


Article

A Matrix Completion Method for Imputing Missing Values of Process Data

Xinyu Zhang, Xiaoyan Sun, Li Xia, Shaohui Tao *  and Shuguang Xiang *

Institute of Process Systems Engineering, Qingdao University of Science and Technology, Qingdao 266042, China
* Correspondence: tau@qust.edu.cn (S.T.); xsg@qust.edu.cn (S.X.)

Abstract: Real-time process data are the foundation for the successful implementation of intelligent manufacturing in the chemical industry. However, in the actual production process, process data may randomly be missing due to various reasons, thus affecting the practical application of intelligent manufacturing technology. Therefore, this paper proposes the application of appropriate matrix completion algorithms to impute the missing values of real-time process data. Considering the characteristics of online missing value imputation problems, this paper proposes an improved method for a matrix completion algorithm that is suitable for real-time missing data imputation. By utilizing real device data, this paper studies the impact of algorithm parameters on the effect of missing value imputing and compares it with several classical missing value imputing methods. The results show that the introduced method achieves higher accuracy in data imputation compared to the baseline method. Furthermore, the proposed enhancement significantly improves the speed performance of algorithms.

Keywords: missing data; matrix completion; chemical process data; data cleaning

1. Introduction

As intelligent manufacturing in the chemical industry gains momentum, the significance of real-time process data escalates. However, transmission interruptions, sensor failures, or database issues may cause random missing data in real-time process data during the actual production process [1], thereby reducing the representativeness of real-time data and affecting the implementation of intelligent manufacturing in the chemical industry [2]. Therefore, the imputation of missing values is critically essential for the successful execution of intelligent manufacturing within the chemical industry.

We commonly use three methods for the analysis of data with missing values: the deletion method, the imputation method, and the model-based method [3]. Commonly used deletion methods for handling missing data include listwise and pairwise. Listwise deletion involves removing all data associated with time points that exhibit anomalies. This method may lead to the deletion of significant amounts of data, thus diminishing statistical efficiency and potentially introducing more uncertainty and bias into parameter estimates [4]. On the other hand, pairwise deletion may lead to inconsistent data lengths due to the varying time points of the deleted data, making it impossible to reconstruct and represent the complete dataset as a matrix. This inconsistency affects the application of numerous algorithms, complicating subsequent tasks in process modeling and statistical inference for monitoring.

The data processed using deletion methods are illustrated in Figure 1.

Thus, when a significant amount of data is missing and the correlation between variables is affected, it is not advisable to use deletion methods. To maximize the number of samples and preserve statistical characteristics for subsequent data mining, it is advisable to employ the strategy of estimating and imputing missing data. While manually selecting values for imputation is possible, automated methods become indispensable for processing



Citation: Zhang, X.; Sun, X.; Xia, L.; Tao, S.; Xiang, S. A Matrix Completion Method for Imputing Missing Values of Process Data. *Processes* **2024**, *12*, 659. <https://doi.org/10.3390/pr12040659>

Academic Editor: Xiong Luo

Received: 27 February 2024

Revised: 21 March 2024

Accepted: 23 March 2024

Published: 26 March 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

the real-time data involved in the deployment of intelligent manufacturing [5]. Common strategies for imputation involve using statistical values, such as the mean or median values of variables [6], or employing a similar pattern from complete data, such as hot-deck imputation [2]. Model-based methods often involve algorithms, such as the Maximum Likelihood (ML) algorithm [7] and the Expectation-Maximization (EM) algorithm [8].

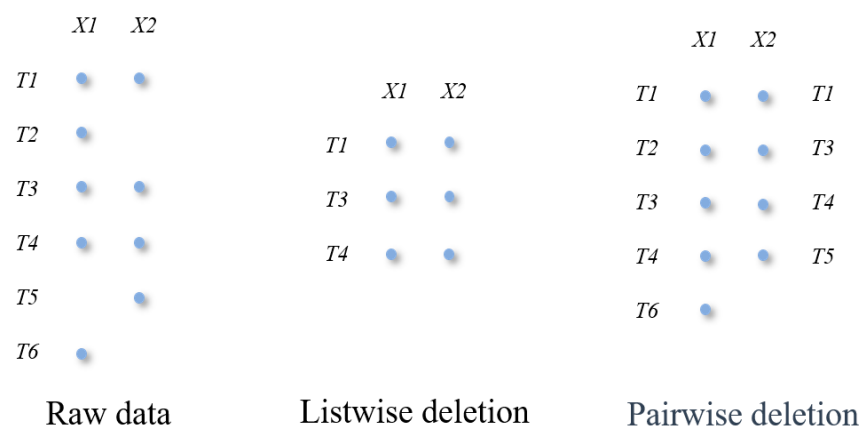


Figure 1. Listwise deletion and pairwise deletion.

To address the issue of missing values in chemical process data, this paper employs a method from machine learning: the Matrix Completion Method (MCM). Due to its remarkable ability to discover and leverage latent patterns or structures, the MCM has found applications in recommendation systems, image restoration, analytical chemistry, thermodynamics, and quantum chemistry. In chemical engineering, the MCM has been used to predict the thermodynamic properties and the models of mixtures. For example, it has been incorporated into the UNIQUAC model to predict activity coefficients at any temperature and composition, surpassing the performance of the best physical models for predicting activity coefficients [9]. Additionally, the MCM has been used to predict Henry's law coefficients based on the sparsity of experimental data matrices, outperforming traditional state equations [10]. It has also found application in calculations involving quantum and variational effects [11].

This paper will introduce existing methods for imputing missing data in Section 2. Section 3 briefly introduces the principle of matrix completion and improves the existing Matrix Completion Method according to the characteristics of the online data imputing problem. Section 4 will evaluate the efficacy of the MCM in processing missing data, thus providing a comparative discussion with other methods. Finally, Section 5 concludes this study.

2. Established Methods for Data Imputation

In handling chemical process data, routine methods for imputing missing values involve utilizing existing data for imputation or employing model-based methods. The former includes mean imputation, median imputation, and hot-deck imputation. These methods aim to recover or estimate missing data values based on available information, thus enhancing the accuracy of portraying the genuine characteristics of the data for further analysis and exploration. The latter encompasses various model-based methods, including the ML and EM algorithms. These methods impute missing values by establishing models and generating data that adhere to specific distributions for imputation. The selection of each method depends on the particular data characteristics and analytical requirements.

2.1. Mean and Median Replacement

Imputing missing data using the mean of data from different time points on the same sensor offers fast processing speed. Let \bar{x}_{obs} represent the mean of the observed data. The missing data x_{mis} are formulated as follows:

$$x_{\text{mis}} = \bar{x}_{\text{obs}} \quad (1)$$

Imputing missing data using the median of data from different time points on the same sensor provides a processing speed similar to mean imputation. Let Me_{obs} represent the median of the observed data. The missing data x_{mis} are given by:

$$x_{\text{mis}} = Me_{\text{obs}} \quad (2)$$

2.2. Hot-Deck Replacement

The hot-deck imputation method for processing missing data involves identifying an object in the existing data that is most similar to the missing data and then imputing the missing values with the values from this similar object. In defining similarity, this experiment treats the data obtained from each time point as individual entities. Comparing the absolute average differences in the observed data x_{obs} among different entities serves as the measure for determining similarity. Assuming there are n sensors of observed data from time point 1 and time point 2, we can express their level of similarity X_s as follows:

$$X_s = \frac{\sum |x_{\text{obs}1} - x_{\text{obs}2}|}{n} \quad (3)$$

When X_s is small, indicating a higher level of similarity, the strategy involves using $x_{\text{obs}2}$ from time point 2 to impute the missing data $x_{\text{mis}1}$ for time point 1.

$$x_{\text{mis}1} = x_{\text{obs}2} \quad (4)$$

2.3. Model-Based Methods

Model-based methods include ML (Maximum Likelihood) and EM (Expectation-Maximization) algorithms. The ML algorithm assumes that we have a dataset X with a distribution described by parameters Θ . The parameter Θ is dictated by the statistical distribution of the data. Given observed data x_{obs} and missing data x_{mis} , the marginal probability density of x_{obs} is defined as follows:

$$P(x_{\text{obs}} | \Theta) = \int P(x_{\text{obs}}, x_{\text{mis}} | \Theta^{(m)}) dx_{\text{mis}} \quad (5)$$

The likelihood definition of Θ based on x_{obs} is as follows:

$$L(x_{\text{obs}} | \Theta) \propto P(x_{\text{obs}} | \Theta) \quad (6)$$

If the likelihood function is differentiable and Θ is known, the estimate of x_{mis} can be obtained by solving the problem illustrated in Equation (7):

$$\frac{\partial \log[L(Y_{\text{obs}} | \Theta)]}{\partial \Theta} = 0 \quad (7)$$

The EM algorithm consists of two steps: an E-step and an M-step. In the E-step, the expectation of the log-likelihood function is calculated. This expectation is equivalent to the conditional distribution of x_{mis} given x_{obs} under the estimation of $\Theta^{(m)}$, and it is expressed as follows:

$$E(x_{\text{mis}} | x_{\text{obs}}, \Theta^{(m)}) = E[\log L(x_{\text{obs}}, x_{\text{mis}} | \Theta^{(m)})] \quad (8)$$

During the M-step, the goal is to determine the parameters $\Theta^{(m+1)}$ that maximize the expectation E. The formula is given by:

$$\Theta^{(m+1)} = \underset{\Theta}{\operatorname{argmax}} E(\Theta | \Theta^{(m)}) \quad (9)$$

The EM algorithm iterates between these two steps until the estimated values converge.

The methods previously discussed, such as mean and median imputation, are simple to calculate but lack accuracy. These methods do not utilize information beyond the data collected by the current sensor, leading to inefficient utilization of information. Hot-deck imputation utilizes information beyond the current sensor. It still uses existing values to impute missing ones, potentially resulting in a single observed value being used to impute multiple time points. For real-time data, it is difficult for us to predict their distribution in advance, and the formulation of the likelihood function in the ML algorithm becomes complex to derive without a definite data distribution. While the EM algorithm does not require the likelihood function expression, it still assumes a specific data distribution and has a slow computation speed.

The MCM can effectively address the issues with the previously mentioned algorithms. The MCM leverages the hidden relationships within the data to calculate new values for imputing missing ones, thus eliminating the need to derive expressions or make assumptions about data distribution.

3. Data Imputation with Improved Matrix Completion Methods

3.1. Principles of Matrix Completion

Matrix completion problems can be considered a type of matrix recovery problem. Specifically, they involve restoring the missing elements in the matrix based on a limited number of known elements [12,13]. Typical applications include estimating missing data, generating recommendations, uncovering hidden structures, conducting image restoration, and classification [14–16].

Matrix completion aims to recover the entire matrix using a limited amount of observed data. We record the positions of the observed data with Ω and search for a matrix X that has the same values as the input data matrix M at the known positions and has the minimum rank. Low-rank matrices preserve a substantial amount of redundant information, which we can use to recover the missing data in the matrix. The operator P_{Ω} sets all elements not in Ω in the matrix to zero, and the elements in Ω are $P_{\Omega}(X) = X$. Therefore, we can formulate the above problem as follows:

$$\underset{X}{\operatorname{minimize}} \operatorname{rank}(X), \text{ s.t. } P_{\Omega}(X) = P_{\Omega}(M) \quad (10)$$

However, the previously discussed problem is an NP-hard problem [17], which is computationally complex and difficult to solve directly. Consequently, it requires replacement with a problem that is easier to solve.

There are various models for matrix completion. These include, for example, models based on nuclear norm relaxation, in which Ma [18] and Toh [19] relaxed the standard problem into a matrix LASSO model. Additionally, the SVT algorithm proposed by Cai et al. [20] enhances the stability of solving matrix completion problems. Another model is based on matrix factorization. For example, the SOR algorithm proposed by Wen et al. [21] can handle large-scale matrix completion problems faster than traditional nuclear norm minimization algorithms. However, this approach requires an initial rank estimate, and due to the non-convex nature of the model, it cannot assure global convergence. Finally, models based on non-convex function relaxation are also an option. These include, for example, the algorithm proposed by Nie et al. [22] using Schatten p-norm and Lp-norm, and the FGSR algorithm proposed by Fan et al. [23]. Moreover, the FGSR algorithm avoids

SVD decomposition, resulting in higher computational efficiency, and is insensitive to the choice of initial rank.

Due to the challenge of estimating the initial rank of the target data, this study selects the SVT (Singular Value Thresholding) algorithm, which does not require initial rank estimation, and the FGSR (Factor Group-Sparse Regularization) algorithm, which is not sensitive to the initial rank. We improve these algorithms by incorporating features for online use to impute missing values.

The SVT algorithm applies convex relaxation to the problem (10), transforming it into the following problem:

$$\underset{X}{\text{minimize}} \|X\|_{*}, \text{ s.t. } P_{\Omega}(X) = P_{\Omega}(M) \quad (11)$$

This optimization problem involves regularization, resulting in the construction of a Lagrangian function. Finally, we use the alternating iterative method to solve the optimization problem, expressing it in the following form:

$$\begin{cases} X^k = D_{\tau}(Y^{k-1}) \\ Y^k = Y^{k-1} + \delta_k P_{\Omega}(M - X^k) \end{cases} \quad (12)$$

In this context, Y is the Lagrange multiplier, δ_k is the step size, and the operation $D_{\tau}(Y^{k-1})$ is represented by the following equation:

$$D_{\tau}(Y^{k-1}) = \begin{cases} [U, S, V] = \text{SVD}(Y^{k-1}) \\ S = \text{sgn}(S) \cdot \max(|S| - \tau, 0) \\ X^k = U * S * V^T \end{cases} \quad (13)$$

where ε is the convergence error, the condition for ending the iteration is:

$$\frac{\|P_{\Omega}(M - X^k)\|_F}{\|P_{\Omega}M\|_F} < \varepsilon \quad (14)$$

The FGSR algorithm uses other parameters as proxies for rank, transforming problem (10) into the following form:

$$\underset{X}{\text{minimize}} \text{FGSR}(X), \text{ s.t. } P_{\Omega}(X) = P_{\Omega}(M) \quad (15)$$

FGSR(X) can be expressed as follows:

$$\text{FGSR}(X) = \frac{2}{3\alpha^{1/3}} \min_{AB=X} \|A\|_{2,1} + \frac{\alpha}{2} \|B\|_F^2 \quad (16)$$

A and B can be represented as follows:

$$\begin{cases} [U_X, S_X, V_X] = \text{SVD}(X) \\ A = \alpha^{1/3} U_X S_X^{2/3} \\ B = \alpha^{-1/3} S_X^{1/3} V_X^T \end{cases} \quad (17)$$

The problem can be represented as follows:

$$\underset{X}{\text{minimize}} \|A\|_{2,1} + \frac{\alpha}{2} \|B\|_F^2, \text{ s.t. } X = AB, P_{\Omega}(X) = P_{\Omega}(M) \quad (18)$$

We can solve the matrix completion problem by addressing the problem (18).

3.2. Data Imputation Based on Matrix Completion

The MCM is used to impute missing values within chemical processes. The fundamental principle is that data from chemical processes include control variable data X_{con} and display variable data X_{var} . The data are in matrix form, with one dimension representing time and the other representing sensor identification. The display variable data X_{var} are returned after being collected by sensors at various positions in the production device, which include observed data x_{obsv} and missing data x_{mis} caused by transmission or sensor failure. Control variable data X_{con} are set manually and contain observed data x_{obsc} . Consequently, the raw data can be perceived as a matrix M with missing elements composed of x_{obsv} , x_{obsc} , and x_{mis} . The form of matrix M is shown in Figure 2.

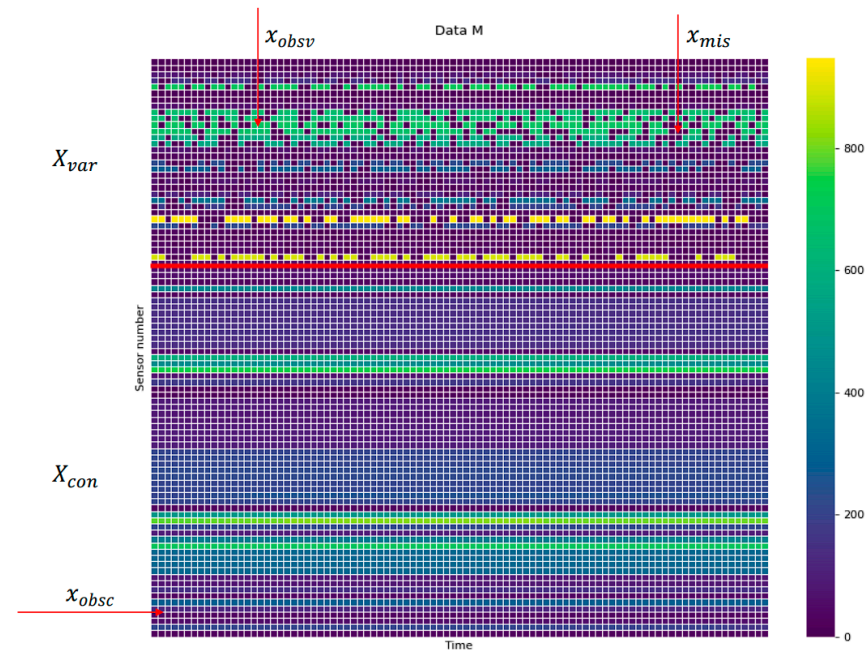


Figure 2. The form of input matrix data M , composed of x_{obsv} , x_{obsc} , and x_{mis} .

From this perspective, the problem of filling in missing values in chemical process data can be transformed into a matrix completion problem, as follows:

$$X = f_{MCM}(M) \tag{19}$$

Here, X represents the dataset after filling, and the specific process of data processing is shown in Figure 3:

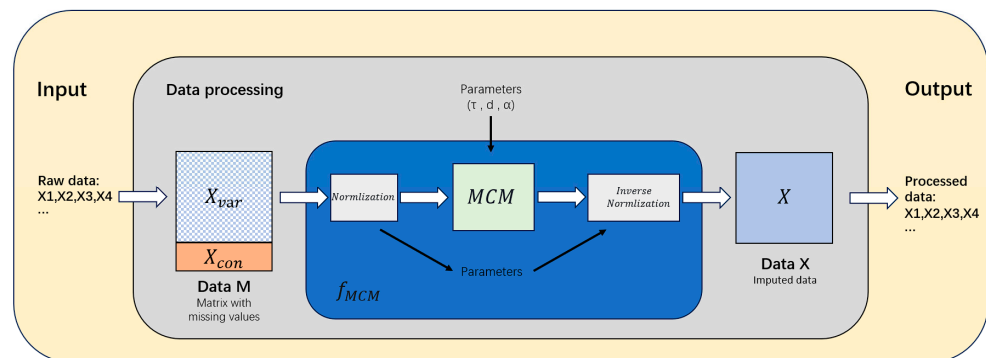


Figure 3. Flowchart of the Matrix Completion Method for imputing missing data.

Given that the equipment in chemical plants operates continuously, generating a constant stream of new data, we have adopted a moving window strategy for online data cleansing in this study. A moving window can reduce the size of the input matrix, thereby accelerating the speed of matrix completion for data processing.

In terms of the selection of parameters used in the algorithm, considering the balance between accuracy and computation time, the convergence error ε of the algorithm is set to $\varepsilon < 10^{-6}$, and the maximum number of iterations is set to 300.

The parameters of the SVT algorithm include the step size δ and the parameter τ . The conventional range of step size δ is 0~2, and this study uses the middle value $\delta = 1$. And, the parameter τ is selected empirically, following reference [20]. The literature sets the parameter τ for an $n \times n$ matrix as $5n$. In this study, for a matrix of size $n_1 \times n_2$, the parameter τ is determined as $5 \times \text{qrt}(n_1 \times n_2)$.

For the FGSR algorithm, parameters include λ , d , and α . Step size λ is set at 0.3. Parameter d represents the rank of the factor matrix after matrix decomposition, and its selection range is $d \leq \min(n_1, n_2)$, where n_1 and n_2 are the row and column numbers of the matrix. In this study, $d = \min(n_1, n_2)$. Parameter α , which adjusts the scale of the two-factor matrices following matrix decomposition, is chosen from the range $\alpha > 0$. In this experiment, we set $\alpha = 2$. The reasons for parameter selection refer to Section 4.3.

3.3. Improved Matrix Completion for Online Computing

In the application of the matrix completion algorithm, there is a need to initialize some data involved in the iterative computation. When processing data using a sliding window, most of the data in two adjacent windows are the same. Consequently, we consider using the data from the last iteration of the previous window's computation as the initial data for the next window to reduce the number of iterative computations for the new window.

We posit that the data window currently under processing is the i -th window. The D_{τ} operation from Equation (13) of the SVT algorithm is improved, leading to the $D_{n_{\tau}}$ operation, as shown in Equation (20).

$$D_{n_{\tau}}(Y^{k-1}) = \begin{cases} [U, S, V] = \text{SVD}(Y^{k-1}), & \text{if } i = 1, 3, 5 \dots \\ [U, S, V] = U^{i-1}, S^{i-1}, V^{i-1}, & \text{if } i = 2, 4, 6 \dots \\ S = \text{sgn}(S) \cdot \max(|S| - \tau, 0) \\ X^k = U * S * V^T \end{cases} \quad (20)$$

U^{i-1} , S^{i-1} , and V^{i-1} represent the final iteration of the prior window. By implementing this improvement and capitalizing on the advantages of online processing, we minimize the number of SVD decompositions, thus lowering the overall time expenditure while ensuring precision. We will identify the SVT algorithm incorporating these changes as the ISVT (Improved Singular Value Thresholding) algorithm in the following text.

For the FGSR algorithm, we begin iterative computation upon each window transition and use A_{i-1} and B_{i-1} from the concluding iteration of the prior window when initializing matrices A_i and B_i for the i -th window's computation ($i \neq 1$) according to (17). The modified initialization of A and B can be denoted by Equation (21).

$$\begin{cases} [A, B] = [A_{i-1}, B_{i-1}], & \text{if } i \neq 1 \\ [U_X, S_X, V_X] = \text{SVD}(X), & \text{if } i = 1 \\ A = \alpha^{\frac{1}{3}} U_X S_X^{\frac{2}{3}} \\ B = \alpha^{\frac{1}{3}} S_X V_X^T \end{cases} \quad (21)$$

Such a practice can effectively decrease the frequency of SVD decompositions and accelerate the computational speed. In the subsequent sections, we will refer to the modified FGSR algorithm as the IFGSR (Improved Factor Group-Sparse Regularization) algorithm.

4. Performance Evaluations

4.1. Studied Case

The experimental data are derived from the DCS system of an atmospheric–vacuum distillation unit in a particular refinery. The atmospheric–vacuum distillation unit, an essential device that finds wide application in areas including petroleum and chemical engineering, plays a significant role in improving oil refining optimization and economic efficiency.

The DCS system records assorted data from the apparatus, including the feed temperature, tower top pressure, and side draw flow rate of the distillation column within the unit. The data referenced in this paper are logged every five minutes, with the initial data being complete. The initial data are divided into two groups: control variable data X_{con} and display variable data X_{var} . The temperature (TI), pressure (PI), and flow rate (FI) data gathered by the DCS serve as display variable data X_{var} , while the input data PIC, TIC, and FIC for the control system function as control variable data X_{con} .

All data preprocessing and subsequent processing tasks are accomplished using MatlabR2021b. The program is operated on a laptop with a 3.20 GHz CPU and 16 GB of memory.

4.2. Evaluation Procedures

This study selects three common methods for processing missing values: mean imputation, median imputation, and hot-deck imputation. Additionally, four matrix completion algorithms, SVT, ISVT, FGSR, and IFGSR, are used to impute the missing values.

In the experiment, we selected data from 600 time points, which formed an original data matrix of size $n_s \times 600$, where n_s represents the number of sensors. The window size for processing the data is determined by the matrix's row count, which corresponds to the sensor number n_s . The window size ranges from $[0.8n_s]$ to $[2.0n_s]$, where $[\cdot]$ represents rounding up. The missing rate for variable data ranges from 10% to 80%, and the positions of the missing data are randomly determined.

The number of sensors displaying variable data for temperature, pressure, and flow rate differs. The specific details are shown in Table 1.

Table 1. Table of variable types and number of sensors.

Variable Type	Variable Data	Control Data	Total
Temperature	245	25	270
Pressure	38	17	55
Flow rate	33	60	93

As different variables have data of different scales and the range of data values varies, we need to perform experiments on each variable independently to evaluate the performance of algorithms.

The detailed steps of the experiment are as follows:

1. First, we convert the complete dataset into a matrix form suitable for algorithm processing. Each column of the data represents the measured values of the variables collected at the current time, and each row represents the sensor number transmitting these variable data.
2. The experiment determines the relevant parameters and the size of the moving window.
3. We set the missing rate and randomly generate missing data in the complete display variable data X_{var} . After preprocessing, the data we obtain will serve as experimental data.
4. We use the MCM to fill in the experimental data.
5. We compare the output with the original data to evaluate the effects of different methods.

Concerning the selection of normalization methods, this experiment opted for quantile normalization. The formula for this approach is provided as follows:

$$X_{scaled} = \frac{X - Q_1(X)}{Q_3(X) - Q_1(X)} \quad (22)$$

where X_{scaled} represents the normalized data, X denotes the original data, and Q_1 and Q_3 are quartiles. Quantile normalization scales data using quartiles, which effectively reduces the impact of outliers on the data.

The evaluation criterion employed is *MAPE* (Mean Absolute Percentage Error), which measures the relative error between imputed values and actual values. *MAPE* results are expressed in percentage, with smaller values indicating higher prediction accuracy. When *MAPE* equals 0, it signifies perfect accuracy in imputed values. The average *MAPE* for all data within a single time point represents the *MAPE* for that time point, denoted as $MAPE_t$. Its formula is as follows:

$$MAPE_t = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{X_i - X_p}{X_i} \right| \quad (23)$$

where X_i represents the actual values of the data, X_p represents the imputed values after MCM processing, and n represents the total count of data points. Subsequently, the *MAPE* for all time points is computed, denoted as $MAPE_a$, with the following formula:

$$MAPE_a = \frac{\sum_{t=1}^{n_t} MAPE_t}{n_t} \quad (24)$$

where n_t represents the number of time points after matrix completion processing.

Finally, using *MAPE* as the standard for evaluating the accuracy of missing value imputation, the MCM is compared with other methods.

4.3. Results and Discussion

In the section on results and the discussion, we divide the algorithms into the MCM, statistical value imputation methods, and the hot-deck method. The initial step involves performing a sensitivity analysis of the parameters within the four algorithms implemented in the MCM.

For the SVT and ISVT algorithms, parameter testing is conducted using the SVT algorithm and by applying the same parameter settings to both in subsequent experiments. We use the standard parameter setting range from reference [20] to adjust and test the value of parameter τ and the size of step length δ . Figure 4 illustrates the results of the test.

As depicted in Figure 4, the errors in pressure and temperature data tend to stabilize when $\delta > 0.25$. Similarly, the error in flow rate data stabilizes after $\delta > 0.75$, reaching a lower value around $\delta = 1$. Therefore, we choose $\delta = 1$ as the step size for the SVT and ISVT algorithms for subsequent testing.

For the FGSR and IFGSR algorithms, parameter testing is conducted using the FGSR algorithm and by applying the same parameter settings to both in subsequent experiments. The parameters of the FGSR algorithm include γ , α , and step size λ . The initial parameter settings are $\alpha = 1$ and step size $\lambda = 0.03$. The parameter γ is related to the initial rank estimation. According to reference [23], the initial rank estimation has a slight effect on the FGSR algorithm, so we directly choose its maximum selectable value, that is, $\min(n_1, n_2)$, representing the smaller value between the number of rows and columns within the window of the data matrix.

In the selection of parameters, this study chooses parameters that exhibit stable performance in processing three types of variable data.

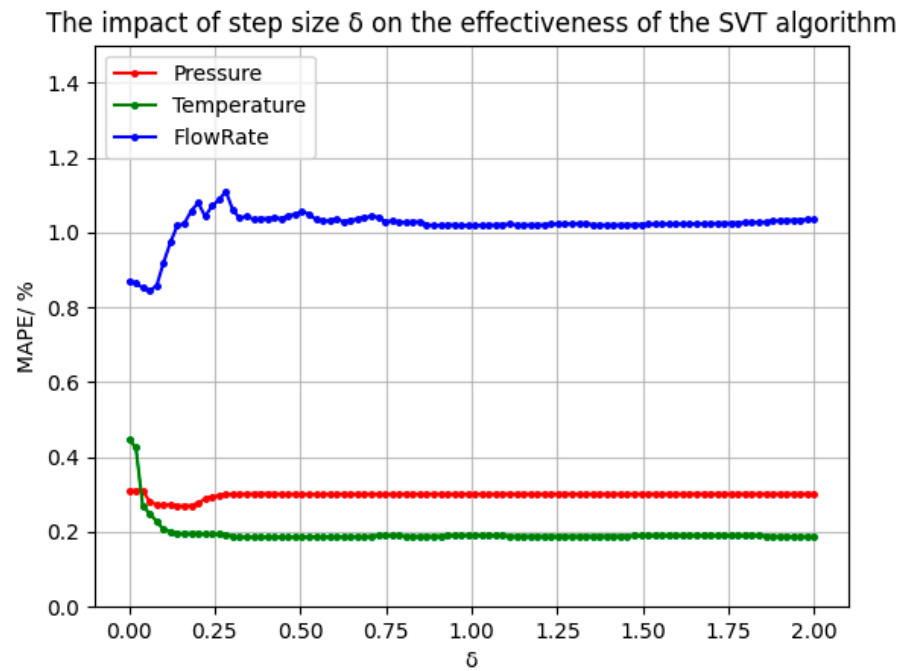


Figure 4. The impact of step size δ on the effectiveness of the SVT algorithm.

When $\lambda = 0.03$, we evaluate the influence of changing the parameter α on the data processing results. The results are illustrated in Figure 5.

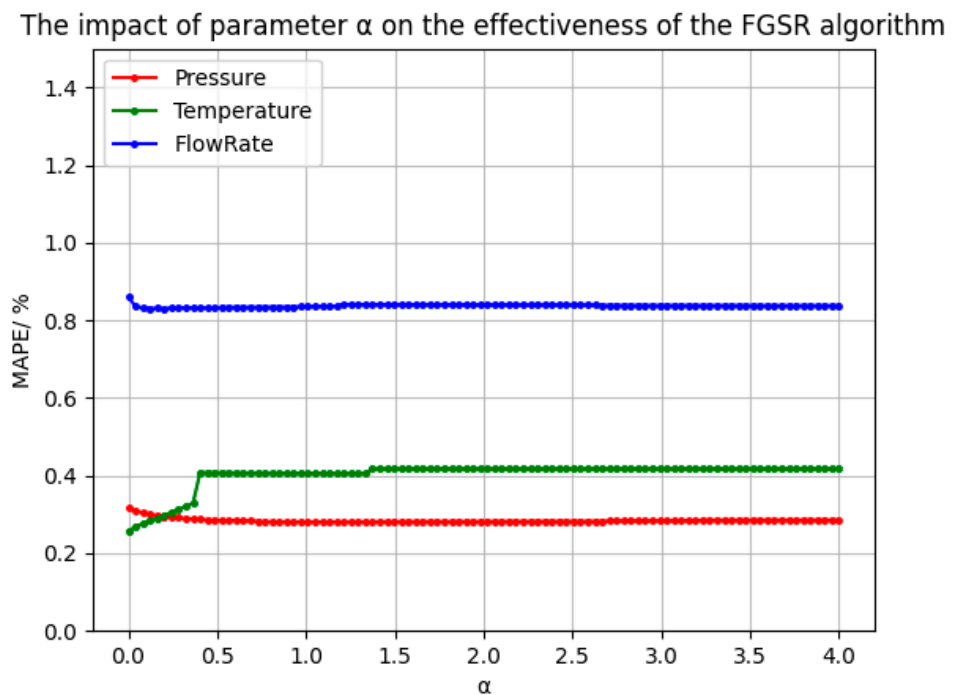


Figure 5. The impact of parameter α on the effectiveness of the FGSR algorithm.

As shown in Figure 5, the *MAPE* of the three variables after processing does not change significantly after $\alpha > 0.5$. In this study, we choose the midpoint value of 2 from the range of 0~4 as the value of α for the following experiments.

Setting $\alpha = 2$, we evaluate the influence of the step increment λ on the results. The results are illustrated in Figure 6.

The impact of step size λ on the effectiveness of the FGSR algorithm

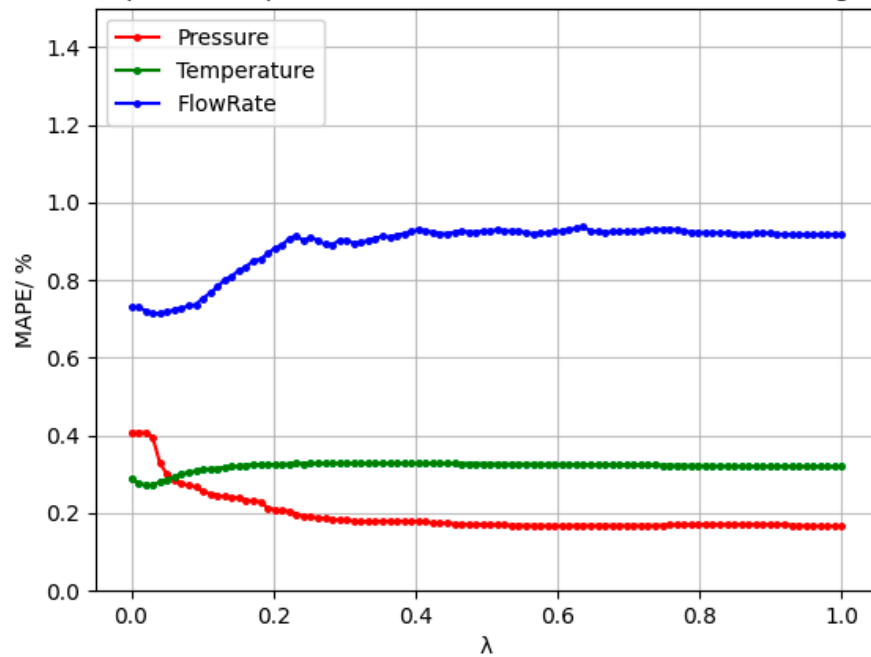


Figure 6. The impact of step size λ on the effectiveness of the FGSR algorithm.

As illustrated in Figure 6, after $\lambda > 0.2$, the MAPE of the three variable data does not change significantly. The MAPE of the pressure data slightly decreases as λ increases, and the MAPE of the flow rate data has a slight fluctuation and is at a lower level near $\lambda = 0.3$. In this study, we chose $\lambda = 0.3$ as the value of λ for subsequent experiments.

The experimental results reveal that the MAPE in the imputation of flow rate data is substantially higher than that in temperature and pressure data. This is a consequence of the data's characteristics, with the flow rate data used in the experiment showing a larger range of change than the other two variables.

After determining the parameters, we then test the size of the data processing window and use a fixed window size in subsequent tests. The subsequent experimental data will be presented in Tables A1–A6 in Appendix A, corresponding, respectively, to Figures 7–12.

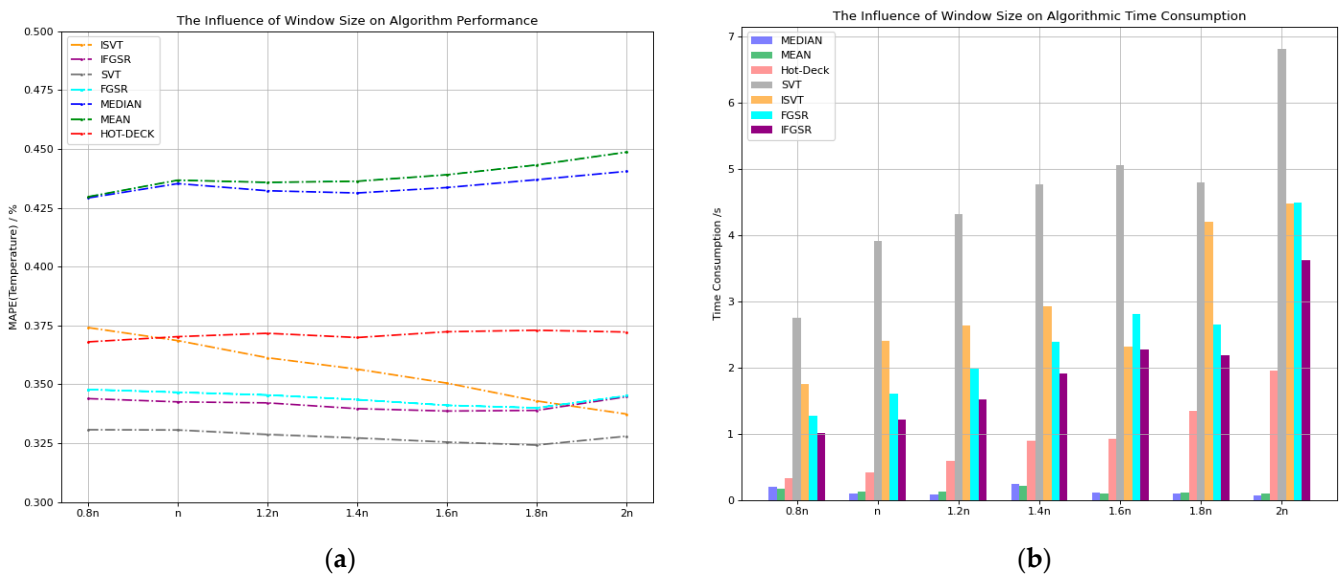


Figure 7. Influence of window size on the processing of temperature data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

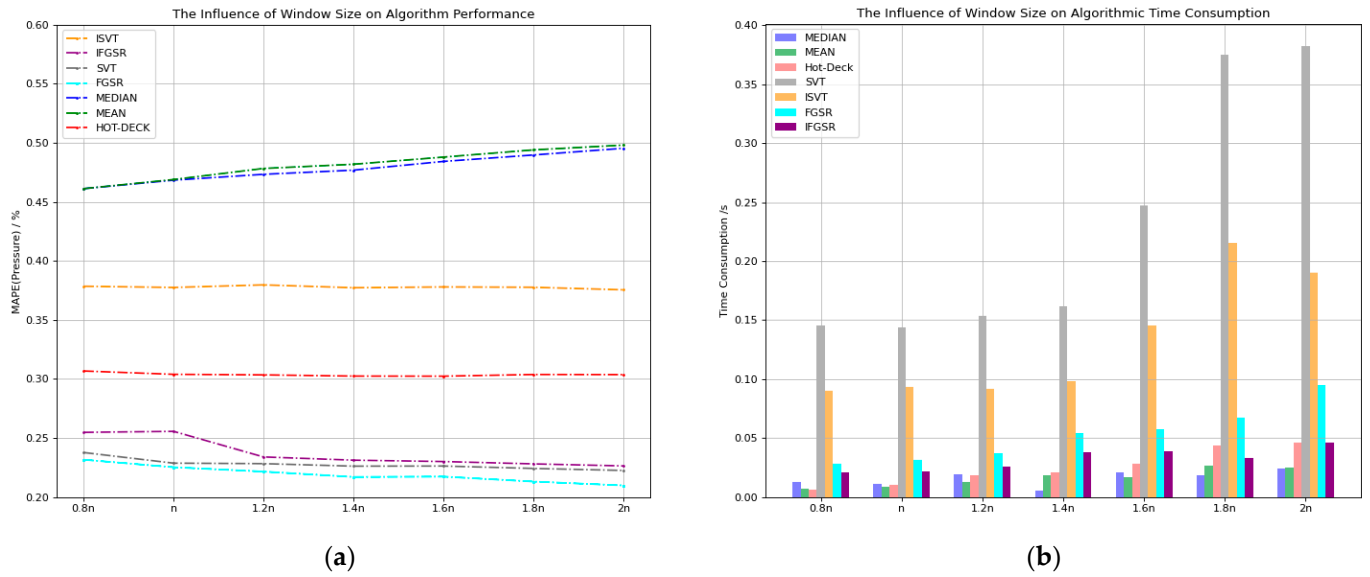


Figure 8. Influence of window size on the processing of pressure Data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

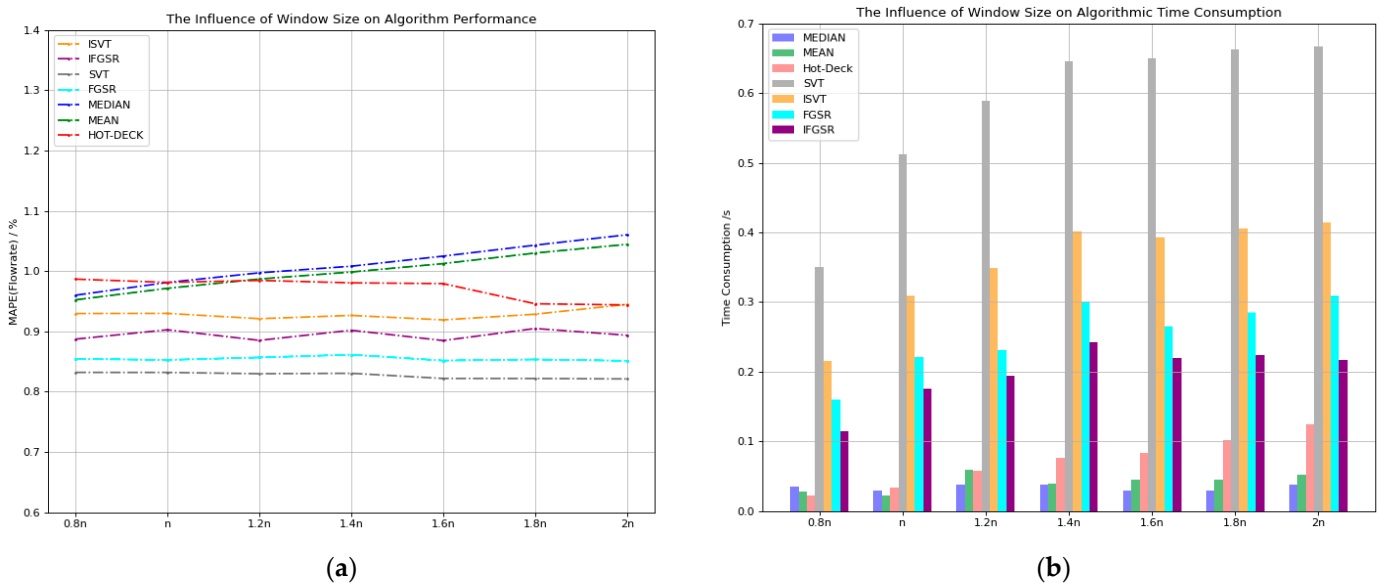


Figure 9. Influence of window size on the processing of flow rate data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

We fix the missing rate of the variable data at 40%, generating missing data randomly. The range for the window size is from 0.8n to 2.0n (where n represents the number of sensors for the current variable, as detailed in Table 1, and the chosen window size is rounded up). We first conduct tests on the temperature data, and Figure 7 presents the results.

As shown in Figure 7a, when we test the temperature data using MAPE as the evaluation standard, the statistical value imputation methods and the hot-deck algorithm exhibit an increase in MAPE with the growing window size. The MAPE of the ISVT algorithm gradually decreases with the increase in the window size. The FGSR, IFGSR, and SVT algorithms have the smallest MAPE when the window size reaches 1.8n. As seen in Figure 7b, the time consumption of the SVT, FGSR, and IFGSR algorithms increases significantly after the window size is 1.8n, and the time consumption of the ISVT algorithm shows little change between the window sizes of 1.8n and 2n. Considering these factors, to

achieve higher accuracy and reduce time consumption, we choose $1.8n$ as the window size for subsequent experiments.

As illustrated in Figure 8a, when we test the pressure data using *MAPE* as the evaluation standard, the *MAPE* of the ISVT algorithm and the hot-deck algorithm remains relatively stable. The *MAPE* values for the FGSR, SVT, and IFGSR algorithms all decrease with an increase in the window size. As seen in Figure 8b, as the time window increases, the time consumption of the hot-deck algorithm and the MCM gradually increases. When the window size is $2n$, the time consumption of the IFGSR algorithm is halved compared to the FGSR algorithm and is close to the hot-deck algorithm. For better accuracy, in subsequent experiments, we use $2n$ as the window size to process data.

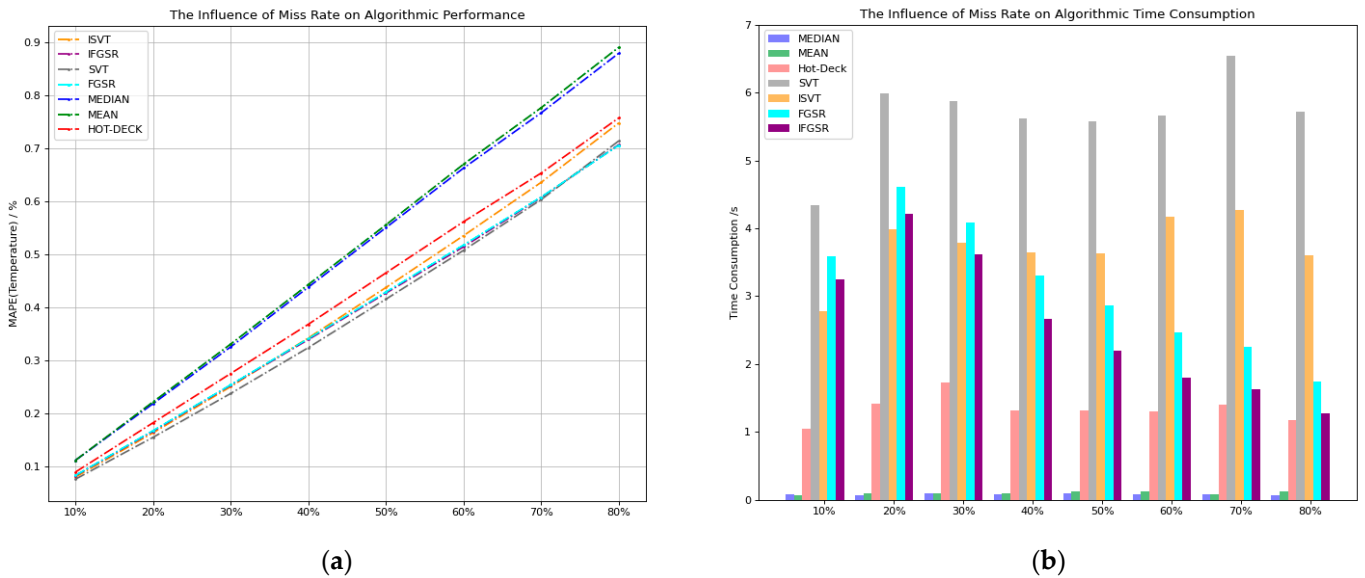


Figure 10. Influence of miss rate on the processing of temperature data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

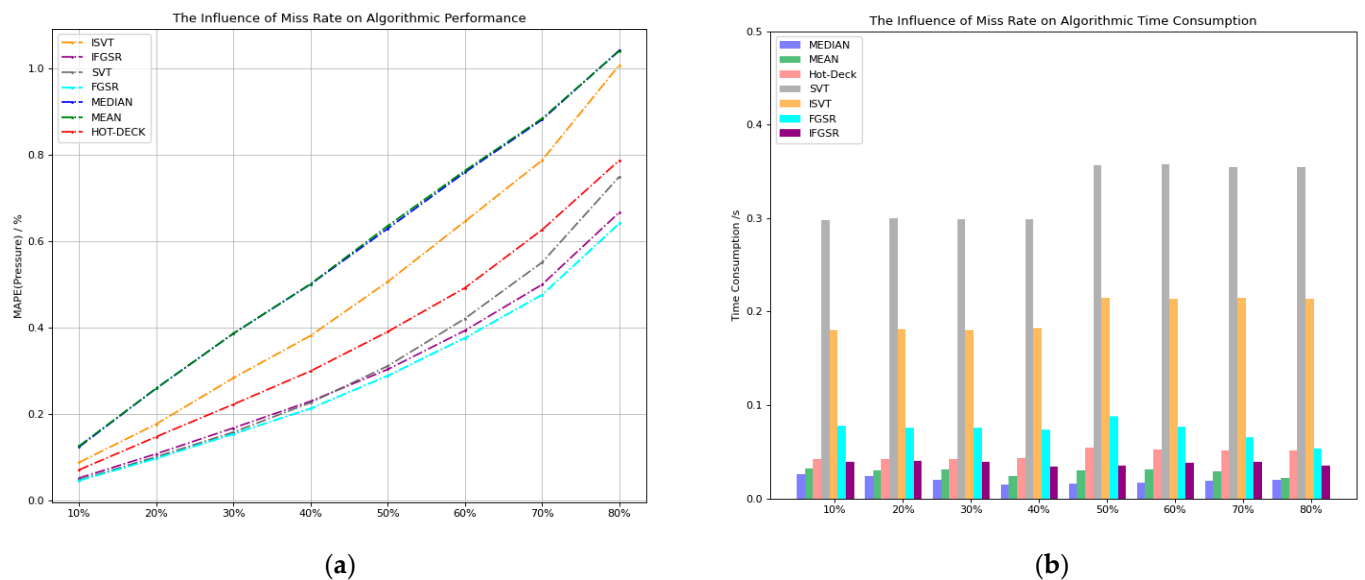


Figure 11. Influence of miss rate on the processing of pressure data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

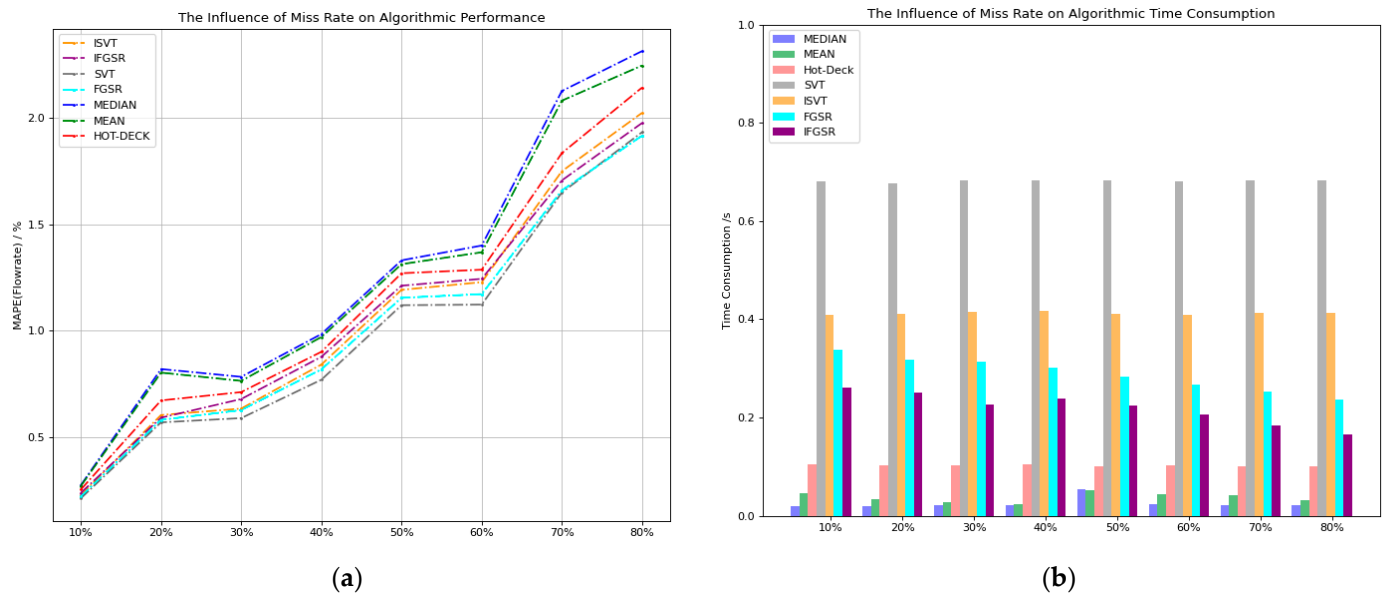


Figure 12. Influence of miss rate on the processing of flow rate data. Subfigure (a) represents the MAPE after data imputation, subfigure (b) represents the time consumption.

As shown in Figure 9a, the MAPE of MCMs exhibits fluctuations with an increase in the window size when different algorithms are applied. The MAPE of ISVT, FGSR, and IFGSR maintains a low level when the window size is $1.6n$. The precision of the SVT algorithm remains stable once the window size exceeds $1.6n$. As seen in Figure 9b, the time consumption of the MCM does not change much after the window size is $1.4n$. Therefore, based on the above results, we select a window size of $1.6n$ for future experiments on flow rate data.

In summary, through testing on three variables, we have chosen $1.8n$ as the window size for temperature data, $2n$ as the window size for pressure data, and $1.6n$ as the window size for flow rate data for subsequent experiments.

After determining the parameters and the size of the test window, we use the same experimental data to test the performance of various methods under different data loss rates. The evaluation metrics include accuracy and time consumption. The results are as follows.

As shown in Figure 10, we first test the temperature data. With a fixed window size, we test the results with a data loss rate of 10% to 80%.

As shown in Figure 10a, when we use MAPE as the evaluation standard, it can be seen that MCMs have the smallest MAPE at any missing rate, ISVT's MAPE is slightly larger than SVT, and the accuracy performance of IFGSR and FGSR is almost the same. As shown in Figure 10b, the highest time consumption in the MCM is the SVT and FGSR algorithms. In comparison, both ISVT and IFGSR show improvements in time consumption.

In summary, the MCM consistently exhibits the smallest MAPE when processing temperature data. Among them, SVT has the lowest MAPE but the highest time consumption; ISVT's accuracy is slightly lower than SVT, and the time consumption is lower. The accuracy of FGSR and IFGSR is almost the same, with both slightly underperforming the SVT algorithm. IFGSR has a lower time consumption compared to FGSR, and both significantly reduce time consumption compared to SVT and ISVT.

For pressure data, Figure 11 shows the results of testing data with missing rates ranging from 10% to 80% at a fixed window size.

As shown in Figure 11a, processing pressure data with MAPE as the evaluation metric reveals that FGSR and IFGSR consistently exhibit the smallest MAPE. Moreover, the difference in MAPE between IFGSR and FGSR is insignificant across all missing rates. Figure 11b illustrates that the IFGSR algorithm consumes less time than the FGSR algorithm,

and its time consumption exceeds only those of the two statistical value imputation methods at high missing rates.

In conclusion, when processing pressure data, the IFGSR algorithm achieves a slight reduction in accuracy compared to the FGSR algorithm but significantly reduces the processing time. Compared to other methods, it also demonstrates higher accuracy and advantages in time consumption.

For flow rate data, Figure 12 shows the results of testing data with missing rates ranging from 10% to 80% at a fixed window size.

Figure 12a illustrates that the MCM exhibits the highest accuracy in processing missing values for flow rate data. The overall accuracy ranking is SVT > FGSR > IFGSR \approx ISVT. As shown in Figure 12b, considering time consumption, ISVT and IFGSR save a substantial amount of time compared to SVT and FGSR.

In conclusion, when using the MCM to impute the missing values of temperature, pressure, and flow rate data, the SVT, FGSR, and IFGSR algorithms in the MCM consistently demonstrate superior accuracy under any conditions compared to traditional methods, thus validating the applicability of matrix completion for real-time missing data imputation. Furthermore, a comparison between the IFGSR and FGSR algorithms reveals a minor difference in accuracy but a significant reduction in computation time for IFGSR, demonstrating the effectiveness of the improvement method proposed in this paper for matrix completion algorithms used for real-time missing data imputation.

5. Conclusions

This study describes a method for using the MCM to fill in missing values in chemical process data. This method employs matrix completion and sliding windows to process missing data in real-time chemical process data. The fundamental principle involves using the MCM to restore matrices that contain missing values within the chemical process data. In addition, an improvement method is proposed, which enhances the performance of the matrix completion algorithm used for real-time missing data imputation. The results show that the MCM performs well in solving the problem of real-time data missing value imputation in chemical processes, and the proposed improvement method also significantly enhances the computational speed of matrix completion algorithms.

The MCM proposed in this paper does not rely heavily on long-term historical data. It does not need to rely on long-term historical data for detailed modeling of specific devices and then use the model to calculate missing data. It only needs short-term data to obtain relatively accurate results for missing value recovery. This method exhibits good versatility and is applicable to different production devices. The MCM can also predict future data of production devices, and, when combined with appropriate constraints, it can identify and replace outliers.

Author Contributions: Conceptualization, X.Z. and S.T.; methodology, X.Z. and S.T.; software, X.Z.; validation, X.Z.; formal analysis, X.Z.; investigation, S.T.; resources, X.S.; data curation, X.Z.; writing—original draft preparation, X.Z.; writing—review and editing, S.T.; visualization, X.Z.; supervision, S.X.; project administration, L.X.; funding acquisition, S.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number No. 22178190.

Data Availability Statement: The data presented in this study are not available due to privacy.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Table A1. The influence of window size on the processing of temperature data.

Method \ Window Size	Error and Time Consumption	0.8n _s	1.0n _s	1.2n _s	1.4n _s	1.6n _s	1.8n _s	2.0n _s
Mean imputation	MAPE/%	0.429689	0.436732	0.435803	0.436275	0.439029	0.443191	0.448638
	Time/s	0.170502	0.12856	0.130309	0.206899	0.090882	0.109137	0.100491
Median imputation	MAPE/%	0.429191	0.435274	0.432243	0.431291	0.433602	0.436931	0.44047
	Time/s	0.201265	0.096462	0.075046	0.235175	0.113331	0.0966	0.060941
Hot-deck	MAPE/%	0.368026	0.370244	0.371674	0.369868	0.372333	0.372959	0.372206
	Time/s	0.321434	0.418652	0.58801	0.895145	0.929394	1.342582	1.954099
SVT	MAPE/%	0.330692	0.330586	0.328675	0.327193	0.325432	0.324194	0.327919
	Time/s	2.759206	3.91747	4.31461	4.770563	5.054205	4.797185	6.812804
ISVT	MAPE/%	0.374091	0.368559	0.361274	0.356427	0.350504	0.342927	0.33737
	Time/s	1.743603	2.402995	2.642217	2.927075	2.322131	4.203153	4.477385
FGSR	MAPE/%	0.347781	0.346603	0.345447	0.343482	0.341073	0.33999	0.34509
	Time/s	1.269798	1.605816	1.98593	2.384596	2.812689	2.657019	4.498306
IFGSR	MAPE/%	0.343915	0.342529	0.342109	0.339652	0.338637	0.338905	0.344688
	Time/s	1.013055	1.208337	1.51925	1.903469	2.269553	2.185626	3.625262

Table A2. The influence of window size on the processing of pressure data.

Method \ Window Size	Error and Time Consumption	0.8n _s	1.0n _s	1.2n _s	1.4n _s	1.6n _s	1.8n _s	2.0n _s
Mean imputation	MAPE/%	0.46116	0.468882	0.478266	0.481818	0.487873	0.493975	0.498029
	Time/s	0.006626	0.008727	0.012539	0.018682	0.016906	0.026218	0.024772
Median imputation	MAPE/%	0.461145	0.468343	0.473304	0.47683	0.484212	0.489664	0.495333
	Time/s	0.012414	0.010906	0.018815	0.005472	0.020775	0.01809	0.023827
Hot-deck	MAPE/%	0.306717	0.303902	0.303436	0.302407	0.302331	0.303755	0.303662
	Time/s	0.006375	0.009966	0.018106	0.020586	0.028261	0.043914	0.046188
SVT	MAPE/%	0.237668	0.228656	0.228244	0.226121	0.226222	0.224198	0.22248
	Time/s	0.145216	0.143519	0.153234	0.161981	0.24699	0.375339	0.382238
ISVT	MAPE/%	0.378499	0.37748	0.379664	0.377244	0.37791	0.37764	0.37548
	Time/s	0.089936	0.09314	0.091471	0.098333	0.145666	0.215231	0.190195
FGSR	MAPE/%	0.231582	0.225196	0.221528	0.216968	0.217246	0.213067	0.209774
	Time/s	0.028133	0.031491	0.036807	0.054531	0.057873	0.067387	0.095115
IFGSR	MAPE/%	0.254744	0.255588	0.233928	0.231192	0.230023	0.228048	0.226329
	Time/s	0.020926	0.021267	0.025544	0.038162	0.039157	0.033097	0.045742

Table A3. The influence of window size on the processing of flow rate data.

Method\Window Size	Error and Time Consumption	0.8n _s	1.0n _s	1.2n _s	1.4n _s	1.6n _s	1.8n _s	2.0n _s
Mean imputation	MAPE/%	0.952342	0.971453	0.986896	0.99851	1.012467	1.029958	1.044433
	Time/s	0.027833	0.022236	0.058512	0.039126	0.044706	0.04536	0.051778
Median imputation	MAPE/%	0.960008	0.981177	0.997188	1.008103	1.024871	1.043059	1.060312
	Time/s	0.035634	0.029653	0.038005	0.037985	0.029285	0.029558	0.038445
Hot-deck	MAPE/%	0.986663	0.981268	0.984567	0.980568	0.979232	0.94581	0.943907
	Time/s	0.022614	0.032845	0.058058	0.075891	0.083756	0.101958	0.124636
SVT	MAPE/%	0.831948	0.831894	0.829771	0.830461	0.821872	0.821781	0.821165
	Time/s	0.350121	0.512551	0.589402	0.64564	0.65025	0.663364	0.666758
ISVT	MAPE/%	0.929614	0.929827	0.921077	0.926509	0.919068	0.928503	0.944799
	Time/s	0.215114	0.308748	0.348895	0.401168	0.392368	0.406407	0.414813
FGSR	MAPE/%	0.854404	0.852655	0.856774	0.861344	0.85159	0.853464	0.85124
	Time/s	0.160431	0.220477	0.23072	0.299942	0.264435	0.285232	0.30906
IFGSR	MAPE/%	0.887159	0.902684	0.885102	0.90192	0.884778	0.904824	0.893551
	Time/s	0.114031	0.175019	0.194166	0.242649	0.220073	0.223618	0.217471

Table A4. The influence of miss rate on the processing of temperature data.

Method\Miss Rate	Error and Time Consumption	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Mean imputation	MAPE/%	0.111415	0.221567	0.330508	0.442765	0.555371	0.669687	0.776217	0.890122
	Time/s	0.064355	0.089104	0.100775	0.092744	0.124961	0.128273	0.083667	0.127447
Median imputation	MAPE/%	0.111443	0.218174	0.32526	0.437772	0.550006	0.66239	0.766547	0.879003
	Time/s	0.0853	0.069159	0.095377	0.078869	0.096246	0.080675	0.078036	0.069624
Hot-deck	MAPE/%	0.089245	0.18221	0.275035	0.367849	0.464774	0.561281	0.653008	0.757046
	Time/s	1.045313	1.420777	1.724894	1.318843	1.309674	1.30266	1.397718	1.169198
SVT	MAPE/%	0.075853	0.154787	0.237423	0.323773	0.415213	0.507319	0.602957	0.713683
	Time/s	4.344451	5.987962	5.880589	5.619251	5.582199	5.668959	6.547173	5.714972
ISVT	MAPE/%	0.079767	0.163287	0.250239	0.34209	0.437198	0.534976	0.635493	0.747522
	Time/s	2.774571	3.986273	3.783541	3.644063	3.635185	4.167742	4.274185	3.599238
FGSR	MAPE/%	0.082696	0.167274	0.253534	0.340791	0.428555	0.5173	0.607727	0.705739
	Time/s	3.593545	4.609731	4.085753	3.306966	2.857307	2.472233	2.248525	1.738816
IFGSR	MAPE/%	0.08226	0.166069	0.252763	0.338917	0.426536	0.513926	0.606955	0.706588
	Time/s	3.248033	4.216341	3.613562	2.67005	2.195556	1.792113	1.629194	1.265752

Table A5. The influence of miss rate on the processing of pressure data.

Method\Miss Rate	Error and Time Consumption	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Mean imputation	MAPE/%	0.125369	0.25891	0.386006	0.50069	0.635019	0.763016	0.883618	1.040332
	Time/s	0.031878	0.030032	0.031171	0.024268	0.03043	0.031451	0.028735	0.02203
Median imputation	MAPE/%	0.1232	0.258929	0.386007	0.50005	0.628795	0.759014	0.880959	1.041563
	Time/s	0.026002	0.023958	0.019762	0.014444	0.016118	0.016942	0.018383	0.020368
Hot-deck	MAPE/%	0.070259	0.146991	0.22211	0.299001	0.390197	0.491641	0.626109	0.786883
	Time/s	0.042708	0.042116	0.04238	0.04279	0.0541	0.051913	0.051356	0.051745
SVT	MAPE/%	0.046817	0.100516	0.157676	0.226004	0.310759	0.420588	0.550662	0.748993
	Time/s	0.297998	0.299693	0.299347	0.298905	0.356411	0.358202	0.354957	0.354273
ISVT	MAPE/%	0.087438	0.176265	0.283108	0.380976	0.505853	0.645433	0.787063	1.007307
	Time/s	0.180578	0.180712	0.180289	0.182614	0.214508	0.213438	0.214417	0.213874
FGSR	MAPE/%	0.045819	0.096758	0.153544	0.212478	0.288276	0.375787	0.475559	0.640863
	Time/s	0.077745	0.075804	0.07524	0.073885	0.088261	0.076434	0.065284	0.0532
IFGSR	MAPE/%	0.051439	0.107375	0.167297	0.229239	0.303	0.392865	0.499357	0.666309
	Time/s	0.039649	0.040591	0.038836	0.033879	0.034644	0.038104	0.039615	0.035125

Table A6. The influence of miss rate on the processing of flow rate data.

Method\Miss Rate	Error and Time Consumption	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Mean imputation	MAPE/%	0.239366	0.479715	0.5937	1.020246	1.498838	1.665591	1.82332	2.043651
	Time/s	0.036863	0.032585	0.037025	0.033544	0.040757	0.03932	0.032436	0.049352
Median imputation	MAPE/%	0.240111	0.484099	0.597288	1.027179	1.51114	1.68408	1.840259	2.071771
	Time/s	0.02477	0.049942	0.056303	0.046061	0.051352	0.017915	0.031472	0.042276
Hot-deck	MAPE/%	0.251902	0.463079	0.642777	0.89669	1.407212	1.681455	1.648385	2.049117
	Time/s	0.027612	0.028242	0.028666	0.028087	0.025683	0.02345	0.023301	0.026054
SVT	MAPE/%	0.215222	0.415987	0.559695	0.951196	1.382286	1.537685	1.727861	1.922707
	Time/s	0.422097	0.418632	0.413529	0.415509	0.40398	0.431114	0.355944	0.3587
ISVT	MAPE/%	0.229068	0.464872	0.594999	1.024654	1.513273	1.722914	1.924457	2.27328
	Time/s	0.257354	0.255716	0.253206	0.256194	0.262943	0.251303	0.214986	0.258442
FGSR	MAPE/%	0.231652	0.438877	0.610154	0.988806	1.385793	1.575441	1.748605	1.937884
	Time/s	0.207726	0.207422	0.198418	0.193925	0.179496	0.169473	0.163923	0.118877
IFGSR	MAPE/%	0.246612	0.463556	0.640926	1.03057	1.425174	1.60768	1.801965	1.964265
	Time/s	0.170172	0.184834	0.15303	0.140324	0.135162	0.113417	0.105556	0.08579

References

- Zhang, K.; Gonzalez, R.; Huang, B.; Ji, G. Expectation–Maximization Approach to Fault Diagnosis with Missing Data. *IEEE Trans. Ind. Electron.* **2015**, *62*, 1231–1240. [\[CrossRef\]](#)
- Zhu, J.; Ge, Z.; Song, Z.; Gao, F. Review and Big Data Perspectives on Robust Data Mining Approaches for Industrial Process Modeling with Outliers and Missing Data. *Annu. Rev. Control* **2018**, *46*, 107–133. [\[CrossRef\]](#)
- Pigott, T.D. A Review of Methods for Missing Data. *Educ. Res. Eval.* **2001**, *7*, 353–383. [\[CrossRef\]](#)
- Chatfield, C.; Little, R.J.A.; Rubin, D.B. Statistical Analysis with Missing Data. *J. R. Stat. Soc. Ser. A* **1988**, *151*, 375. [\[CrossRef\]](#)
- Xu, S.; Lu, B.; Baldea, M.; Edgar, T.F.; Wojsznis, W.; Blevins, T.; Nixon, M. Data Cleaning in the Process Industries. *Rev. Chem. Eng.* **2015**, *31*, 453–490. [\[CrossRef\]](#)

6. Luo, L.; Bao, S.; Peng, X. Robust Monitoring of Industrial Processes Using Process Data with Outliers and Missing Values. *Chemom. Intell. Lab. Syst.* **2019**, *192*, 103827. [[CrossRef](#)]
7. Allison, P.D. *Handling Missing Data by Maximum Likelihood*; Statistical Horizons: Haverford, PA, USA, 2012.
8. Walczak, B.; Massart, D.L. Dealing with Missing Data: Part II. *Chemom. Intell. Lab. Syst.* **2001**, *58*, 29–42. [[CrossRef](#)]
9. Jirasek, F.; Bamler, R.; Fellenz, S.; Bortz, M.; Kloft, M.; Mandt, S.; Hasse, H. Making Thermodynamic Models of Mixtures Predictive by Machine Learning: Matrix Completion of Pair Interactions. *Chem. Sci.* **2022**, *13*, 4854–4862. [[CrossRef](#)] [[PubMed](#)]
10. Hayer, N.; Jirasek, F.; Hasse, H. Prediction of Henry’s Law Constants by Matrix Completion. *AIChE J.* **2022**, *68*, e17753. [[CrossRef](#)]
11. Bac, S.; Quiton, S.J.; Kron, K.J.; Chae, J.; Mitra, U.; Mallikarjun Sharada, S. A Matrix Completion Algorithm for Efficient Calculation of Quantum and Variational Effects in Chemical Reactions. *J. Chem. Phys.* **2022**, *156*, 184119. [[CrossRef](#)] [[PubMed](#)]
12. Recht, B. A Simpler Approach to Matrix Completion. *J. Mach. Learn. Res.* **2011**, *12*, 3413–3430.
13. Candès, E.J.; Recht, B. Exact Matrix Completion via Convex Optimization. *Found. Comput. Math.* **2009**, *9*, 717–772. [[CrossRef](#)]
14. Liu, Y.; Zheng, W.-F.; Zhang, Y. Optimization of Data Matrix Completion by Symmetric Weighting Algorithm. *J. Sichuan Univ. Nat. Sci. Ed.* **2021**, *58*, 73–80. [[CrossRef](#)]
15. Cabral, R.; De la Torre, F.; Costeira, J.P.; Bernardino, A. Matrix Completion for Weakly-Supervised Multi-Label Image Classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 121–135. [[CrossRef](#)] [[PubMed](#)]
16. Li, W.; Zhao, L.; Lin, Z.; Xu, D.; Lu, D. Non-Local Image Inpainting Using Low-Rank Matrix Completion. *Comput. Graph. Forum* **2015**, *34*, 111–122. [[CrossRef](#)]
17. Chistov, A.L.; Grigor’ev, D.Y. Complexity of Quantifier Elimination in the Theory of Algebraically Closed Fields. In Proceedings of the Mathematical Foundations of Computer Science, Praha, Czechoslovakia, 3–7 September 1984; Chytil, M.P., Koubek, V., Eds.; Springer: Berlin/Heidelberg, Germany, 1984; pp. 17–31.
18. Ma, S.; Goldfarb, D.; Chen, L. Fixed Point and Bregman Iterative Methods for Matrix Rank Minimization. *Math. Program.* **2011**, *128*, 321–353. [[CrossRef](#)]
19. Toh, K.-C.; Yun, S. An Accelerated Proximal Gradient Algorithm for Nuclear Norm Regularized Linear Least Squares Problems. *Pac. J. Optim.* **2010**, *6*, 15.
20. Cai, J.-F.; Candès, E.J.; Shen, Z. A Singular Value Thresholding Algorithm for Matrix Completion. *SIAM J. Optim.* **2010**, *20*, 1956–1982. [[CrossRef](#)]
21. Wen, Z.; Yin, W.; Zhang, Y. Solving a Low-Rank Factorization Model for Matrix Completion by a Nonlinear Successive over-Relaxation Algorithm. *Math. Program. Comput.* **2012**, *4*, 333–361. [[CrossRef](#)]
22. Nie, F.; Wang, H.; Huang, H.; Ding, C. Joint Schatten (p) -norm and (ℓ_1) -norm robust matrix completion for missing value recovery. *Knowl. Inf. Syst.* **2015**, *42*, 525–544. [[CrossRef](#)]
23. Fan, J.; Ding, L.; Chen, Y.; Udell, M. Factor Group-Sparse Regularization for Efficient Low-Rank Matrix Recovery. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.