{ip}\}, & I_k \neq \emptyset \\ 0, & I_k = \emptyset \end{cases}$$

such that $i \in \{1,2,\dots,m-1\}$ and $I_k \coloneqq \{p \mid \exists i, c_{0p}c_{ip} \neq 0, (k-1)n < p \leq kn\}$

Step 4. Obtain the set $\{u_k \mid s_{k1} = \max\limits_i (s_{i1})\}$

Preferably, the set $\{{}^{s_{i1}}u_i | u_i \in U\}$ or $\{{}^{\mu(u_k)}u_k | u_k \in U\}$ can be attained such that $\mu(u_k) = \frac{s_{k1}}{\max_i s_{i1}}$.

3. Soft Decision-Making Methods: EMC19a and EMC19o

In this section, we first propose an algorithm denoted by EMC19a.

Step 1. Construct two $fpfs$-matrices $[a_{ij}]$ and $[b_{ik}]$

Step 2. Obtain score matrix $[s_{i1}]$ defined by

$$s_{i1} \coloneqq \begin{cases} \min\left\{\max\limits_{j\in I_a}\{a_{0j}a_{ij}\}, \min\limits_{k\in I_b}\{b_{0k}b_{ik}\}\right\}, & I_a \neq \emptyset \text{ and } I_b \neq \emptyset \\ 0, & \text{otherwise} \end{cases}$$

such that $i \in \{1,2,\dots,m-1\}$, $I_a \coloneqq \{j \mid \exists i, a_{0j}a_{ij} \neq 0\}$, and $I_b \coloneqq \{k \mid \exists i, b_{0k}b_{ik} \neq 0\}$

Step 3. Obtain the decision set $\{{}^{s_{i1}}u_i | u_i \in U\}$

It is clear that the values s_{i1} give a ranking order over u_i. Therefore, the decision maker can choose the proper ones of the alternatives.

Theorem 4.1. *EMC19a is equivalent to CE12a under the condition that first rows of the* $fpfs$-*matrices are binary.*

Proof. Let us consider the functions s_{i1} provided in CE12a and EMC19a, and show $I_k \neq \emptyset \Leftrightarrow (I_a \neq \emptyset \wedge I_b \neq \emptyset)$. Then,

$$\begin{aligned} I_k \neq \emptyset &\Leftrightarrow \exists i, c_{0p}c_{ip} \neq 0 \\ &\Leftrightarrow \exists i, (c_{0p} \neq 0 \wedge c_{ip} \neq 0) \end{aligned}$$

$$\begin{aligned}
&\Leftrightarrow \ \exists i, (\min\{a_{0j}, b_{0k}\} \neq 0 \wedge \min\{a_{ij}, b_{ik}\} \neq 0)\\
&\Leftrightarrow \ \exists i, (a_{0j} \neq 0 \wedge b_{0k} \neq 0 \wedge a_{ij} \neq 0 \wedge b_{ik} \neq 0)\\
&\Leftrightarrow \ \exists i, (a_{0j}a_{ij} \neq 0 \wedge b_{0k}b_{ik} \neq 0)\\
&\Leftrightarrow \ I_a \neq \emptyset \wedge I_b \neq \emptyset
\end{aligned}$$

In a similar way,

$$\begin{aligned}
I_k = \emptyset &\Leftrightarrow \ \forall i, c_{0p}c_{ip} = 0\\
&\Leftrightarrow \ \forall i, (c_{0p} = 0 \vee c_{ip} = 0)\\
&\Leftrightarrow \ \forall i, (\min\{a_{0j}, b_{0k}\} = 0 \vee \min\{a_{ij}, b_{ik}\} = 0)\\
&\Leftrightarrow \ \forall i, (a_{0j} = 0 \vee b_{0k} = 0 \vee a_{ij} = 0 \vee b_{ik} = 0)\\
&\Leftrightarrow \ \forall i, (a_{0j}a_{ij} = 0 \vee b_{0k}b_{ik} = 0)\\
&\Leftrightarrow \ I_a = \emptyset \vee I_b = \emptyset
\end{aligned}$$

Here, $i \in \{1,2,\dots,m-1\}$, $j,k \in \{1,2,\dots,n\}$, $p = n(j-1)+k$, and $(k-1)n < p \leq kn$.

Suppose that first rows of the $fpfs$-matrices are binary, $I_k = \{t_1^k, t_2^k, \dots, t_{r(k)}^k\}$, $I_a = \{a_1, a_2, \dots, a_s\}$, and $I_b = \{b_1, b_2, \dots, b_t\}$. The functions s_{i1} provided in CE12a and EMC19a are equal in the event that $I_a = \emptyset$ or $I_b = \emptyset$. Assume that $I_a \neq \emptyset$ and $I_b \neq \emptyset$. Since $a_{0j} = 1$ and $b_{0k} = 1$, for all $j \in I_a$ and $k \in I_b$,

$$\begin{aligned}
\max_k\left\{\min_{p\in I_k}(c_{0p}c_{ip})\right\} = \ &\max\left\{\min\left\{c_{0t_1^1}\, c_{it_1^1}, c_{0t_2^1}\, c_{it_2^1}, \dots, c_{0t_{r(1)}^1}\, c_{it_{r(1)}^1}\right\},\right.\\
&\quad \min\left\{c_{0t_1^2}\, c_{it_1^2}, c_{0t_2^2}\, c_{it_2^2}, \dots, c_{0t_{r(2)}^2}\, c_{it_{r(2)}^2}\right\}, \dots,\\
&\quad \left.\min\left\{c_{0t_1^k}\, c_{it_1^k}, c_{0t_2^k}\, c_{it_2^k}, \dots, c_{0t_{r(k)}^k}\, c_{it_{r(k)}^k}\right\}\right\}\\
= \ &\max\left\{\min\left\{c_{it_1^1}, c_{it_2^1}, \dots, c_{it_{r(1)}^1}\right\},\right.\\
&\quad \min\left\{c_{it_1^2}, c_{it_2^2}, \dots, c_{it_{r(2)}^2}\right\}, \dots,\\
&\quad \left.\min\left\{c_{it_1^k}, c_{it_2^k}, \dots, c_{it_{r(k)}^k}\right\}\right\}\\
= \ &\max\left\{\min\left\{\min\{a_{ia_1}, b_{ib_1}\}, \min\{a_{ia_1}, b_{ib_2}\}, \dots, \min\{a_{ia_1}, b_{ib_t}\}\right\},\right.\\
&\quad \min\left\{\min\{a_{ia_2}, b_{ib_1}\}, \min\{a_{ia_2}, b_{ib_2}\}, \dots, \min\{a_{ia_2}, b_{ib_t}\}\right\}, \dots,\\
&\quad \left.\min\left\{\min\{a_{ia_s}, b_{ib_1}\}, \min\{a_{ia_s}, b_{ib_2}\}, \dots, \min\{a_{ia_s}, b_{ib_t}\}\right\}\right\}\\
= \ &\max\left\{\min\left\{a_{ia_1}, \min\{b_{ib_1}, b_{ib_2}, \dots, b_{ib_t}\}\right\},\right.\\
&\quad \min\left\{a_{ia_2}, \min\{b_{ib_1}, b_{ib_2}, \dots, b_{ib_t}\}\right\}, \dots,\\
&\quad \left.\min\left\{a_{ia_s}, \min\{b_{ib_1}, b_{ib_2}, \dots, b_{ib_t}\}\right\}\right\}\\
= \ &\min\left\{\max\{a_{ia_1}, a_{ia_2}, \dots, a_{ia_s}\}, \min\{b_{ib_1}, b_{ib_2}, \dots, b_{ib_t}\}\right\}\\
= \ &\min\left\{\max_{j\in I_a} a_{ij}, \min_{k\in I_b} b_{ik}\right\}
\end{aligned}$$

$$= \min\left\{\max_{j\in I_a}\{a_{0j}a_{ij}\}, \min_{k\in I_b}\{b_{0k}b_{ik}\}\right\}$$

QED

Secondly, we propose another algorithm denoted by EMC19o.

Step 1. Construct two $fpfs$-matrices $[a_{ij}]$ and $[b_{ik}]$

Step 2. Obtain score matrix $[s_{i1}]$ defined by

$$s_{i1} := \begin{cases} \max\left\{\max_{j\in I_a}\{a_{0j}a_{ij}\}, \min_{k\in I_b}\{b_{0k}b_{ik}\}\right\}, & I_a \neq \emptyset \text{ and } I_b \neq \emptyset \\ 0, & \text{otherwise} \end{cases}$$

such that $i \in \{1,2,\ldots,m-1\}$, $I_a := \{j\,|\exists i, a_{0j}a_{ij} \neq 0\}$, and $I_b := \{k\,|\exists i, b_{0k}b_{ik} \neq 0\}$

Step 3. Obtain the decision set $\{^{s_{i1}}u_i|u_i \in U\}$

It is clear that the values s_{i1} give a ranking order over u_i. Therefore, the decision maker can choose the proper ones of the alternatives.

4. Simulation Results

In this section, we first compare the running times of CE12a and EMC19a by using MATLAB R2018b. So long as it has not been encountered a difficulty, we use a laptop with 2.6 GHz i5 Dual Core CPU and 4 GB RAM to compare the methods. However, in this study, we use a workstation with I(R) Xeon(R) CPU E5-1620 v4 @ 3.5 GHz and 64 GB RAM because the computer is insufficient to run CE12a if the parameters are more than 5000.

We present the running times of CE12a and EMC19a in Table 1 and Fig. 1 for 10 objects and the parameters ranging from 10 to 100. Even though the difference of running times between these methods is low, EMC19a is about 70 times faster than CE12a in 100 parameters and 10 objects.

Table 1. The running times of the methods for 10 objects and 10-100 parameters (In Second)

Parameter Count	10	20	30	40	50	60	70	80	90	100
CE12a	0.01811	0.00548	0.00166	0.00273	0.00604	0.00780	0.00941	0.00839	0.01225	0.01640
EMC19a	0.00789	0.00237	0.00039	0.00038	0.00211	0.00081	0.00042	0.00019	0.00032	0.00024
Difference	0.0102	0.0031	0.0013	0.0024	0.0039	0.0070	0.0090	0.0082	0.0119	0.0162
Advantage (%)	56.4254	56.6633	76.2652	86.2857	65.0508	89.6560	95.5149	97.7886	97.4096	98.5428

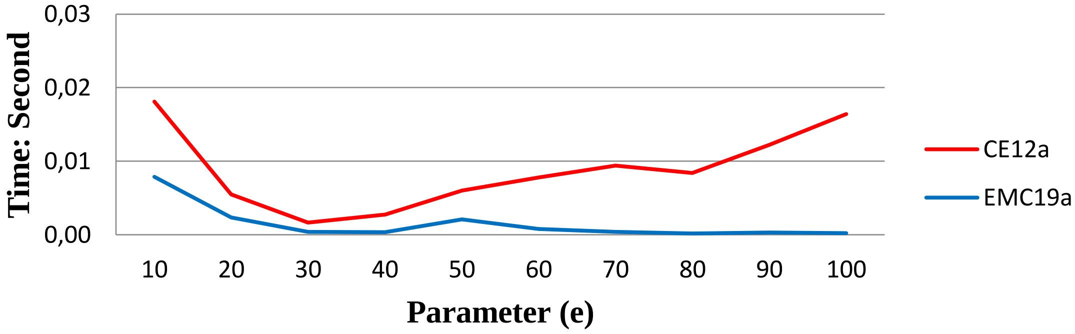

Figure 1. The figure for Table 1

We then give the running times of CE12a and EMC19a in Table 2 and Fig. 2 for 10 objects and the parameters ranging from 1000 to 10000. It must be noted that the difference in running times between these methods is increasing seriously. 278-second running time shows CE12a is not appropriate for any real-time software processing a large amount of data.

Table 2. The running times of the methods for 10 objects and 1000-10000 parameters (In Second)

Parameter Count	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
CE12a	0.7436	3.3679	9.9886	22.4843	40.8677	66.2469	102.8899	147.6173	202.8367	278.8861
EMC19a	0.0080	0.0027	0.0013	0.0015	0.0034	0.0026	0.0019	0.0024	0.0041	0.0026
Difference	0.7356	3.3652	9.9873	22.4828	40.8642	66.2443	102.8880	147.6149	202.8326	278.8835
Advantage (%)	98.9276	99.9201	99.9867	99.9933	99.9916	99.9960	99.9981	99.9984	99.9980	99.9991

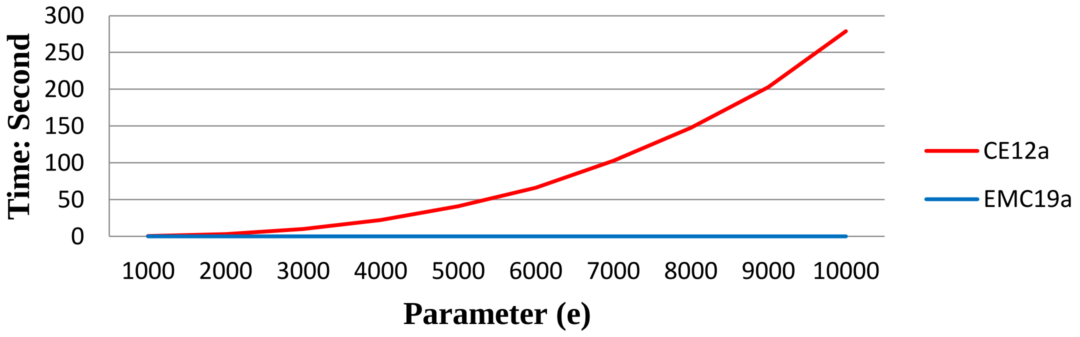

Figure 2. The figure for Table 2

We then give their running times in Table 3 and Fig. 3 for 10 parameters and the objects ranging from 10 to 100. Despite the low difference of running times between these methods, EMC19a is about 5 times faster than CE12a in 10 parameters and 100 objects.

Table 3. The running times of the methods for 10-100 objects and 10 parameters (In Second)

Object Count	10	20	30	40	50	60	70	80	90	100
CE12a	0.0141	0.0047	0.0022	0.0014	0.0032	0.0037	0.0041	0.0054	0.0033	0.0024
EMC19a	0.0074	0.0020	0.0005	0.0003	0.0017	0.0011	0.0009	0.0008	0.0005	0.0005
Difference	0.0067	0.0027	0.0017	0.0011	0.0015	0.0026	0.0032	0.0046	0.0028	0.0019
Advantage (%)	47.2867	58.0965	77.5393	78.0378	47.8422	71.2486	78.3838	85.0085	84.7237	78.6004

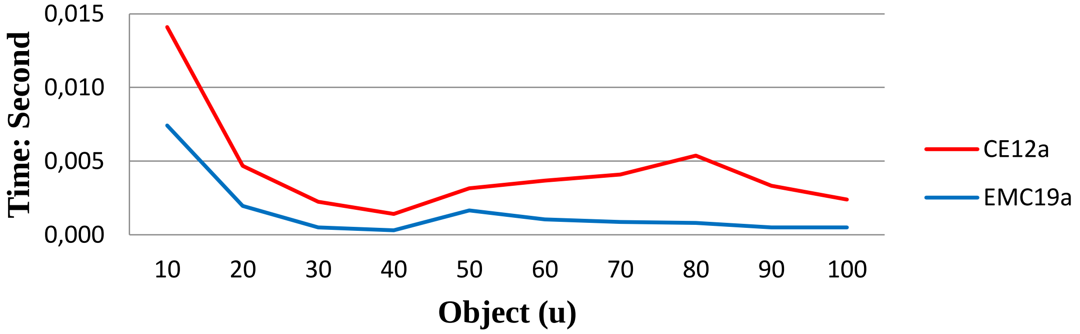

Figure 3. The figure for Table 3

We then give their running times in Table 4 and Fig. 4 for 10 parameters and the objects ranging from 1000 to 10000. The results show that only increasing the objects do not affect the running time as much as only increasing the parameters. Besides, in a large number of parameters, EMC19a works faster than in a large number of objects.

Table 4. The running times of the methods for 1000-10000 objects and 10 parameters (In Second)

Object Count	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
CE12a	0.0602	0.1216	0.2389	0.3778	0.5634	0.7539	1.0909	1.3661	1.6618	1.9791
EMC19a	0.0116	0.0129	0.0175	0.0242	0.0342	0.0414	0.0492	0.0590	0.0687	0.0812
Difference	0.0486	0.1087	0.2214	0.3536	0.5292	0.7125	1.0417	1.3071	1.5931	1.8979
Advantage (%)	80.7231	89.3771	92.6951	93.5828	93.9242	94.5032	95.4878	95.6795	95.8664	95.8986

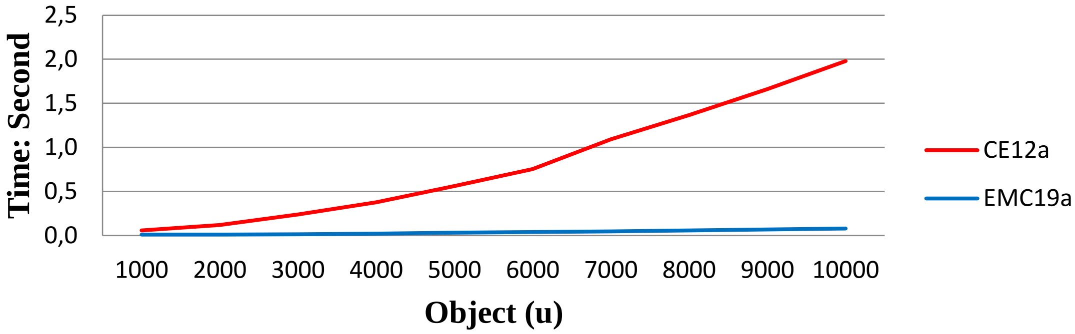

Figure 4. The figure for Table 4

We then give their running times in Table 5 and Fig. 5 for the parameters and the objects ranging from 10 to 100. Although the difference of running times between these methods is low, EMC19a is up to 130 times faster than CE12a.

Table 5. The running times of the methods for 10-100 objects and parameters (In Second)

Count	10	20	30	40	50	60	70	80	90	100
CE12a	0.0140	0.0055	0.0044	0.0076	0.0151	0.0233	0.0349	0.0447	0.0653	0.0905
EMC19a	0.0057	0.0017	0.0009	0.0006	0.0022	0.0011	0.0006	0.0006	0.0007	0.0007
Difference	0.0083	0.0037	0.0035	0.0070	0.0129	0.0222	0.0343	0.0441	0.0647	0.0897
Advantage (%)	59.2037	68.2321	80.3370	92.0232	85.3565	95.0726	98.2459	98.5621	98.9913	99.2141

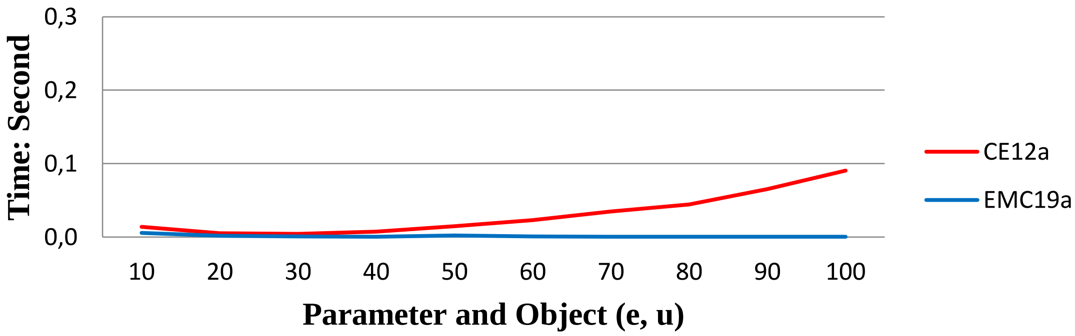

Figure 5. The figure for Table 5

We then give their running times in Table 6 and Fig. 6 for the parameters and the objects ranging from 100 to 1000. 0.0206-second and 616-second running times shows EMC19a is more appropriate than CE12a for any real-time software.

Table 6. The running times of the methods for 100-1000 objects and parameters (In Second)

Count	100	200	300	400	500	600	700	800	900	1000
CE12a	0.1447	1.3295	5.9489	17.0769	39.5983	79.2793	145.0186	244.9221	399.8934	616.6867
EMC19a	0.0101	0.0036	0.0038	0.0048	0.0083	0.0091	0.0114	0.0134	0.0172	0.0206
Difference	0.1346	1.3258	5.9451	17.0721	39.5900	79.2702	145.0072	244.9088	399.8762	616.6662
Advantage (%)	92.9952	99.7266	99.9355	99.9720	99.9791	99.9885	99.9921	99.9945	99.9957	99.9967

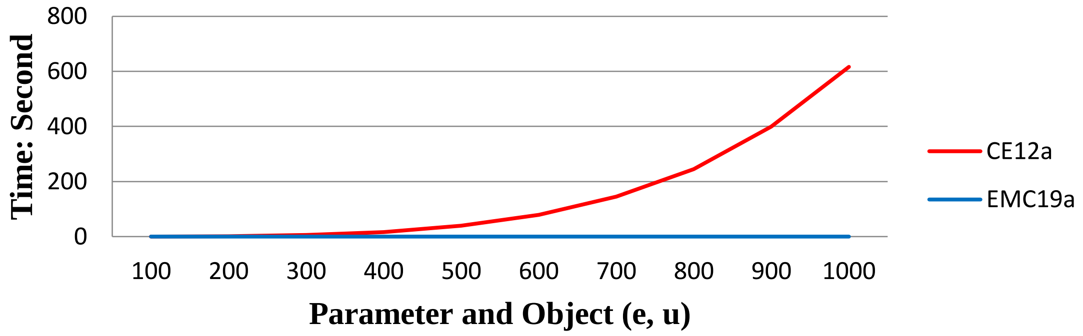

Figure 6. The figure for Table 6. The results show that EMC19a outperforms than CE12a in any number of data.

Secondly, we compare the running times of CE12o and EMC19o by using MATLAB R2018b and a workstation with I(R) Xeon(R) CPU E5-1620 v4 @ 3.5 GHz and 64 GB RAM because the computer mentioned above is insufficient to run CE12o if the parameters are more than 5000.

We present the running times of CE12o and EMC19o in Table 7 and Fig. 7 for 10 objects and the parameters ranging from 10 to 100. Even though the difference of running times between these methods is low, EMC19o is about 100 times faster than CE12o in 100 parameters and 10 objects.

Table 7. The running times of the methods for 10 objects and 10-100 parameters (In Second)

Parameter Count	10	20	30	40	50	60	70	80	90	100
CE12o	0.0117	0.0054	0.0023	0.0030	0.0085	0.0086	0.0112	0.0143	0.0162	0.0203
EMC19o	0.0066	0.0024	0.0004	0.0003	0.0025	0.0005	0.0001	0.0002	0.0002	0.0002
Difference	0.0052	0.0030	0.0019	0.0026	0.0061	0.0081	0.0111	0.0140	0.0159	0.0200
Advantage (%)	43.9452	55.0417	82.2197	88.7153	70.9732	94.4846	98.7038	98.4619	98.6299	98.8234

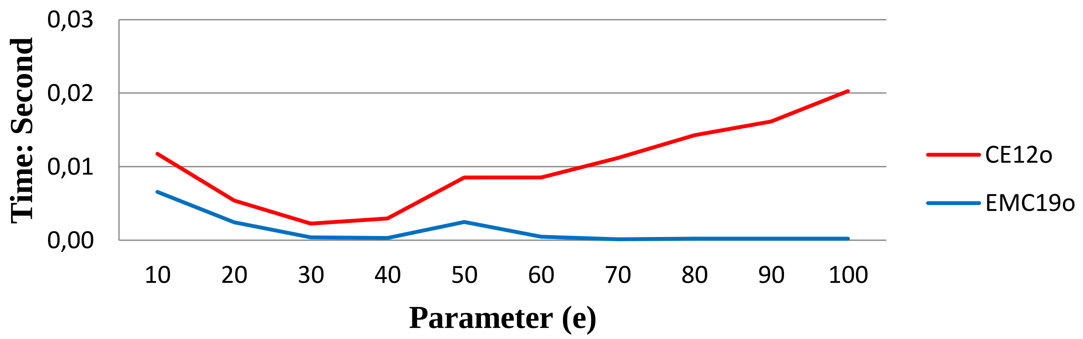

Figure 7. The figure for Table 7

We then give the running times of CE12o and EMC19o in Table 8 and Fig. 8 for 10 objects and the parameters ranging from 1000 to 10000. It must be noted that the difference in running times between these methods is increasing seriously. 515-second running time shows CE12o is not appropriate for any real-time software processing a large amount of data.

Table 8. The running times of the methods for 10 objects and 1000-10000 parameters (In Second)

Parameter Count	1000	2000	3000	4000	5000	6000	7000	8000	9000	10000
CE12o	0.8761	6.0139	18.1107	39.1060	72.1961	117.6709	184.9469	271.8735	373.6522	515.0063
EMC19o	0.0074	0.0028	0.0014	0.0016	0.0034	0.0027	0.0020	0.0023	0.0025	0.0027
Difference	0.8687	6.0111	18.1093	39.1044	72.1927	117.6682	184.9449	271.8712	373.6497	515.0036
Advantage (%)	99.1569	99.9539	99.9923	99.9959	99.9952	99.9977	99.9989	99.9991	99.9993	99.9995

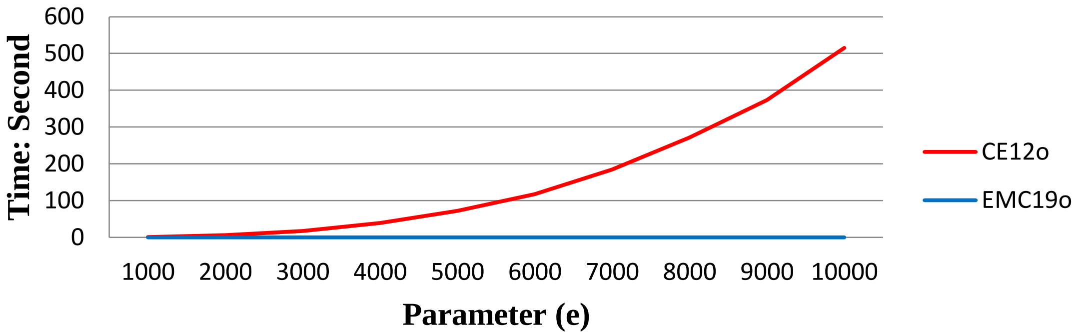

Figure 8. The figure for Table 8

We then give their running times in Table 9 and Fig. 9 for 10 parameters and the objects ranging from 10 to 100. Despite the low difference of running times between these methods, EMC19o is about 7 times faster than CE12o in 10 parameters and 100 objects.

Table 9. The running times of the methods for 10-100 objects and 10 parameters (In Second)

Object Count	10	20	30	40	50	60	70	80	90	100
CE12o	0.0113	0.0044	0.0017	0.0018	0.0039	0.0053	0.0056	0.0062	0.0038	0.0043
EMC19o	0.0058	0.0020	0.0004	0.0003	0.0016	0.0019	0.0008	0.0006	0.0005	0.0006
Difference	0.0056	0.0025	0.0013	0.0015	0.0023	0.0034	0.0048	0.0056	0.0033	0.0038
Advantage (%)	49.2078	55.1922	73.7450	82.3905	57.8698	63.7274	84.9306	90.7323	86.8001	87.2092

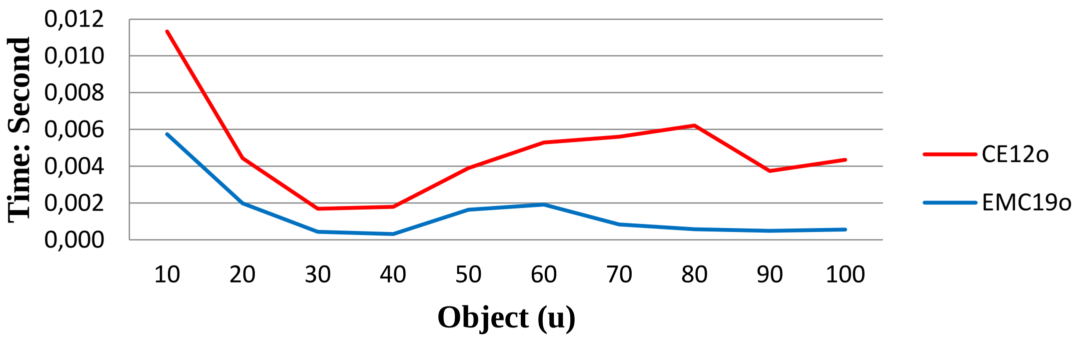

Figure 9. The figure for Table 9

We then give their running times in Table 10 and Fig. 10 for 10 parameters and the objects ranging from 1000 to 10000. The results show that only increasing the objects do not affect the running time as much as only increasing the parameters. Besides, in a large number of parameters, EMC19o works faster than in a large number of objects.

Table 10. The running times of the methods for 1000-10000 objects and 10 parameters (In Second)

Object Count	**1000**	**2000**	**3000**	**4000**	**5000**	**6000**	**7000**	**8000**	**9000**	**10000**
CE12o	0.0761	0.1539	0.2690	0.4271	0.6235	0.8606	1.1178	1.4193	1.7636	2.1721
EMC19o	0.0116	0.0135	0.0172	0.0263	0.0338	0.0413	0.0497	0.0598	0.0691	0.0816
Difference	0.0645	0.1404	0.2518	0.4008	0.5898	0.8193	1.0682	1.3594	1.6945	2.0905
Advantage (%)	84.7582	91.2375	93.5980	93.8512	94.5847	95.1976	95.5574	95.7834	96.0829	96.2443

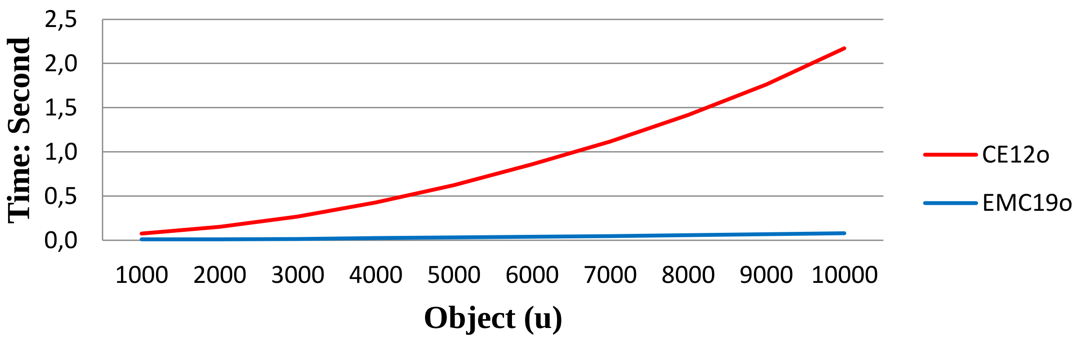

Figure 10. The figure for Table 10

We then give their running times in Table 11 and Fig. 11 for the parameters and the objects ranging from 10 to 100. Although the difference of running times between these methods is low, EMC19o is up to 150 times faster than CE12o.

Table 11. The running times of the methods for 10-100 objects and parameters (In Second)

Count	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
CE12o	0.0106	0.0059	0.0050	0.0103	0.0197	0.0353	0.0447	0.0685	0.0829	0.1030
EMC19o	0.0053	0.0018	0.0009	0.0006	0.0022	0.0012	0.0006	0.0006	0.0007	0.0007
Difference	0.0053	0.0041	0.0041	0.0097	0.0175	0.0341	0.0441	0.0679	0.0822	0.1023
Advantage (%)	50.0941	69.9820	82.4312	93.8611	88.6337	96.5521	98.5871	99.1218	99.1953	99.3082

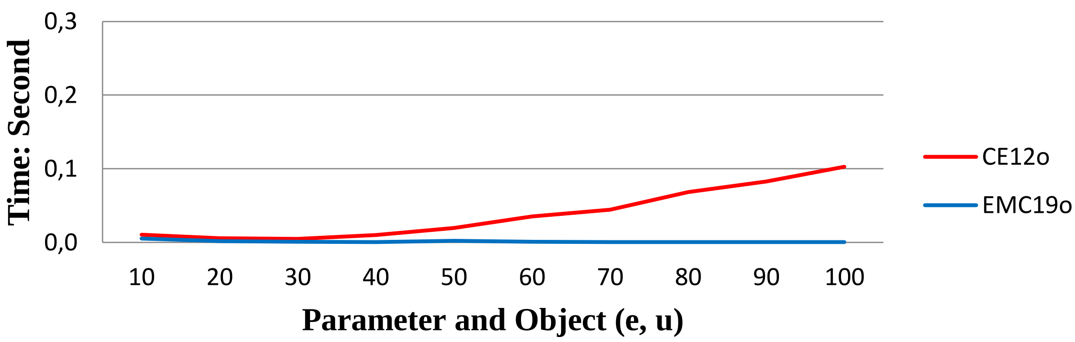

Figure 11. The figure for Table 11

We then give their running times in Table 12 and Fig. 12 for the parameters and the objects ranging from 100 to 1000. 0.0201-second and 637-second running times shows EMC19o is more appropriate than CE12o for any real-time software

Table 12. The running times of the methods for 100-1000 objects and parameters (In Second)

Parameter Count	100	200	300	400	500	600	700	800	900	1000
CE12o	0.1635	1.4368	6.5425	18.5807	42.0813	83.9797	151.8452	257.2460	415.8905	637.8363
EMC19o	0.0071	0.0037	0.0034	0.0048	0.0083	0.0089	0.0113	0.0137	0.0162	0.0201
Difference	0.1564	1.4331	6.5392	18.5760	42.0731	83.9708	151.8339	257.2323	415.8743	637.8162
Advantage (%)	95.6327	99.7408	99.9485	99.9744	99.9804	99.9894	99.9925	99.9947	99.9961	99.9969

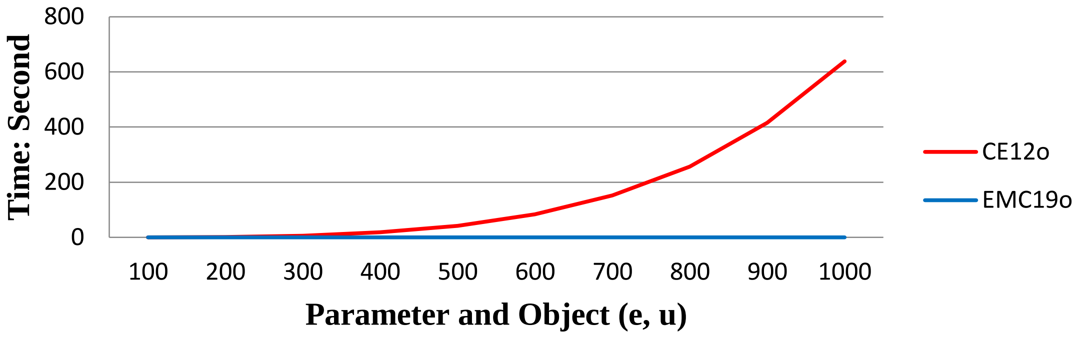

Figure 12. The figure for Table 12

The results show that EMC19o outperforms than CE12o in any number of data.

5. An Application of EMC19o

In this section, we apply EMC19o to sort some filters used in image denoising concerning noise removal performance. Even though sorting these filters is to be more difficult in the event that the filters perform variously in different noise densities, EMC19o overcomes this difficulty. To illustrate, let us consider mean-SSIM results (Table 13) and mean-PSNR results (Table 14) provided in [41].

Table 13. The mean-SSIM results of the filters for the 15 traditional images

Noise Density	10%	20%	30%	40%	50%	60%	70%	80%	90%
PSMF	0.9028	0.8715	0.8018	0.6988	0.4903	0.1882	0.0633	0.0318	0.0139
DBA	0.9079	0.8664	0.8097	0.7376	0.6521	0.5552	0.4567	0.3623	0.2937
MDBUTMF	0.8841	0.7994	0.7443	0.7657	0.7963	0.7880	0.7501	0.6443	0.3052
NAFSM	0.9147	0.8916	0.8669	0.8409	0.8124	0.7796	0.7403	0.6872	0.5736
DAMF	0.9253	0.9113	0.8946	0.8752	0.8523	0.8244	0.7892	0.7398	0.6572

Table 14. The mean-PSNR results of the filters for the 15 traditional images

Noise Density	10%	20%	30%	40%	50%	60%	70%	80%	90%
PSMF	31.6100	28.3800	25.2900	22.2500	18.1900	12.6000	9.1700	7.4600	6.0800
DBA	32.8200	29.0500	26.1800	23.7200	21.4900	19.2700	17.0900	14.8100	12.1600
MDBUTMF	31.2400	28.0500	26.6800	26.7300	26.8400	26.1800	24.8600	21.4200	14.0700
NAFSM	33.8800	31.0500	29.3000	28.0400	26.9500	25.9300	24.8900	23.5900	20.7800
DAMF	37.4800	34.1400	31.9500	30.3000	28.9100	27.6300	26.3200	24.8000	22.7100

Assume that the success in high noise densities is more important than in the others. In that case, the values given in Table 13 and the values normalized via maximum entry of Table 14 given in Table 14 can be represented with two $fpfs$-matrices as follows:

$$[a_{ij}] := \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 & 0.5 & 0.6 & 0.7 & 0.8 & 0.9 \\ 0.9028 & 0.8715 & 0.8018 & 0.6988 & 0.4903 & 0.1882 & 0.0633 & 0.0318 & 0.0139 \\ 0.9079 & 0.8664 & 0.8097 & 0.7376 & 0.6521 & 0.5552 & 0.4567 & 0.3623 & 0.3623 \\ 0.8841 & 0.7994 & 0.7443 & 0.7657 & 0.7963 & 0.7880 & 0.7501 & 0.6443 & 0.3052 \\ 0.9147 & 0.8916 & 0.8669 & 0.8409 & 0.8124 & 0.7796 & 0.7403 & 0.6872 & 0.5736 \\ 0.9253 & 0.9113 & 0.8946 & 0.8752 & 0.8523 & 0.8244 & 0.7892 & 0.7398 & 0.6572 \end{bmatrix}$$

and

$$[b_{ik}] := \begin{bmatrix} 0.1 & 0.2 & 0.3 & 0.4 & 0.5 & 0.6 & 0.7 & 0.8 & 0.9 \\ 0.8434 & 0.7572 & 0.6748 & 0.5936 & 0.4853 & 0.3362 & 0.2447 & 0.1990 & 0.1622 \\ 0.8757 & 0.7751 & 0.6985 & 0.6329 & 0.5734 & 0.5141 & 0.4560 & 0.3951 & 0.3244 \\ 0.8335 & 0.7484 & 0.7118 & 0.7132 & 0.7161 & 0.6985 & 0.6633 & 0.5715 & 0.3754 \\ 0.9039 & 0.8284 & 0.7818 & 0.7481 & 0.7191 & 0.6918 & 0.6641 & 0.6294 & 0.5544 \\ 1.0000 & 0.9109 & 0.8525 & 0.8084 & 0.7713 & 0.7372 & 0.7022 & 0.6617 & 0.6059 \end{bmatrix}$$

If we apply EMC19o to the $fpfs$-matrices $[a_{ij}]$ and $[b_{ik}]$, then the score matrix and the decision set are as follows:

$$[s_{i1}] = [0.2795 \quad 0.3331 \quad 0.5251 \quad 0.5498 \quad 0.5918]^T$$

and

$$\{{}^{0.4723}\text{PSMF},\ {}^{0.5629}\text{DBA},\ {}^{0.8872}\text{MDBUTMF},\ {}^{0.9289}\text{NAFSM},\ {}^{1}\text{DAMF}\}$$

The scores show that DAMF outperforms the others and the ranking order DAMF, NAFSM, MDBUTMF, DBA, and PSMF is valid.

Assume that the success in low noise densities is more important than in the others. In that case, the values given in Table 13 and the values normalized via maximum entry of Table 14 given in Table 14 can be represented with two $fpfs$-matrices as follows:

$$[c_{ij}] := \begin{bmatrix} 0.9 & 0.8 & 0.7 & 0.6 & 0.5 & 0.4 & 0.3 & 0.2 & 0.1 \\ 0.9028 & 0.8715 & 0.8018 & 0.6988 & 0.4903 & 0.1882 & 0.0633 & 0.0318 & 0.0139 \\ 0.9079 & 0.8664 & 0.8097 & 0.7376 & 0.6521 & 0.5552 & 0.4567 & 0.3623 & 0.3623 \\ 0.8841 & 0.7994 & 0.7443 & 0.7657 & 0.7963 & 0.7880 & 0.7501 & 0.6443 & 0.3052 \\ 0.9147 & 0.8916 & 0.8669 & 0.8409 & 0.8124 & 0.7796 & 0.7403 & 0.6872 & 0.5736 \\ 0.9253 & 0.9113 & 0.8946 & 0.8752 & 0.8523 & 0.8244 & 0.7892 & 0.7398 & 0.6572 \end{bmatrix}$$

and

$$[d_{ik}] := \begin{bmatrix} 0.9 & 0.8 & 0.7 & 0.6 & 0.5 & 0.4 & 0.3 & 0.2 & 0.1 \\ 0.8434 & 0.7572 & 0.6748 & 0.5936 & 0.4853 & 0.3362 & 0.2447 & 0.1990 & 0.1622 \\ 0.8757 & 0.7751 & 0.6985 & 0.6329 & 0.5734 & 0.5141 & 0.4560 & 0.3951 & 0.3244 \\ 0.8335 & 0.7484 & 0.7118 & 0.7132 & 0.7161 & 0.6985 & 0.6633 & 0.5715 & 0.3754 \\ 0.9039 & 0.8284 & 0.7818 & 0.7481 & 0.7191 & 0.6918 & 0.6641 & 0.6294 & 0.5544 \\ 1.0000 & 0.9109 & 0.8525 & 0.8084 & 0.7713 & 0.7372 & 0.7022 & 0.6617 & 0.6059 \end{bmatrix}$$

If we apply EMC19o to the $fpfs$-matrices $[c_{ij}]$ and $[d_{ik}]$, then the score matrix and the decision set are as follows:

$$[s_{i1}] = [0.8125 \ \ 0.8171 \ \ 0.7957 \quad 0.8232 \quad 0.8328]^T$$

and

$$\{{}^{0.9757}\text{PSMF},\ {}^{0.9812}\text{DBA},\ {}^{0.9555}\text{MDBUTMF},\ {}^{0.9885}\text{NAFSM},\ {}^{1}\text{DAMF}\}$$

The scores show that DAMF outperforms the others and the ranking order DAMF, NAFSM, DBA, PSMF, and MDBUTMF is valid.

6. Conclusion

The soft max-min decision-making method SMmDM was defined in 2010 [5] and the fuzzy soft max-min decision-making method FSMmDM [12], being a generalisation of SMmDM, was defined in 2012. Lately, since such methods cannot model decision-making problems in the event that the parameters have uncertainties, these methods have been configured [34] via $fpfs$-matrices [13,40]. However, the configured method has a drawback such as its incapability of processing a large number of parameters on such a standard computer with 2.6 GHz i5 Dual Core CPU and 4GB RAM.

In this study, we have proposed the method EMC19a, which is faster than CE12a and the method EMC19o, which is faster than CE12o. Of course, for other products, simplifications of these methods can be investigated.

Also, we have compared the methods mentioned above in terms of their running times. Besides the results in Section 6, the results in Table 15 and 16 too show that EMC19a and EMC19o perform better than CE12a and CE12o, respectively.

Table 15. The mean advantage, max advantage, and max difference values of EMC19a over CE12a

Location	Objects	Parameters	Mean Advantage %	Max Advantage %	Max Difference
Table 1	10	10-100	81.9602	98.5428	0.0162
Table 2	10	1000-10000	99.8809	99.9991	278.8835
Table 3	10-100	10	70.6767	85.0085	0.0067
Table 4	1000-	10	92.7738	95.8986	1.8979
Table 5	10-100	10-100	87.5239	99.2141	0.0897
Table 6	100-1000	100-1000	99.2576	99.9967	616.6662

Table 16. The mean advantage, max advantage, and max difference values of EMC19o over CE12o

Location	Objects	Parameters	Mean Advantage %	Max Advantage %	Max Difference
Table 7	10	10-100	82.9999	98.8234	0.0200
Table 8	10	1000-10000	99.9089	99.9995	515.0036
Table 9	10-100	10	73.1805	90.7323	0.0056
Table 10	1000-	10	93.6895	96.2443	2.0905
Table 11	10-100	10-100	87.7767	99.3082	0.1023
Table 12	100-1000	100-1000	99.5246	99.9969	637.8162

Finally, it is clear that the methods constructed by minimum-maximum (min-max), max-max, and min-min decision functions are also worth to study.

Acknowledgements

The authors thank Dr. Uğur Erkan for technical support. This work was supported by the Office of Scientific Research Projects Coordination at Çanakkale Onsekiz Mart University, Grant number: FBA-2018-1367.

References

[1]. Molodtsov, D., Soft set theory-first results, Computers and Mathematics with Applications, 1999, 37(4-5), 19-31.

[2]. Maji, P. K., Biswas, R., Roy, A. R., Fuzzy soft sets, The Journal of Fuzzy Mathematics, 2001, 9(3), 589-602.

[3]. Maji, P. K., Roy, A. R., Biswas, R., An application of soft sets in a decision making problem, Computer and Mathematics with Applications, 2002, 44(8-9), 1077-1083.

[4]. Maji, P. K., Biswas, R., Roy, A. R., Soft set theory, Computer and Mathematics with Applications, 2003, 45(4-5), 555-562.

[5]. Çağman, N., Enginoğlu, S., Soft matrix theory and its decision making, Computer and Mathematics with Applications, 2010, 59(10), 3308-3314.

[6]. Çağman, N., Enginoğlu, S., Soft set theory and uni-int decision making, European Journal of Operational Research, 2010, 207(2), 848-855.

[7]. Çağman, N., Çıtak, F., Enginoğlu, S., Fuzzy parameterized fuzzy soft set theory and its applications, Turkish Journal of Fuzzy Systems, 2010, 1(1), 21-35.

[8]. Çağman, N., Çıtak, F., Enginoğlu, S., FP-soft set theory and its applications, Annals of Fuzzy Mathematics and Informatics, 2011, 2(2), 219-226.

[9]. Çağman, N., Enginoğlu, S., Çıtak, F., Fuzzy soft set theory and its applications, Iranian Journal of Fuzzy Systems, 2011, 8(3), 137-147.

[10]. Çağman, N., Deli, İ., Means of FP-soft sets and their applications, Hacettepe Journal of Mathematics and Statistics, 2012, 41(5), 615-625.

[11]. Çağman, N., Deli, İ., Products of FP-soft sets and their applications, Hacettepe Journal of Mathematics and Statistics, 2012, 41(3), 365-374.

[12]. Çağman, N., Enginoğlu, S., Fuzzy soft matrix theory and its application in decision making, Iranian Journal of Fuzzy Systems, 2012, 9(1), 109-119.

[13]. Enginoğlu, S., Soft matrices, Doctoral Dissertation, Gaziosmanpaşa University, Tokat, Turkey 2012.

[14]. Atmaca, S., Zorlutuna, İ., On topological structures of fuzzy parametrized soft sets, The Scientific World Journal, 2014, Article ID 164176, 8 pages.

[15]. Deli, İ., Çağman, N., Relations on FP-soft sets applied to decision making problems, Journal of New Theory, 2015, 3, 98-107.

[16]. Çıtak, F., Çağman, N., Soft int-rings and its algebraic applications, Journal of Intelligent and Fuzzy Systems, 2015, 28, 1225-1233.

[17]. Enginoğlu, S., Çağman, N., Karataş, S., Aydın, T., On soft topology, El-Cezerî Journal of Science and Engineering, 2015, 2(3), 23-38.

[18]. Karaaslan, F., Soft classes and soft rough classes with applications in decision making, Mathematical Problems in Engineering, 2016, Article ID 1584528, 11 pages.

[19]. Muştuoğlu, E., Sezgin, A., Türk, Z. K., Some characterizations on soft uni-groups and normal soft uni-groups, International Journal of Computer Applications, 2016, 155(10), 1-8.

[20]. Sezgin, A., A new approach to semigroup theory I: Soft union semigroups, ideals and bi-ideals, Algebra Letters, 2016, 2016(3), 1-46.

[21]. Şenel, G., A new approach to Hausdorff space theory via the soft sets, Mathematical Problems in Engineering, 2016, Article ID 2196743, 6 pages.

[22]. Tunçay, M., Sezgin, A., Soft union ring and its applications to ring theory, International Journal of Computer Applications, 2016, 151(9), 7-13.

[23]. Zorlutuna, İ., Atmaca, S., Fuzzy parametrized fuzzy soft topology, New Trends in Mathematical Sciences, 2016, 4(1), 142-152.

[24]. Atmaca, S., Relationship between fuzzy soft topological spaces and (X,τ_e) parameter spaces, Cumhuriyet Science Journal, 2017, 38(4), Supplement, 813-821.

[25]. Bera, S., Roy, S. K., Karaaslan, F., Çağman, N., Soft congruence relation over lattice, Hacettepe Journal of Mathematics and Statistics, 2017, 46(6), 1035-1042.

[26]. Çıtak, F., Çağman, N., Soft k-int-ideals of semirings and its algebraic structures, Annals of Fuzzy Mathematics and Informatics, 2017, 13(4), 531-538.

[27]. Riaz, M., Hashmi, R., Fuzzy parameterized fuzzy soft topology with applications, Annals of Fuzzy Mathematics and Informatics, 2017, 13(5), 593-613.

[28]. Şenel, G., A comparative research on the definition of soft point, International Journal of Computer Applications, 2017, 163(2), 1-4.

[29]. Riaz, M., Hashmi, R., Fuzzy parameterized fuzzy soft compact spaces with decision-making, Punjab University Journal of Mathematics, 2018, 50(2), 131-145.

[30]. Riaz, M., Hashmi, R., Farooq, A., Fuzzy parameterized fuzzy soft metric spaces, Journal of Mathematical Analysis, 2018, 9(2), 25-36.

[31]. Şenel, G., Analyzing the locus of soft spheres: Illustrative cases and drawings. European Journal of Pure and Applied Mathematics, 2018, 11(4), 946-957.

[32]. Ullah, A., Karaaslan, F., Ahmad, I., Soft uni-Abel-Grassmann's groups, European Journal of Pure and Applied Mathematics, 2018, 11(2), 517-536.

[33]. Sezgin, A., Çağman, N., Çıtak, F., α-Inclusions applied to group theory via soft set and logic, Communications Faculty of Sciences University of Ankara Series A1 Mathematics and Statistics, 2019, 68(1), 334-352.

[34]. Enginoğlu, S., Memiş, S. A configuration of some soft decision-making algorithms via $fpfs$-matrices, Cumhuriyet Science Journal, 2018, 39(4), 871-881.

[35]. Enginoglu, S., Memiş, S., Comment on "Fuzzy soft sets" [The Journal of Fuzzy Mathematics, 9(3), 2001, 589–602], International Journal of Latest Engineering Research and Applications, 2018, 3(9), 1-9.

[36]. Enginoğlu, S., Memiş, S., A review on an application of fuzzy soft set in multicriteria decision making problem [P.K. Das, R. Borgohain, International Journal of Computer Applications 38 (12) (2012) 33-37]. In: Akgül, M., Yılmaz, İ., İpek, A. (eds.) Proceeding of The International Conference on Mathematical Studies and Applications 2018, pp. 173-178, Karaman, Turkey.

[37]. Enginoğlu, S., Memiş, S., A review on some soft decision-making methods. In: Akgül, M., Yılmaz, İ., İpek, A. (eds.) Proceeding of The International Conference on Mathematical Studies and Applications 2018, pp. 437-442, Karaman, Turkey.

[38]. Enginoglu, S., Memiş, S., Arslan, B., Comment (2) on soft set theory and uni-int decision making [European Journal of Operational Research, (2010) 207, 848-855], Journal of New Theory, 2018, 25, 85-102.

[39]. Enginoglu, S., Memiş, S., Öngel, T., Comment on soft set theory and uni-int decision making [European Journal of Operational Research, (2010) 207, 848-855], Journal of New Results in Science, 2018, 7(3), 28-43.

[40]. Enginoğlu, S., Çağman, N., Fuzzy parameterized fuzzy soft matrices and their application in decision-making, TWMS Journal of Applied and Engineering Mathematics, In Press.

[41]. Erkan, U., Gökrem, L., Enginoğlu, S., Different applied median filter in salt and pepper noise, Computers and Electrical Engineering, 2018, 70, 789-798.