Vol.0 1–9 (Oct. 2024)

Missing Value Imputation via Mathematical Optimization with Instance-and-Feature Neighborhoods

Yuta Kawakami[1] Takumi Matsumoto[1] Yuichi Takano[2]

Received: xx xx, xxxx, Accepted: xx xx, xxxx

Abstract: Datasets collected for analysis often contain a certain amount of incomplete instances, where some feature values are missing. Since many statistical analyses and machine learning algorithms depend on complete datasets, missing values need to be imputed in advance. Bertsimas et al. (2018) proposed a high-performance method that combines machine learning and mathematical optimization algorithms for imputing missing values. We extensively revise this imputation method based on the nearest neighbors algorithm by using not only neighborhoods of data instances but also neighborhoods of features. Specifically, we first formulate an optimization model using the instance-and-feature neighborhoods for missing value imputation. We next design an alternating optimization algorithm to find high-quality solutions to our optimization model for missing value imputation. We also develop a warm-start strategy to efficiently find a sequence of solutions for various neighborhood sizes. Experimental results demonstrate the excellent imputation accuracy of our method with instance-and-feature neighborhoods and the computational efficiency of our alternating optimization algorithm with the warm-start strategy.

Keywords: missing value imputation, mathematical optimization, nearest neighbors, alternating optimization, warm start

1. Introduction

1.1 Background

In recent years, dramatic advances in information and communication technology have made a variety of datasets easily available. Against this background, data mining and machine learning tasks are becoming increasingly important for improving operational efficiency and business performance. In many situations, however, the datasets collected for analysis contain a certain amount of incomplete instances, where some feature values are missing [24].

Such missing values can arise for a variety of reasons [12], including human error in data processing, machine error due to equipment failure, refusal to answer questions, dropout from the survey, and merging unrelated data. For example, it is known that gene expression data frequently have missing values due to experimental reasons [22].

Since many statistical analyses and machine learning algorithms depend on complete datasets, missing values need to be imputed in advance. However, imputing missing values can degrade prediction accuracy, impede data analysis, and bias outcomes due to differences between missing and actual values [2].

1.2 Related Work

Various methods for missing value imputation have been proposed to deal with incomplete datasets. These methods can be categorized into statistical and machine learning techniques [24]. Statistical techniques include mean/mode imputation, least squares methods, linear/logistic regression, and expectation–maximization (EM) algorithms, whereas machine learning techniques include clustering, decision trees, nearest neighbors algorithms, and random forests; see systematic reviews [12], [24] for detailed references to these imputation techniques.

Recently, deep learning methods have been actively applied to missing value imputation [33]; these include variational autoencoders [8], [27] and generative adversarial networks [23], [30], [36]. Although deep learning methods have demonstrated overwhelming performance in tasks such as image recognition and natural language processing, several studies [17], [21], [33] have reported that conventional imputation techniques (e.g., the EM algorithm [11], [25], nearest neighbors algorithm [3], and random forests [31]) can perform as well as powerful deep learning methods.

In contrast to these statistical and machine learning techniques, Bertsimas et al. [5] proposed a high-performance method that combines machine learning and mathematical optimization algorithms for missing value imputation. They formulated an optimization model for imputing missing values based on machine learning techniques (e.g., the nearest neighbors algorithm). This optimization model was solved by alternating optimization, which repeats imputing missing values and training a machine learning model.

Bertsimas et al. [5] reported that in terms of the imputation accuracy, their method outperformed state-of-the-art methods for missing value imputation [6], [7], [10], [29] on 84 types of machine learning datasets [19]. Bertsimas et al. [4] applied this method to imputing clinical covariates in multivariate panel data.

[1] Graduate School of Science and Technology, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan
[2] Institute of Systems and Information Engineering, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan

They also reported that for real-world clinical datasets, their method achieved higher accuracy than did state-of-the-art methods for missing value imputation [7], [14], [29].

Methods for missing value imputation have also been used for rating prediction in recommender systems [1]. Indeed, rating prediction, which involves predicting users' ratings for their unknown items, is amount to imputing missing values in the user–item rating matrix. To improve the accuracy of rating prediction, Wang et al. [34] devised a collaborative filtering (nearest neighbors) algorithm that uses neighborhoods of both users and items.

1.3 Our Contribution

The goal of this paper is to develop a high-performance method for missing value imputation based on mathematical optimization. Inspired by Wang et al. [34], we extensively revise the imputation method proposed by Bertsimas et al. [5], by using not only neighborhoods of data instances but also neighborhoods of features in the nearest neighbors algorithm.

Main contributions of our research are threefold. First, we formulate an optimization model using the neighborhoods of both instances and features to impute missing values. Second, we design an alternating optimization algorithm to find high-quality solutions to our optimization model for missing value imputation. Third, we develop a warm-start strategy to efficiently find a sequence of solutions for various neighborhood sizes.

We conduced computational experiments using incomplete datasets generated based on three types of missing data mechanisms [25] from real-world datasets. Computational results demonstrate that our method with instance-and-feature neighborhoods can perform very well especially for some missing data mechanisms. Moreover, our warm-start strategy can greatly reduce the computation time required by the alternating optimization algorithm for large-sized neighborhoods.

2. Optimization Model

In this section, we formulate our optimization model using neighborhoods of both instances and features for missing value imputation. Throughout this paper, we denote the set of consecutive integers from 1 to n as $[n] \coloneqq \{1, 2, \ldots, n\}$.

2.1 Incomplete Data Matrix

We focus on the following incomplete data matrix with some missing entries:

$$\boldsymbol{X} \coloneqq (x_{ij})_{(i,j)\in[n]\times[p]} \in \mathbb{R}^{n\times p},$$

where x_{ij} denotes a (observed or missing) value of feature $j \in [p]$ for instance $i \in [n]$. We assume that $\boldsymbol{X}$ is a matrix of numerical (i.e., quantitative) data, if necessary, by transforming categorical (i.e., qualitative) features into one-hot or distributed representations [9], [28]. We also assume that each feature is standardized to have mean zero and standard deviation one.

For such an incomplete data matrix $\boldsymbol{X}$, we define index sets of observed and missing entries as

$$\mathcal{O} \coloneqq \{(i, j) \in [n] \times [p] \mid x_{ij} \text{ is observed}\},$$
$$\mathcal{M} \coloneqq \{(i, j) \in [n] \times [p] \mid x_{ij} \text{ is missing}\}.$$

We also define index sets of instances and features containing missing values as

$$\mathcal{M}^{(\mathrm{I})} \coloneqq \{i \in [n] \mid \exists j \in [p], (i, j) \in \mathcal{M}\},$$
$$\mathcal{M}^{(\mathrm{J})} \coloneqq \{j \in [p] \mid \exists i \in [n], (i, j) \in \mathcal{M}\}.$$

2.2 Design Variables

We first introduce a design variable representing a complete data matrix after missing value imputation as

$$\boldsymbol{W} \coloneqq (w_{ij})_{(i,j)\in[n]\times[p]} \in \mathbb{R}^{n\times p},$$

where w_{ij} denotes an imputed value of feature $j \in [p]$ for instance $i \in [n]$.

We next introduce design variables for determining neighborhoods of instances and features as

$$\boldsymbol{Z}^{(\mathrm{I})} \coloneqq (z^{(\mathrm{I})}_{ik})_{(i,k)\in\mathcal{M}^{(\mathrm{I})}\times[n]} \in \{0, 1\}^{|\mathcal{M}^{(\mathrm{I})}|\times n},$$
$$\boldsymbol{Z}^{(\mathrm{J})} \coloneqq (z^{(\mathrm{J})}_{j\ell})_{(j,\ell)\in\mathcal{M}^{(\mathrm{J})}\times[p]} \in \{0, 1\}^{|\mathcal{M}^{(\mathrm{J})}|\times p},$$

where each entry of these matrices are defined as

$$z^{(\mathrm{I})}_{ik} \coloneqq \begin{cases} 1 & \text{instance } k \text{ is a neighbor of instance } i, \\ 0 & \text{otherwise,} \end{cases}$$
$$z^{(\mathrm{J})}_{j\ell} \coloneqq \begin{cases} 1 & \text{feature } \ell \text{ is a neighbor of feature } j, \\ 0 & \text{otherwise.} \end{cases}$$

2.3 Objective Function

We consider minimizing the following objective function based on the collaborative filtering (nearest neighbors) algorithm [34]:

$$f(\boldsymbol{W}, \boldsymbol{Z}^{(\mathrm{I})}, \boldsymbol{Z}^{(\mathrm{J})}) \coloneqq (1-\lambda) \sum_{i\in\mathcal{M}^{(\mathrm{I})}} \sum_{k=1}^{n} z^{(\mathrm{I})}_{ik} \sum_{j=1}^{p} (w_{ij} - w_{kj})^2 + \lambda \sum_{j\in\mathcal{M}^{(\mathrm{J})}} \sum_{\ell=1}^{p} z^{(\mathrm{J})}_{j\ell} \sum_{i=1}^{n} (w_{ij} - w_{i\ell})^2, \quad (1)$$

where $\lambda \in [0, 1]$ is a trade-off parameter between instance and feature neighbors.

In Eq. (1), the first term represents the sum of distances between an incomplete instance $i \in \mathcal{M}^{(\mathrm{I})}$ and its neighbor instances, and the second term represents the sum of distances between an incomplete feature $j \in \mathcal{M}^{(\mathrm{J})}$ and its neighbor features. Missing values are imputed by minimizing Eq. (1) such that imputed values get close between neighbor instances and between neighbor features. Note that the first term in Eq. (1) corresponds to the objective function adopted in Bertsimas et al. [5], and that the second term is newly introduced by us.

2.4 Formulation

Let $K^{(\mathrm{I})} \in [n-1]$ and $K^{(\mathrm{J})} \in [p-1]$ be parameters for specifying the neighborhood sizes of instances and features, respectively. Then, our optimization model for imputing missing values based on the instance-and-feature neighborhoods can be formulated as follows:

$$\begin{aligned}
\text{minimize} \quad & f(\boldsymbol{W}, \boldsymbol{Z}^{(\mathrm{I})}, \boldsymbol{Z}^{(\mathrm{J})}) && (2)\\
\text{subject to} \quad & w_{ij} = x_{ij} \quad ((i,j) \in \mathcal{O}), && (3)\\
& z^{(\mathrm{I})}_{ii} = 0 \quad (i \in \mathcal{M}^{(\mathrm{I})}), && (4)\\
& z^{(\mathrm{J})}_{jj} = 0 \quad (j \in \mathcal{M}^{(\mathrm{J})}), && (5)\\
& \sum_{k=1}^{n} z^{(\mathrm{I})}_{ik} = K^{(\mathrm{I})} \quad (i \in \mathcal{M}^{(\mathrm{I})}), && (6)\\
& \sum_{\ell=1}^{p} z^{(\mathrm{J})}_{j\ell} = K^{(\mathrm{J})} \quad (j \in \mathcal{M}^{(\mathrm{J})}), && (7)\\
& \boldsymbol{W} \in \mathbb{R}^{n\times p}, && (8)\\
& \boldsymbol{Z}^{(\mathrm{I})} \in \{0,1\}^{|\mathcal{M}^{(\mathrm{I})}|\times n}, && (9)\\
& \boldsymbol{Z}^{(\mathrm{J})} \in \{0,1\}^{|\mathcal{M}^{(\mathrm{J})}|\times p}. && (10)
\end{aligned}$$

Here, the observed values are fixed in Eq. (3). Neither instances nor features are selected as their own neighbors due to Eqs. (4)–(5). The neighborhood sizes are specified in Eqs. (6)–(7). Design variables are listed in Eqs. (8)–(10).

3. Alternating Optimization Algorithm

Our optimization model (2)–(10), which is a mixed-integer optimization problem with the nonconvex objective function in Eq. (1), is very difficult to solve exactly. We thus develop a revised version of the alternating optimization algorithm employed in Bertsimas et al. [5].

3.1 Outline

Our alternating optimization algorithm is described in Algorithm 1. We begin by initializing a complete data matrix $\boldsymbol{W} \in \mathbb{R}^{n\times p}$. According to conventions, we substitute the mean of the corresponding column into each missing entry.

We next alternate between updating neighborhoods (i.e., $\boldsymbol{Z}^{(\mathrm{I})}$ and $\boldsymbol{Z}^{(\mathrm{J})}$) and updating missing values (i.e., $\boldsymbol{W}$). We will explain this update procedure in more detail in the next two subsections.

Let $f_t \in \mathbb{R}$ be an incumbent objective value (Eq. (1)) in the t-th iteration. Then, we terminate the algorithm if the objective value is not sufficiently improved, namely

$$f_t > f_{t-1} - \varepsilon,$$

where $\varepsilon \in \mathbb{R}_+$ is a sufficiently small positive number.

3.2 Updating Neighborhoods

Our algorithm at each iteration updates the neighborhoods (i.e., $\boldsymbol{Z}^{(\mathrm{I})}$ and $\boldsymbol{Z}^{(\mathrm{J})}$) while keeping a given data matrix $\boldsymbol{W}$ fixed.

We first focus on the procedure of updating $\boldsymbol{Z}^{(\mathrm{I})} \in \{0,1\}^{|\mathcal{M}^{(\mathrm{I})}|\times n}$. The corresponding optimization problem can be decomposed into problems for each $i \in \mathcal{M}^{(\mathrm{I})}$ as

$$\begin{aligned}
\text{minimize} \quad & \sum_{k=1}^{n} d^{(\mathrm{I})}_{ik} z^{(\mathrm{I})}_{ik} && (11)\\
\text{subject to} \quad & z^{(\mathrm{I})}_{ii} = 0, && (12)\\
& \sum_{k=1}^{n} z^{(\mathrm{I})}_{ik} = K^{(\mathrm{I})}, && (13)\\
& \boldsymbol{z}^{(\mathrm{I})}_{i} \in \{0,1\}^{n}, && (14)
\end{aligned}$$

Algorithm 1 Alternating Optimization for Problem (2)–(10)

Input:
Incomplete data matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$,
Threshold for termination $\varepsilon \in \mathbb{R}_+$,
Instance–feature trade-off $\lambda \in [0,1]$,
Neighborhood sizes $K^{(\mathrm{I})} \in [n-1]$, $K^{(\mathrm{J})} \in [p-1]$.

Initialize:
Complete data matrix $\boldsymbol{W} \in \mathbb{R}^{n\times p}$, ▷ mean imputation
Initial objective value $f_0 \coloneqq +\infty$,
Iteration count $t \leftarrow 0$.

repeat
  Set $t \leftarrow t+1$.
  for all $i \in \mathcal{M}^{(\mathrm{I})}$ **do**
    Update $\boldsymbol{z}^{(\mathrm{I})}_i$ as in Eq. (15). ▷ updating $\boldsymbol{Z}^{(\mathrm{I})}$
  end for
  for all $j \in \mathcal{M}^{(\mathrm{J})}$ **do**
    Update $\boldsymbol{z}^{(\mathrm{J})}_j$ as in Eq. (20). ▷ updating $\boldsymbol{Z}^{(\mathrm{J})}$
  end for
  for all $(\alpha,\beta) \in \mathcal{M}$ **do**
    Update $w_{\alpha\beta}$ as in Eq. (21). ▷ updating $\boldsymbol{W}$
  end for
  Set $f_t \coloneqq f(\boldsymbol{W}, \boldsymbol{Z}^{(\mathrm{I})}, \boldsymbol{Z}^{(\mathrm{J})})$. ▷ Eq. (1)
until $f_t > f_{t-1} - \varepsilon$. ▷ termination condition

Output: Complete data matrix $\boldsymbol{W} \in \mathbb{R}^{n\times p}$.

where

$$d^{(\mathrm{I})}_{ik} \coloneqq \sum_{j=1}^{p} (w_{ij} - w_{kj})^2 \quad ((i,k) \in \mathcal{M}^{(\mathrm{I})} \times [n]),$$
$$\boldsymbol{z}^{(\mathrm{I})}_i \coloneqq (z^{(\mathrm{I})}_{ik})_{k\in[n]} \quad (i \in \mathcal{M}^{(\mathrm{I})}).$$

Problem (11)–(14) can be solved easily by setting $z^{(\mathrm{I})}_{ik} = 1$ in ascending order of $d^{(\mathrm{I})}_{ik}$ for $k \in [n]\backslash\{i\}$ until Eq. (13) is satisfied. Specifically, we define a bijective function for each $i \in \mathcal{M}^{(\mathrm{I})}$ as

$$\sigma : [n-1] \to [n]\backslash\{i\}$$

such that

$$d^{(\mathrm{I})}_{i\sigma(1)} \le d^{(\mathrm{I})}_{i\sigma(2)} \le \cdots \le d^{(\mathrm{I})}_{i\sigma(n-1)}.$$

We then set

$$z_{ik} = \begin{cases} 1 & \text{if } k \in \{\sigma(k') \mid k' \in [K^{(\mathrm{I})}]\},\\ 0 & \text{otherwise} \end{cases} \qquad (k \in [n]). \qquad (15)$$

We next move on to the procedure of updating $\boldsymbol{Z}^{(\mathrm{J})} \in \{0,1\}^{|\mathcal{M}^{(\mathrm{J})}|\times p}$. The corresponding optimization problem can be decomposed into problems for each $j \in \mathcal{M}^{(\mathrm{J})}$ as

$$\begin{aligned}
\text{minimize} \quad & \sum_{\ell=1}^{p} d^{(\mathrm{J})}_{j\ell} z^{(\mathrm{J})}_{j\ell} && (16)\\
\text{subject to} \quad & z^{(\mathrm{J})}_{jj} = 0, && (17)\\
& \sum_{\ell=1}^{p} z^{(\mathrm{J})}_{j\ell} = K^{(\mathrm{J})}, && (18)\\
& \boldsymbol{z}^{(\mathrm{J})}_{j} \in \{0,1\}^{p}, && (19)
\end{aligned}$$

where

$$d^{(\mathrm{J})}_{j\ell} \coloneqq \sum_{i=1}^{n} (w_{ij} - w_{i\ell})^2 \quad ((j,\ell) \in \mathcal{M}^{(\mathrm{J})} \times [p]),$$
$$\boldsymbol{z}^{(\mathrm{J})}_j \coloneqq (z^{(\mathrm{J})}_{j\ell})_{\ell\in[p]} \quad (j \in \mathcal{M}^{(\mathrm{J})}).$$

Problem (16)–(19) can be solved easily in the same manner as problem (11)–(14). In this case, we define a bijective function for each $j \in \mathcal{M}^{(\mathrm{J})}$ as

$$\sigma : [p-1] \to [p] \backslash \{j\}$$

such that

$$d^{(\mathrm{J})}_{j\sigma(1)} \leq d^{(\mathrm{J})}_{j\sigma(2)} \leq \cdots \leq d^{(\mathrm{J})}_{j\sigma(p-1)}.$$

We then set

$$z^{(\mathrm{J})}_{j\ell} = \begin{cases} 1 & \text{if } \ell \in \{\sigma(\ell') \mid \ell' \in [K^{(\mathrm{J})}]\}, \\ 0 & \text{otherwise} \end{cases} \qquad (\ell \in [p]). \tag{20}$$

3.3 Updating Missing Values

Our algorithm at each iteration updates a complete data matrix $\boldsymbol{W}$ while keeping given neighborhoods (i.e., $\boldsymbol{Z}^{(\mathrm{I})}$ and $\boldsymbol{Z}^{(\mathrm{J})}$) fixed. For each $(\alpha, \beta) \in \mathcal{M}$, differentiating the objective function in Eq. (1) with respect to $w_{\alpha\beta}$ yields the following first-order optimality condition:

$$2(1-\lambda)\Big(\sum_{k=1}^{n} z^{(\mathrm{I})}_{\alpha k}(w_{\alpha\beta} - w_{k\beta}) - \sum_{i \in \mathcal{M}^{(\mathrm{I})}} z^{(\mathrm{I})}_{i\alpha}(w_{i\beta} - w_{\alpha\beta})\Big)$$
$$+ 2\lambda\Big(\sum_{\ell=1}^{p} z^{(\mathrm{J})}_{\beta\ell}(w_{\alpha\beta} - w_{\alpha\ell}) - \sum_{j \in \mathcal{M}^{(\mathrm{J})}} z^{(\mathrm{J})}_{j\beta}(w_{\alpha j} - w_{\alpha\beta})\Big) = 0.$$

By collecting the terms containing $w_{\alpha\beta}$ onto the left-hand side, we obtain

$$\underbrace{\Big((1-\lambda)\Big(K^{(\mathrm{I})} + \sum_{i \in \mathcal{M}^{(\mathrm{I})}} z^{(\mathrm{I})}_{i\alpha}\Big) + \lambda\Big(K^{(\mathrm{J})} + \sum_{j \in \mathcal{M}^{(\mathrm{J})}} z^{(\mathrm{J})}_{j\beta}\Big)\Big) w_{\alpha\beta}}_{D_{\alpha\beta}} \quad \because \text{Eqs. (6)–(7)}$$
$$= \underbrace{(1-\lambda)\Big(\sum_{k=1}^{n} z^{(\mathrm{I})}_{\alpha k} w_{k\beta} + \sum_{i \in \mathcal{M}^{(\mathrm{I})}} z^{(\mathrm{I})}_{i\alpha} w_{i\beta}\Big) + \lambda\Big(\sum_{\ell=1}^{p} z^{(\mathrm{J})}_{\beta\ell} w_{\alpha\ell} + \sum_{j \in \mathcal{M}^{(\mathrm{J})}} z^{(\mathrm{J})}_{j\beta} w_{\alpha j}\Big)}_{N_{\alpha\beta}}.$$

Consequently, we derive an entry-wise analytical solution of missing values in matrix $\boldsymbol{W}$ as follows:

$$w_{\alpha\beta} = \frac{N_{\alpha\beta}}{D_{\alpha\beta}} \quad ((\alpha, \beta) \in \mathcal{M}). \tag{21}$$

3.4 Warm-Start Strategy

We develop a warm-start strategy to efficiently find a sequence of solutions for various neighborhood sizes $(K^{(\mathrm{I})}, K^{(\mathrm{J})}) \in [K_1] \times [K_2]$, where $K_1 \in [n-1]$ and $K_2 \in [p-1]$ are maximum neighborhood sizes.

Let $\widehat{\boldsymbol{W}}(k_1, k_2)$ denote a solution of matrix $\boldsymbol{W}$ to problem (2)–(10) with neighborhood sizes $(K^{(\mathrm{I})}, K^{(\mathrm{J})}) = (k_1, k_2)$. Our basic strategy is to speed up the computation of $\widehat{\boldsymbol{W}}(k_1, k_2)$ for $(k_1, k_2) \in [K_1] \times [K_2]$ by starting our alternating optimization algorithm (Algorithm 1) from

$$\frac{1}{2}\Big(\widehat{\boldsymbol{W}}(k_1 - 1, k_2) + \widehat{\boldsymbol{W}}(k_1, k_2 - 1)\Big) \tag{22}$$

as the initial solution. Our warm-start strategy for Algorithm 1 is described in Algorithm 2.

Algorithm 2 Warm-Start for Algorithm 1

Input:
Maximum neighborhood sizes $K_1 \in [n-1], K_2 \in [p-1]$.
Initialize:
Complete data matrix $\widehat{\boldsymbol{W}}(0,0) \in \mathbb{R}^{n \times p}$. ▷ mean imputation
for all $k_1 \in [K_1]$ **do**
  Compute $\widehat{\boldsymbol{W}}(k_1, 0)$ by starting Algorithm 1 from $\widehat{\boldsymbol{W}}(k_1 - 1, 0)$.
end for
for all $k_2 \in [K_2]$ **do**
  Compute $\widehat{\boldsymbol{W}}(0, k_2)$ by starting Algorithm 1 from $\widehat{\boldsymbol{W}}(0, k_2 - 1)$.
end for
for all $k_2 \in [K_2]$ **do**
  for all $k_1 \in [K_1]$ **do**
    Compute $\widehat{\boldsymbol{W}}(k_1, k_2)$ by starting Algorithm 1 from Eq. (22).
  end for
end for
Output: Complete data matrices $\widehat{\boldsymbol{W}}(k_1, k_2)$ for $(k_1, k_2) \in (\{0\} \cup [K_1]) \times (\{0\} \cup [K_2])$.

4. Experiments

In this section, we report experimental results of our method for missing value imputation. All computations were performed on a Windows computer with an Intel Core i9-9900K CPU (3.60 GHz) and 32 GB of RAM.

4.1 Datasets

We downloaded three real-world datasets from the UC Irvine Machine Learning Repository [19]. Table 1 lists the datasets, where n is the number of data instances, and p is the number of features. These datasets contain only numerical features with no missing values. We standardized each feature to have mean zero and standard deviation one.

Table 1 Datasets

Name	n	p	Original dataset
Rice	3810	7	Rice (Cammeo and Osmancik)[20]
Breast	569	30	Breast Cancer Wisconsin (Diagnostic) [32]
QSAR	1055	41	QSAR biodegradation [26]

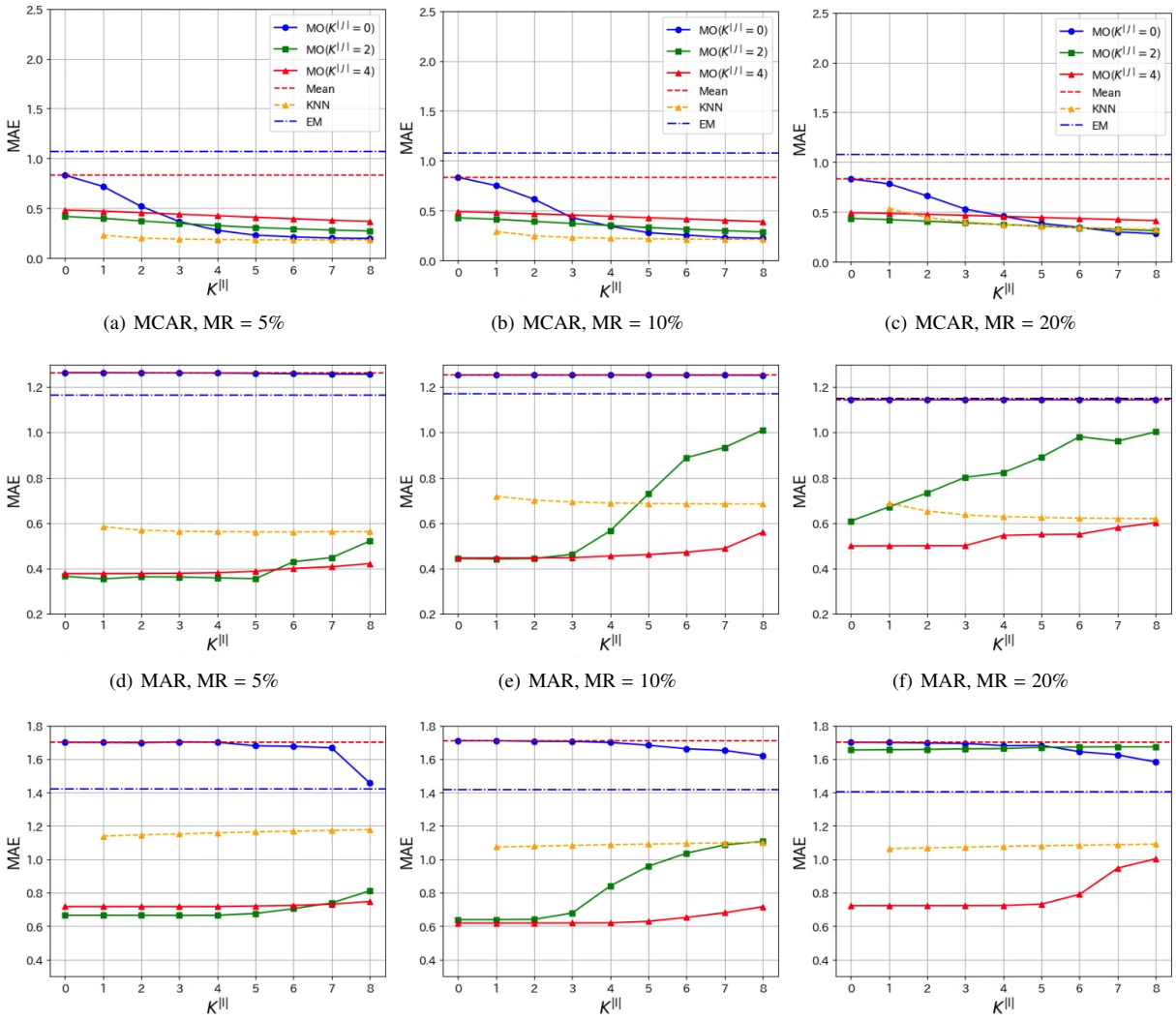
4.2 Missing Data Mechanisms

There are three types of missing data mechanisms [25]: missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR). Table 2 lists statistical assumptions of these missing data mechanisms, where $\boldsymbol{X}_O := (x_{ij})_{(i,j) \in O}$ and $\boldsymbol{X}_{\mathcal{M}} := (x_{ij})_{(i,j) \in \mathcal{M}}$ are observed and missing entries of data matrix $\boldsymbol{X}$, respectively.

Table 2 Missing data mechanisms

Name	Assumption
MCAR	$\Pr(\mathcal{M} \mid \boldsymbol{X}) = \Pr(\mathcal{M})$
MAR	$\Pr(\mathcal{M} \mid \boldsymbol{X}) = \Pr(\mathcal{M} \mid \boldsymbol{X}_O)$
NMAR	$\Pr(\mathcal{M} \mid \boldsymbol{X}) = \Pr(\mathcal{M} \mid \boldsymbol{X}_O, \boldsymbol{X}_{\mathcal{M}})$

By following prior studies [13], [15], we generated incomplete datasets based on the three mechanisms from the real-world complete datasets (Table 1), where $\bar{x}_j := (\sum_{i=1}^{n} x_{ij})/n$ for $j \in [p]$.



(a) MCAR, MR = 5% (b) MCAR, MR = 10% (c) MCAR, MR = 20%

(d) MAR, MR = 5% (e) MAR, MR = 10% (f) MAR, MR = 20%

(g) NMAR, MR = 5% (h) NMAR, MR = 10% (i) NMAR, MR = 20%

Fig. 1 Mean absolute errors on the Rice dataset

MCAR Mechanism: We randomly chose a subset $\mathcal{M} \subseteq [n] \times [p]$ of a specified size and treated these entries as missing values:

$$x_{ij} = \text{missing} \quad ((i,j) \in \mathcal{M}).$$

MAR Mechanism: We randomly sampled $o, m \in [p]$ $(o \neq m)$ and made missing entries in the m-th column as

$$x_{io} \leq \bar{x}_o \;\Rightarrow\; x_{im} = \text{missing} \quad (i \in [n]).$$

We repeated this operation by resampling $m \in [p]$ without replacement until the total number of missing entries reached a specified number.

NMAR Mechanism: We randomly sampled $m \in [p]$ and made missing entries in the m-th column as

$$x_{im} \leq \bar{x}_m \;\Rightarrow\; x_{im} = \text{missing} \quad (i \in [n]).$$

We repeated this operation by resampling $m \in [p]$ without replacement until the total number of missing entries reached a specified number.

4.3 Experimental Setup

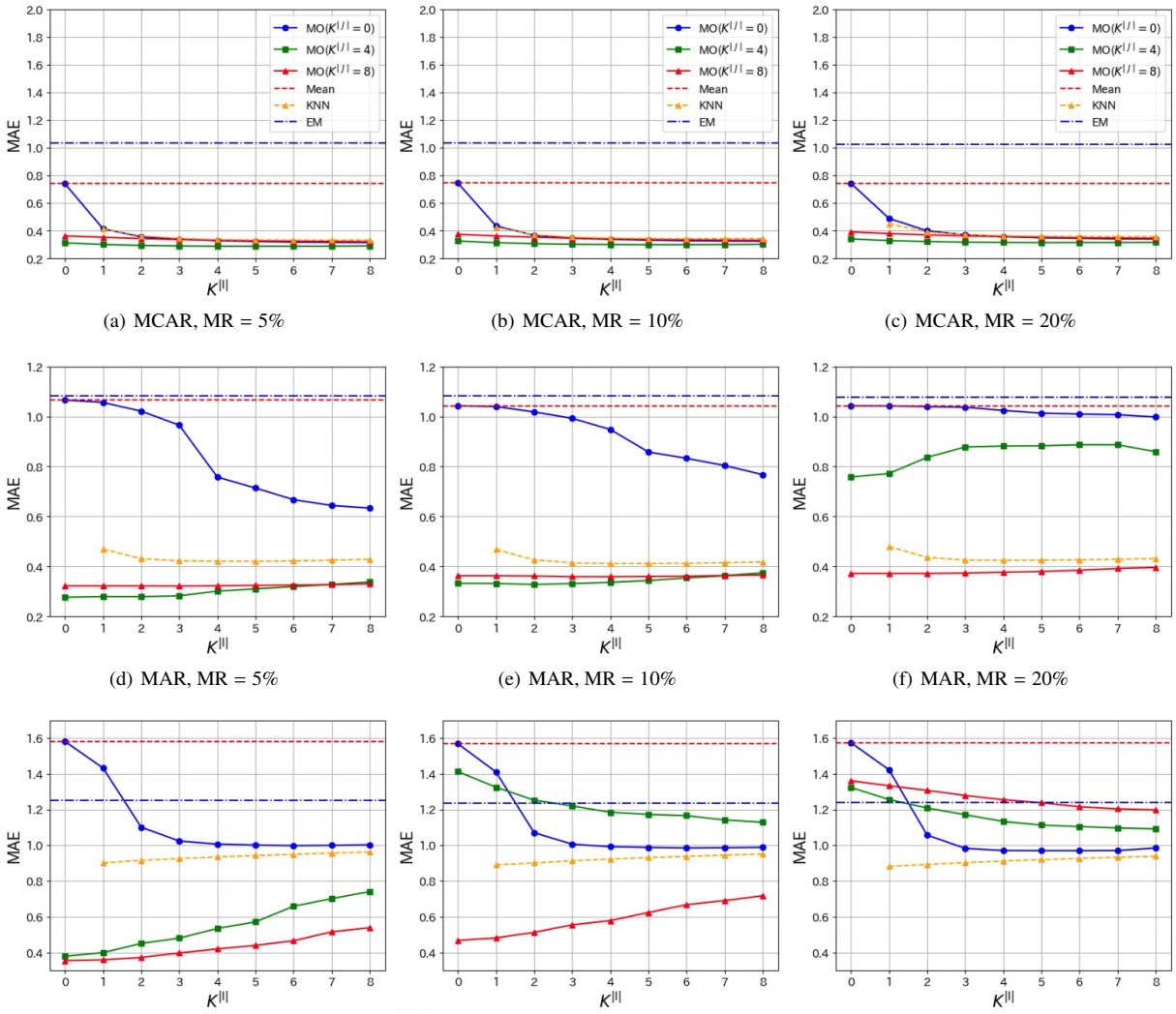
We compared the imputation accuracy of the following methods:

- **MO**($K^{(J)} = *$): Our method (Algorithm 1) for solving problem (2)–(10);
- **MO+WS**($K^{(J)} = *$): Our method (Algorithm 1) with the warm-start strategy (Algorithm 2) for solving problem (2)–(10);
- **Mean**: Imputing missing values with the corresponding column mean;
- **KNN**: Nearest neighbors algorithm [3] with the instance neighborhood size $K^{(I)}$ for missing value imputation;
- **EM**: EM algorithm [11], [25] for missing value imputation;

where $K^{(J)}$ is the neighborhood size of features. We set $\varepsilon = 0.01$ as the threshold of termination, and $\lambda = 0.5$ as the instance–feature trade-off in Algorithm 1. Considering the total number of features (i.e., p) in Table 1, we set $K^{(J)} \in \{0, 2, 4\}$ for the Rice dataset and $K^{(J)} \in \{0, 4, 8\}$ for the Breast and QSAR datasets. The KNN and EM algorithms were implemented using the `Impyute` library in the Python programming language.

The missing ratio (MR) of an incomplete data matrix is given by

$$\mathbf{MR} := \frac{|\mathcal{M}|}{np}.$$



(a) MCAR, MR = 5% (b) MCAR, MR = 10% (c) MCAR, MR = 20%

(d) MAR, MR = 5% (e) MAR, MR = 10% (f) MAR, MR = 20%

(g) NMAR, MR = 5% (h) NMAR, MR = 10% (i) NMAR, MR = 20%

Fig. 2 Mean absolute errors on the Breast dataset

For each pair of missing data mechanism (i.e., MCAR, MAR, or NMAR) and missings ratio (MR $\in \{5\%, 10\%, 20\%\}$), we generated incomplete datasets 10 times and averaged results over the corresponding 10 trials.

We used the mean absolute error (MAE) as a measure of inaccuracy in missing value imputation:

$$\mathbf{MAE} \coloneqq \frac{1}{|\mathcal{M}|} \sum_{(i,j)\in\mathcal{M}} \left| w_{ij} - x_{ij} \right|.$$

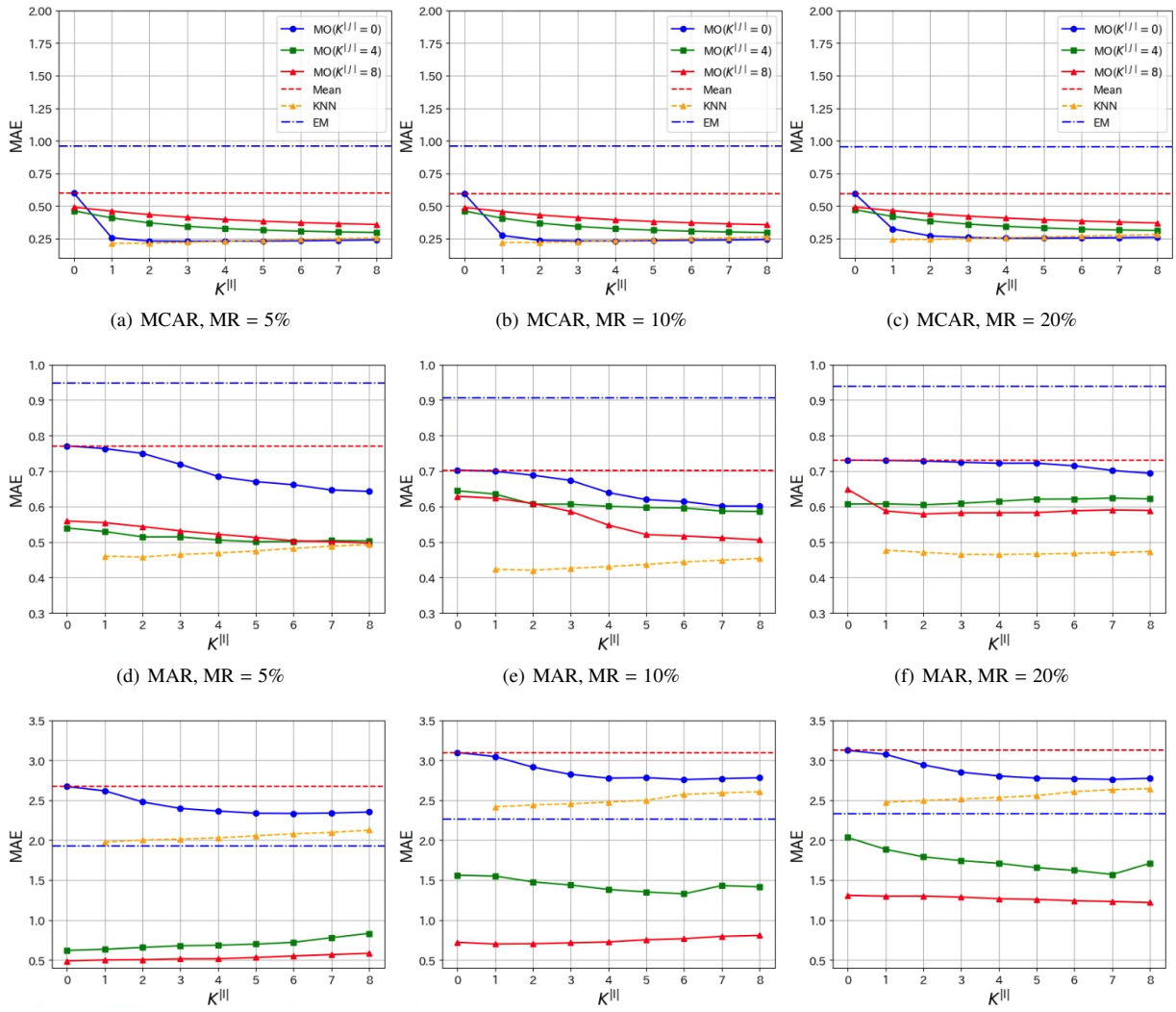
4.4 Results of the Imputation Accuracy

Figs. 1–3 show the mean absolute errors given by each imputation method on the Rice, Diag, and QSAR datasets, respectively. Here, the horizontal axis $K^{(\mathrm{I})}$ is the neighborhood size of instances employed in the MO and KNN methods. We have omitted the results of our method with the warm-start strategy (i.e., MO+WS) because there was no significant difference in the mean absolute errors between the MO and MO+WS methods. Note that the MO method with the feature neighborhood size $K^{(\mathrm{J})} = 0$ corresponds to the previous method proposed by Bertsimas et al. [5].

On the Rice Dataset (Fig. 1): For the MCAR mechanism, the KNN method performed best overall, and our MO methods performed better than the Mean and EM methods. Notably, our MO methods with $K^{(\mathrm{J})} \in \{2, 4\}$ achieved good imputation accuracy even when $K^{(\mathrm{I})}$ was very small. For the MAR and NMAR mechanisms, our MO methods with $K^{(\mathrm{J})} \in \{2, 4\}$ substantially outperformed other methods when MR $\in \{5\%, 10\%\}$, and our MO method with $K^{(\mathrm{J})} = 4$ remained the best accuracy even when MR = 20%.

On the Breast Dataset (Fig. 2): For the MCAR mechanism, our MO method with $K^{(\mathrm{J})} = 4$ outperformed other methods. Additionally, the MO methods with $K^{(\mathrm{J})} \in \{0, 8\}$ and the KNN method showed similar imputation accuracy when $K^{(\mathrm{I})}$ was large. For the MAR mechanism, our MO methods with $K^{(\mathrm{J})} \in \{4, 8\}$ outperformed other methods when MR $\in \{5\%, 10\%\}$, and our MO method with $K^{(\mathrm{J})} = 8$ remained the best accuracy even when MR = 20%. For the NMAR mechanism, our MO method with $K^{(\mathrm{J})} = 8$ attained great accuracy when MR $\in \{5\%, 10\%\}$, whereas the KNN method performed best when MR = 20%.

On the QSAR Dataset (Fig. 3): For the MCAR mechanism, the previous MO (with $K^{(\mathrm{J})} = 0$) and KNN methods performed best overall. For the MAR mechanism, the KNN method performed best overall, and our MO methods with $K^{(\mathrm{J})} \in \{4, 8\}$ per-



(a) MCAR, MR = 5% (b) MCAR, MR = 10% (c) MCAR, MR = 20%

(d) MAR, MR = 5% (e) MAR, MR = 10% (f) MAR, MR = 20%

(g) NMAR, MR = 5% (h) NMAR, MR = 10% (i) NMAR, MR = 20%

Fig. 3 Mean absolute errors on the QSAR dataset

formed better than other methods. For the NMAR mechanism, our MO methods with $K^{(\mathrm{J})} \in \{4, 8\}$ substantially outperformed other methods.

From these results on the imputation accuracy, we can conclude that our method with instance-and-feature neighborhoods performed relatively good on the whole. In particular, our method performed very well for the MAR and NMAR mechanisms when the missing rate was not large.

4.5 Results of the Computation Time

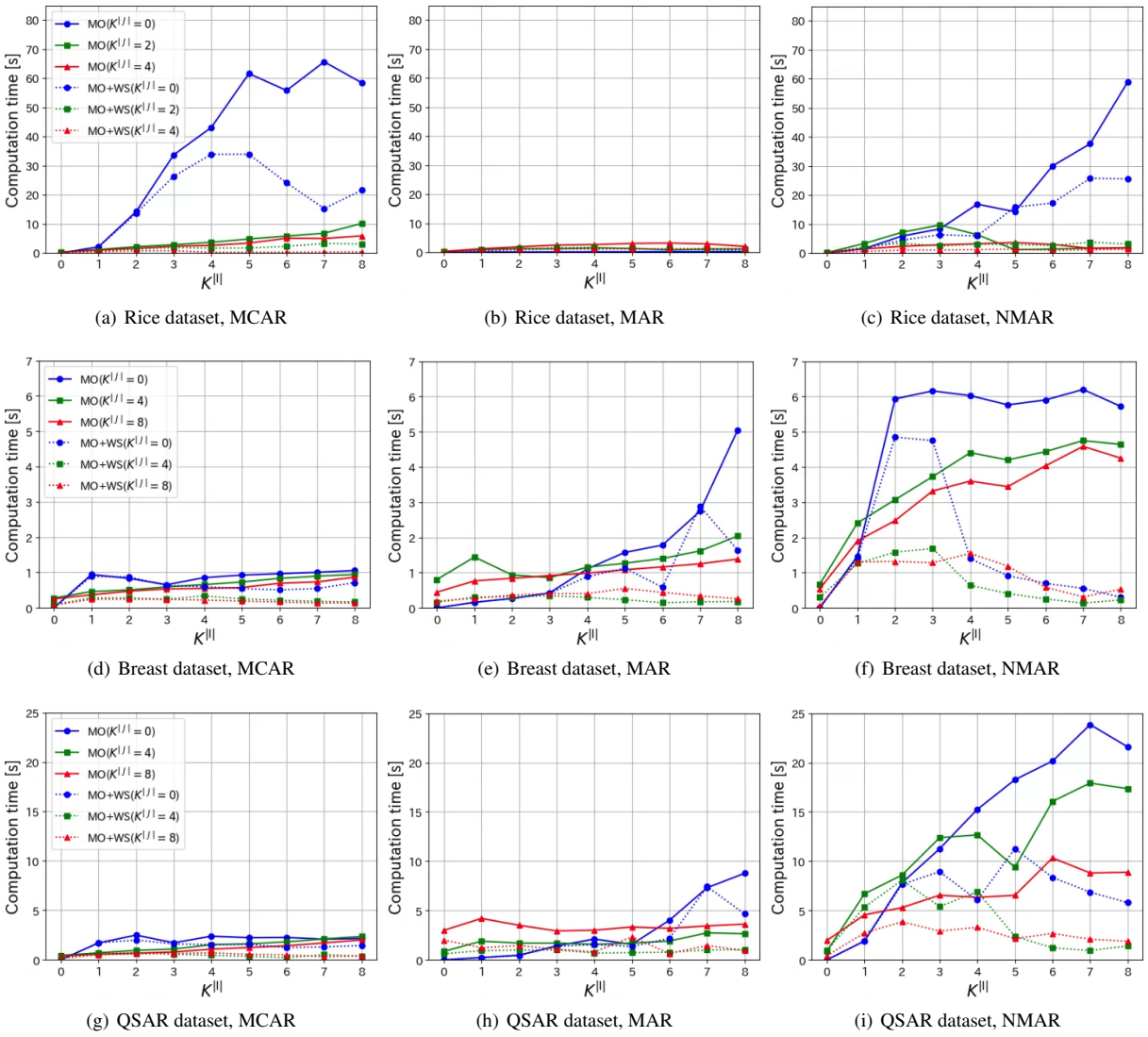
Fig. 4 shows the computation times (in seconds) required by our method with and without the warm-start strategy when the missing rate was MR = 20%. In some cases, the computation time of the MO methods (without the warm-start strategy) sharply increased as the instance neighborhood size $K^{(\mathrm{I})}$ increased. In contrast, the increase in computation time of the MO+WS methods was reduced by using the warm-start strategy. A typical example is Fig. 4(f): when $K^{(\mathrm{I})}$ was large, the computation times were much longer without the warm-start strategy than with the warm-start strategy.

5. Conclusion

We considered methods for missing value imputation using mathematical optimization based on the nearest neighbors algorithm. We formulated an optimization model using the neighborhoods of both instances and features to impute missing values. We designed an alternating optimization algorithm to find high-quality solutions to our optimization model. We also developed a warm-start strategy to efficiently find a sequence of solutions for various neighborhood sizes.

Experimental results demonstrate that our method performed very well especially for the MAR and NMAR missing data mechanisms when the missing rate was not large. Moreover, our warm-start strategy successfully reduced the computation time of the alternating optimization algorithm with large-sized neighborhoods. It is known that missing values can be particularly harmful for certain applications, especially when the distribution of missing entries is not uniform, as in the case of MAR and NMAR mechanisms [13]. This fact highlights the importance of our method being valid for MAR and NMAR mechanisms.

A future direction of study will be to develop an algorithm



(a) Rice dataset, MCAR (b) Rice dataset, MAR (c) Rice dataset, NMAR

(d) Breast dataset, MCAR (e) Breast dataset, MAR (f) Breast dataset, NMAR

(g) QSAR dataset, MCAR (h) QSAR dataset, MAR (i) QSAR dataset, NMAR

Fig. 4 Computation times (in seconds) required by our methods for the three datasets with MR = 20%

that finds a solution to our optimization problem with a proof of global optimality. Another direction for future research will be to extend our imputation method to collaborative data analysis on distributed datasets [16], [18], [35].

References

[1] Aggarwal, C. C. (2016). Recommender Systems. Springer International Publishing.

[2] Ayilara, O. F., Zhang, L., Sajobi, T. T., Sawatzky, R., Bohm, E., & Lix, L. M. (2019). Impact of missing data on bias and precision when estimating change in patient-reported outcomes from a clinical registry. Health and Quality of Life Outcomes, 17, 1–9.

[3] Batista, G. E., & Monard, M. C. (2003). An analysis of four missing data treatment methods for supervised learning. Applied Artificial Intelligence, 17(5-6), 519–533.

[4] Bertsimas, D., Orfanoudaki, A., & Pawlowski, C. (2021). Imputation of clinical covariates in time series. Machine Learning, 110(1), 185–248.

[5] Bertsimas, D., Pawlowski, C., & Zhuo, Y. D. (2018). From predictive methods to missing data imputation: An optimization approach. Journal of Machine Learning Research, 18(196), 1–39.

[6] Brás, L. P., & Menezes, J. C. (2007). Improving cluster-based missing value estimation of DNA microarray data. Biomolecular Engineering, 24(2), 273–282.

[7] van Buuren, S., & Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in R. Journal of Statistical Software, 45, 1–67.

[8] Camino, R. D., Hammerschmidt, C. A., & State, R. (2019). Improving missing data imputation with deep generative models. arXiv preprint arXiv:1902.10666.

[9] Chen, F., Wang, Y. C., Wang, B., & Kuo, C. C. J. (2020). Graph representation learning: A survey. APSIPA Transactions on Signal and Information Processing, 9, e15.

[10] Caruana, R. (2001). A non-parametric EM-style algorithm for imputing missing values. In International Workshop on Artificial Intelligence and Statistics (pp. 35–40). PMLR.

[11] Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B (Methodological), 39(1), 1–22.

[12] Emmanuel, T., Maupong, T., Mpoeleng, D., Semong, T., Mphago, B., & Tabona, O. (2021). A survey on missing data in machine learning. Journal of Big Data, 8, 1–37.

[13] Garciarena, U., & Santana, R. (2017). An extensive analysis of the interaction between missing data types, imputation methods, and supervised classifiers. Expert Systems with Applications, 89, 52–65.

[14] Honaker, J., King, G., & Blackwell, M. (2011). Amelia II: A program for missing data. Journal of Statistical Software, 45, 1–47.

[15] Huang, J., Keung, J. W., Sarro, F., Li, Y. F., Yu, Y. T., Chan, W. K., & Sun, H. (2017). Cross-validation based K nearest neighbor imputation for software quality datasets: An empirical study. Journal of Systems and Software, 132, 226–252.

[16] Imakura, A., & Sakurai, T. (2020). Data collaboration analysis framework using centralization of individual intermediate representations for distributed data sets. ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering, 6(2), 04020018.

[17] Jäger, S., Allhorn, A., & Bießmann, F. (2021). A benchmark for data

imputation methods. Frontiers in Big Data, 4, 693674.

[18] Kawakami, Y., Takano, Y., & Imakura, A. (2024). New solutions based on the generalized eigenvalue problem for the data collaboration analysis. arXiv preprint arXiv:2404.14164.

[19] Kelly, M., Longjohn, R., & Nottingham, K. The UCI Machine Learning Repository. `https://archive.ics.uci.edu`.

[20] Koklu, M., Cinar, I., & Taspinar, Y. S. (2021). Classification of rice varieties with deep learning methods. Computers and Electronics in Agriculture, 187, 106285.

[21] Lalande, F., & Doya, K. (2022). Numerical data imputation: Choose kNN over deep learning. In International Conference on Similarity Search and Applications (pp. 3–10). Cham: Springer International Publishing.

[22] Liew, A. W. C., Law, N. F., & Yan, H. (2011). Missing value imputation for gene expression data: Computational techniques to recover missing data from available information. Briefings in Bioinformatics, 12(5), 498–513.

[23] Li, S. C. X., Jiang, B., & Marlin, B. (2019). MisGAN: Learning from incomplete data with generative adversarial networks. In International Conference on Learning Representations.

[24] Lin, W. C., & Tsai, C. F. (2020). Missing value imputation: A review and analysis of the literature (2006–2017). Artificial Intelligence Review, 53, 1487–1509.

[25] Little, R. J., & Rubin, D. B. (2019). Statistical Analysis with Missing Data (Vol. 793). John Wiley & Sons.

[26] Mansouri, K., Ringsted, T., Ballabio, D., Todeschini, R., & Consonni, V. (2013). Quantitative structure–activity relationship models for ready biodegradability of chemicals. Journal of Chemical Information and Modeling, 53(4), 867–878.

[27] McCoy, J. T., Kroon, S., & Auret, L. (2018). Variational autoencoders for missing data imputation with application to a simulated milling circuit. IFAC-PapersOnLine, 51(21), 141–146.

[28] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems, 26.

[29] Oba, S., Sato, M. A., Takemasa, I., Monden, M., Matsubara, K. I., & Ishii, S. (2003). A Bayesian missing value estimation method for gene expression profile data. Bioinformatics, 19(16), 2088–2096.

[30] Shang, C., Palmer, A., Sun, J., Chen, K. S., Lu, J., & Bi, J. (2017). VIGAN: Missing view imputation with generative adversarial networks. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 766–775). IEEE.

[31] Stekhoven, D. J., & Bühlmann, P. (2012). MissForest—Nonparametric missing value imputation for mixed-type data. Bioinformatics, 28(1), 112–118.

[32] Street, W. N., Wolberg, W. H., & Mangasarian, O. L. (1993). Nuclear feature extraction for breast tumor diagnosis. In Biomedical Image Processing and Biomedical Visualization (Vol. 1905, pp. 861–870). SPIE.

[33] Sun, Y., Li, J., Xu, Y., Zhang, T., & Wang, X. (2023). Deep learning versus conventional methods for missing data imputation: A review and comparative study. Expert Systems with Applications, 227, 120201.

[34] Wang, J., De Vries, A. P., & Reinders, M. J. (2006). Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 501–508).

[35] Yanagi, T., Ikeda, S., Sukegawa, N., & Takano, Y. (2024). Privacy-preserving recommender system using the data collaboration analysis for distributed datasets. arXiv preprint arXiv:2406.01603.

[36] Yoon, J., Jordon, J., & Schaar, M. (2018). Gain: Missing data imputation using generative adversarial nets. In International Conference on Machine Learning (pp. 5689–5698). PMLR.