Computers and Chemical Engineering 180 (2024) 108448

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/cace

Automated outlier detection and estimation of missing data

Jinwook Rhyu[a], Dragana Bozinovski[b], Alexis B. Dubs[a], Naresh Mohan[b], Elizabeth M. Cummings Bende[b], Andrew J. Maloney[a], Miriam Nieves[b], Jose Sangerman[b], Amos E. Lu[a], Moo Sun Hong[a], Anastasia Artamonova[b], Rui Wen Ou[c], Paul W. Barone[b], James C. Leung[b], Jacqueline M. Wolfrum[b], Anthony J. Sinskey[b,c], Stacy L. Springs[b], Richard D. Braatz[a,b,*]

[a] *Department of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, United States of America*
[b] *Center for Biomedical Innovation, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, United States of America*
[c] *Department of Biology, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, United States of America*

ARTICLE INFO

Dataset link: https://github.com/JinwookRhyu/Automated-Outlier-Detection-and-Estimation-of-Missing-Data

Keywords:
Multivariate statistics
Statistical quality control
Principal component analysis
Otlier detection
Biomanufacturing

ABSTRACT

The majority of algorithms used for data imputation are based on latent variable methods. The presence of outliers in process data, however, misleads the latent relations among variables, resulting in an inaccurate estimation of missing values. This article proposes an approach for automatically detecting outliers using T^2 and Q contributions and estimating missing data using various general-purpose algorithms while reducing the impact of outliers. The software is validated using biomanufacturing data from the production of a monoclonal antibody produced by Chinese hamster ovary cells in a perfusion bioreactor for five missingness cases including missing completely at random, sensor drop-out, multi-rate, patterned, and censoring. Based on the normalized root mean squared error and the three proposed metrics corresponding to feasibility, plausibility, and rapidity, respectively, matrix completion methods are the most effective, except for the censoring case in which probabilistic principal component analysis-based methods are the most effective.

1. Introduction

Most process datasets contain missing values, especially in biomanufacturing where multiple sensors are used to monitor a complex, dynamic system. The common missingness patterns in process data are (1) random missingness, which exhibits no explicit pattern, (2) sensor drop-out, in which the missing values are correlated in time, (3) multi-rate, in which the missingness happens periodically, and (4) censoring, in which there exist the thresholds for censoring so that the measurements outside the range are not recorded (Imtiaz and Shah, 2008; Severson et al., 2017).

The presence of missing values inhibits the use of data in process modeling (Nelson et al., 1996; Bridewell et al., 2006), analysis (Imtiaz and Shah, 2008; Zhu et al., 2018), and control (Arteaga and Ferrer, 2002). The simplest way to deal with the missingness is to only consider the observations with full measurements. Removing observations, however, can cause significant data loss in the specific time period when the missing values are agglomerated, which makes capturing the process dynamics challenging.

Several studies have explored the use of the expectation–maximization (EM) algorithm to fill in missing data points (Isaksson, 1993; Shumway and Stoffer, 2000; Raghavan et al., 2006; Gopaluni, 2010; Barazandegan et al., 2015). Severson et al. (2017) reviewed ten general-purpose imputation algorithms that are based on principal component analysis (PCA) and conducted case studies on the synthetic Gaussian data and the Tennessee Eastman problem. These imputation algorithms, however, are vulnerable to outliers (Stevens, 1984; Bollen, 1987; Pison et al., 2003; Mavridis and Moustaki, 2008). Not removing the outliers before filling in missing values emphasizes the effect of outliers and degrades the accuracy and reliability of results obtained by subsequent data analytics. There are also several studies on fault detection in the presence of outliers and missing data (Li et al., 2000; Fan et al., 2021). However, the objective of these studies was more focused on detecting observation-wise faults rather than cleaning the dataset element-wise so that the data points in the cleaned dataset are as close to the true values as possible.

Closely related to this problem, there is an active field called *robust PCA* (RPCA) which aims to recover the matrix while assuming that the

[*] Correspondence to: Department of Chemical Engineering, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Room E19-551, Cambridge, MA 02139, United States of America.
E-mail address: braatz@mit.edu (R.D. Braatz).

https://doi.org/10.1016/j.compchemeng.2023.108448
Received 14 May 2023; Received in revised form 22 August 2023; Accepted 4 October 2023
Available online 17 October 2023

observed values are not necessarily clean (i.e., a portion of elements are corrupted) (Wright et al., 2009; Lin et al., 2010; Candès et al., 2011; Chandrasekaran et al., 2011). RPCA approaches were successfully applied to a wide range of fields such as video surveillance (Candès et al., 2011; Bouwmans and Zahzah, 2014; Bouwmans et al., 2018), face recognition (Candès et al., 2011; Luan et al., 2014; Cao et al., 2019), and motion correction (Hamy et al., 2014; Bie et al., 2019). However, as discussed in the review by Vaswani and Narayanamurthy (2018), RPCA-based algorithms assume that the outlier location is uniform random (e.g., principal component pursuit (PCP) from Candès et al. (2011) and modified-PCP from Zhan and Vaswani (2015)) or require that the user select values for heuristic parameters (e.g., AltProj from Netrapalli et al. (2014), RPCA-GD from Yi et al. (2016), NO-RMC from Cherapanamjeri et al. (2017), and ReProCS-NORST from Qiu et al. (2014)). As the outliers do not necessarily appear randomly in the majority of chemical and biological processes (Fig. 2) and the user requirement of selection of heuristic parameters reduces automation, a case study on applying the RPCA-based algorithm was moved to Supplementary Material.

In order to find the best way to clean the dataset with outliers and missing values, this article introduces an approach and software that automatically detects outliers, fills in missing values, and evaluates each imputation algorithm used for matrix recovery. To the best of our knowledge, there has been no software that simultaneously detects outliers and fills in missing values while reducing the impact of outliers in an automated way. Section 2 describes the framework of outlier detection and missing value estimation used in this software. Section 3 demonstrates how the software works using a dataset collected from a continuous biomanufacturing pilot facility at the Massachusetts Institute of Technology. The data are from the production of a monoclonal antibody produced by Chinese hamster ovary cells in a perfusion bioreactor. Section 4 validates the performance of the software using a subset of the above dataset for which all of the measurements are available, followed by the conclusions in Section 5.

2. Methods

This section explains the methods of detecting outliers and filling in missing values used in this software (Fig. 1). Section 2.1 briefly describes principal component analysis (PCA), which is the basic method for the detection of outliers and the estimation of missing values. Section 2.2 describes three steps used in preprocessing (Step A) – temporary imputation of missing values (Step A-1), outlier detection based on T^2 and Q contributions (Step A-2), and elimination of low-quality observations (Step A-3). Section 2.3 describes multiple approaches used in Step B to deal with missing values and explains how to determine the number of principal components under the presence of missing values. Finally, Section 2.4 proposes several metrics for comparing the performance of imputation algorithms in Step C.

2.1. Introduction to PCA

PCA is a tool that projects a matrix into a lower dimensional space that captures most of the variability in the process variables. The objective of PCA is to model the significant correlations among variables while ignoring noise. For a given data matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, where d is the number of variables and n is the number of observations, the decomposition used by PCA for the matrix $\mathbf{X}$ is

$$\mathbf{T} = \mathbf{X}\mathbf{P}, \tag{1}$$

where $\mathbf{T} \in \mathbb{R}^{n\times a}$ is the score matrix, $\mathbf{P} \in \mathbb{R}^{d\times a}$ is the loading matrix that maps from principal components to the data, and a is the number of principal components.[1] For process data, the matrix $\mathbf{X}$ is usually standardized so that each variable has zero mean and unit norm before performing PCA (Abdi and Williams, 2010). Then the matrix $\mathbf{X}$ can be reconstructed using these score and loading matrices, which are expressed as

$$\mathbf{X} = \mathbf{T}\mathbf{P}^{\top} + \mathbf{E} = \hat{\mathbf{X}} + \mathbf{E}, \tag{2}$$

where $\hat{\mathbf{X}} \in \mathbb{R}^{n\times d}$ is a reconstructed matrix and the matrix $\mathbf{E} \in \mathbb{R}^{n\times d}$ is a reconstruction error.

The T^2 statistic (Hotelling, 1947), which is a scaled squared 2-norm of the observation vector $\mathbf{x} \in \mathbb{R}^{d\times 1}$ from its mean (Chiang et al., 2000), is a widely used metric to determine whether the observed data is an outlier. For the given score matrix $\mathbf{T}$ and the loading matrix $\mathbf{P}$, the T^2 statistic of the observation vector $\mathbf{x}$ is calculated as (Hotelling, 1947)

$$T^2 = \mathbf{x}^{\top}\mathbf{P}\boldsymbol{\Sigma}_a^{-2}\mathbf{P}^{\top}\mathbf{x}, \tag{3}$$

where $\boldsymbol{\Sigma}_a^{-2} \in \mathbb{R}^{a\times a}$ is the diagonal matrix whose main diagonal elements are the inverse of σ_k^2, which is the column-wise variance of $\mathbf{T}$ (Chiang et al., 2000; Zhu and Braatz, 2014). While the T^2 statistic explores the observation space explained by the column space of $\mathbf{P}$, the Q statistic explores the residual space of $\mathbf{P}$ which has the dimension of $d - a$ (Jackson and Mudholkar, 1979). The Q statistic is calculated as

$$Q = \mathbf{r}^{\top}\mathbf{r}, \tag{4}$$

where $\mathbf{r} \in \mathbb{R}^{d\times 1}$ is the residual expressed as $(\mathbf{I}_d - \mathbf{P}\mathbf{P}^{\top})\mathbf{x}$. Since the Q statistic is the total sum of variation in the residual space, the main advantage of using the Q statistic is that it is not affected by small singular values that are close to zero (Jackson and Mudholkar, 1979).

2.2. Preprocessing (Step A)

The presence of outliers, which are the values that are far from the major trend, can result in poor performance in estimating missing values using PCA, probabilistic PCA (PPCA), or matrix completion-based algorithms (Stanimirova et al., 2007; Rousseeuw and Hubert, 2011). As such, those outliers should be identified and removed before applying imputation algorithms for better accuracy (Chiang et al., 2000). In this step, the given dataset is preprocessed by detecting outliers and converting those values into missing values.

2.2.1. Temporary imputation of missing values (Step A-1)

The calculation of T^2 and Q statistics requires that the data matrix has no missing values. As such, missing values are temporarily imputed using interpolation, mean imputation, or the last observed values, which do not require information on latent relationships among measurement variables.

For the time series dataset, the interpolation method is recommended as it fills in missing values that capture the dynamic characteristics inside the given data. The mean imputation method dilutes the dynamic characteristic by replacing every missing value with identical values, whereas the last observed method emphasizes the weight of the element observed before a long period of missing, which can result in a significant bias, including unrealistic step changes in variables. In contrast, mean imputation is recommended for non-time series datasets as it treats each observation independently.

2.2.2. Outlier detection based on T^2 and Q contributions (Step A-2)

After the temporary imputation in Step A-1, the contributions for the T^2 and Q statistics are calculated for outlier detection. Based on the expression for the T^2 contribution and the Q contribution of jth variable on ith observation (Miller et al., 1998; Chiang et al., 2000),

$$\mathrm{cont}_{ij}^{T^2} = \sum_{k=1}^{a} \frac{t_{ik}p_{jk}}{\sigma_k^2} x_{ij}, \tag{5}$$

$$\mathrm{cont}_{ij}^{Q} = \left(x_{ij} - \sum_{k=1}^{a} t_{ik}p_{jk} \right)^2, \tag{6}$$

[1] The transposed version of the matrix $\mathbf{X}$ is used in some publications.

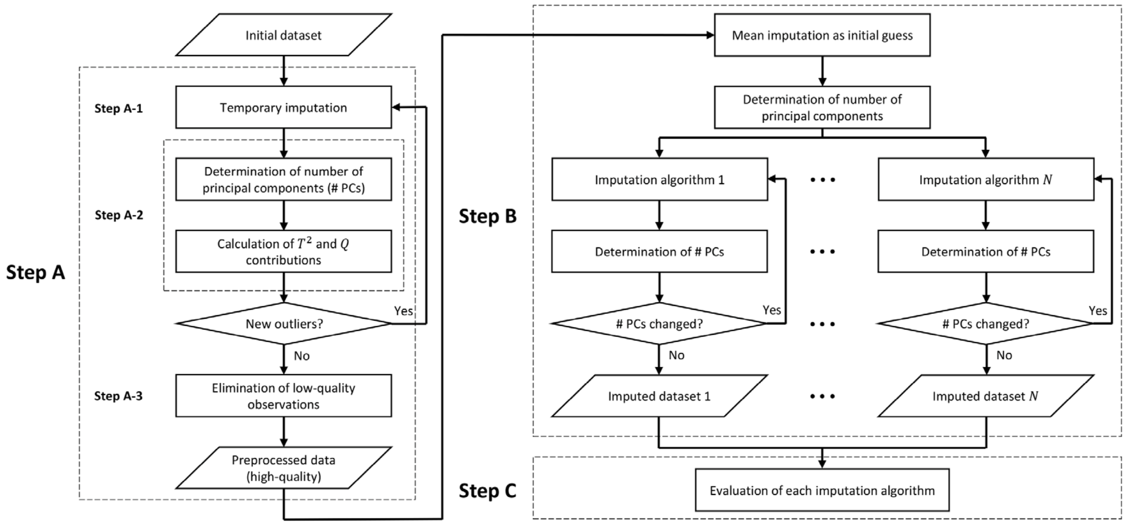

Initial dataset
Step A
Step A-1
Temporary imputation
Step A-2
Determination of number of principal components (# PCs)
Calculation of T^2 and Q contributions
New outliers?
Yes
No
Step A-3
Elimination of low-quality observations
Preprocessed data (high-quality)
Step B
Mean imputation as initial guess
Determination of number of principal components
Imputation algorithm 1
Imputation algorithm N
Determination of # PCs
Determination of # PCs
# PCs changed?
# PCs changed?
Yes
Yes
No
No
Imputed dataset 1
Imputed dataset N
Step C
Evaluation of each imputation algorithm

Fig. 1. Process diagram of the software. Preprocessing (Step A) is composed of temporary imputation of missing values (Step A-1), outlier detection (Step A-2), and elimination of low-quality observations (Step A-3). Multiple imputation algorithms are implemented in parallel in Step B, and each imputation algorithm is evaluated based on the imputation result in Step C.

the values that are outside of the column-wise distribution are considered outliers. For example, setting the confidence limit to 0.9999 means that the indices with contribution values outside the 99.99% range of the normal distribution are considered outliers. Further discussion on the confidence limit is addressed in Appendix A. These detected outliers are converted to missing values and the code goes back to Step A-1 for further outlier detection. Steps A-1 and A-2 are iterated until there are no newly detected outliers in the given dataset.

The software allows the user to specify any data in the dataset as not being outliers. For example, the initial observations in a dataset with dynamics may be protected from being considered outliers, e.g., by having similar values to known initial conditions for the experiment.

2.2.3. *Elimination of low-quality observations (Step A-3)*

Before moving to further steps, a decision needs to be made about how to deal with the observations having a few variables survived after the iteration of Steps A-1 and A-2 (i.e., the majority of measured variables are initially missed or detected as outliers). Possessing these *low-quality* observations significantly increases the missingness level of the dataset, leading to poor data imputation. Especially for the algorithms that require the calculation of inverse matrices, the existence of low-quality observations might make the corresponding matrix singular. While including low-quality observations decreases the quality of the dataset, removing a large number of observations decreases the quantity of the dataset. The balance between the quality and quantity of the dataset is specified by an appropriate cutoff for eliminating low-quality observations. In this article, the criterion for determining the low-quality observation is that the number of survived variables be less than the number of principal components of the dataset after completing outlier detection steps. The number of principal components is determined by using the algorithm shown in Section 2.3.5 with the missing values temporarily imputed using methods in Section 2.2.1. A more detailed analysis and discussion of the cutoff for the observation removal is in Appendix B.

2.3. *Imputation algorithms for missing values (Step B)*

This section describes various general-purpose imputation algorithms that are implemented in the software for imputing the missing values in the preprocessed data. These algorithms include mean imputation, three PCA approaches (Alternating, SVDImpute, and PCADA), three probabilistic PCA approaches (PPCA, PPCA-M, and BPCA), and two matrix completion approaches (SVT and ALM). Note that these imputation algorithms have a hyperparameter, which is the number of principal components. An approach for determining the number of principal components of the dataset with missing values is described in Section 2.3.5.

2.3.1. *Mean imputation*

Mean imputation (MI) is the simplest way to deal with missing data which imputes column-wise mean values into the missing values. This algorithm does not require any complex calculation or iteration and can be applied to any type of missingness. Because this approach does not consider any correlation among observations or variables, the algorithm can result in models of low accuracy. As such, mean imputation should only be used as an initial guess for other approaches. This section describes several approaches that fill in missing data while taking into account patterns among the observed values.

2.3.2. *PCA approaches*

PCA can be used to estimate missing values as the results are expected to capture the major variability among process variables. Wiberg (1976) proposed a cost function that considers only the values at the indices where x_{ij} was observed:

$$c = \sum_{i,j} o_{ij}\left(x_{ij} - \sum_{k=1}^{a} t_{ik}p_{jk}\right)^2, \tag{7}$$

where o_{ij} is the (i,j)th element of $\mathbf{O} \in \mathbb{R}^{n\times d}$ which is an indication matrix that has 1 at the entries with observed data points x_{ij} and 0 elsewhere. Grung and Manne (1998) proposed an alternating least-squares (ALS) type of approach that minimizes the cost function c by alternating between the estimation of the matrix $\mathbf{P}$ with fixed matrix $\mathbf{T}$ and the estimation of the matrix $\mathbf{T}$ with fixed matrix $\mathbf{P}$. Ilin and Raiko (2010) included a bias term to obtain an accurate least-squares solution. The update rules are

$$\mathbf{t}_i = (\mathbf{P}^{(i)\top}\mathbf{P}^{(i)})^{-1}\mathbf{P}^{(i)\top}(\mathring{\mathbf{X}}_{i:}^{\top} - \mathbf{m}^{(i)}), \qquad i = 1,\dots,n, \tag{8}$$

$$m_j = \frac{1}{|\mathbf{O}_{:j}|}\sum_{i\in\mathbf{O}_{:j}}(x_{ij} - \mathbf{t}_i^{\top}\mathbf{p}_j), \qquad j = 1,\dots,d, \tag{9}$$

$$\mathbf{p}_j^{\top} = (\mathring{\mathbf{X}}_{:j} - m_j\mathbf{1})^{\top}\mathbf{T}^{(j)}(\mathbf{T}^{(j)\top}\mathbf{T}^{(j)})^{-1}, \qquad j = 1,\dots,d, \tag{10}$$

where $\mathbf{t}_i \in \mathbb{R}^{a\times 1}$ is the transpose of ith row of the matrix $\mathbf{T}$, $\mathring{\mathbf{X}} \in \mathbb{R}^{n\times d}$ is a data matrix $\mathbf{X}$ with its missing values replaced by zeros, m_j is the jth element of the vector $\mathbf{m}$, $\mathbf{O}_{:j} \in \mathbb{R}^{n\times 1}$ is the jth column of the matrix $\mathbf{O}$, $|\mathbf{O}_{:j}|$ is the 1-norm of $\mathbf{O}_{:j}$ which is the same as the number of observed values, $\mathbf{p}_j \in \mathbb{R}^{a\times 1}$ is the transpose of jth row of the matrix $\mathbf{P}$, and $\mathbf{1} \in \mathbb{R}^{n\times 1}$ is the one vector, respectively. Here, the derived matrices $\mathbf{T}^{(j)} \in \mathbb{R}^{n\times a}$, $\mathbf{m}^{(i)} \in \mathbb{R}^{d\times 1}$, and $\mathbf{P}^{(i)} \in \mathbb{R}^{d\times a}$ are calculated from

$$\mathbf{T}^{(j)}_{i:} = \begin{cases} \mathbf{t}_i^{\mathsf{T}}, & \text{for } o_{ij} = 1,\\ \mathbf{0} \in \mathbb{R}^{1\times a}, & \text{for } o_{ij} = 0, \end{cases} \qquad j = 1, \dots, d, \tag{11}$$

$$m^{(i)}_j = \begin{cases} m_j, & \text{for } o_{ij} = 1,\\ 0, & \text{for } o_{ij} = 0, \end{cases} \qquad i = 1, \dots, n, \tag{12}$$

$$\mathbf{P}^{(i)}_{j:} = \begin{cases} \mathbf{p}_j^{\mathsf{T}}, & \text{for } o_{ij} = 1,\\ \mathbf{0} \in \mathbb{R}^{1\times a}, & \text{for } o_{ij} = 0, \end{cases} \qquad i = 1, \dots, n. \tag{13}$$

An alternating least-squares algorithm could be implemented following (8)–(13) (Alternating) or using MATLAB's pca command (ALS). While the MATLAB version of this software contains 10 imputation algorithms, the Python version has 9 algorithms excluding ALS. An important consideration, however, is that the same algorithm can result in different outputs due to differences in the software implementation (Severson et al., 2017). Given that the pca command starts with random initial guesses which significantly affects the imputation results, the ALS algorithm was run multiple times and the most *plausible* imputation result was selected as the final result for ALS. To quantify the plausibility of the imputation result, $\text{cont}^{T^2}_{ij}$ values (Zhu and Braatz, 2014; Severson et al., 2016) as expressed in (5) were calculated for each imputed element. Among the $\text{cont}^{T^2}_{ij}$ values at every imputed element, the maximum value was chosen as the representative when comparing the imputation results from different initial guesses. Given that the high $\text{cont}^{T^2}_{ij}$ values imply that the imputed values are not following the major latent relationship, the imputation result with the smallest $\max_{o_{ij}=0} \text{cont}^{T^2}_{ij}$ value was chosen as the final result for ALS algorithm. Note that $o_{ij} = 0$ means only the indices with imputed values were considered. Five runs were used for ALS algorithm in this article where the number five was obtained from the heuristics and may depend on the size of the input matrix and the level of missingness.

Troyanskaya et al. (2001) imputed missing values by using the singular value decomposition (SVD)-based PCA approach (SVDImpute) for calculating $\mathbf{P}$ and $\mathbf{T}$. When performing the SVD, the data matrix is expressed as

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{T}}, \tag{14}$$

where $\mathbf{U} \in \mathbb{R}^{n\times n}$ and $\mathbf{V} \in \mathbb{R}^{d\times d}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times d}$ is a pseudo-diagonal matrix whose main diagonal elements are the singular values. The PCA solution can be obtained by taking the a largest singular values from $\boldsymbol{\Sigma}$ and constructing the matrices $\mathbf{T}$ and $\mathbf{P}$ by taking the columns of $\mathbf{U}\boldsymbol{\Sigma}$ and columns of $\mathbf{V}$ that match with the chosen singular values, respectively (Jolliffe, 1986). The same solution also can be obtained by applying PCA to the covariance matrix,

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^{\mathsf{T}}\mathbf{X} = \mathbf{V}\mathbf{D}\mathbf{V}^{\mathsf{T}}, \tag{15}$$

where $\mathbf{D} = \frac{1}{n-1}\boldsymbol{\Sigma}^{\mathsf{T}}\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$. Here, the matrix $\mathbf{P}$ is constructed from the first a columns of the matrix $\mathbf{V}$, whereas the matrix $\mathbf{T}$ is computed from the matrix multiplication $\mathbf{XP}$ as shown in (1). Column-wise mean values are imputed at the initial step as the matrix should be complete for SVD. Then the SVD steps and the imputation steps are iterated until the reconstructed matrix converges. This algorithm also minimizes the cost function c in (7) (Ilin and Raiko, 2010).

Imtiaz and Shah (2008) proposed PCA-data augmentation (PCADA) algorithm which adds a bootstrap resampling step to the above algorithm to consider the measurement errors during the imputation step. In this algorithm, a matrix with true values without any noise, $\mathbf{X}^{\text{nf}} \in \mathbb{R}^{n\times d}$ where nf stands for noise-free, is introduced that can be calculated as

$$\mathbf{X}^{\text{nf}} = \hat{\mathbf{X}}\hat{\mathbf{P}}\hat{\mathbf{P}}^{\mathsf{T}}, \tag{16}$$

where $\hat{\mathbf{X}}$ and $\hat{\mathbf{P}}$ are the reconstructed matrix and loading matrix obtained using bootstrap resampling, respectively. Defining the residual at $\mathbf{O}$ indices as

$$r_{ij} = \begin{cases} 0, & \text{for } o_{ij} = 0,\\ x_{ij} - x^{\text{nf}}_{ij}, & \text{for } o_{ij} = 1, \end{cases} \tag{17}$$

each reconstructed matrix $\hat{\mathbf{X}}^{(k)}$ in the K-bootstrap dataset is obtained from

$$\hat{x}^{(k)}_{ij} = \begin{cases} x^{\text{nf}}_{ij} + r_{lj}, & \text{for } o_{ij} = 0,\\ x_{ij}, & \text{for } o_{ij} = 1, \end{cases} \tag{18}$$

where l is a random integer in $\mathbf{O}_{:j}$. SVD is performed on each $\hat{\mathbf{X}}^{(1)}, \dots, \hat{\mathbf{X}}^{(K)}$ to find the loading matrices $\hat{\mathbf{P}}^{(1)}, \dots, \hat{\mathbf{P}}^{(K)}$. Then, the matrices $\hat{\mathbf{X}}$ and $\hat{\mathbf{P}}$ are calculated from

$$\hat{\mathbf{X}} = \frac{1}{K}\sum_{k=1}^{K} \hat{\mathbf{X}}^{(k)}, \tag{19}$$

$$\hat{\mathbf{P}} = \frac{1}{K}\sum_{k=1}^{K} \hat{\mathbf{P}}^{(k)}, \tag{20}$$

respectively, which are used in (16) for iteration until the sum of squared errors at $\mathbf{O}$ indices defined as

$$\text{SSE}_{\text{obs}} = \sum_{i,j} (x_{ij} - x^{\text{nf}}_{ij})^2, \quad \text{for } o_{ij} = 1, \tag{21}$$

converges. Imtiaz and Shah (2008) reported that PCADA including (21) converges even for datasets that have very low signal-to-noise ratios and up to 25% missing data. However, since the random noise is added in (18), Imtiaz and Shah (2008) also noted that the convergence of PCADA is not smooth like other PCA-based methods, which can be observed in Fig. 9 in their paper. If PCADA does not converge, it is recommended to increase the number of bootstraps (K), as the variance in $\hat{\mathbf{X}}$ and $\hat{\mathbf{P}}$ in (19) and (20) decreases based on the central limit theorem, which would be beneficial for the convergence of (21). Similar to the SVDImpute algorithm, column-wise mean values are imputed at the initial step for the PCADA algorithm.

2.3.3. *Probabilistic PCA (PPCA) approaches*

Probabilistic PCA (PPCA) approaches assume there is a distribution of latent variables with respect to the observation (Tipping and Bishop, 1999). The most common model used in PPCA is a factor analysis based on

$$\mathbf{x} = \mathbf{Pt} + \boldsymbol{\mu} + \boldsymbol{\epsilon}, \tag{22}$$

where $\mathbf{x} \in \mathbb{R}^{d\times 1}$ is the observation vector, $\mathbf{P} \in \mathbb{R}^{d\times a}$ is the loading matrix, $\mathbf{t} \in \mathbb{R}^{a\times 1}$ is the vector of latent variables, $\boldsymbol{\mu} \in \mathbb{R}^{d\times 1}$ is the mean, and $\boldsymbol{\epsilon} \in \mathbb{R}^{d\times 1}$ is the noise. The noise is typically assumed to follow the multivariate normal distribution $\boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I}_d)$, resulting in

$$\mathbf{x}|\mathbf{t} \sim N(\mathbf{Pt} + \boldsymbol{\mu}, \sigma^2\mathbf{I}_d). \tag{23}$$

With an additional assumption that the latent variables follows the multivariable normal distribution $\mathbf{t} \sim N(\mathbf{0}, \mathbf{I}_a)$, the distribution of the observation vector is given by

$$\mathbf{x} \sim N(\boldsymbol{\mu}, \mathbf{PP}^{\mathsf{T}} + \sigma^2\mathbf{I}_d). \tag{24}$$

Given that $\mathbf{S} \in \mathbb{R}^{d\times d}$ is the sample covariance matrix of $\mathbf{x}_n$ and is expressed as

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{x}_n - \boldsymbol{\mu})(\mathbf{x}_n - \boldsymbol{\mu})^{\mathsf{T}}, \tag{25}$$

the maximum likelihood estimates (MLE) for $\mathbf{P}$ and σ^2 are (Tipping and Bishop, 1999)

$$\tilde{\mathbf{P}} = \mathbf{SP}(\sigma^2\mathbf{I}_a + (\mathbf{P}^{\mathsf{T}}\mathbf{P} + \sigma^2\mathbf{I}_a)^{-1}\mathbf{P}^{\mathsf{T}}\mathbf{SP})^{-1}, \tag{26}$$

$$\tilde{\sigma}^2 = \frac{1}{d}\text{tr}(\mathbf{S} - \mathbf{SP}(\mathbf{P}^{\mathsf{T}}\mathbf{P} + \sigma^2\mathbf{I}_a)^{-1}\tilde{\mathbf{P}}^{\mathsf{T}}), \tag{27}$$

where $\tilde{\mathbf{P}}$ and $\tilde{\sigma}^2$ are the updated values of $\mathbf{P}$ and σ^2, respectively. The latent variable $\mathbf{t}_n$ can be estimated using the MLE of the matrix $\mathbf{P}$ from the conditional equation (Tipping and Bishop, 1999)

$$\langle \mathbf{t}_n | \mathbf{x}_n \rangle = (\mathbf{P}_{\text{MLE}}{}^{\top}\mathbf{P}_{\text{MLE}} + \sigma^2 \mathbf{I}_a)^{-1} \mathbf{P}_{\text{MLE}}{}^{\top}(\mathbf{x}_n - \boldsymbol{\mu}). \tag{28}$$

Then, the missing values can be reconstructed using the loading matrix and the latent variables. Similar to the pca command, PPCA can be performed by using MATLAB's ppca command.

Yu et al. (2010) extended this approach by considering the missing elements of the data matrix $\mathbf{X}$ as well as the latent variables ($\mathbf{t}_n$), the loading matrix ($\mathbf{P}$), and the residual variance (σ^2) as the unknown variables. This algorithm, which is called PPCA-M (Severson et al., 2017), is given by

$$\langle \mathbf{t}_i \rangle = (\mathbf{W}^{(i)})^{-1} \sum_{j \in \mathbf{O}_{i:}} \mathbf{p}_j (x_{ij} - \mu_j), \tag{29}$$

$$\langle x_{ij} \rangle = \begin{cases} \langle \mathbf{t}_i \rangle^{\top} \mathbf{p}_j + \mu_j, & \text{for } o_{ij} = 0, \\ x_{ij}, & \text{for } o_{ij} = 1, \end{cases} \tag{30}$$

$$\langle \mathbf{t}_i \mathbf{t}_i^{\top} \rangle = \sigma^2 (\mathbf{W}^{(i)})^{-1} + \langle \mathbf{t}_i \rangle \langle \mathbf{t}_i \rangle^{\top}, \tag{31}$$

$$\langle \mathbf{x}_i \mathbf{x}_i^{\top} \rangle_{jk} = \begin{cases} \sigma^2 (\mathbf{p}_j^{\top} (\mathbf{W}^{(i)})^{-1} \mathbf{p}_k) + \langle x_{ij} \rangle \langle x_{ik} \rangle, & \text{for } o_{ij} = o_{ik} = 0,\ \forall j \neq k, \\ \sigma^2 (1 + \mathbf{p}_j^{\top} (\mathbf{W}^{(i)})^{-1} \mathbf{p}_k) + \langle x_{ij} \rangle \langle x_{ik} \rangle, & \text{for } o_{ij} = o_{ik} = 0,\ \forall j = k, \\ \langle x_{ij} \rangle x_{ik}, & \text{for } o_{ij} = 0,\ o_{ik} = 1, \\ x_{ij} \langle x_{ik} \rangle, & \text{for } o_{ij} = 1,\ o_{ik} = 0, \\ x_{ij} x_{ik}, & \text{for } o_{ij} = o_{ik} = 1, \end{cases} \tag{32}$$

$$\langle \mathbf{x}_i \mathbf{t}_i^{\top} \rangle_{j:} = \begin{cases} \sigma^2 \mathbf{p}_j^{\top} (\mathbf{W}_{(i)})^{-1} + \langle x_{ij} \rangle \langle \mathbf{t}_i \rangle^{\top}, & \text{for } o_{ij} = 0, \\ x_{ij} \langle \mathbf{t}_i \rangle^{\top}, & \text{for } o_{ij} = 1, \end{cases} \tag{33}$$

with the other parameters updated by

$$\mathbf{W}^{(i)} = \sum_{j \in \mathbf{O}_{i:}} \mathbf{p}_j \mathbf{p}_j^{\top} + \sigma^2 \mathbf{I}_a, \tag{34}$$

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} (\langle \mathbf{x}_i \rangle - \mathbf{P} \langle \mathbf{t}_i \rangle), \tag{35}$$

$$\mathbf{P} = \left(\sum_{i=1}^{n} (\langle \mathbf{x}_i \mathbf{t}_i^{\top} \rangle - \boldsymbol{\mu} \langle \mathbf{t}_i \rangle^{\top}) \right) \left(\sum_{i=1}^{n} \langle \mathbf{t}_i \mathbf{t}_i^{\top} \rangle \right)^{-1}, \tag{36}$$

$$\sigma^2 = \frac{1}{nd} \sum_{i=1}^{n} \text{tr}(\langle \mathbf{x}_i \mathbf{x}_i^{\top} \rangle - 2 \langle \mathbf{x}_i \mathbf{t}_i^{\top} \rangle \mathbf{P}^{\top} - 2 \boldsymbol{\mu} \langle \mathbf{x}_i \rangle^{\top} + 2 \boldsymbol{\mu} \langle \mathbf{t}_i \rangle^{\top} \mathbf{P}^{\top} + \mathbf{P} \langle \mathbf{t}_i \mathbf{t}_i^{\top} \rangle \mathbf{P}^{\top} + \boldsymbol{\mu} \boldsymbol{\mu}^{\top}), \tag{37}$$

where $\mathbf{O}_{i:} \in \mathbb{R}^{1 \times d}$ is the ith row of the matrix $\mathbf{O}$.

Oba et al. (2003) introduced a Bayesian method for PPCA (BPCA) which estimates both the model parameters $\boldsymbol{\theta} = \{\mathbf{P}, \boldsymbol{\mu}, \sigma^2\}$ and the missing values using a variational Bayes algorithm (Attias, 2013). BPCA has three steps: principal component regression, Bayesian estimation, and an EM-like repetitive algorithm. Oba et al. (2003) noted that the posterior distributions of parameters and missing values are likely to approach the global optimal due to the simple landscape of the objective function.

2.3.4. Matrix completion approaches

Approaches used in the matrix completion literature often assume that the given matrix has a low rank. The assumption of low rank is the same as in the PCA approach, in which a few principal components are significant and the others are considered to be noise. The matrix completion approach determines the missing values by solving the optimization (Candès and Recht, 2012)

$$\begin{aligned} &\min_{\mathbf{A}} \ \|\mathbf{A}\|_* \\ &\text{subject to } a_{ij} = x_{ij}, \quad \text{for } o_{ij} = 1, \end{aligned} \tag{38}$$

where $\|\cdot\|_*$ is the nuclear norm, which is the sum of the singular values. By defining the orthogonal projector $\mathcal{P}_{\boldsymbol{\Omega}}$ as

$$[\mathcal{P}_{\Omega}(\mathbf{A})]_{ij} = \begin{cases} a_{ij}, & \text{for } \boldsymbol{\Omega}_{ij} = 1, \\ 0, & \text{for } \boldsymbol{\Omega}_{ij} = 0, \end{cases} \tag{39}$$

(38) can be expressed as

$$\begin{aligned} &\min_{\mathbf{A}} \ \|\mathbf{A}\|_* \\ &\text{subject to } \mathcal{P}_{\mathbf{O}}(\mathbf{A}) = \mathcal{P}_{\mathbf{O}}(\mathbf{X}). \end{aligned} \tag{40}$$

Cai et al. (2010) proposed a singular value thresholding (SVT) algorithm, which ignores singular values less than a threshold τ, to minimize the nuclear norm. The SVD of a matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with rank r is expressed as (14) with only r main diagonal elements of the matrix $\boldsymbol{\Sigma}$ being nonzero. Following this idea, Cai et al. (2010) introduced the soft-thresholding operator $\mathcal{D}_\tau$ defined by

$$\mathcal{D}_\tau(\mathbf{X}) = \mathbf{U} \mathcal{D}_\tau(\boldsymbol{\Sigma}) \mathbf{V}^{\top}, \quad \mathcal{D}_\tau(\boldsymbol{\Sigma}) = \text{diag}((\sigma_i - \tau)_+), \tag{41}$$

where σ_i is the ith singular value and $t_+ = \max\{0, t\}$. Cai et al. (2010) proved that, for each $\tau > 0$ and $\mathbf{X} \in \mathbb{R}^{n \times d}$, the operator $\mathcal{D}_\tau$ obeys

$$\mathcal{D}_\tau(\mathbf{X}) = \arg\min_{\mathbf{Y}} \tfrac{1}{2} \|\mathbf{X} - \mathbf{Y}\|_F^2 + \tau \|\mathbf{Y}\|_*, \tag{42}$$

where $\|\cdot\|_F$ is the Frobenius norm which is the square root of the sum of the squares of its elements. The SVT algorithm iterates the equations:

$$\mathbf{A} = \mathcal{D}_\tau(\mathbf{Z}), \tag{43}$$

$$\mathbf{Z} = \mathbf{Z} + \delta \mathcal{P}_{\mathbf{O}}(\mathbf{X} - \mathbf{A}), \tag{44}$$

where the scalar $\delta > 0$ is a step size usually set as $1.2 \frac{nd}{|\mathbf{O}|}$, the initial $\mathbf{Z}$ is set to $k_0 \delta \mathcal{P}_{\mathbf{O}}(\mathbf{X})$, and k_0 is the smallest integer that is larger than $\frac{\tau}{\delta \|\mathcal{P}_{\mathbf{O}}(\mathbf{X})\|_2}$. After iterating (43) and (44) until the convergence at sufficiently large τ, the matrix $\mathbf{A}$ converges to the solution for (40). In this paper, $\tau = 5n$ was used based on the heuristics (Cai et al., 2010).

Lin et al. (2010) proposed an inexact augmented Lagrange multiplier (ALM) algorithm that solves a reformulated (40),

$$\begin{aligned} &\min_{\mathbf{A}, \mathbf{E}} \ \|\mathbf{A}\|_* \\ &\text{subject to } \mathbf{A} + \mathbf{E} = \mathbf{X}, \quad \mathcal{P}_{\mathbf{O}}(\mathbf{E}) = 0. \end{aligned} \tag{45}$$

This optimization can be solved by finding matrices $\mathbf{A}$ and $\mathbf{E}$ that minimize the partial augmented Lagrangian function,

$$\mathcal{L}(\mathbf{A}, \mathbf{E}, \mathbf{Z}, \mu) = \|\mathbf{A}\|_* + \langle \mathbf{Z}, \mathbf{X} - \mathbf{A} - \mathbf{E} \rangle + \tfrac{\mu}{2} \|\mathbf{X} - \mathbf{A} - \mathbf{E}\|_F^2, \tag{46}$$

where $\langle \cdot, \cdot \rangle$ is an inner product and $\mathbf{Z}$ is a Lagrange multiplier (Lin et al., 2010). Defining the soft-thresholding operator $\mathcal{S}_\epsilon$ as

$$[\mathcal{S}_\epsilon(\mathbf{A})]_{ij} = \begin{cases} a_{ij} - \epsilon, & \text{for } a_{ij} > \epsilon, \\ a_{ij} + \epsilon, & \text{for } a_{ij} < -\epsilon, \\ 0, & \text{otherwise}, \end{cases} \tag{47}$$

the solution can be found by iterating the following update rules (Lin et al., 2010):

$$(\mathbf{U}, \mathbf{S}, \mathbf{V}) = \text{svd}(\mathbf{X} - \mathbf{E}_{\text{old}} + \mu^{-1} \mathbf{Z}), \tag{48}$$

$$\mathbf{A} = \mathbf{U} \mathcal{S}_{\mu^{-1}}[\mathbf{S}] \mathbf{V}^{\top}, \tag{49}$$

$$\mathbf{E}_{\text{new}} = \mathcal{P}_{\bar{\mathbf{O}}}(\mathbf{X} - \mathbf{A} + \mu^{-1} \mathbf{Z}), \tag{50}$$

$$\mathbf{Z} = \mathbf{Z} + \mu(\mathbf{X} - \mathbf{A} - \mathbf{E}_{\text{new}}), \tag{51}$$

$$\mu = \begin{cases} \rho \mu, & \text{if } \min(\mu, \sqrt{\mu}) \frac{\|\mathbf{E}_{\text{new}} - \mathbf{E}_{\text{old}}\|_F}{\|\mathbf{X}\|_F} < \epsilon_2, \\ \mu, & \text{otherwise}, \end{cases} \tag{52}$$

$$\mathbf{E}_{\text{old}} = \mathbf{E}_{\text{new}}, \tag{53}$$

where $\bar{\mathbf{O}}$ is a complement of a set $\mathbf{O}$ (i.e., $\bar{\mathbf{O}} = \mathbf{1} - \mathbf{O}$) and the parameters with fixed values are given as $\rho = 1.2172 + 1.8588 \frac{|\mathbf{O}|}{nd}$ and $\epsilon_2 = 10^{-6}$. The initial values for other variables are given as $\mathbf{E} = \mathbf{Z} = \mathbf{0} \in \mathbb{R}^{n \times d}$ and $\mu = \frac{1}{\|\mathbf{X}\|_F}$ (Lin et al., 2010).

2.3.5. Determination of the number of principal components of the dataset with missing values

There are several methods for determining the number of principal components, a, used in the PCA and PPCA-based algorithms (Chiang et al., 2000). The *percent variance test method* chooses a as the minimum number of loading vectors needed to cover some pre-defined proportion of the total variance, typically 90%. The *scree test* plots the variance in decreasing order and chooses a where an elbow is detected. *Parallel analysis* (Horn, 1965) compares the variance profile of the given data to the profile obtained from data for which the measurement variables are uncorrelated. The value of a can also be determined by using *cross-validation* (Wold, 1978) with the prediction residual sum of squares (PRESS) statistic which is defined as

$$\mathrm{PRESS}^{(i)} = \frac{1}{nd} \|\mathbf{X} - \hat{\mathbf{X}}\|_F^2, \tag{54}$$

where $\hat{\mathbf{X}}$ is the reconstructed matrix of $\mathbf{X}$ using i loading vectors.

This article uses the cross-validation method to determine the number of principal components. As the imputed values for the missing indices, the loading matrix ($\mathbf{P}$), and the number of principal components (a) are mutually dependent, three steps were iterated until convergence of a: (1) imputation, (2) calculation of the matrix $\mathbf{P}$, and (3) determination of the integer a.[2]

2.4. Metrics for evaluation of imputation algorithms (Step C)

This article compares 9 algorithms for filling in the missing values: Mean imputation (MI), three PCA algorithms (Alternating, SVDImpute, and PCADA), three PPCA algorithms (PPCA, PPCA-M, and BPCA), and two matrix completion algorithms (SVT and ALM).[3] If the ground truth is known for the missing values, well-known metrics such as normalized root mean square error (NRMSE) or subspace angle (Björck and Golub, 1973) between the true and recovered $\mathbf{P}$ can be used for model evaluation (Severson et al., 2017). However, the ground truth is typically unknown for the process data with missing values. To compare the algorithms without knowing the ground truth, this section suggests the following three criteria which are in order of priority: feasibility, plausibility, and rapidity.

First, the imputed values can be checked whether they are within the pre-defined boundaries, which is related to the feasibility of the solution. Some process variables, such as the flow rate or concentration of each component, should satisfy nonnegativity constraints. In this case, a simple filter that checks the positiveness of each element can be used to quickly evaluate the feasibility of the solution. This concept can be extended to the data with some variables having specific upper or lower bounds. For the censoring case, the sensor threshold could be set as the bound to check whether the imputed values are outside the boundaries.

The second criterion is to check whether the imputed values are considered outliers, which is related to the plausibility of the solution. Random variables with unknown distribution are typically assumed to have a Gaussian distribution (Casella and Berger, 2002). As such, missing values collected during normal operations have a low likelihood of being outliers among the observed values. Whether an estimate is considered an outlier can be assessed by calculating the T^2 and Q statistic contribution (Zhu and Braatz, 2014) at the missing indices using (5) and (6), respectively.

The last algorithm evaluation metric is the computation time, which is related to the rapidity of the algorithm to deal with missing values. Among imputation algorithms with similar accuracy, the algorithm with the shortest computation time is always preferred as it can quickly deal with information loss due to missing values during process control.

For the MATLAB version, each algorithm was additionally checked as to whether it ran without error because some commands in the PROPACK package (Larsen, 2004) such as `lansvd` occasionally induced errors due to convergence failures. Here, `try-catch` statements were used to detect the error when running the code.

3. Demonstration

This section demonstrates how the software imputes the values at the missing indices using the 9 algorithms described in Section 2.3 in three steps: preprocessing (Step A), imputation of missing values (Step B), and evaluation of imputation algorithms (Step C). A dataset of monoclonal antibody production from Chinese hamster ovary cells collected from a continuous biomanufacturing testbed at the Massachusetts Institute of Technology was used in this section. Fourteen measurements (cell diameter, total cell density, viable cell density, osmolality (Osmo), $\mathrm{pCO_2}$, pH, and the concentrations of calcium (Ca), glutamine (Gln), glutamate (Glu), glucose (Gluc), potassium (K), lactose (Lac), sodium (Na), and ammonium ($\mathrm{NH_4}$)) were recorded every 2 or 4 h over the course of a 30-day run, resulting in total 293 observations. All software and data will be released on github upon acceptance of the manuscript for journal publication.

While dynamic characteristics are well captured in some variables (e.g., total cell density, viable cell density, and concentrations of glutamate, glucose, and ammonium), it can be seen from Fig. 2 that most of the variables contain outliers around Day 25, which implies data contamination. In addition, missing values are clearly observed in osmolality between Days 7 and 9. The number of missing values at each variable is reported in Table 1.

3.1. Step A. Preprocessing

Step A-1. Temporary imputation of missing values. In this step, we choose one of the three options listed in Section 2.2.1 to temporarily fill in missing values to obtain the complete matrix that will be further used for outlier detection. Due to the explanation in Section 2.2.1, interpolation method was used for this demonstration (Fig. 3) while the results obtained by using the two other methods (mean imputation and last observed) can be found in Appendix C.

Step A-2. Outlier detection based on T^2 and Q contributions. T^2 and Q contributions ((5) and (6)) are calculated for the complete matrix obtained after step A-1. The confidence limit of 0.9999 was used to determine whether the observed values are outliers (Fig. 4). For this dataset, the observations during the first six days were protected from being considered outliers as values changed rapidly during the startup of the equipment. Nine iterations of Steps A-1 and A-2 were performed to identify outliers in all of the measurements (Table 2), with many grouped at around Days 20 and 25 (Fig. 5). The osmolality and glucose and ammonium concentrations had the fewest outliers. Glucose is a measurement commonly used in bioprocess control systems, so even a small number of outliers during operations can be problematic (Gambhir et al., 1999; Wlaschin and Hu, 2006; Craven et al., 2014). The pH, which is one of the most important measurements from an operational and control perspective, had a large number of outliers and is an even more important concern (Stephanopoulos and San, 1984; Alford, 2006; Konstantinov and Cooney, 2015; Hong et al., 2021). Most of the other measurements are not used in most process control systems, but are

[2] The different imputation algorithms can converge to different values for a.

[3] MATLAB version software includes the ALS algorithm which uses `pca` command, resulting in 10 algorithms.

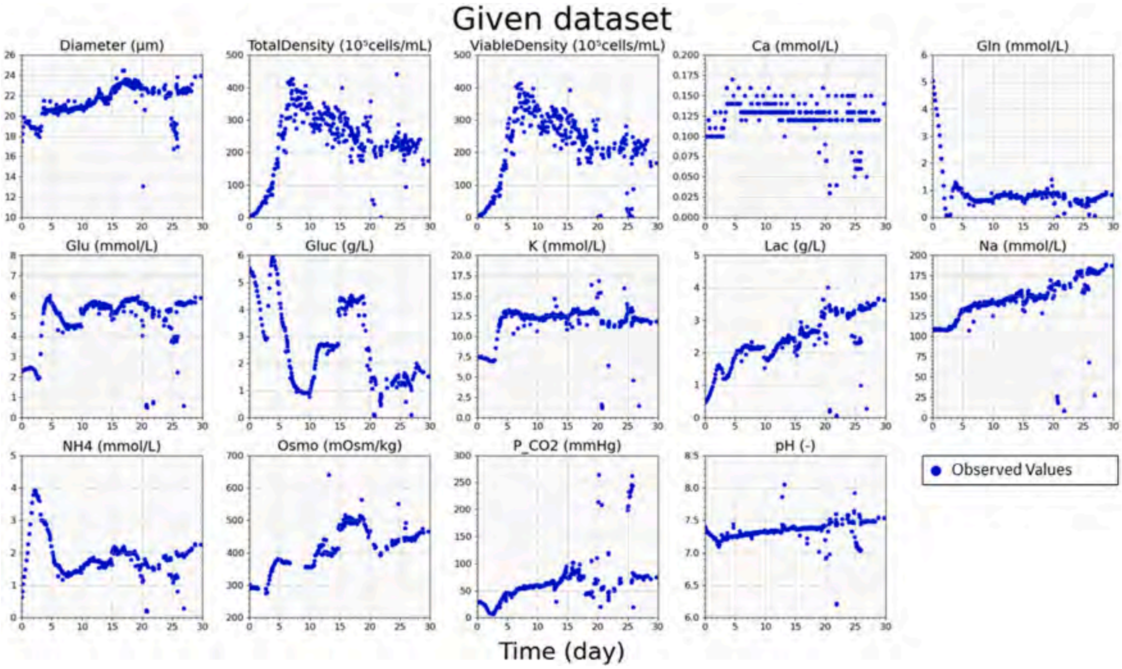


Fig. 2. Original biomanufacturing dataset of monoclonal antibody production of Chinese hamster ovary cells. Fourteen measurement variables were recorded every 2 or 4 h over the course of a 30-day run, resulting in total 293 observations.

Table 1
Number of missing values at each variable in the given dataset (Fig. 2). TotalD and ViableD stand for total density and viable density, respectively.

Diameter	TotalD	ViableD	Ca	Gln	Glu	Gluc	K	Lac	Na	NH_4	Osmo	P_{CO_2}	pH
7	4	7	0	3	0	0	0	1	0	0	68	23	5

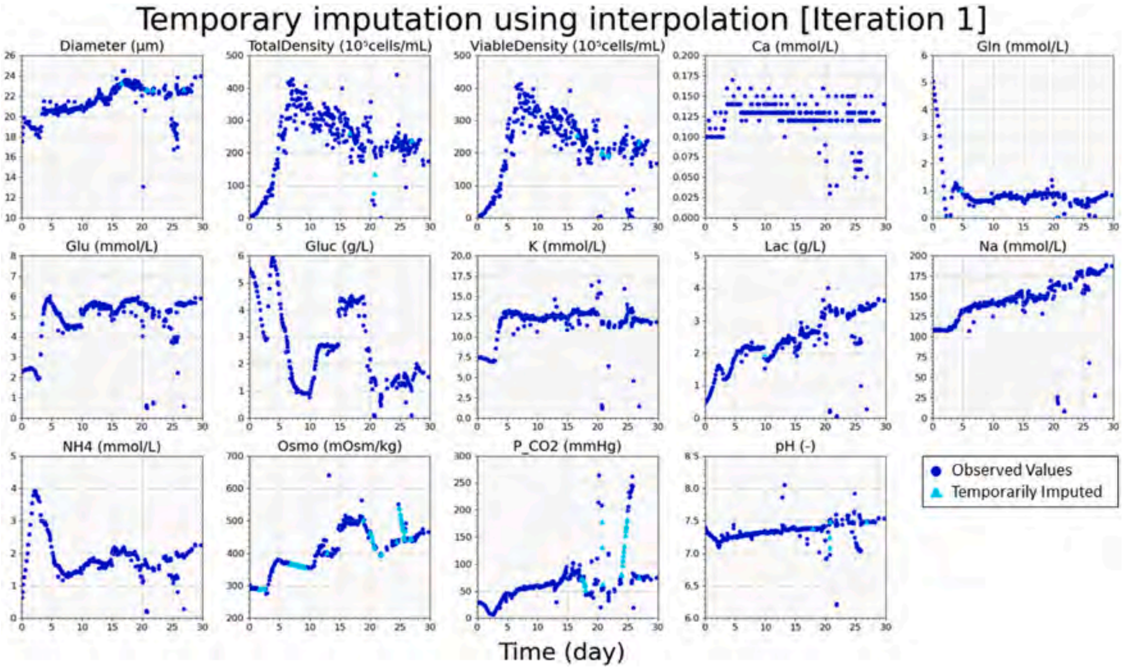


Fig. 3. Temporary imputation of missing values using interpolation. Preliminary imputation based on interpolation applied to the dataset in Fig. 2. The blue dots indicate the original observed values whereas the light blue triangles indicate the temporarily imputed values using interpolation.

useful for process modeling or process estimation (Fujimori et al., 1990; Heinzle et al., 2007; Biechele et al., 2015).

Step A-3. Elimination of low-quality observations. As can be seen from Fig. 5, there are (a) some data points that were detected as outliers but still seems to follow the main trend within the dataset (e.g., the point around Day 13 in Diameter) as well as the opposite cases where (b) the data points outside the major trend were not detected as outliers (e.g., points around Day 25 in Na and NH_4), which can be considered as

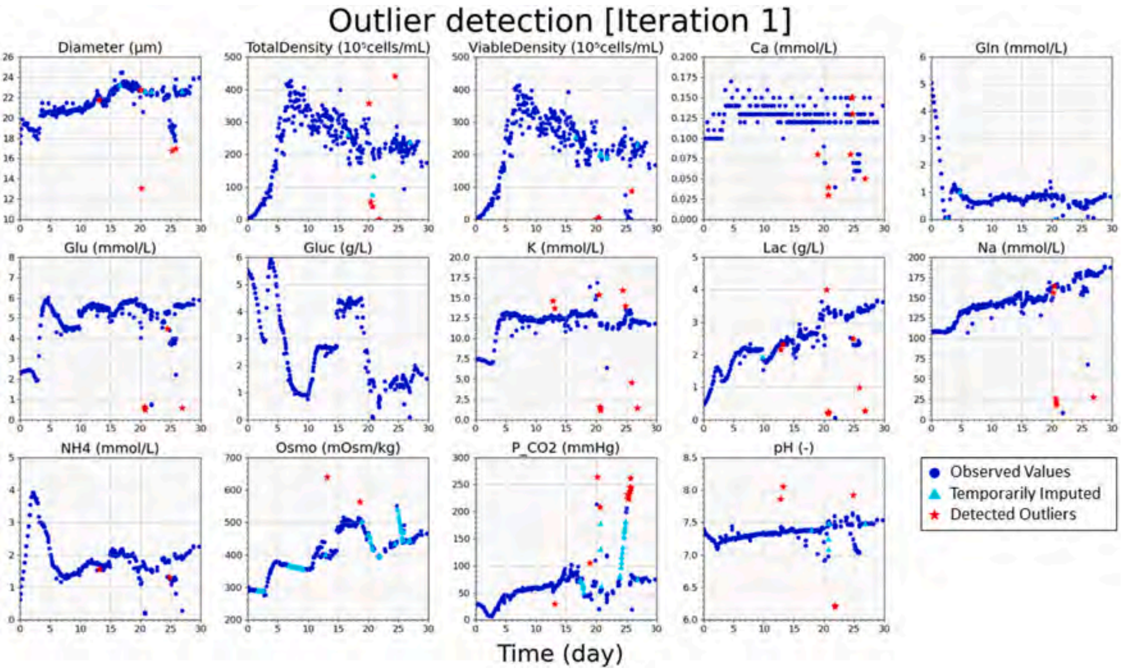


Fig. 4. Preprocessed dataset at Step A-2 in the first iteration. Indices with T^2 and Q statistic contributions exceeding the threshold based on the confidence limit of 0.9999 are considered outliers. Blue dots indicate the observed values, light blue triangles indicate the temporarily imputed values using interpolation, and red stars indicate the detected outliers.

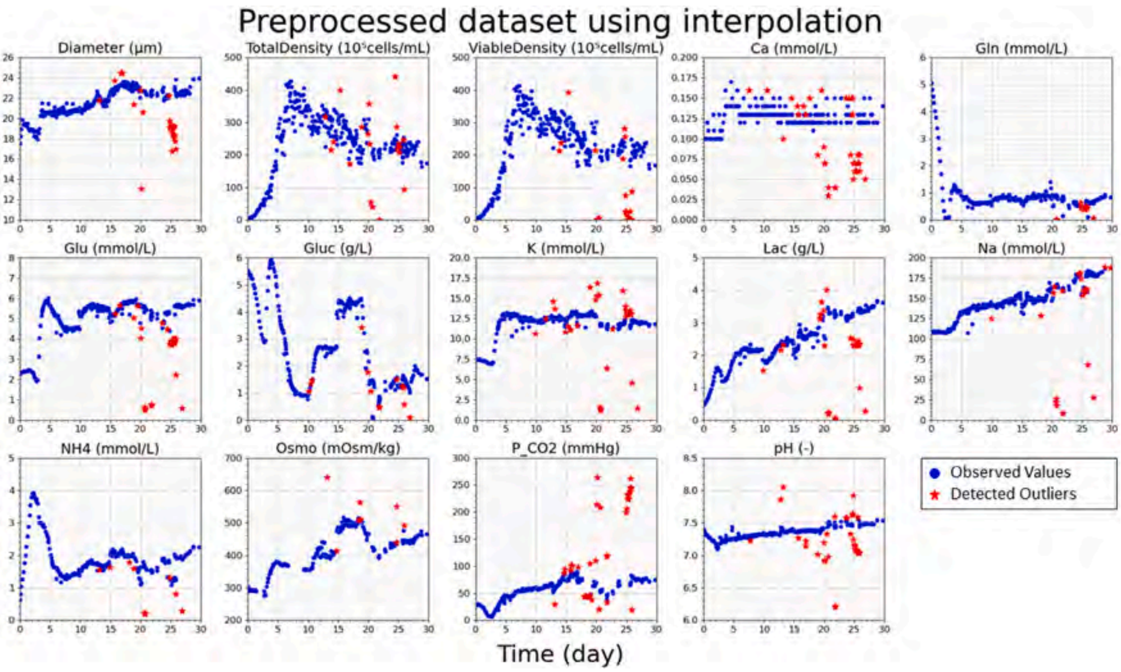


Fig. 5. The final result of Step A after 9 iterations of Step A-1 using interpolation and Step A-2. Blue dots indicate the observed values and red stars indicate the detected outliers.

Table 2
Number of detected outliers at each variable after nine iterations of Steps A-1 and A-2. TotalD and ViableD stand for total density and viable density, respectively.

Diameter	TotalD	ViableD	Ca	Gln	Glu	Gluc	K	Lac	Na	NH_4	Osmo	P_{CO_2}	pH
23	22	24	41	15	27	14	35	30	25	12	8	37	34

Table 3
The portion of raw data showing the false positive happened on Day 13 in Diameter. Cells with red colors indicate that the values were detected as outliers while blank cells indicate missing values.

Day	Diameter	TotalD	ViableD	Ca	Gln	Glu	Gluc	K	Lac	Na	NH_4	Osmo	P_{CO_2}	pH
12.31597	21.8	352.67	340.28	0.12	0.83	5.58	2.65	12.68	2.28	145.3	1.75	442	66.8	7.326
12.39931	21.91	320.88	312.69	0.13	0.81	5.62	2.66	12.63	2.29	146.9	1.75	403	63.8	7.327
12.56597	21.9	340.41	329.67	0.15	0.82	5.63	2.69	12.88	2.32	147	1.75	399	68	7.323
12.64931	22.05	288.98	282.65	0.14	0.83	5.54	2.67	12.53	2.32	148.5	1.74	440	67.6	7.315
12.73264	21.82	324.38	318.01	0.13	0.82	5.62	2.69	12.62	2.34	147.6	1.72	403	64	7.341
12.81597	22.26	318.05	273.22	0.13	0.71	5.13	2.36	14.66	2.16	143.8	1.54		51.1	7.859
12.83125	22.66	225.77	220.71	0.12	0.89	5.52	2.73	12.1	2.35	147.8	1.77	400	55.4	7.432
12.85764	22.38	321.42	315.76	0.13	0.95	5.39	2.71	12.68	2.38	148.4	1.76	398	60.9	7.403
12.89931	21.91	317.68	312.83	0.13	0.85	5.67	2.72	12.64	2.43	149.1	1.72	394	60.6	7.381
12.98264	21.96	338.79	332.43	0.13	0.82	5.58	2.68	12.53	2.35	147.7	1.72	398	61.1	7.374
13.06597	22.16	333.78	325.56	0.13	0.82	5.49	2.66	12.63	2.31	145.3	1.72	396	57.5	7.413
13.14931	21.8	283.56	272.14	0.1	0.76	5.38	2.47	13.73	2.32	148.4	1.56	640	29.7	8.048
13.23264	22.12	321.48	314.17	0.13	0.79	5.6	2.67	12.64	2.4	149.6	1.7	400	61.6	7.372
13.31597	21.87	268.2	262.04	0.13	0.78	5.6	2.65	12.53	2.39	145.8	1.65	396	63.5	7.356
13.39931	21.85	310.54	305.11	0.13	0.79	5.55	2.65	12.51	2.39	147.7	1.69	395	62.5	7.361
13.56597	21.53	281.07	274.73	0.14	0.77	5.64	2.67	12.5	2.43	146.6	1.65	390	65.6	7.361

Table 4
The portion of raw data showing the false negative happened on Day 25 in Na and NH_4. Cells with red colors indicate that the values were detected as outliers while blank cells indicate missing values.

Day	Diameter	TotalD	ViableD	Ca	Gln	Glu	Gluc	K	Lac	Na	NH_4	Osmo	P_{CO_2}	pH
24.69722	22.27	286.49	281.54	0.15	0.63	5.27	1.5	12.37	3.2	177.6	1.84	440		7.643
24.73333	22.2	261.83	256.38	0.13	0.63	5.31	1.5	12.26	3.29	181.2	1.78	551		7.613
24.81667	19.76	220.34	74.64	0.05	0.43	4.39	1.29	13.97	2.48	174.2	1.31			7.918
24.83472	19.11	220.1	26.47	0.07	0.44	3.76	1.25	12.74	2.31	154.6	1.31			7.243
24.98333	19.27	236.1	22.6	0.06	0.45	3.83	1.23	13.48	2.3	157.8	1.23		198.9	7.161
25.06667	18.72	207	17.95	0.06	0.45	3.69	1.2	13.02	2.3	153.7	1.09		206.5	7.134
25.15	16.73	225.29	0.2	0.07	0.42	3.91	1.28	13.17	2.3	163.9	1.19		232.8	7.094
25.23333	19.35	218.62	8.08	0.06	0.42	3.9	1.29	13.11	2.3	158.4	1.24		222.8	7.074
25.31667	18.88	231.22	6.26	0.06	0.39	3.98	1.3	13.15	2.29	159.5	1.23		224.4	7.093
25.4	18.48	220.37	4.88	0.06	0.41	3.97	1.29	13.31	2.31	160.1	1.24		232.8	7.071
25.56667	18.08	208.68	1.34	0.08	0.41	3.99	1.3	13.15	2.31	160.4	1.27		245.5	7.043

Table 5
Number of missing values inside the preprocessed dataset after Step A. Note that the missing values in the preprocessed dataset include the missing values in the given dataset (Missing) and the outliers detected in Step A (Outliers), which are the indices that are colored in light blue and red in Fig. 6, respectively, while excluding the removed rows that are indicated as black. TotalD and ViableD stand for total density and viable density, respectively.

	Diameter	TotalD	ViableD	Ca	Gln	Glu	Gluc	K	Lac	Na	NH_4	Osmo	P_{CO_2}	pH
Missing	1	1	1	0	2	0	0	0	1	0	0	44	17	2
Outliers	6	7	6	17	2	4	6	10	7	6	3	7	18	13
Total	7	8	7	17	4	4	6	10	8	6	3	51	35	15

Table 6
Three metrics used for algorithm evaluation. Feasibility criterion indicates the number of imputed elements outside the boundaries, plausibility criterion indicates the number of imputed elements considered outliers, and rapidity criterion indicates the computation time in seconds.

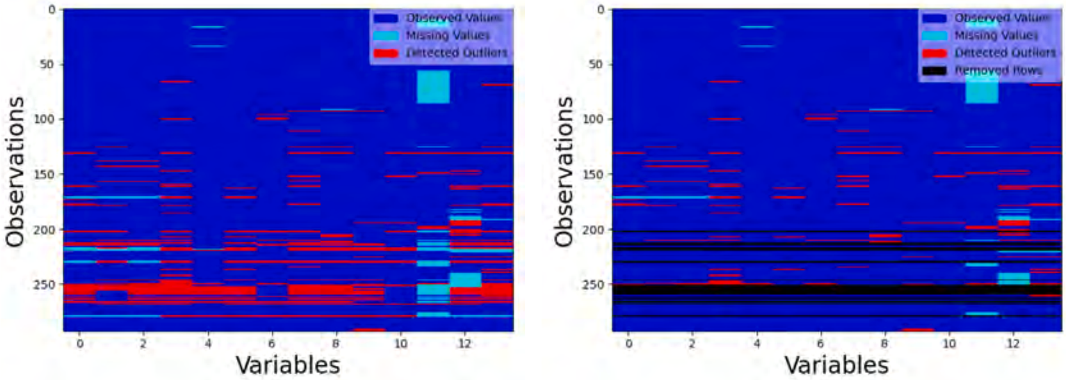
	MI	Alternating	SVDImpute	PCADA	PPCA	PPCA-M	BPCA	SVT	ALM
Feasibility	0	0	0	0	0	0	0	0	0
Plausibility	16	6	2	3	2	2	1	2	2
Rapidity (s)	0.00676	0.141	0.0355	51.3	66.1	26.9	1.12	4.68	0.0741

false positive and negative, respectively. However, it should be noted that most of these cases happen within low-quality observations as shown in Tables 3 and 4, respectively. These false cases were automatically removed during the elimination of low-quality observations in this step.

As mentioned in Section 3.1, the low-quality observations were determined based on whether the number of survived variables is larger than the number of principal components after the iteration of Steps A-1 and A-2. The number of principal components was determined to be 5 using the algorithm in Section 2.3.5 with interpolation as the imputation method. Fig. 6a displays the indication matrix of the dataset in Fig. 5 whereas Fig. 6b shows the indication matrix after removing observations that have less than five survived measurement variables which are shown in black. As can be seen from Figs. 6ab, observations with a high proportion of outliers (observations 210–220 and 250–270 which correspond to Days 20 and 25, respectively) were removed. Overall, 25 out of 293 observations were eliminated, resulting in a total of 268 observations being used for Step B. The number of missing values inside the preprocessed dataset after Step A (Fig. 6b) is shown in Table 5.

3.2. Step B. Imputation of missing values

This step imputes the missing values of the dataset that is preprocessed in Step A (Fig. 6) using the algorithms introduced in Section 2.3. Fig. 7 illustrates the imputation results using (a) MI, (b) Alternating, (c) SVDImpute, (d) PCADA, (e) PPCA, (f) PPCA-M, (g) BPCA, (h) SVT, and (i) ALM algorithms where observed values, imputed missing values, and replaced outliers are indicated as blue dots, green triangles, and green stars, respectively. Note that the imputed missing values or replaced outliers outside the fixed y-axis limits are not shown in Fig. 7, where most of them should have activated either feasibility or plausibility criteria.



(a) Indication matrix of missing values and outliers.

(b) Indication matrix after elimination of low-quality observations.

Fig. 6. Indication matrices of Fig. 5 when setting the threshold for the number of survived elements as 5. Observed values, missing values, detected outliers, and removed rows are indicated in blue, light blue, red, and black, respectively. Variables are in Diameter, TotalDensity, ViableDensity, Ca, Gln, Glu, Gluc, K, Lac, Na, NH_4, Osmo, P_{CO_2}, and pH order as in the previous figures.

3.3. Step C. Evaluation of imputation algorithms

This step evaluates each algorithm used in Step B which helps the user to determine which model to use for the given dataset. Table 6 displays the three evaluation metrics described in Section 2.4 where the lower bound for the feasibility criterion was set to zero to check the positiveness, and the confidence limit of 0.9999 was used for the plausibility criterion.

Based on Table 6, all nine algorithms had no imputed values outside the boundary, meaning that all missing values were estimated to be positive. As all algorithms satisfied the feasibility criterion, the plausibility metric then can be used for the next step. MI showed the maximum number of imputed values considered outliers, which is mainly due to the fact that MI is the only algorithm that does not take the latent relationship among variables into account. Alternating algorithm also showed a relatively high value of the plausibility metric, implying that the algorithm might not work well on the given dataset.

It can be seen from Table 6 that the values are similar for the plausibility metrics except for MI and Alternating. Therefore, the operator may decide whether to finalize the algorithm with the minimum plausibility metric, which is BPCA, or move on to the rapidity metric for further comparison. For the sensitive system where a fast response is necessary, algorithms with a low computation time such as SVDImpute or ALM could be preferred over BPCA where the computation time is an order of magnitude higher.

It should be noted that the three metrics, feasibility, plausibility, and rapidity, can only work as indirect criteria for algorithm evaluation. Having small numbers in Table 6 does not necessarily imply good performance, especially for feasibility and plausibility criteria when the missingness was due to censoring. However, the first two criteria can still work as soft guidance for model evaluation in the case of a time-series dataset with slow dynamics where an abrupt change in measurements is not expected.

4. Validation

A complete dataset where all measurements are recorded is required in order to validate whether the algorithms accurately impute the missing values. Such dataset for this section was constructed based on the preprocessed dataset after Step A (Fig. 6b) with missing values filled in using interpolation, resulting in 268 observations. The reason for not using only the complete observations is that doing so results in the elimination of 98 out of 293 observations, which is a data loss of approximately one-third. The complete dataset used in this section is illustrated in Fig. 8. Note that this dataset still contains the outliers indicated in red in Fig. 6b.

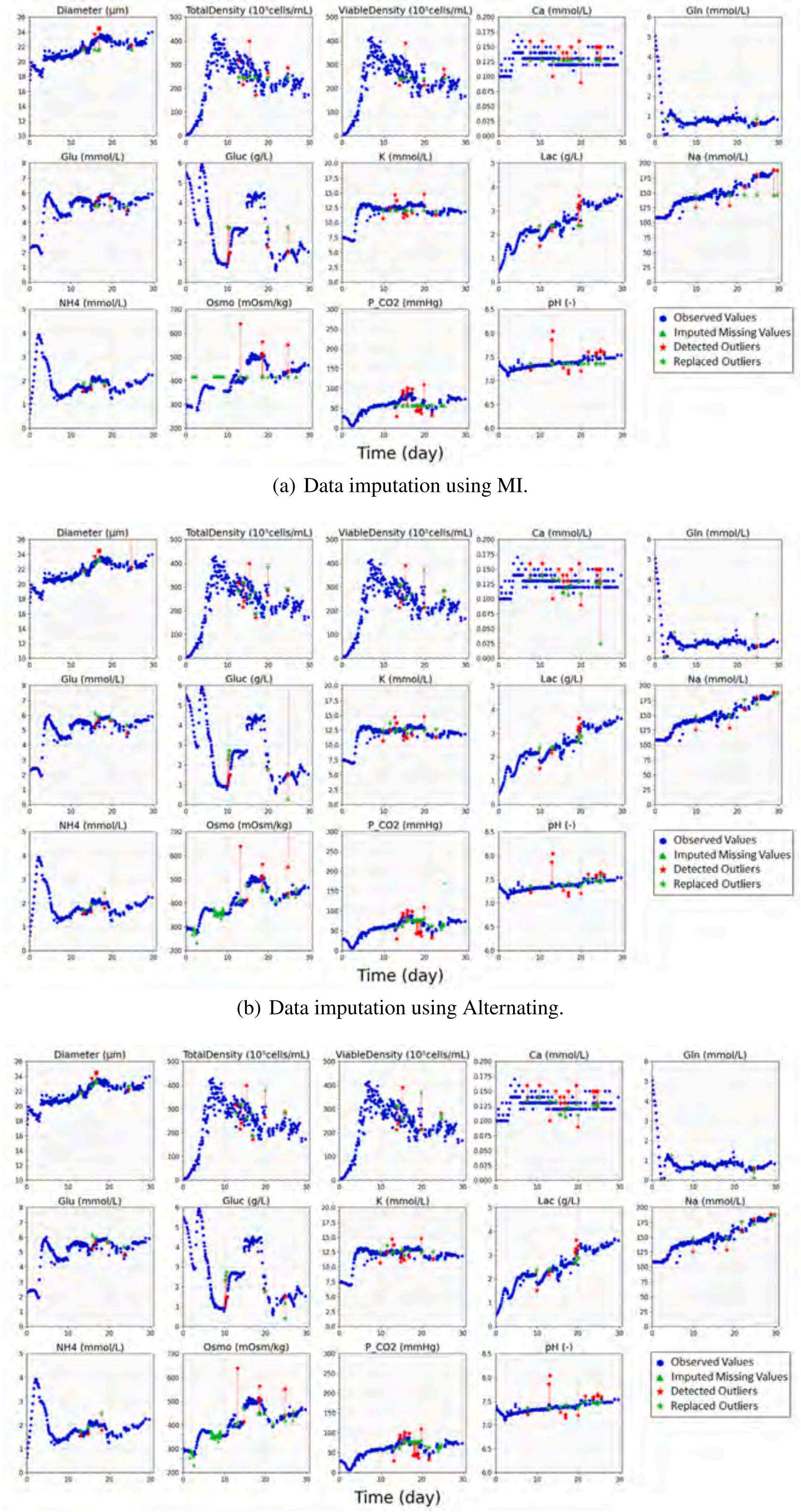
The software developed by Severson et al. (2017) was used to add various types of missingness to this dataset. This software enables the user to select the missingness type (e.g. missing completely at random (MCAR), sensor drop-out, multi-rate missingness, censoring, and patterned) and level to be added to the dataset. In this section, all five types of missingness with the level of 10% were added to the dataset, respectively, as shown in Fig. 9 to check how well each algorithm performs in each missingness scenario. The confidence limit of 0.9999 for outlier detection, the lower bound of zeros for the feasibility criterion, and the confidence limit of 0.9999 for the plausibility criterion were used, which are the same settings as in the Demonstration part (Section 3). Each scenario was repeated 50 times to obtain the distributions on the normalized root mean squared error (NRMSE) of each variable, three algorithm evaluation metrics including feasibility, plausibility, and rapidity, and the number of principal components. The representative imputation results of each algorithm at each missingness type are shown in Appendix D.

Variable-wise NRMSE can be defined as

$$\text{NRMSE}_j = \frac{\text{RMSE}_j}{\sigma_j} = \frac{1}{\sigma_j}\sqrt{\frac{\sum_{k\notin \hat{\mathbf{O}}_{:j}}(\hat{x}_{kj}-x_{kj})^2}{n-|\hat{\mathbf{O}}_{:j}|}}, \quad \text{for } j = 1,\ldots,d, \tag{55}$$

where σ_j is a variable-wise standard deviation of the full dataset, $\hat{\mathbf{O}}$ is an indication matrix of observed data after adding missingness, and $\hat{x}_{ij}$ is an imputed value using the software. NRMSE $= 1$ indicates that the imputed values are generally one-sigma away from the true values. In this section, variable-wise NRMSE was used instead of the overall NRMSE for algorithm evaluation as the importance of each variable might vary. For example, control variables may have higher significance than other output variables as the process might undergo a transition between stages with respect to the trend of control variables. As mentioned in Section 3.1, pH is one of the most important measurement variables for the biomanufacturing dataset used in this section. Therefore, the NRMSE bar graph of pH should be considered an important factor when evaluating the performance of each algorithm.

In addition to the variable-wise NRMSE, three criteria introduced in Section 3.3 were used to evaluate each algorithm: feasibility, plausibility, and rapidity. A feasible solution is not always a plausible one. The mean imputation (MI) is a good example of this case where it always passes the feasibility test while it is not a plausible solution as it does not consider any dynamics. Conversely, a plausible solution does not necessarily mean that it is a feasible solution. For example, as the lower boundary for the feasibility criterion was set to zero for this dataset, the number of elements outside the boundaries is the same as the number of negative values in this section. In fact, the possibility of imputed elements having negative values depends on the distribution of

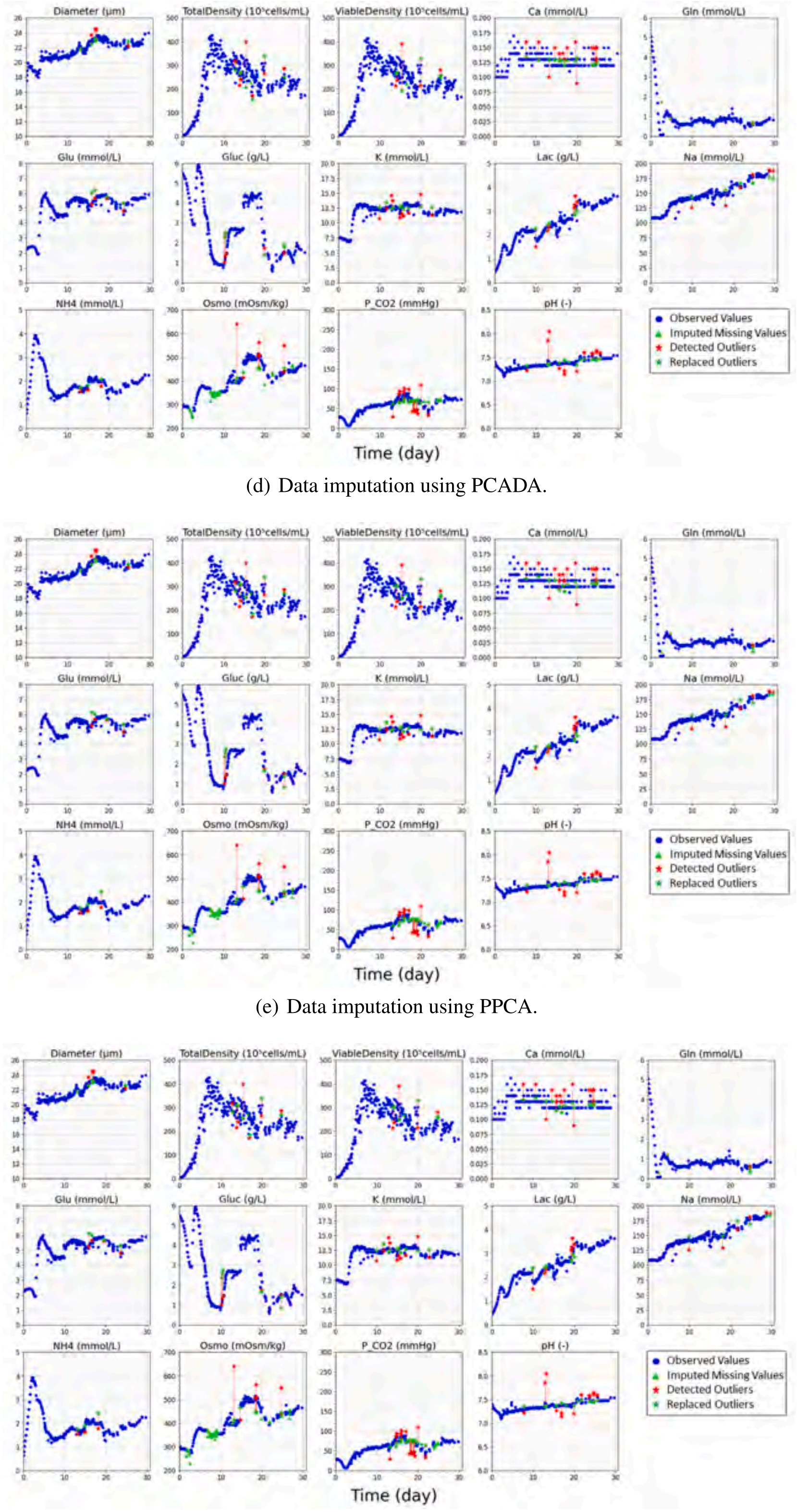


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. 7. Outlier detection and data imputation results of the biomanufacturing dataset using the software. Data imputation results using 9 algorithms where the observed values are marked as blue dots, imputed missing values are marked as green triangles, and the replaced outliers are marked as green stars.



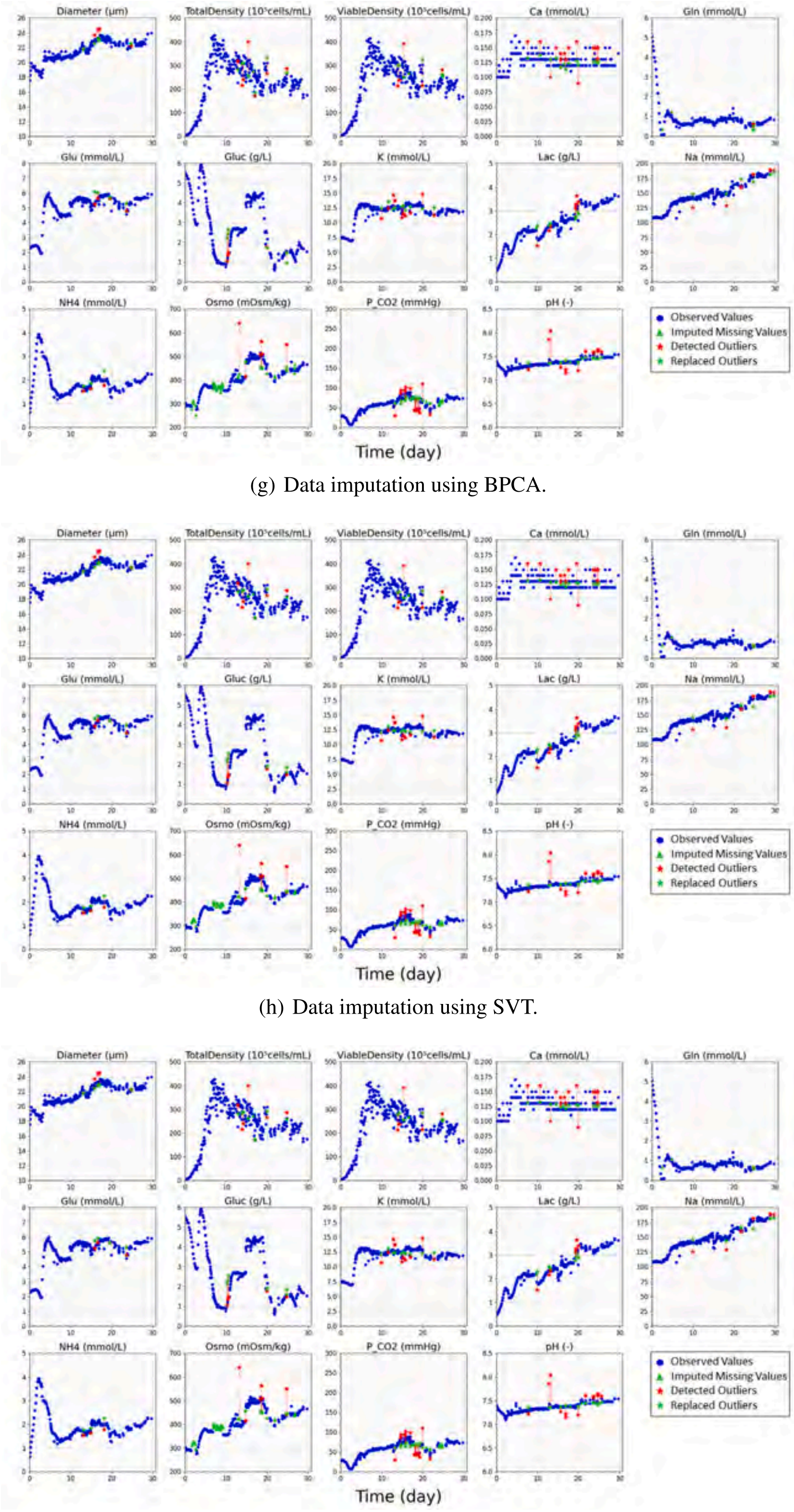
(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. 7. (*continued*).

the variable measurements. For instance, the distributions of variables such as total cell density, viable cell density, glutamine concentration, and pCO_2 have a small ratio of mean to variance. If the missingness happens near the elements very close to zero, there is a high possibility of negative values being imputed for that missing value. Thus, an ideal solution should satisfy both the feasibility and plausibility criteria.



(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. 7. (*continued*).

Computation time, which is used for the rapidity criterion, is also an important metric for algorithm evaluation. In the case of online control, a rapid response with respect to disturbances and controls is required for high-quality products. In order to prevent the data imputation step

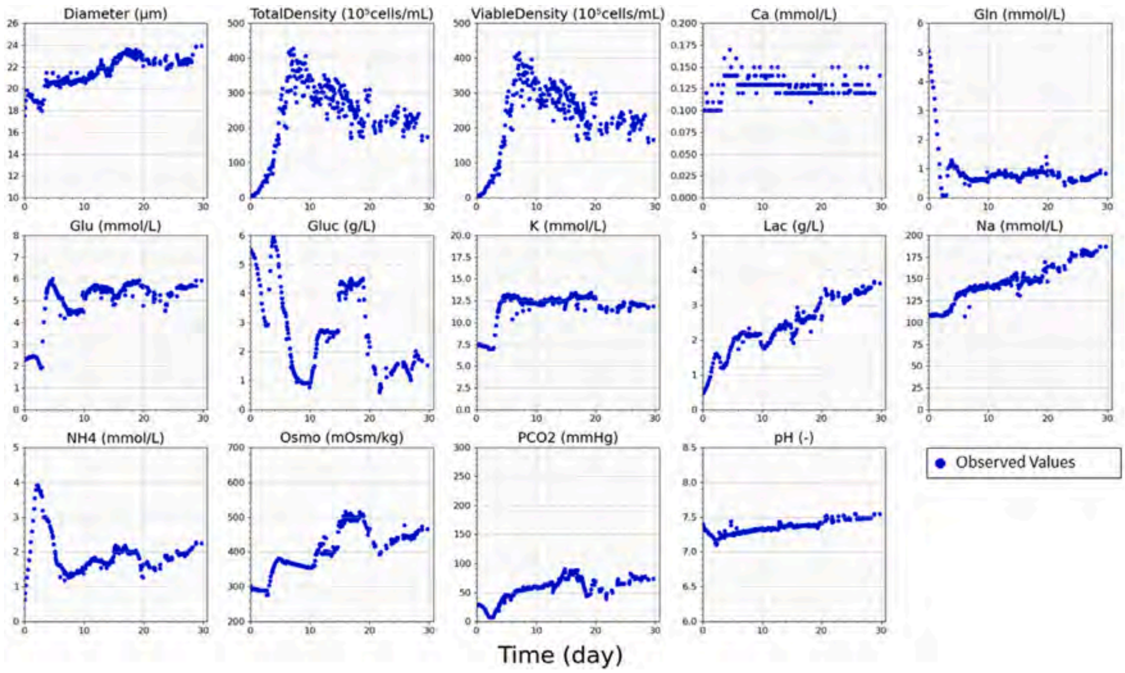
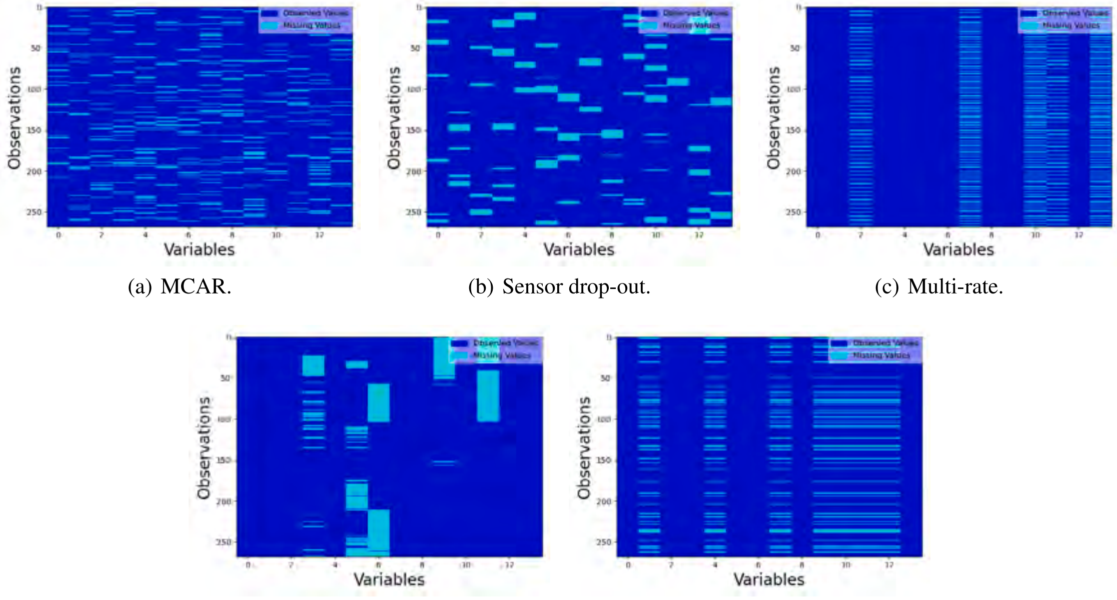


Fig. 8. Complete dataset used in Section 4. Rows indicated as black in Fig. 6b were eliminated and the missing values where the numbers are presented in the first row of Table 5 were filled in using interpolation. A total of 268 out of 293 observations in the original dataset were used in the dataset for validation.



(a) MCAR.

(b) Sensor drop-out.

(c) Multi-rate.

(d) Censoring.

(e) Patterned.

Fig. 9. Demonstration of how each of five missingness cases investigated in Section 4 with the level of 10% were added to a dataset using the software developed by Severson et al. (2017). This action was repeated 50 times in Section 4. Observed values and missing values are indicated as blue and light blue, respectively.

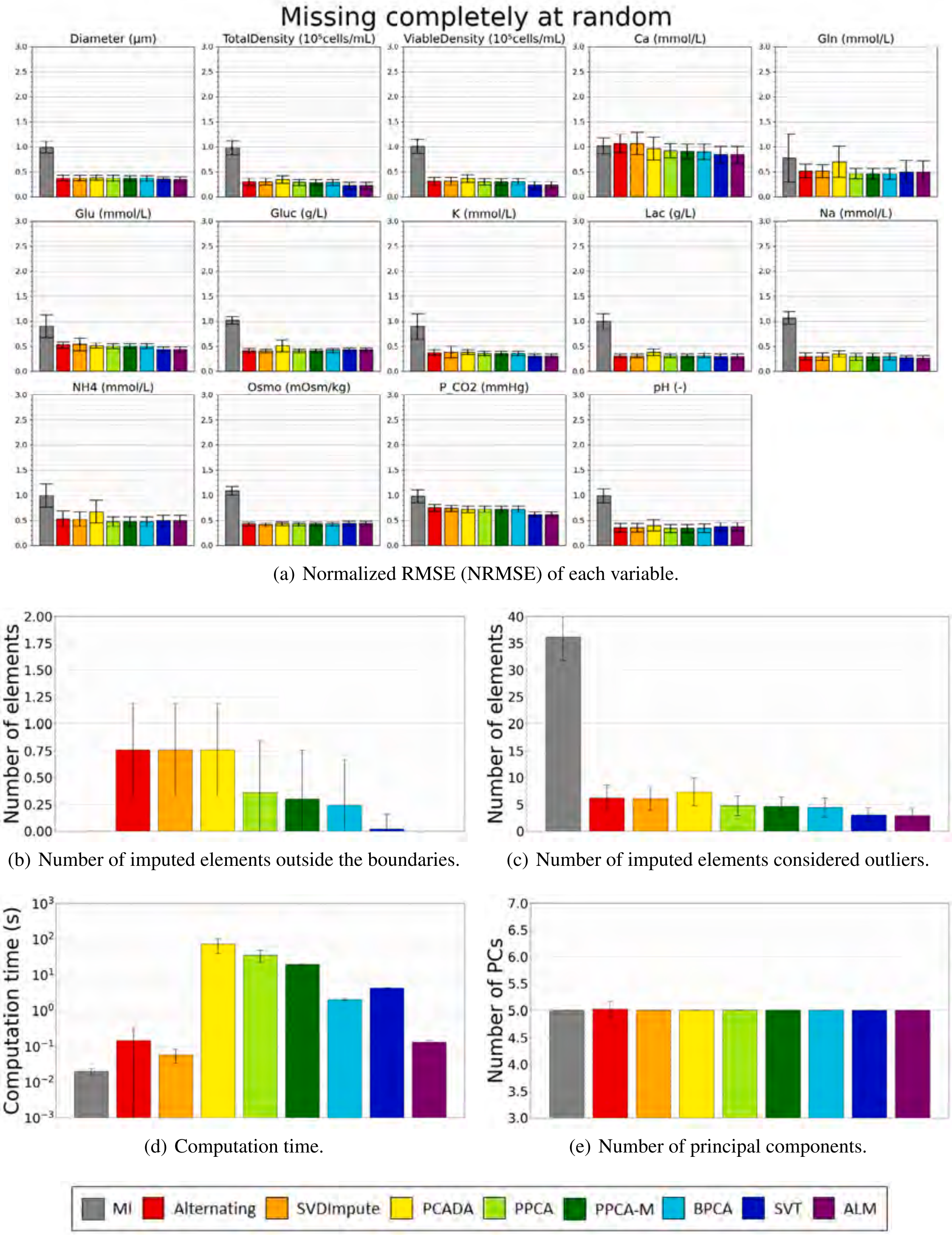
from being a bottleneck for updating the setpoint, the algorithm with a shorter computation time is recommended, especially in the case where the amounts of observations and variables are large. Severson et al. (2017) compared the computational costs of each algorithm used in this paper as shown in Table 7. The dataset used in this section possesses d of 14, n of 268, K of 50. As mentioned in Section 2.3.5, the number of principal components, a, might vary based on which imputation algorithm is used, which can be checked by comparing the number of principal components determined by each algorithm.

Figs. 10–14 display the bar graph of (a) variable-wise NRMSE, (b) number of imputed elements outside the boundaries (feasibility), (c) number of imputed elements considered outliers (plausibility), (d) computation time (rapidity), and (e) the number of principal components for each missingness type of MCAR, sensor drop-out, multi-rate, censoring, and patterned, respectively.

The MI algorithm works as a good reference for the NRMSE plot as the values are close to 1 for all missingness cases except the censoring case due to the way the NRMSE was defined in (55). Note that the standard deviation is the square root of the average of squared deviations from the mean. The performance of algorithms can be preliminarily assessed by checking whether the NRMSE values are smaller than 1 upon first inspection. In fact, all algorithms excluding MI showed

Table 7
The computational cost of each algorithm (Severson et al., 2017). d indicates the number of measurement variables, n indicates the number of observations, and K indicates the number of bootstrap used in PCADA. The dataset used in this section possesses d of 14, n of 268, K of 50.

ALS/Alternating/PPCA/BPCA	SVDImpute/SVT/ALM	PCADA	PPCA-M
$O(na^3 + nda^2 + da^3)$	$O(\min(nd^2, n^2d))$	$O(\min(Knd^2, Kn^2d))$	$O(na^3 + nda^2)$



(a) Normalized RMSE (NRMSE) of each variable.

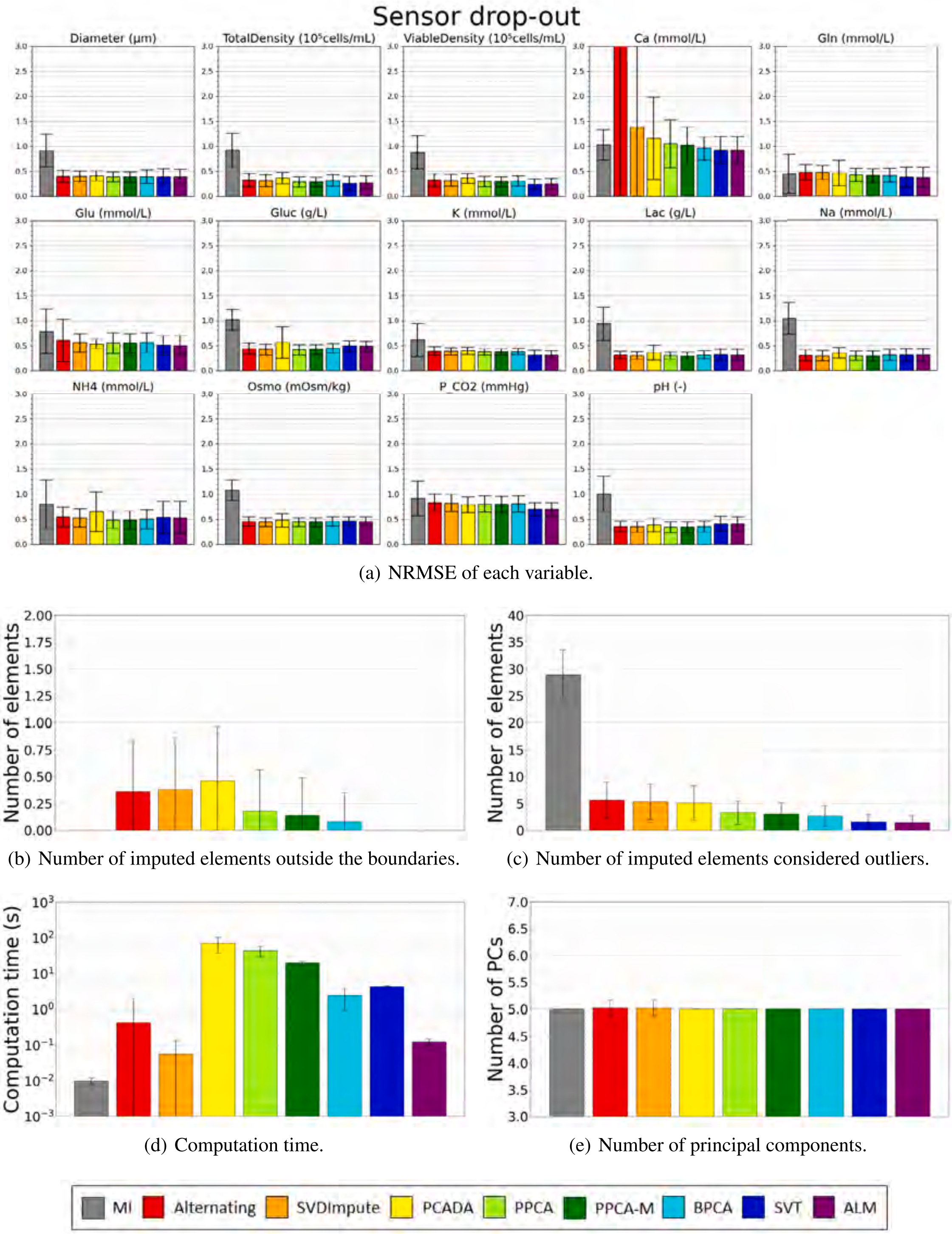
(b) Number of imputed elements outside the boundaries.

(c) Number of imputed elements considered outliers.

(d) Computation time.

(e) Number of principal components.

Fig. 10. Evaluation of imputation algorithms in the missing completely at random (MCAR) case for the continuous biomanufacturing dataset. The bar and error bar indicates the mean and the standard deviation, respectively, of the measured values in 50 simulations. The upper limit for the NRMSE graph in (a) was set to 3.0 as the values higher than that imply that the imputed values are generally outside the 3 sigma range, indicating poor performance.



(a) NRMSE of each variable.

(b) Number of imputed elements outside the boundaries.

(c) Number of imputed elements considered outliers.

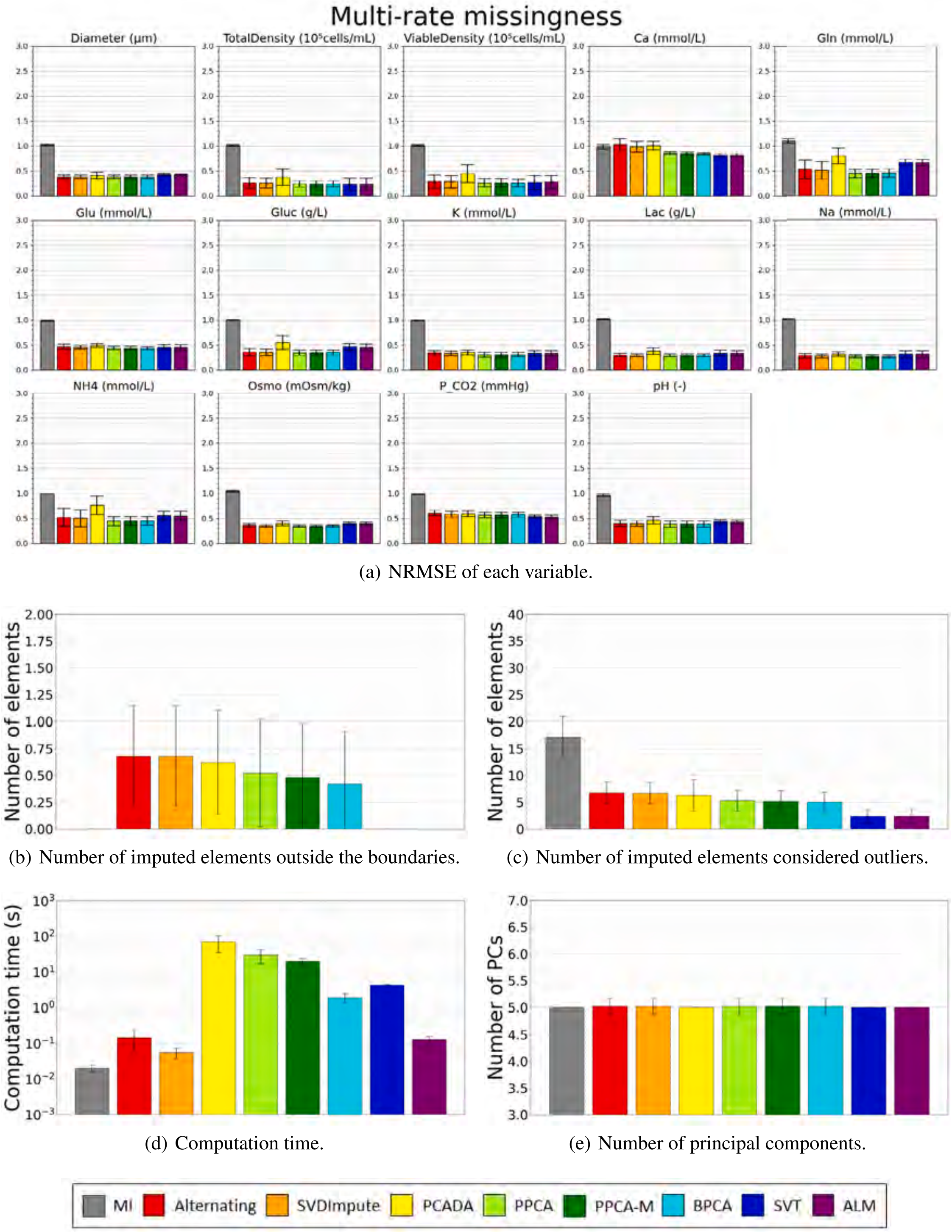
(d) Computation time.

(e) Number of principal components.

Fig. 11. Evaluation of imputation algorithms in the sensor drop-out case for the continuous biomanufacturing dataset. The bar and error bar indicates the mean and the standard deviation, respectively, of the measured values in 50 simulations. The upper limit for the NRMSE graph in (a) was set to 3.0 as the values higher than that imply that the imputed values are generally outside the 3 sigma range, indicating poor performance.

NRMSE values smaller than 1 in most variables in Fig. 10, implying that those algorithms at least perform better than the MI algorithm in the MCAR case. For the elements with weak dynamics (i.e. having no clear trend as a function of time) such as Calcium concentration, NRMSE tends to have higher values as it shows less dependency on other elements.

According to Figs. 10–12 and 14, all algorithms excluding the Alternating algorithm maintained similar order of magnitude of NRMSE values in different missingness cases, implying that the performance does not significantly depend on the missingness type for those algorithms. The Alternating algorithm shows NRMSE values larger than 3 in sensor drop-out and patterned missingness cases, indicating poor performance. Having NRMSE values larger than 3 means that the



(a) NRMSE of each variable.

(b) Number of imputed elements outside the boundaries.

(c) Number of imputed elements considered outliers.

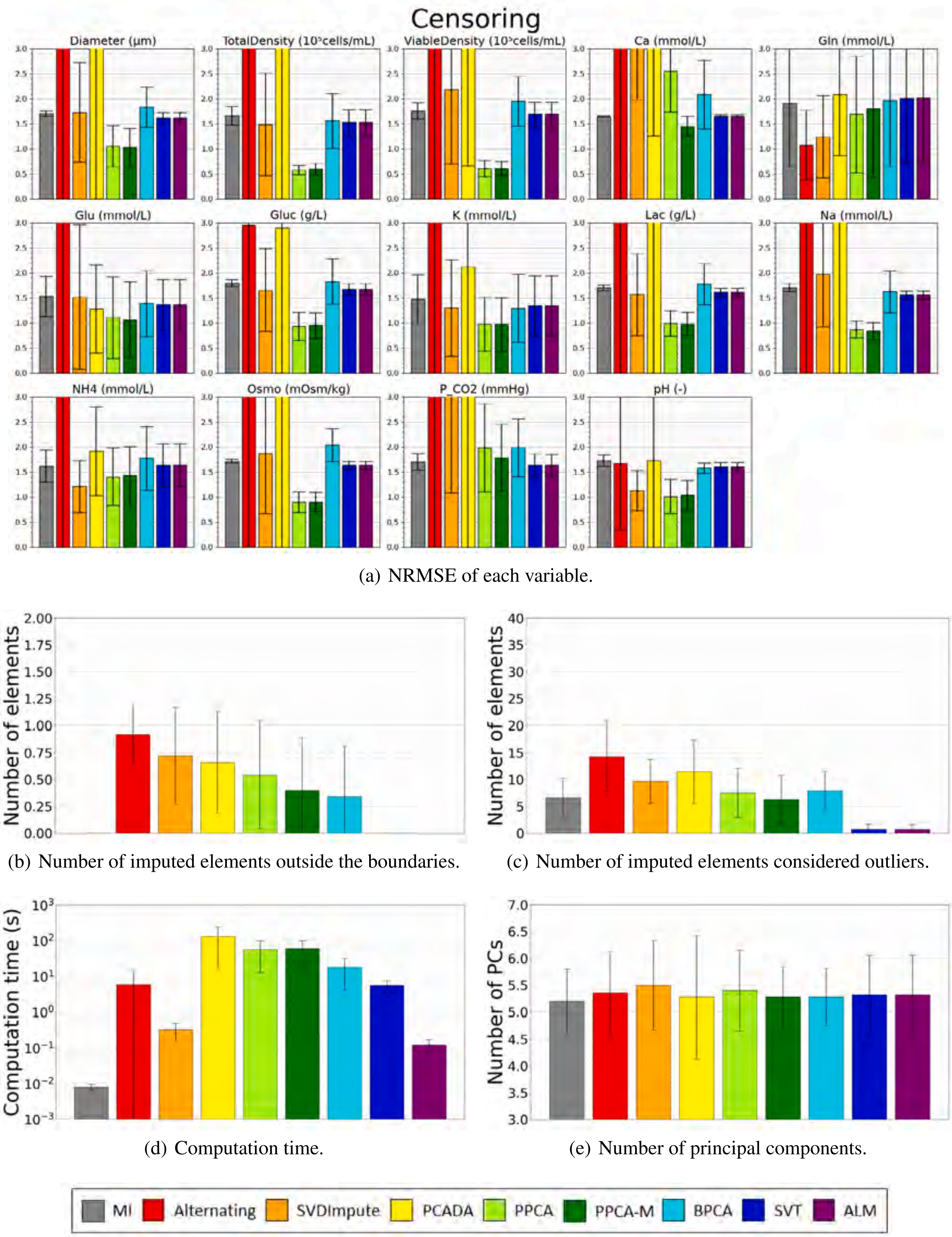
(d) Computation time.

(e) Number of principal components.

Fig. 12. Evaluation of imputation algorithms in the multi-rate missingness case for the continuous biomanufacturing dataset. The bar and error bar indicates the mean and the standard deviation, respectively, of the measured values in 50 simulations. The upper limit for the NRMSE graph in (a) was set to 3.0 as the values higher than that imply that the imputed values are generally outside the 3 sigma range, indicating poor performance.

imputed values are more than 3 sigmas away from the true values, implying that the algorithm is not worth trying.

From Fig. 13, it can be seen that most algorithms did not work well in the censoring case, yielding the highest NMRSE values among all missingness cases. This might be due to the nature of having a worse imputation accuracy in the extrapolation case than in the interpolation case. In the censoring case, the values outside the boundaries have to be estimated using only the values inside the boundaries, which significantly complicates an accurate imputation. In addition to the nature of extrapolation, having a large chunk of missing values make the data



(a) NRMSE of each variable.

(b) Number of imputed elements outside the boundaries.

(c) Number of imputed elements considered outliers.

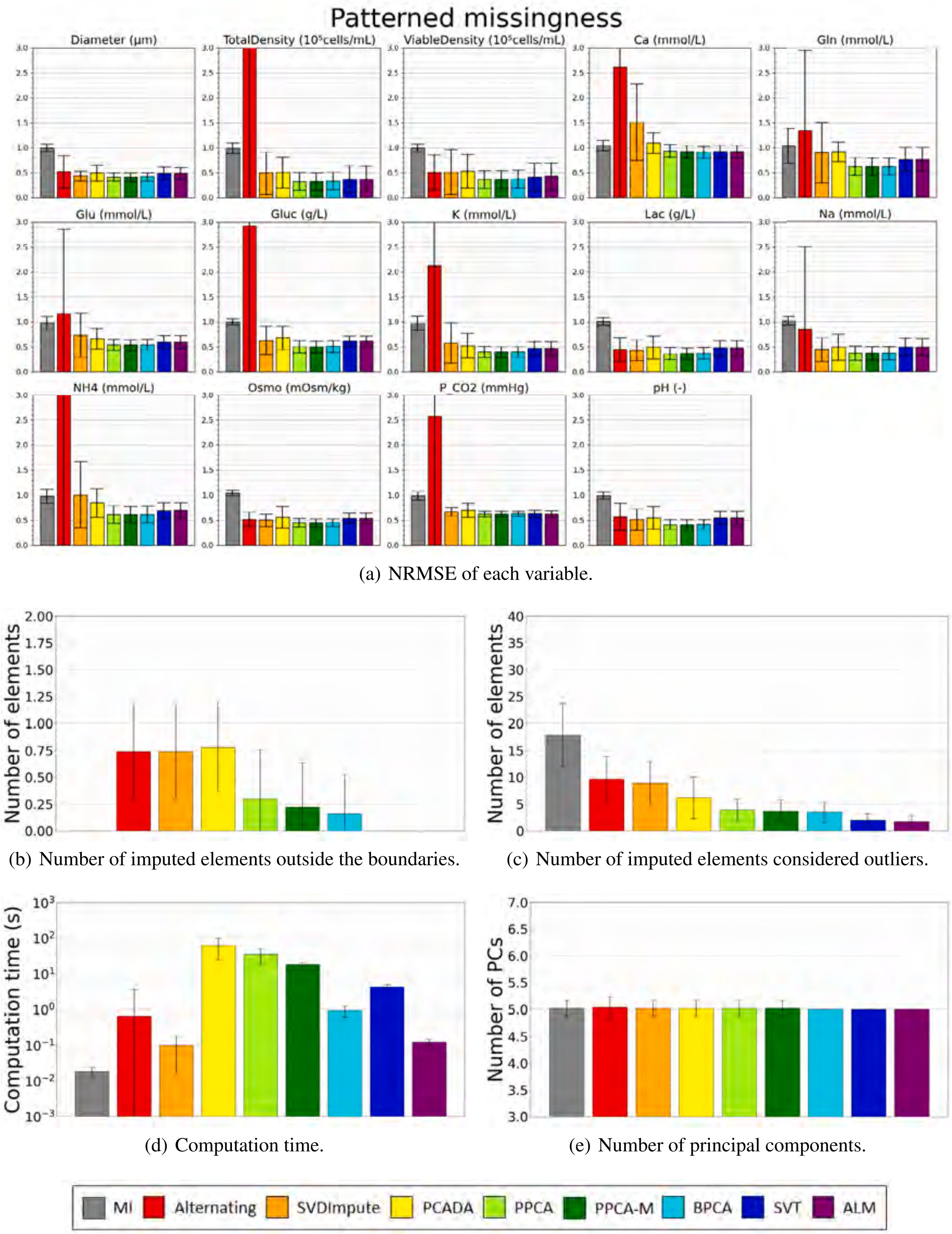
(d) Computation time.

(e) Number of principal components.

Fig. 13. Evaluation of imputation algorithms in the censoring case for the continuous biomanufacturing dataset. The bar and error bar indicates the mean and the standard deviation, respectively, of the measured values in 50 simulations. The upper limit for the NRMSE graph in (a) was set to 3.0 as the values higher than that imply that the imputed values are generally outside the 3 sigma range, indicating poor performance.

imputation more difficult, especially for the dynamic process data as the relation between the missed variables and observed variables within the time period is unknown. Comparing the NRMSE values in Fig. 13, PPCA and PPCA-M were the only algorithms to have better performance than MI.

Missingness type can be determined from the arrangement of missing values in the indication matrix. For example, the biomanufacturing dataset used in this article seems to possess a combination of patterned missingness and censoring. Based on the validation result, PPCA and PPCA-M would be recommended for filling in missing values in this



(a) NRMSE of each variable.

(b) Number of imputed elements outside the boundaries.

(c) Number of imputed elements considered outliers.

(d) Computation time.

(e) Number of principal components.

Fig. 14. Evaluation of imputation algorithms in the patterned missingness case for the continuous biomanufacturing dataset. The bar and error bar indicates the mean and the standard deviation, respectively, of the measured values in 50 simulations. The upper limit for the NRMSE graph in (a) was set to 3.0 as the values higher than that imply that the imputed values are generally outside the 3 sigma range, indicating poor performance.

case. The following are the overall reports of each algorithm for five missingness scenarios in the biomanufacturing dataset.

MI, which works as the reference method, has the shortest computation time as there is no further calculation for data imputation. While MI has no imputed elements outside the boundaries, it typically results in the largest number of imputed elements considered outliers as the dynamics and the latent relations among variables are not considered at all.

Alternating showed a smaller computation time by an order of 1–2 compared to PPCA-based methods. Alternating worked pretty well in

MCAR and multi-rate cases, but there was at least one variable with an average NRMSE larger than 3 in other missingness types, which implies a poor imputation. It also has the largest number of imputed elements outside the boundaries, which is probably due to failure to converge before the maximum iteration number. SVDImpute had the lowest computation time while maintaining a decent performance among PCA-based methods. PCADA had the largest computation time among all algorithms due to K bootstrap resampling. Larger K should be used as the sample size increases to ensure the representativeness of the bootstrap resampling sets, which will result in a longer computation time.

While PPCA and PPCA-M algorithms showed similar NRMSE values in all missingness scenarios, PPCA-M had roughly half of the computation time as PPCA. These two algorithms showed the smallest NRMSE in pH, especially in censoring and patterned cases. BPCA had a shorter computation time than PPCA-M, which was around one order of magnitude less. BPCA had a similar performance as PPCA and PPCA-M in most missingness cases except for the censoring case.

SVT and ALM, which are the matrix completion methods, had the smallest number of imputed elements outside the boundaries and those considered outliers in every missingness scenario. ALM shows around one order of magnitude smaller computation time than SVT.

5. Conclusion

The missing values in process data, which typically happen due to several reasons such as sensor dropout, having variables being measured at different rates, and censoring, should be filled in for process modeling, analysis, and control. However, the presence of outliers inhibits capturing appropriate latent relationships among variables, which might prevent accurate data imputation. To deal with this problem, a software that automatically detects outliers in an iterative way and fills in missing values using various general-purpose algorithms was proposed in this article. In Section 2, the methods for detecting outliers using T^2 and Q statistic contributions and estimating missing values using 9 algorithms (MI, Alternating, SVDImptue, PCADA, PPCA, PPCA-M, BPCA, SVT, and ALM) were explained in detail. Each algorithm was evaluated by three criteria which are in order of priority: feasibility, plausibility, and rapidity. The demonstration of this software was performed in Section 3. The validation of this software was performed in Section 4 using the complete dataset, which is a subset of the given dataset. Five types of missingness (MCAR, sensor drop-out, multi-rate, censoring, and patterned) were added to the complete dataset using the software introduced by Severson et al. (2017). Based on the NRMSE values as well as the three criteria, SVT and ALM performed the best in all missingness scenarios except for the censoring case where PPCA and PPCA-M performed better. As the given biomanufacturing dataset had a combination of patterned missingness and censoring based on the indication matrix, PPCA and PPCA-M were recommended from the software. The future plan for extending this study is to check whether the software could be applied to non-time-series datasets and time-series datasets with a number of measurement variables (d) much larger than the number of observations (n).

CRediT authorship contribution statement

Jinwook Rhyu: Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Dragana Bozinovski:** Data curation, Investigation, Writing – review & editing. **Alexis B. Dubs:** Data curation, Investigation, Writing – original draft, Writing – review & editing. **Naresh Mohan:** Data curation, Investigation. **Elizabeth M. Cummings Bende:** Data curation, Investigation. **Andrew J. Maloney:** Data curation, Investigation, Writing – review & editing. **Miriam Nieves:** Data curation, Investigation. **Jose Sangerman:** Data curation, Investigation. **Amos E. Lu:** Data curation, Investigation, Writing – review & editing. **Moo Sun Hong:** Data curation, Investigation, Writing – review & editing. **Anastasia Artamonova:** Data curation, Investigation. **Rui Wen Ou:** Data curation, Investigation. **Paul W. Barone:** Data curation, Investigation, Writing – review & editing. **James C. Leung:** Data curation, Formal analysis, Investigation, Writing – review & editing. **Jacqueline M. Wolfrum:** Data curation, Formal analysis, Investigation, Writing – review & editing. **Anthony J. Sinskey:** Data curation, Funding acquisition, Investigation, Writing – review & editing. **Stacy L. Springs:** Data curation, Funding acquisition, Investigation, Writing – review & editing. **Richard D. Braatz:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data and code availability

The data and code are available on the corresponding GitHub repository: https://github.com/JinwookRhyu/Automated-Outlier-Detection-and-Estimation-of-Missing-Data. The first dataset "mAb_dataset_demonstration.xlsx" contains the dataset used in the Demonstration part (Section 3) with 298 observations and 14 variables. Note that the last five observations were not used as those points are after the process termination, resulting in a total of 293 observations used in this article. The second dataset "mAb_dataset_validation.xlsx" used in the Validation part (Section 4) is the preprocessed version of the first dataset after Step A (Section 2.2), containing 268 observations with 14 variables. The code comes with an open-source MIT license.

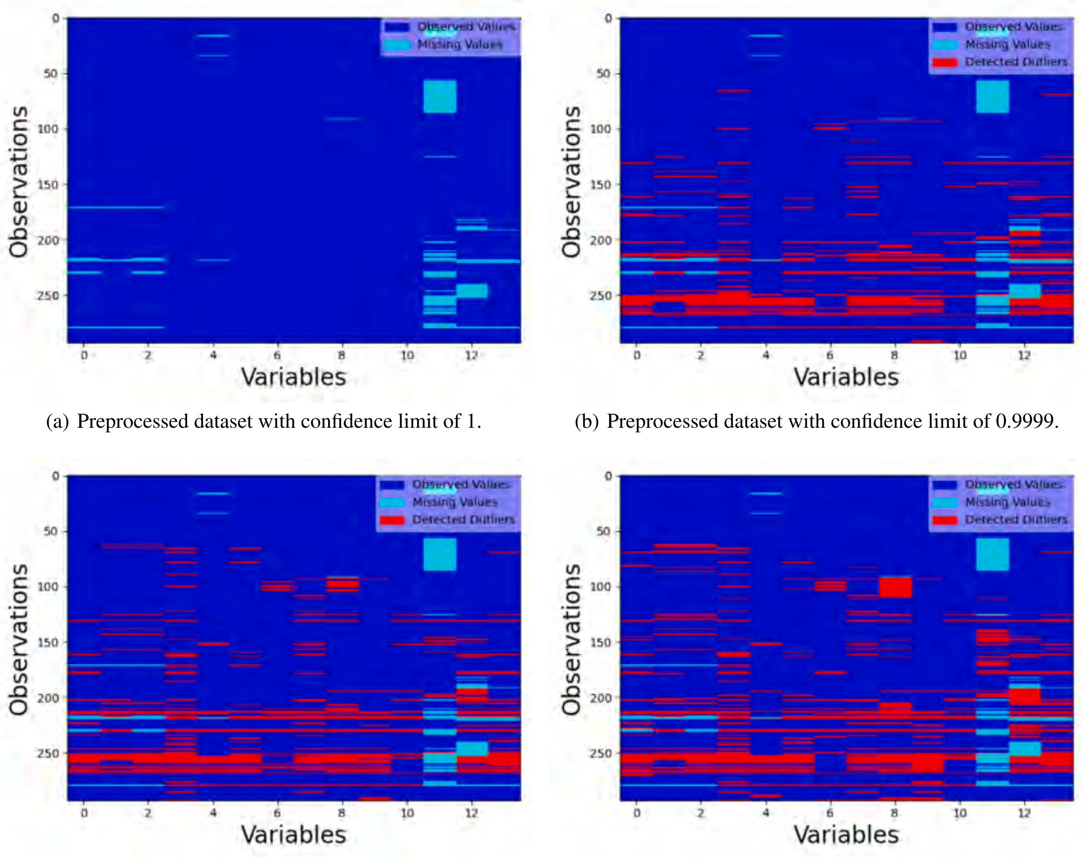
Acknowledgments

This study is supported by the U.S. Food and Drug Administration, Contract No. 75F40121C00090. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the financial sponsor. MIT thanks Sartorius Stedim Cellca GMBH for the generous support of the adalimumab-producing CHO cell line.

Appendix A. Confidence limit for outlier detection

The confidence limit plays an important role in determining whether the value should be considered an outlier. In this article, T^2 and Q contributions are assumed to follow a normal distribution, and the values higher than the threshold based on the confidence limit are considered outliers. For example, indices marked as red in Fig. A.1b imply that they have T^2 or Q contribution higher than 99.99th percentile in the normal distribution of column-wise T^2 or Q contributions. Decreasing the confidence limit results in an increase in the chance of exceeding the threshold (i.e., being caught as outliers). This trend is demonstrated in Fig. A.1 where the number of detected outliers increases to 0, 347, 422, and 527 as the confidence limit decreases to 1, 0.9999, 0.999, and 0.99. Fig. A.1a is an extreme case of completely trusting the measurements so that there is no single outlier in the given dataset.

Assuming a Gaussian distribution, setting the confidence limit to $1 - 1/n$ would cause roughly one measurement in each variable to be considered an outlier. In order to avoid false-positive in detecting outliers, it is recommended to select the confidence limit in a more conservative manner (i.e., value close to 1). In this article, the confidence limit was set to 0.9999, which is higher than $1 - 1/293 \approx 0.9966$.



(a) Preprocessed dataset with confidence limit of 1.

(b) Preprocessed dataset with confidence limit of 0.9999.

(c) Preprocessed dataset with confidence limit of 0.999.

(d) Preprocessed dataset with confidence limit of 0.99.

Fig. A.1. Outlier detection results using different confidence limit values: (a) 1, (b) 0.9999, (c) 0.999, and (d) 0.99. Observed values, missing values, and detected outliers are indicated in blue, light blue, and red, respectively. The number of detected outliers is 0, 347, 422, and 527, respectively.

The optimal confidence limit could be determined based on n and domain knowledge as different criteria may apply to outlier detection in different process fields.

Appendix B. Threshold for the number of survived elements

In this section, different values of the threshold for determining low-quality observations were tested to explore the tradeoff between the quantity and the quality of a dataset. Fig. B.1 demonstrates the results of row elimination with thresholds of (a) 1, (b) 3, (c) 5, and (d) 11, respectively. Setting a loose threshold for determining low-quality observations ensures the quantity of a given dataset while there exists a high proportion of outliers. For example, Fig. B.1a and Fig. B.1b show that most observations around days 20–25 are not eliminated in Fig. 5, which misleads the latent relationship among variables. On the other hand, a tight threshold ensures the quality of a given dataset although it also results in a larger data loss. In the extreme case where the threshold is equal to the number of variables, only the observations with the full measurement that do not contain any outliers will be survived.

In this article, we set the threshold of row elimination as the number of principal components of the preprocessed dataset. This criterion ensures that the number of measured variables in every observation is larger or equal to the number of principal components, which satisfies the main objective of using PCA-based methods: reducing dimensionality. However, the optimal threshold for row elimination may be determined using domain knowledge as the relationship between the quality and the quantity may depend on the type of process system.

Appendix C. Temporary imputation using mean imputation and last observed values

See Fig. C.1.

Appendix D. Results at each missingness scenario using each algorithm

D.1. Missing completely at random (MCAR)

See Fig. D.1.

D.2. Sensor drop-out

See Fig. D.2.

D.3. Multi-rate missingness

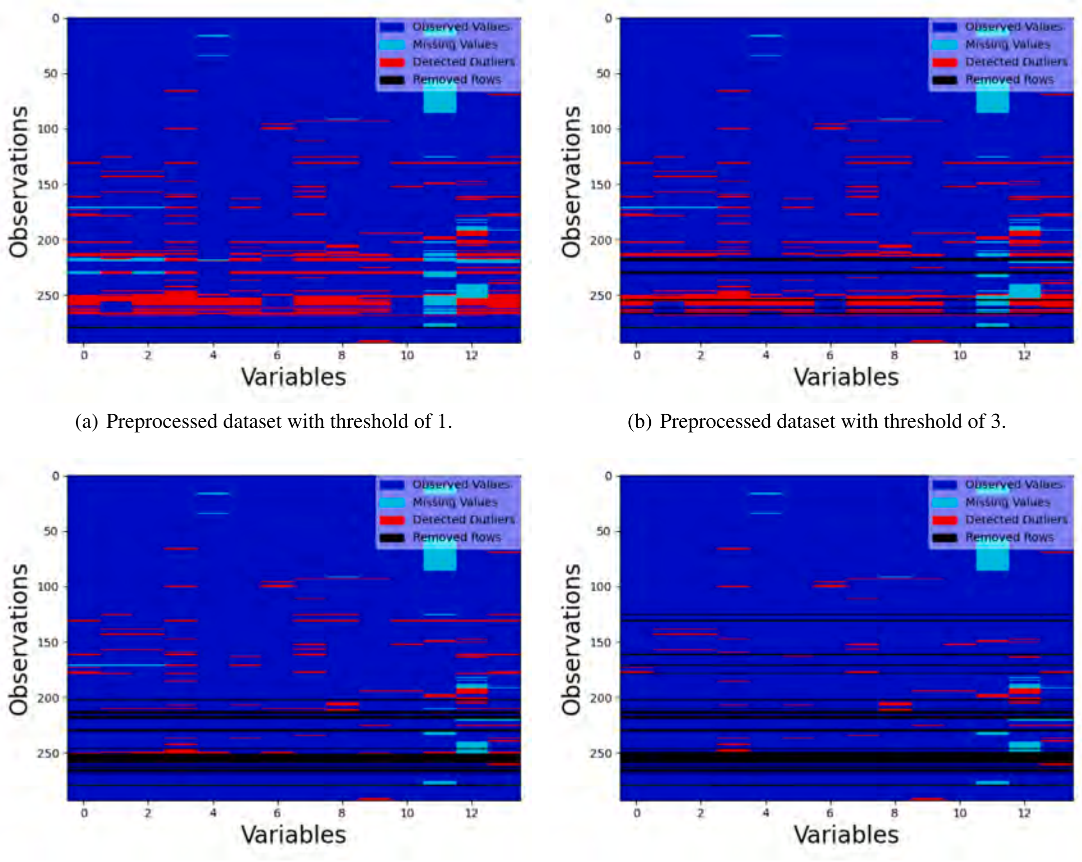
See Fig. D.3.

D.4. Censoring

See Fig. D.4.

D.5. Patterned missingness

See Fig. D.5.



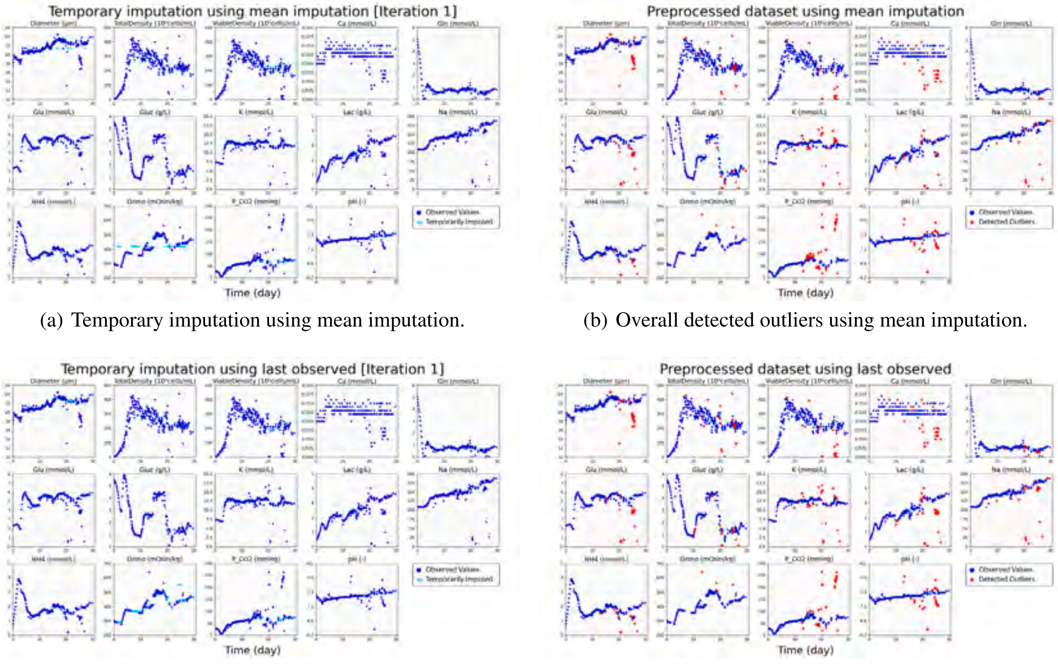
(a) Preprocessed dataset with threshold of 1.

(b) Preprocessed dataset with threshold of 3.

(c) Preprocessed dataset with threshold of 5.

(d) Preprocessed dataset with threshold of 11.

Fig. B.1. Results of row elimination with different thresholds for the number of survived elements: (a) 1, (b) 3, (c) 5, and (d) 11. Observed values, missing values, detected outliers, and removed rows are indicated in blue, light blue, red, and black, respectively. The threshold of T^2 and Q contributions for outlier detection was set to 0.9999 and the interpolation was used for the temporary imputation method. The number of removed rows is 1, 9, 25, and 33, respectively.



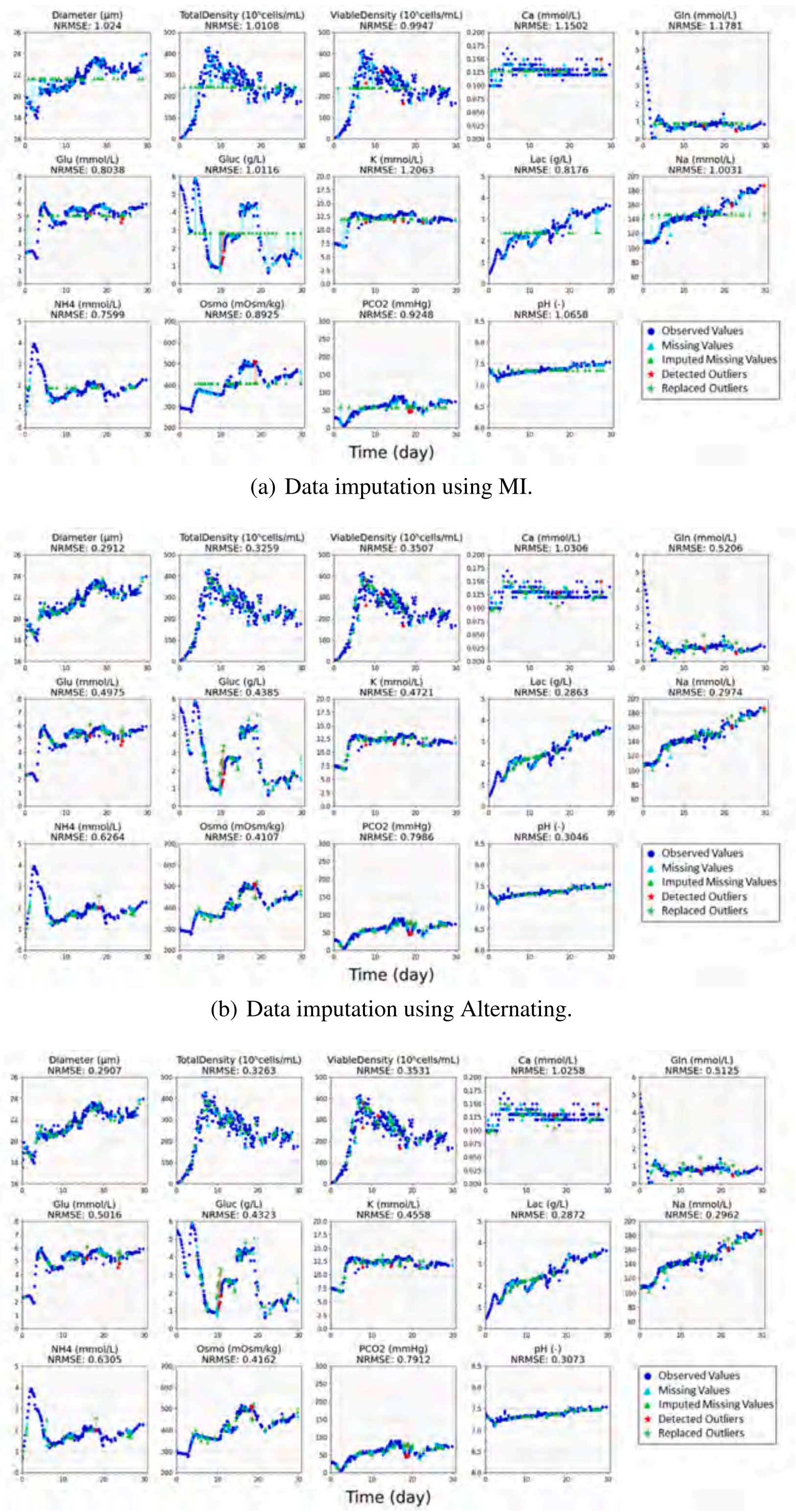
(a) Temporary imputation using mean imputation.

(b) Overall detected outliers using mean imputation.

(c) Temporary imputation using last observed values.

(d) Overall detected outliers using last observed values.

Fig. C.1. Demonstration of Step A-1 using (a) mean imputation and (c) last observed value. The final result of Step A after (b) 12 iterations of Step A-1 using mean imputation and Step A-2, and (d) 11 iterations of Step A-1 using last observed value and Step A-2. Blue dots indicate the observed values, light blue dots indicate the temporarily imputed values using interpolation, and red dots indicate the detected outliers.

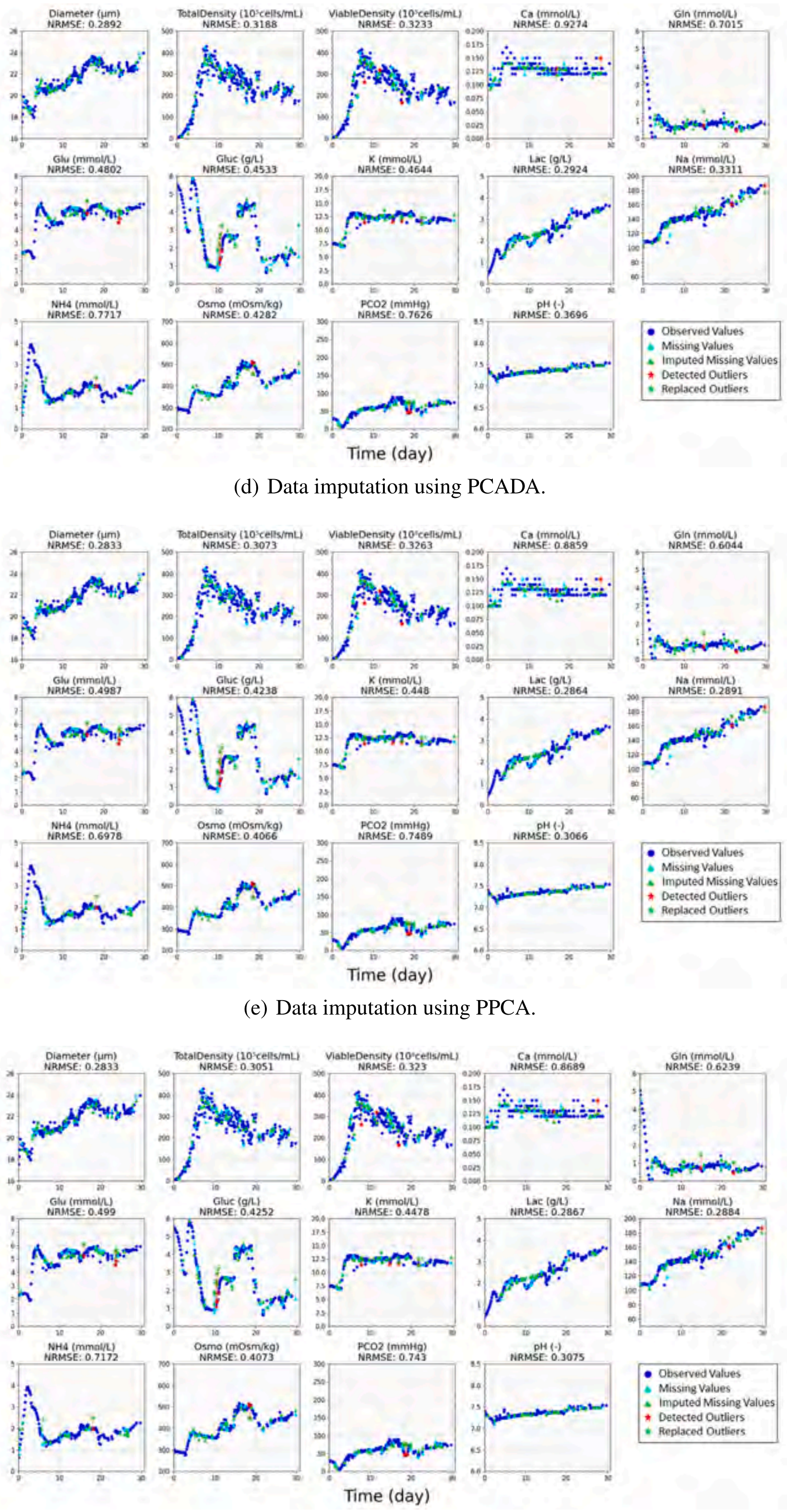


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. D.1. Data imputation results using 9 algorithms excluding ALS in the missing completely at random (MCAR) case.

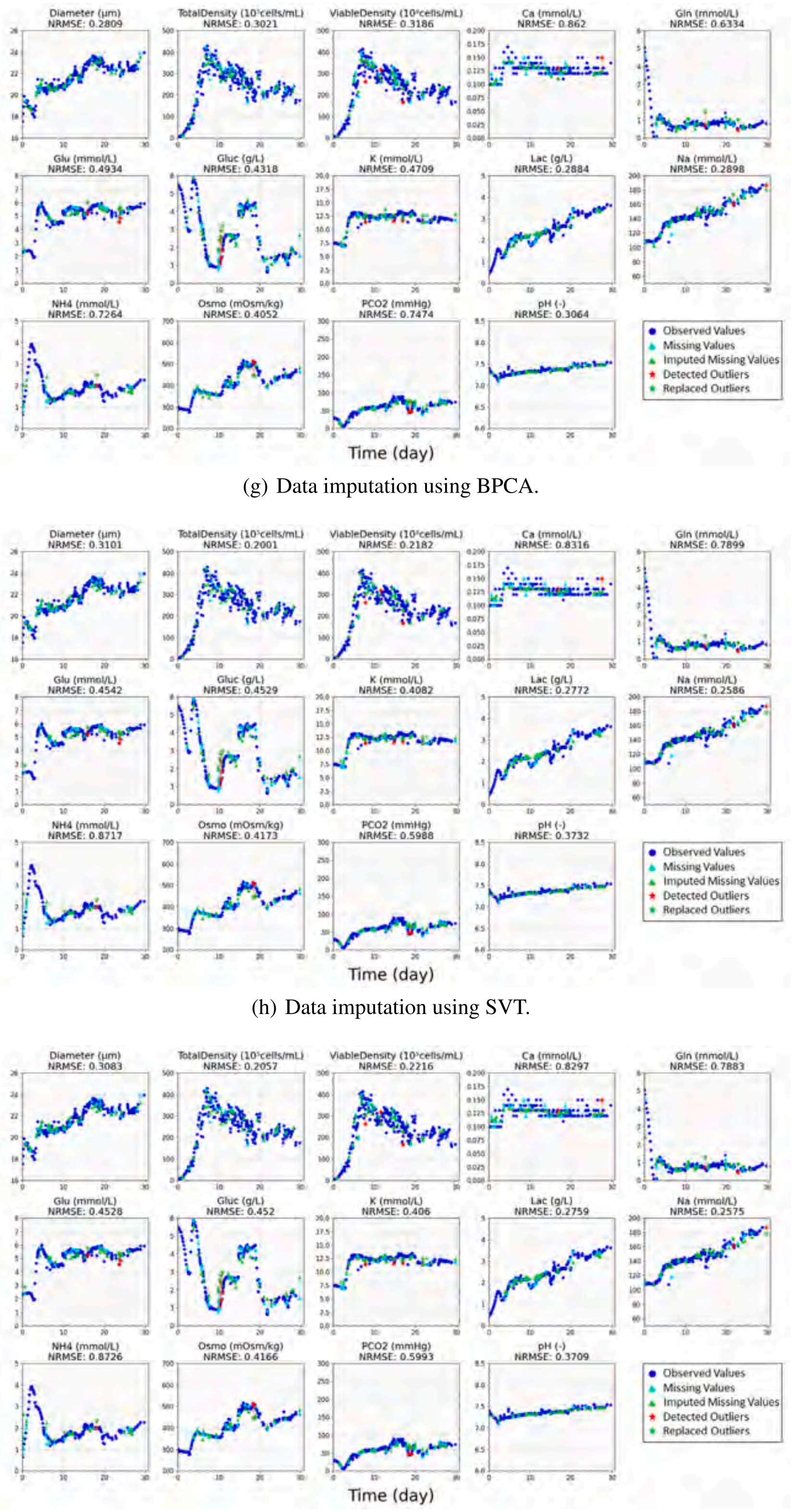


(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. D.1. (*continued*).

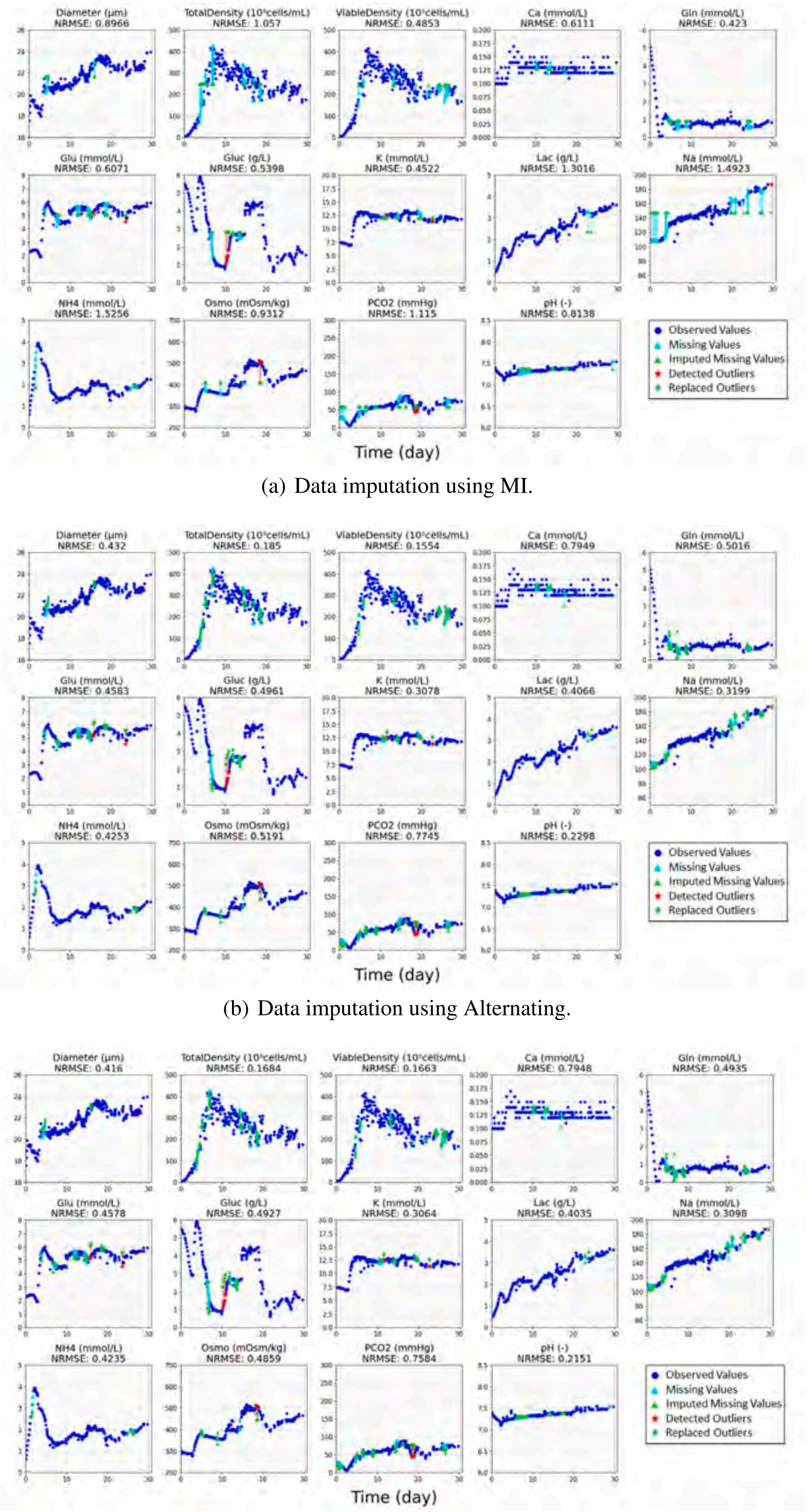


(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. D.1. (*continued*).

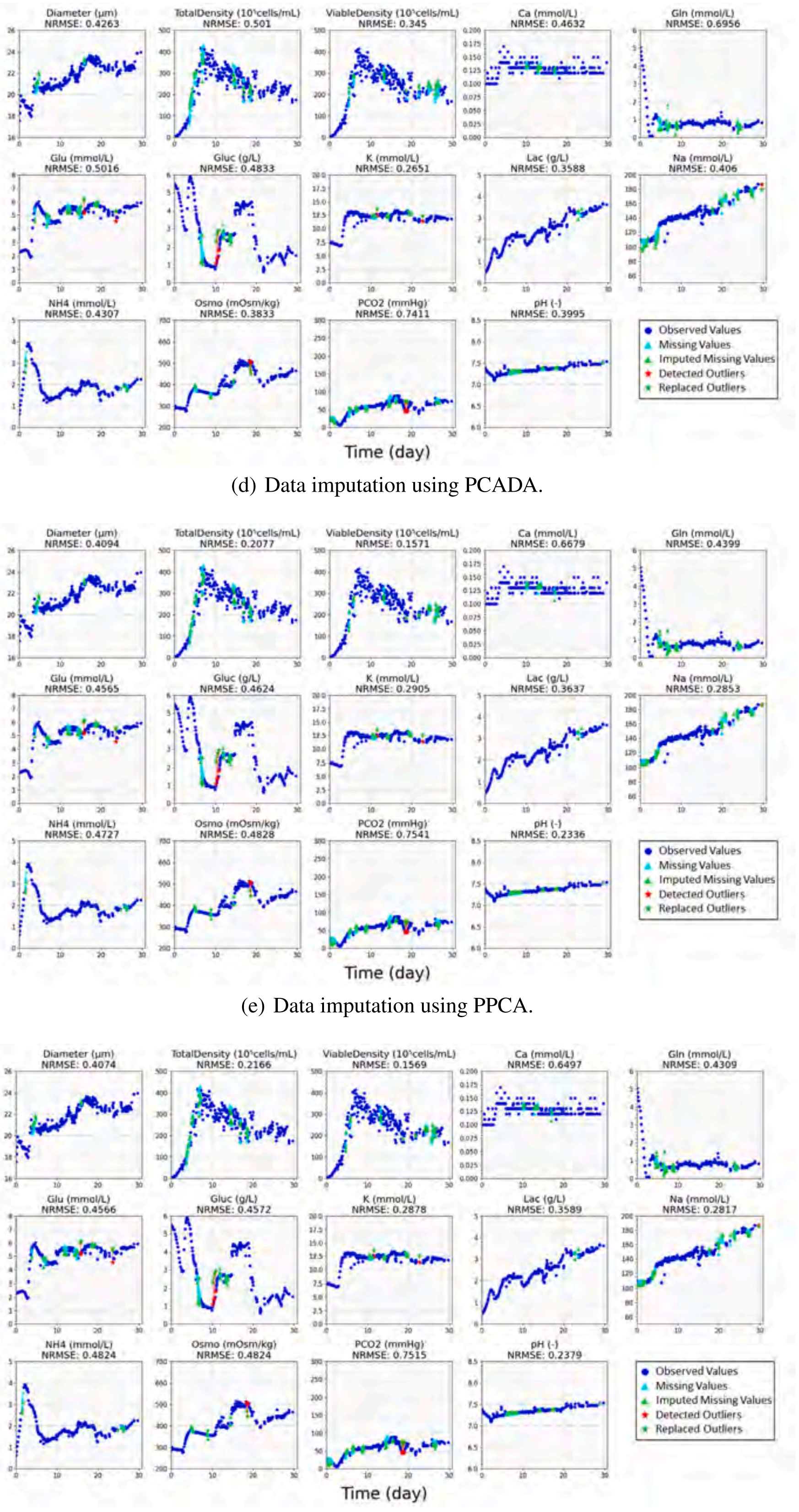


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. D.2. Data imputation results using 9 algorithms excluding ALS in the sensor drop-out case.

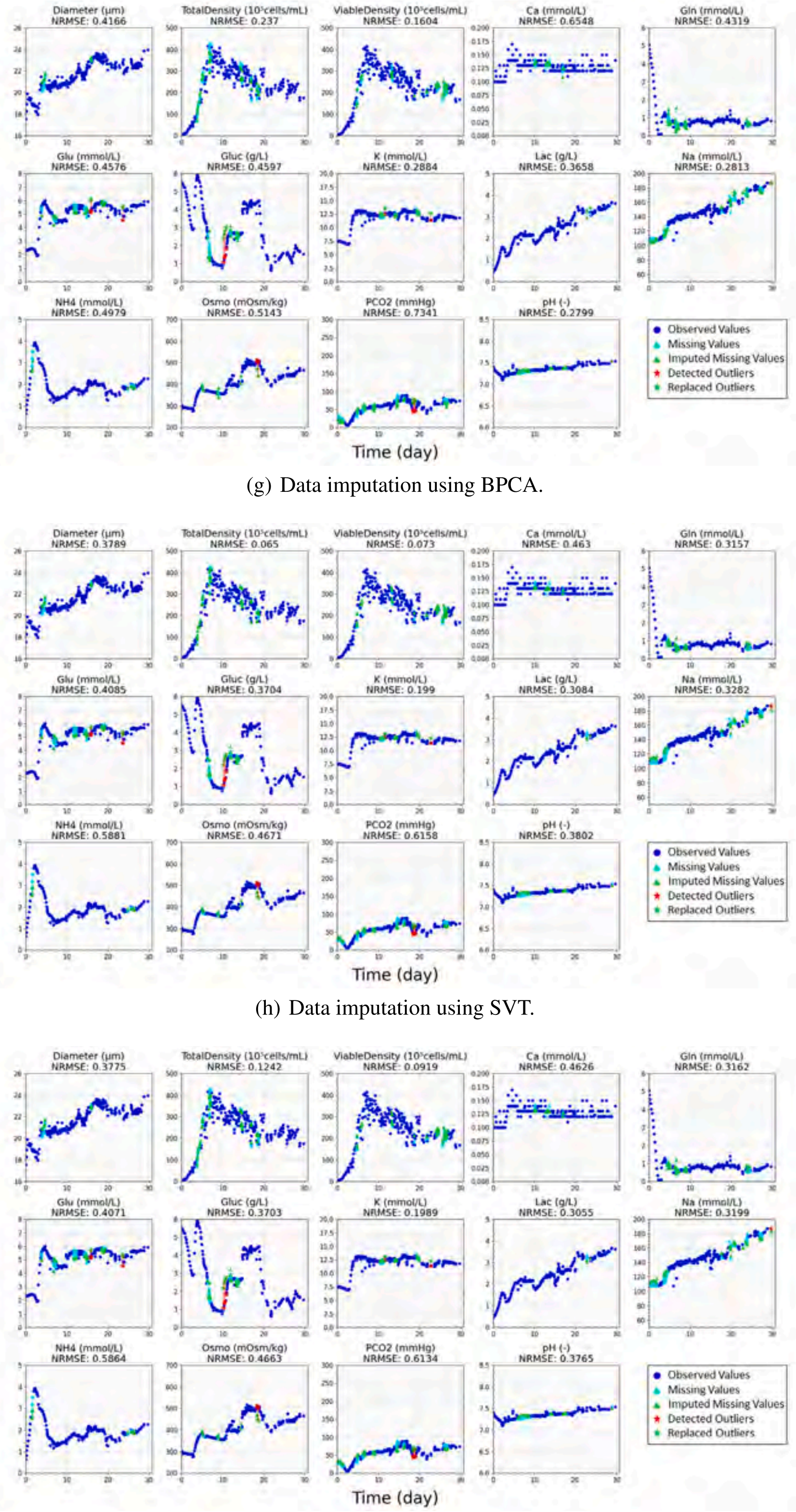


(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. D.2. (*continued*).

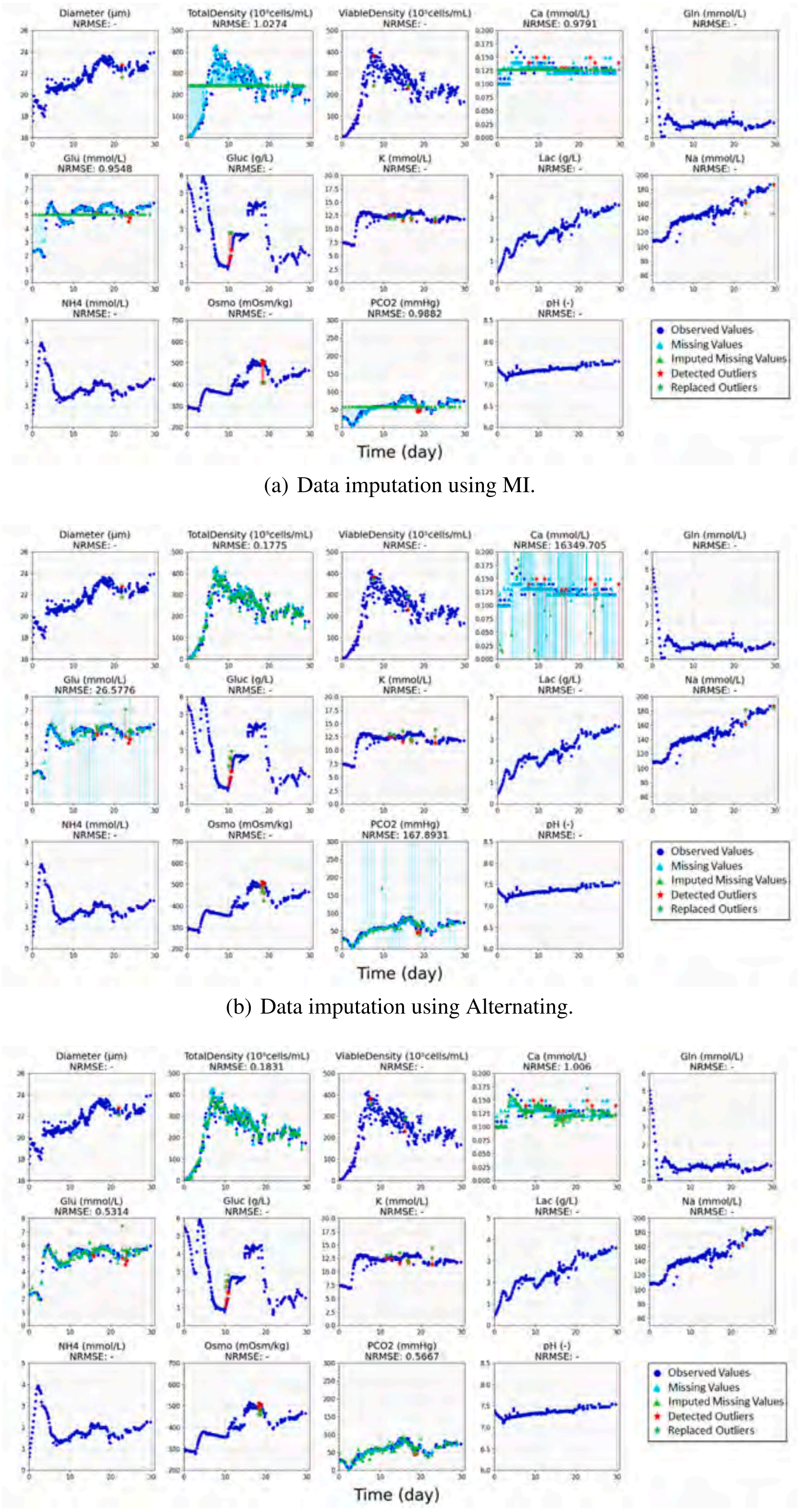


(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. D.2. (*continued*).

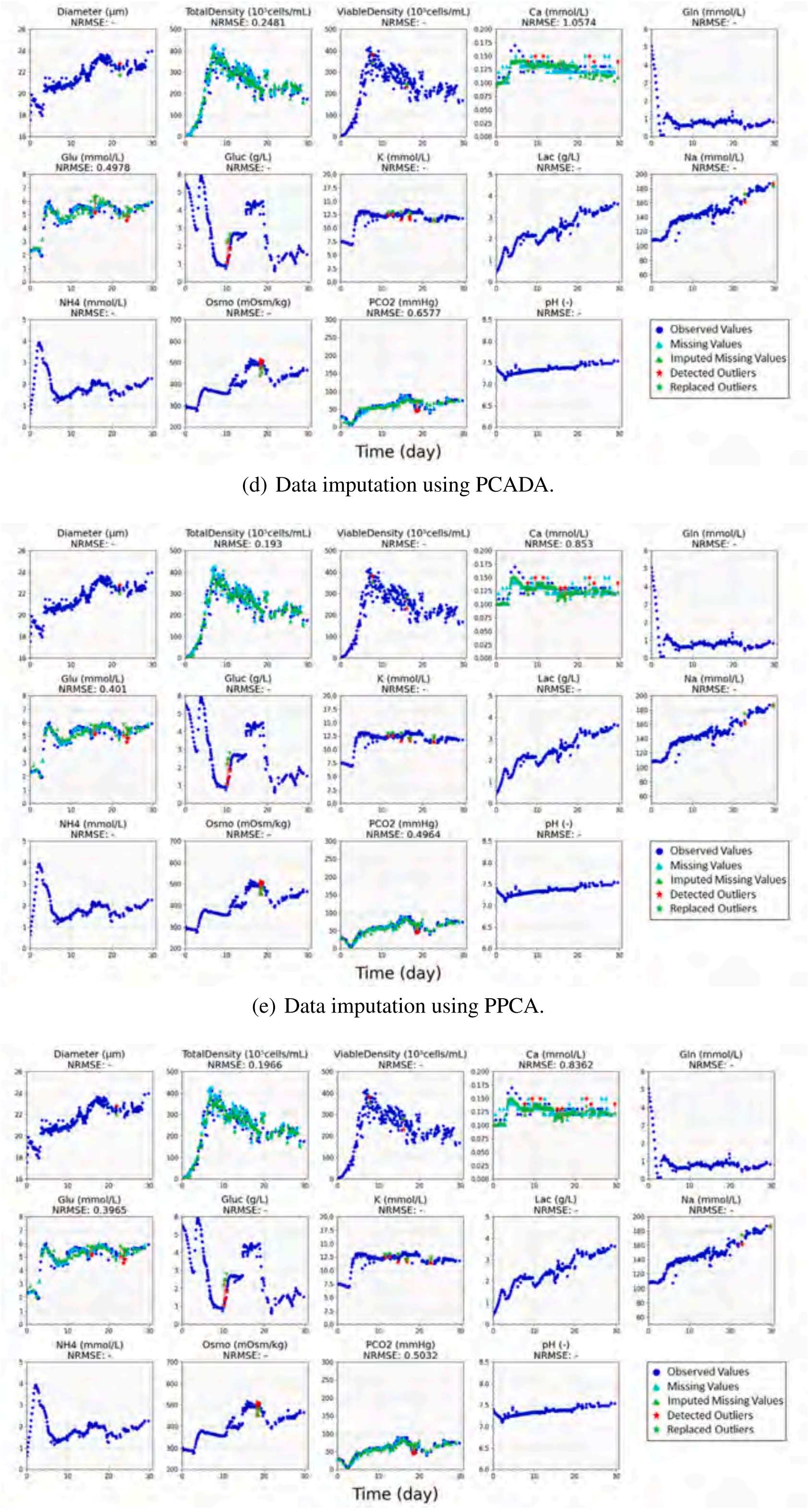


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. D.3. Data imputation results using 9 algorithms excluding ALS in the multi-rate missingness case.

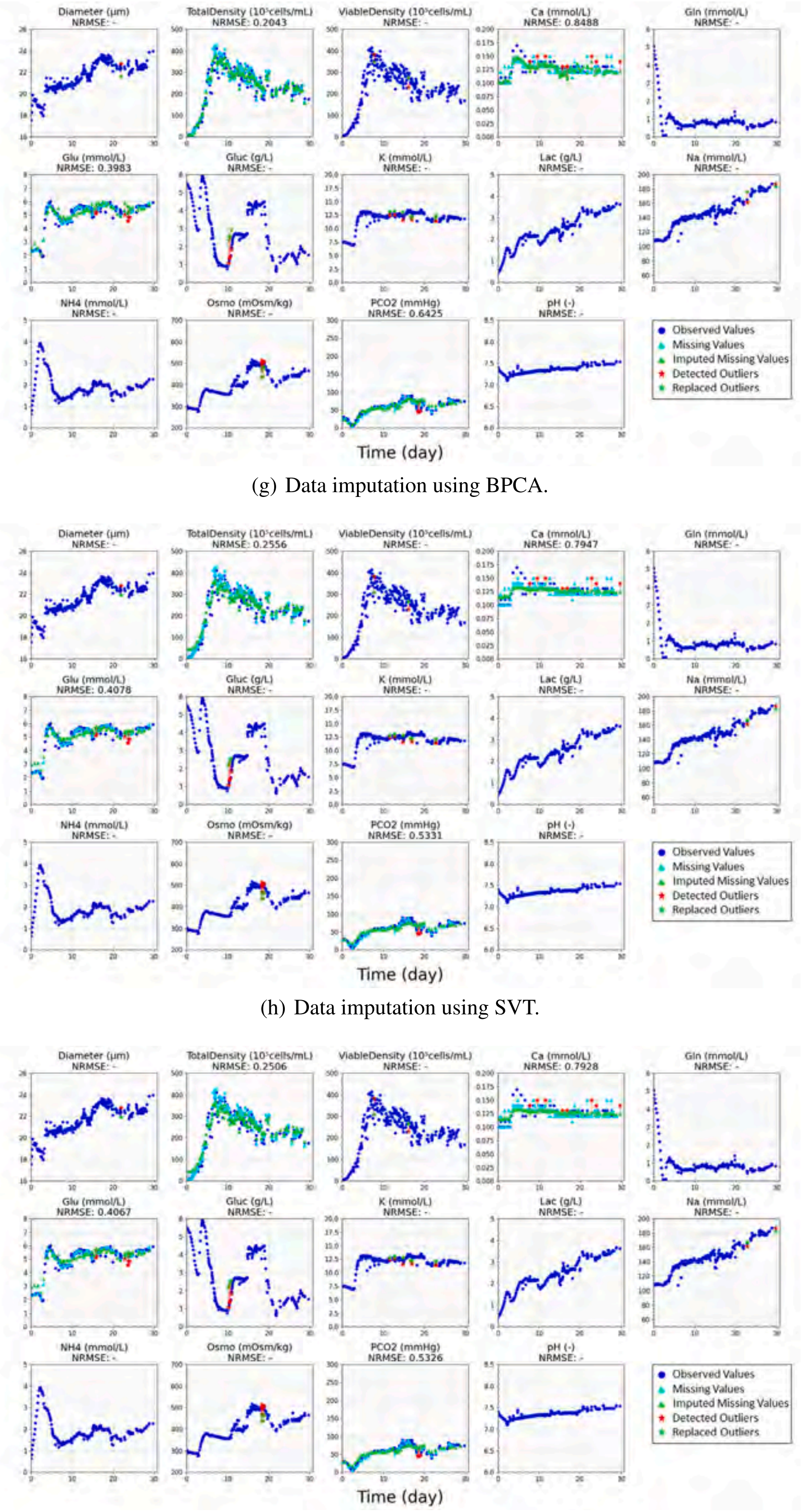


(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. D.3. (*continued*).

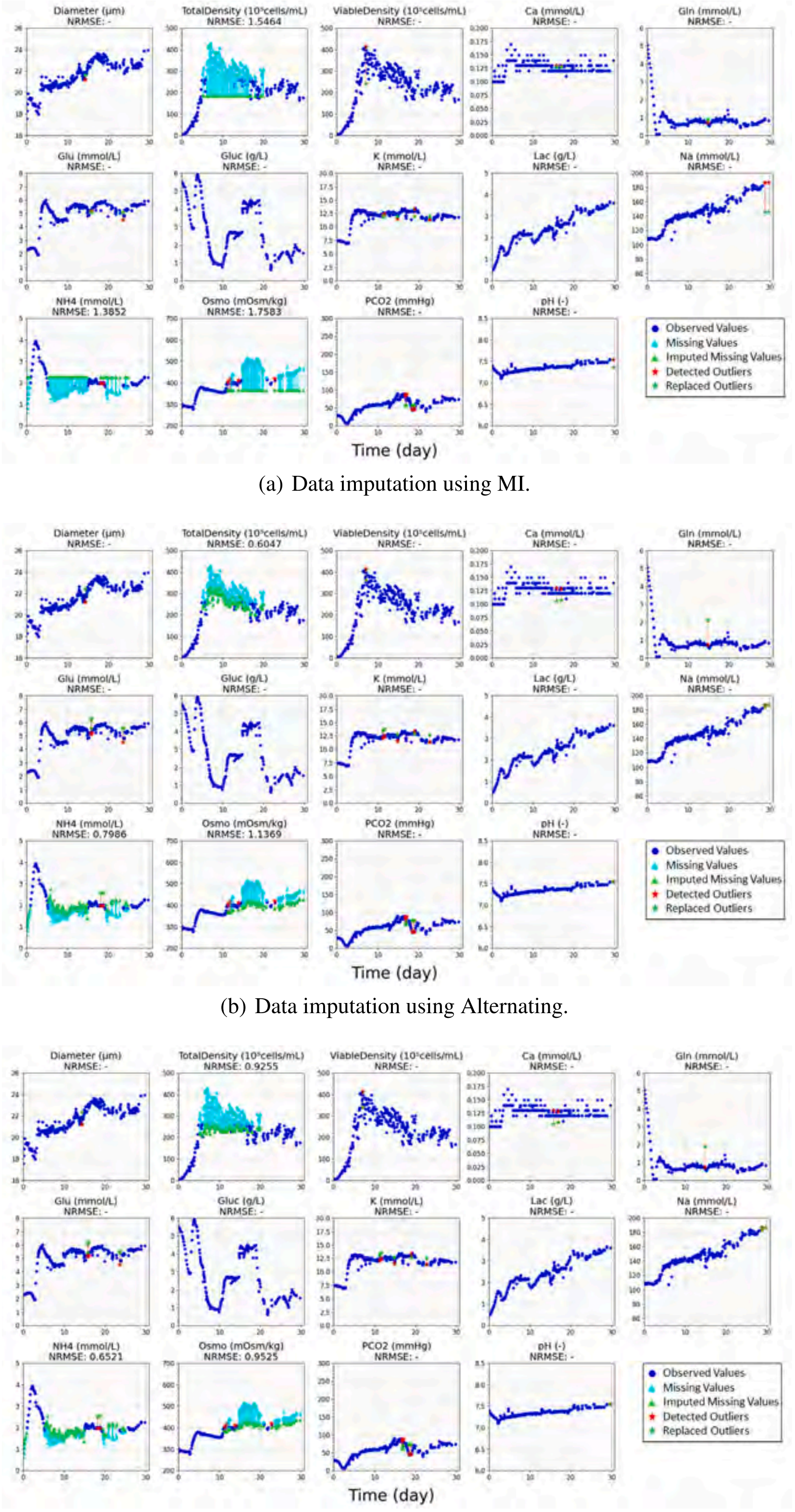


(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. D.3. (*continued*).

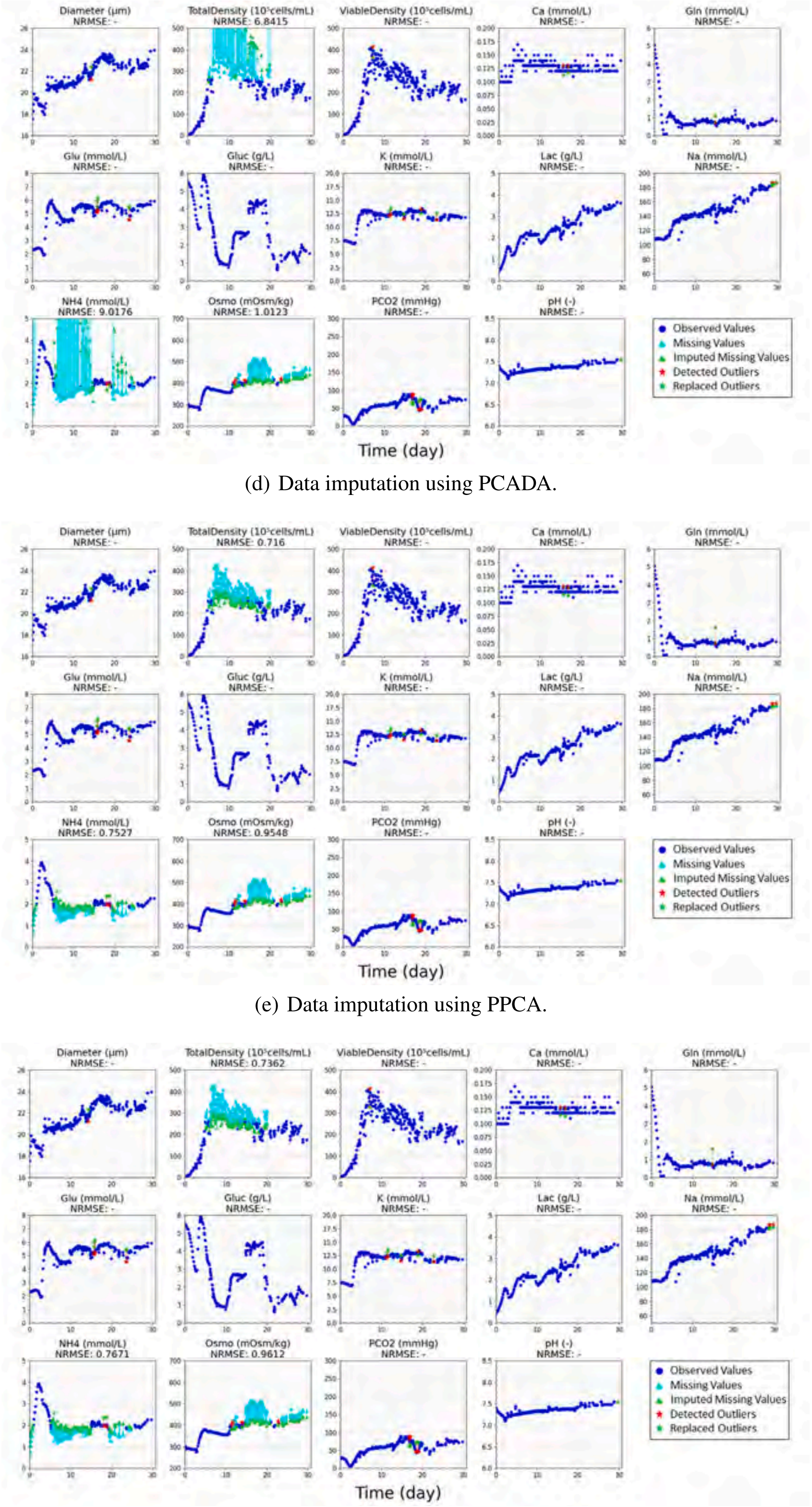


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. D.4. Data imputation results using 9 algorithms excluding ALS in the censoring case.

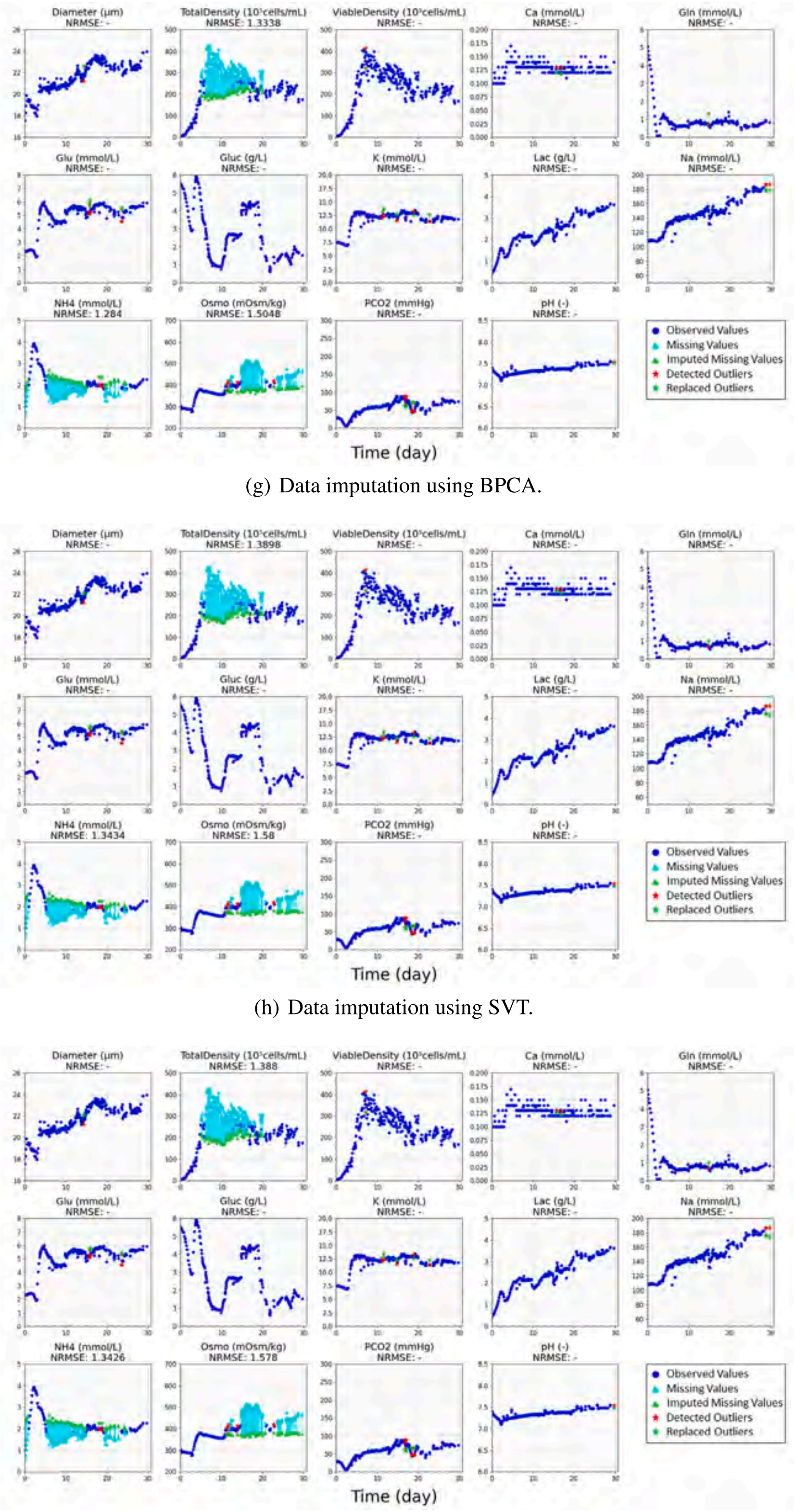


(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. D.4. (*continued*).

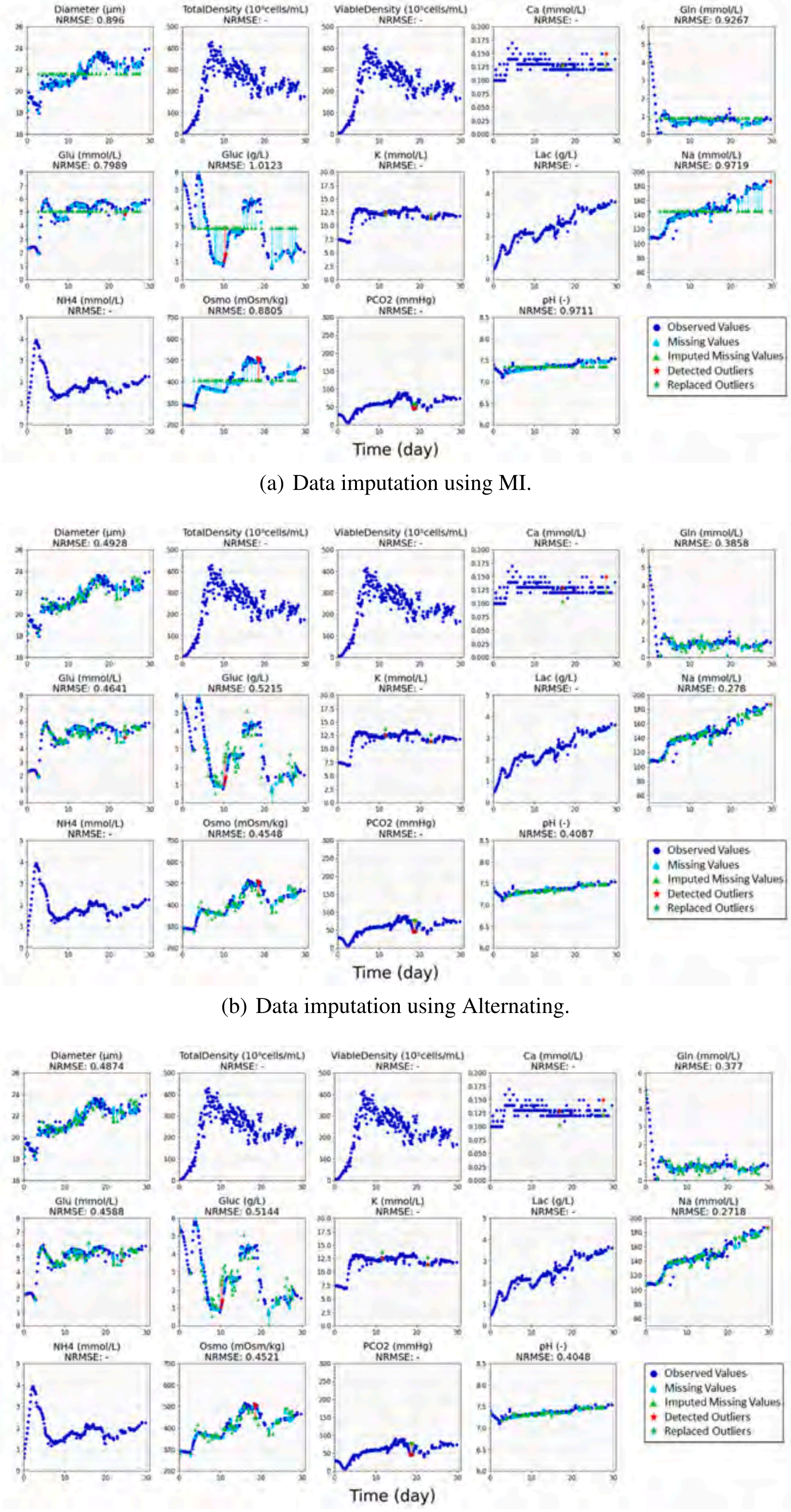


(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. D.4. (*continued*).

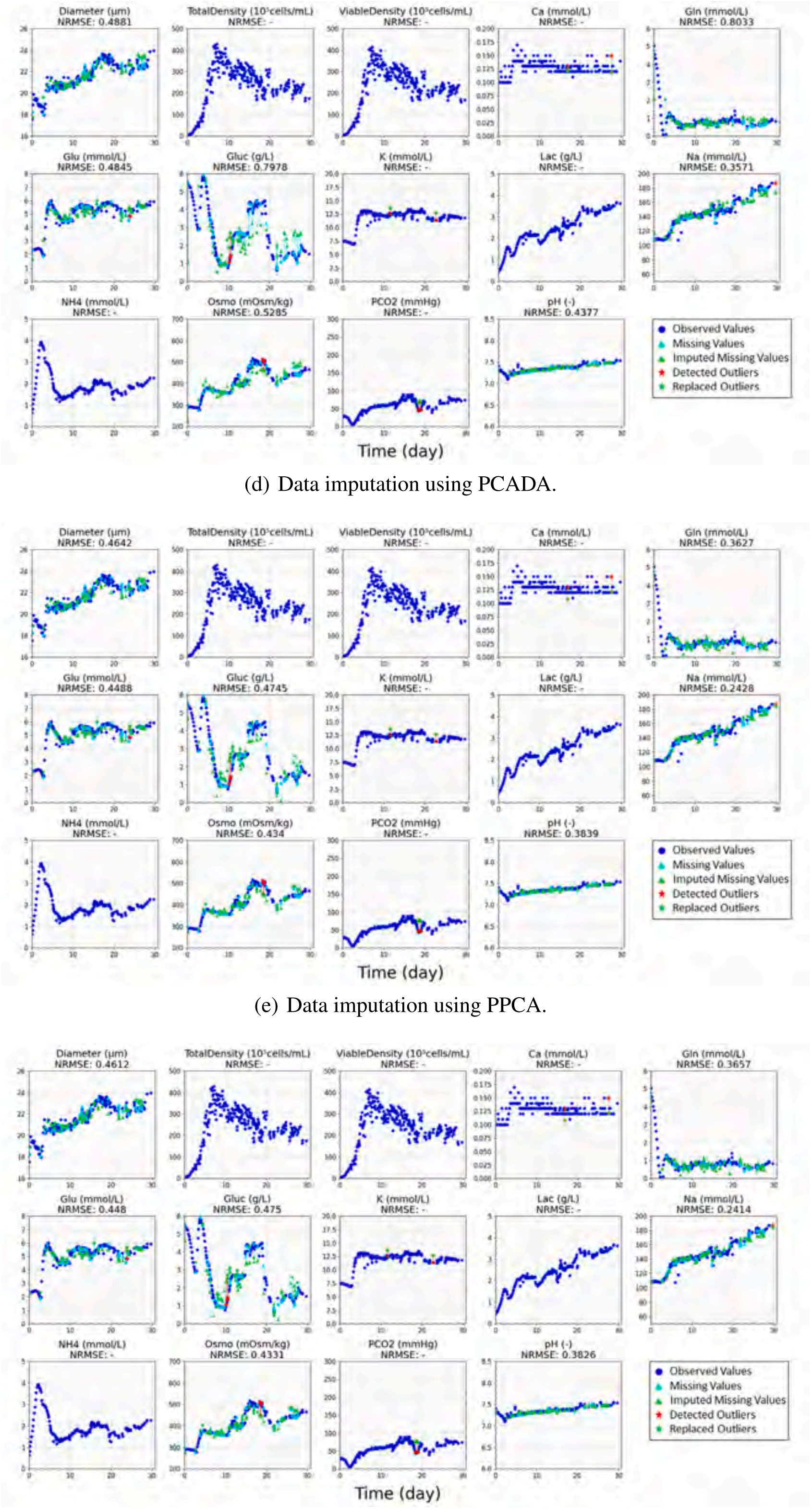


(a) Data imputation using MI.

(b) Data imputation using Alternating.

(c) Data imputation using SVDImpute.

Fig. D.5. Data imputation results using 9 algorithms excluding ALS in the patterned missingness case.

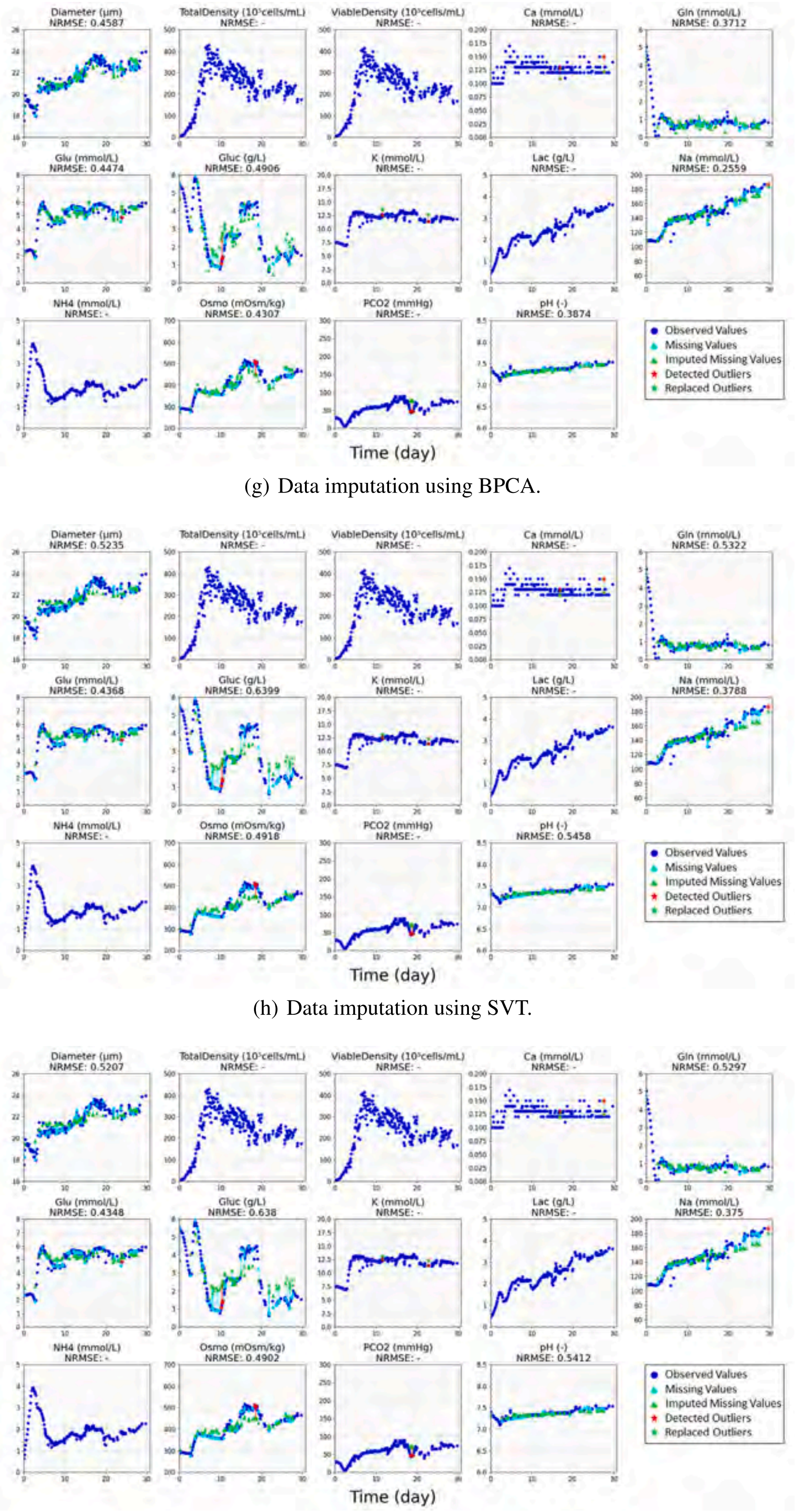


(d) Data imputation using PCADA.

(e) Data imputation using PPCA.

(f) Data imputation using PPCA-M.

Fig. D.5. (*continued*).



(g) Data imputation using BPCA.

(h) Data imputation using SVT.

(i) Data imputation using ALM.

Fig. D.5. (*continued*).

Appendix E. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.compchemeng.2023.108448.

References

Abdi, H., Williams, L.J., 2010. Principal component analysis. Wiley Interdiscip. Rev. Comput. Stat. 2 (4), 433–459. http://dx.doi.org/10.1002/wics.101.

Alford, J.S., 2006. Bioprocess control: Advances and challenges. Comput. Chem. Eng. 30 (10–12), 1464–1475. http://dx.doi.org/10.1016/j.compchemeng.2006.05.039.

Arteaga, F., Ferrer, A., 2002. Dealing with missing data in MSPC: Several methods, different interpretations, some examples. J. Chemom. 16 (8–10), 408–418. http://dx.doi.org/10.1002/cem.750.

Attias, H., 2013. Inferring parameters and structure of latent variable models by variational Bayes. http://dx.doi.org/10.48550/arXiv.1301.6676, arXiv:1301.6676.

Barazandegan, M., Ekram, F., Kwok, E., Gopaluni, B., Tulsyan, A., 2015. Assessment of type II diabetes mellitus using irregularly sampled measurements with missing data. Bioprocess Biosyst. Eng. 38, 615–629. http://dx.doi.org/10.1007/s00449-014-1301-7.

Bie, C., Liang, Y., Zhang, L., Zhao, Y., Chen, Y., Zhang, X., He, X., Song, X., 2019. Motion correction of chemical exchange saturation transfer MRI series using robust principal component analysis (RPCA) and PCA. Quant. Imaging Med. Surg. 9 (10), 1697. http://dx.doi.org/10.21037/qims.2019.09.14.

Biechele, P., Busse, C., Solle, D., Scheper, T., Reardon, K., 2015. Sensor systems for bioprocess monitoring. Eng. Life Sci. 15 (5), 469–488. http://dx.doi.org/10.1002/elsc.201500014.

Björck, E., Golub, G.H., 1973. Numerical methods for computing angles between linear subspaces. Math. Comp. 27 (123), 579–594. http://dx.doi.org/10.2307/2005662.

Bollen, K.A., 1987. Outliers and improper solutions: A confirmatory factor analysis example. Sociol. Methods Res. 15 (4), 375–384. http://dx.doi.org/10.1177/0049124187015004002.

Bouwmans, T., Javed, S., Zhang, H., Lin, Z., Otazo, R., 2018. On the applications of robust PCA in image and video processing. Proc. IEEE 106 (8), 1427–1457. http://dx.doi.org/10.1109/JPROC.2018.2853589.

Bouwmans, T., Zahzah, E.H., 2014. Robust PCA via principal component pursuit: A review for a comparative evaluation in video surveillance. Comput. Vis. Image Underst. 122, 22–34. http://dx.doi.org/10.1016/j.cviu.2013.11.009.

Bridewell, W., Langley, P., Racunas, S., Borrett, S., 2006. Learning process models with missing data. In: Machine Learning: ECML 2006: 17th European Conference on Machine Learning, Berlin, Germany, September 18-22, 2006 Proceedings 17. Springer, pp. 557–565. http://dx.doi.org/10.1007/11871842_52.

Cai, J.-F., Candès, E.J., Shen, Z., 2010. A singular value thresholding algorithm for matrix completion. SIAM J. Optim. 20 (4), 1956–1982. http://dx.doi.org/10.1137/080738970.

Candès, E.J., Li, X., Ma, Y., Wright, J., 2011. Robust principal component analysis? J. ACM 58 (3), 1–37. http://dx.doi.org/10.1145/1970392.1970395.

Candès, E., Recht, B., 2012. Exact matrix completion via convex optimization. Commun. ACM 55 (6), 111–119. http://dx.doi.org/10.1007/s10208-009-9045-5.

Cao, L., Li, H., Guo, H., Wang, B., 2019. Robust PCA for face recognition with occlusion using symmetry information. In: 2019 IEEE 16th International Conference on Networking, Sensing and Control. ICNSC, Banff, AB, Canada, May 9-11, 2019, IEEE, pp. 323–328. http://dx.doi.org/10.1109/ICNSC.2019.8743225.

Casella, G., Berger, R.L., 2002. Statistical Inference. Duxbury Press, Pacific Grove, CA.

Chandrasekaran, V., Sanghavi, S., Parrilo, P.A., Willsky, A.S., 2011. Rank-sparsity incoherence for matrix decomposition. SIAM J. Optim. 21 (2), 572–596. http://dx.doi.org/10.1137/090761793.

Cherapanamjeri, Y., Gupta, K., Jain, P., 2017. Nearly optimal robust matrix completion. In: International Conference on Machine Learning. Sydney, Australia, August 6-11, 2017, PMLR, pp. 797–805, https://proceedings.mlr.press/v70/cherapanamjeri17a.html.

Chiang, L.H., Russell, E.L., Braatz, R.D., 2000. Fault Detection and Diagnosis in Industrial Systems. Springer Science & Business Media, London, UK.

Craven, S., Whelan, J., Glennon, B., 2014. Glucose concentration control of a fed-batch mammalian cell bioprocess using a nonlinear model predictive controller. J. Process Control 24 (4), 344–357. http://dx.doi.org/10.1016/j.jprocont.2014.02.007.

Fan, J., Chow, T.W.S., Qin, S.J., 2021. Kernel-based statistical process monitoring and fault detection in the presence of missing data. IEEE Trans. Ind. Inform. 18 (7), 4477–4487. http://dx.doi.org/10.1109/TII.2021.3119377.

Fujimori, K., Covell, D.G., Fletcher, J.E., Weinstein, J.N., 1990. A modeling analysis of monoclonal antibody percolation through tumors: A binding-site barrier. J. Nucl. Med. 31 (7), 1191–1198, https://pubmed.ncbi.nlm.nih.gov/2362198/.

Gambhir, A., Europa, A.F., Hu, W.-S., 1999. Alteration of cellular metabolism by consecutive fed-batch cultures of mammalian cells. J. Biosci. Bioeng. 87 (6), 805–810. http://dx.doi.org/10.1016/s1389-1723(99)80157-1.

Gopaluni, R.B., 2010. Nonlinear system identification under missing observations: The case of unknown model structure. J. Process Control 20 (3), 314–324. http://dx.doi.org/10.1016/j.jprocont.2009.12.008.

Grung, B., Manne, R., 1998. Missing values in principal component analysis. Chemometr. Intell. Lab. Syst. 42 (1–2), 125–139. http://dx.doi.org/10.1016/S0169-7439(98)00031-8.

Hamy, V., Dikaios, N., Punwani, S., Melbourne, A., Latifoltojar, A., Makanyanga, J., Chouhan, M., Helbren, E., Menys, A., Taylor, S., Atkinson, D., 2014. Respiratory motion correction in dynamic MRI using robust data decomposition registration–application to DCE-MRI. Med. Image Anal. 18 (2), 301–313. http://dx.doi.org/10.1016/j.media.2013.10.016.

Heinzle, E., Biwer, A.P., Cooney, C.L., 2007. Development of Sustainable Bioprocesses: Modeling and Assessment. John Wiley & Sons, Chichester, England.

Hong, M.S., Velez-Suberbie, M.L., Maloney, A.J., Biedermann, A., Love, K.R., Love, J.C., Mukhopadhyay, T.K., Braatz, R.D., 2021. Macroscopic modeling of bioreactors for recombinant protein producing *Pichia pastoris* in defined medium. Biotechnol. Bioeng. 118 (3), 1199–1212. http://dx.doi.org/10.1002/bit.27643.

Horn, J.L., 1965. A rationale and test for the number of factors in factor analysis. Psychometrika 30, 179–185. http://dx.doi.org/10.1007/BF02289447.

Hotelling, H., 1947. Multivariate quality control. In: Eisenhart, C., Hastay, M.W., Wallis, W.A. (Eds.), Selected Techniques of Statistical Analysis. McGraw-Hill, New York, pp. 111–184.

Ilin, A., Raiko, T., 2010. Practical approaches to principal component analysis in the presence of missing values. J. Mach. Learn. Res. 11, 1957–2000, https://jmlr.csail.mit.edu/papers/volume11/ilin10a/ilin10a.pdf.

Imtiaz, S.A., Shah, S.L., 2008. Treatment of missing values in process data analysis. Can. J. Chem. Eng. 86 (5), 838–858. http://dx.doi.org/10.1002/cjce.20099.

Isaksson, A.J., 1993. Identification of ARX-models subject to missing data. IEEE Trans. Automat. Control 38 (5), 813–819. http://dx.doi.org/10.1109/9.277253.

Jackson, J.E., Mudholkar, G.S., 1979. Control procedures for residuals associated with principal component analysis. Technometrics 21 (3), 341–349. http://dx.doi.org/10.2307/1267757.

Jolliffe, I., 1986. Principal Component Analysis. Springer, New York.

Konstantinov, K.B., Cooney, C.L., 2015. White paper on continuous bioprocessing May 20–21, 2014 Continuous Manufacturing Symposium. J. Pharm. Sci. 104 (3), 813–820. http://dx.doi.org/10.1002/jps.24268.

Larsen, R.M., 2004. Propack-software for large and sparse SVD calculations. pp. 2008–2009, Available online. URL http://sun.stanford.edu/~rmunk/PROPACK.

Li, W., Yue, H.H., Valle-Cervantes, S., Qin, S.J., 2000. Recursive PCA for adaptive process monitoring. J. Process Control 10 (5), 471–486. http://dx.doi.org/10.1016/S0959-1524(00)00022-6.

Lin, Z., Chen, M., Ma, Y., 2010. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. http://dx.doi.org/10.48550/arXiv.1009.5055, arXiv:1009.5055.

Luan, X., Fang, B., Liu, L., Yang, W., Qian, J., 2014. Extracting sparse error of robust PCA for face recognition in the presence of varying illumination and occlusion. Pattern Recognit. 47 (2), 495–508. http://dx.doi.org/10.1016/j.patcog.2013.06.031.

Mavridis, D., Moustaki, I., 2008. Detecting outliers in factor analysis using the forward search algorithm. Multivar. Behav. Res. 43 (3), 453–475. http://dx.doi.org/10.1080/00273170802285909.

Miller, P., Swanson, R.E., Heckler, C.E., 1998. Contribution plots: A missing link in multivariate quality control. Appl. Math. Comput. Sci. 8 (4), 775–792, http://zbc.uz.zgora.pl/repozytorium/Content/58067/AMCS_1998_8_4_5.pdf.

Nelson, P., Taylor, P.A., MacGregor, J.F., 1996. Missing data methods in PCA and PLS: Score calculations with incomplete observations. Chemometr. Intell. Lab. Syst. 35 (1), 45–65. http://dx.doi.org/10.1016/S0169-7439(96)00007-X.

Netrapalli, P., Niranjan, U., Sanghavi, S., Anandkumar, A., Jain, P., 2014. Non-convex robust PCA. Adv. Neural Inf. Process. Syst. 27, https://papers.nips.cc/paper_files/paper/2014/file/443cb001c138b2561a0d90720d6ce111-Paper.pdf.

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. http://dx.doi.org/10.1093/bioinformatics/btg287.

Pison, G., Rousseeuw, P.J., Filzmoser, P., Croux, C., 2003. Robust factor analysis. J. Multivariate Anal. 84 (1), 145–172. http://dx.doi.org/10.1016/S0047-259X(02)00007-6.

Qiu, C., Vaswani, N., Hogben, L., 2014. Recursive robust PCA or recursive sparse recovery in large but structured noise. IEEE Trans. Inform. Theory 60 (8), 5007–5039. http://dx.doi.org/10.1109/ICASSP.2013.6638807.

Raghavan, H., Tangirala, A.K., Gopaluni, R.B., Shah, S.L., 2006. Identification of chemical processes with irregular output sampling. Control Eng. Pract. 14 (5), 467–480. http://dx.doi.org/10.1016/j.conengprac.2005.01.015.

Rousseeuw, P.J., Hubert, M., 2011. Robust statistics for outlier detection. Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 1 (1), 73–79. http://dx.doi.org/10.1002/widm.2.

Severson, K., Chaiwatanodom, P., Braatz, R.D., 2016. Perspectives on process monitoring of industrial systems. Annu. Rev. Control 42, 190–200. http://dx.doi.org/10.1016/j.ifacol.2015.09.646.

Severson, K.A., Molaro, M.C., Braatz, R.D., 2017. Principal component analysis of process datasets with missing values. Processes 5 (3), 38. http://dx.doi.org/10.3390/pr5030038.

Shumway, R.H., Stoffer, D.S., 2000. Time Series Analysis and Its Applications. Springer, New York.

Stanimirova, I., Daszykowski, M., Walczak, B., 2007. Dealing with missing values and outliers in principal component analysis. Talanta 72 (1), 172–178. http://dx.doi.org/10.1016/j.talanta.2006.10.011.

Stephanopoulos, G., San, K.-Y., 1984. Studies on on-line bioreactor identification. I. Theory. Biotechnol. Bioeng. 26 (10), 1176–1188. http://dx.doi.org/10.1002/bit.260261006.

Stevens, J.P., 1984. Outliers and influential data points in regression analysis. Psychol. Bull. 95 (2), 334–344. http://dx.doi.org/10.1037/0033-2909.95.2.334.

Tipping, M.E., Bishop, C.M., 1999. Probabilistic principal component analysis. J. R. Stat. Soc. Ser. B Stat. Methodol. 61 (3), 611–622. http://dx.doi.org/10.1111/1467-9868.00196.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B., 2001. Missing value estimation methods for DNA microarrays. Bioinformatics 17 (6), 520–525. http://dx.doi.org/10.1093/bioinformatics/17.6.520.

Vaswani, N., Narayanamurthy, P., 2018. Static and dynamic robust PCA and matrix completion: A review. Proc. IEEE 106 (8), 1359–1379. http://dx.doi.org/10.1109/JPROC.2018.2844126.

Wiberg, T., 1976. Computation of principal components when data are missing. In: Proceedings of the Second Symposium on Computational Statistics. pp. 229–236, https://link.springer.com/article/10.1007/BF01902863.

Wlaschin, K.F., Hu, W.-S., 2006. Fedbatch culture and dynamic nutrient feeding. Adv. Biochem. Eng. Biotechnol. 101, 43–74. http://dx.doi.org/10.1007/10_015.

Wold, S., 1978. Cross-validatory estimation of the number of components in factor and principal components models. Technometrics 20 (4), 397–405. http://dx.doi.org/10.2307/1267639.

Wright, J., Ganesh, A., Rao, S., Peng, Y., Ma, Y., 2009. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. Adv. Neural Inf. Process. Syst. 22, https://proceedings.neurips.cc/paper_files/paper/2009/file/c45147dee729311ef5b5c3003946c48f-Paper.pdf.

Yi, X., Park, D., Chen, Y., Caramanis, C., 2016. Fast algorithms for robust PCA via gradient descent. Adv. Neural Inf. Process. Syst. 29, https://papers.nips.cc/paper_files/paper/2016/file/b5f1e8fb36cd7fbeb7988e8639ac79e9-Paper.pdf.

Yu, L., Snapp, R.R., Ruiz, T., Radermacher, M., 2010. Probabilistic principal component analysis with expectation maximization (PPCA-EM) facilitates volume classification and estimates the missing data. J. Struct. Biol. 171 (1), 18–30. http://dx.doi.org/10.1016/j.jsb.2010.04.002.

Zhan, J., Vaswani, N., 2015. Robust PCA with partial subspace knowledge. IEEE Trans. Signal Process. 63 (13), 3332–3347. http://dx.doi.org/10.1109/ISIT.2014.6875222.

Zhu, X., Braatz, R.D., 2014. Two-dimensional contribution map for fault identification. IEEE Control Syst. 34 (5), 72–77. http://dx.doi.org/10.1109/MCS.2014.2333295.

Zhu, J., Ge, Z., Song, Z., Gao, F., 2018. Review and big data perspectives on robust data mining approaches for industrial process modeling with outliers and missing data. Annu. Rev. Control 46, 107–133. http://dx.doi.org/10.1016/j.arcontrol.2018.09.003.