water

MDPI

Article

Double Tensor-Decomposition for SCADA Data Completion in Water Networks

Pere Marti-Puig [1,*,†], Arnau Martí-Sarri [1,2,†] and Moisès Serra-Serra [3]

[1] Data and Signal Processing Group, U Science Tech, University of Vic—Central University of Catalonia, c/de la Laura 13, 08500 Vic, Catalonia, Spain; arnau.marti@uvic.cat
[2] Aigues de Vic S.A., c/Santiago Ramon y Cajal 60, 08500 Vic, Catalonia, Spain
[3] MECAMAT Group, U Science Tech, University of Vic—Central University of Catalonia, c/de la Laura 13, 08500 Vic, Catalonia, Spain; moises.serra@uvic.cat
* Correspondence: pere.marti@uvic.cat; Tel.: +34-93-881-55-19
† These authors contributed equally to this work.

Received: 30 November 2019; Accepted: 19 December 2019; Published: 24 December 2019

Abstract: Supervisory Control And Data Acquisition (SCADA) systems currently monitor and collect a huge among of data from all kind of processes. Ideally, they must run without interruption, but in practice, some data may be lost due to a sensor failure or a communication breakdown. When it happens, given the nature of these failures, information is lost in bursts, that is, sets of consecutive samples. When this occurs, it is necessary to fill out the gaps of the historical data with a reliable data completion method. This paper presents an *ad hoc* method to complete the data lost by a SCADA system in case of long bursts. The data correspond to levels of drinking water tanks of a Water Network company which present fluctuation patterns on a daily and a weekly scale. In this work, a new *tensorization* process and a novel completion algorithm mainly based on two tensor decompositions are presented. Statistical tests are realised, which consist of applying the data reconstruction algorithms, by deliberately removing bursts of data in verified historical databases, to be able to evaluate the real effectiveness of the tested methods. For this application, the presented approach outperforms the other techniques found in the literature.

Keywords: water networks; SCADA data; tensor completion; tensor decomposition

1. Introduction

Currently, the data collection has made a real breakthrough with the many varieties of sensors and devices which have the possibility of transmitting information from anywhere. With the current increase in data storage capacity, more data can be stored than can be processed. In practice, when processing this amount of information, the problem of incomplete or missing data has to be addressed. The management of data from water networks [1] or from hydrological resources [2–4] is no exception. The problem of data loss is especially challenging when it occurs in long bursts of consecutive values.

Aigues de Vic S. A. (AVSA) decided three years ago to renew its SCADA (Supervisory Control And Data Acquisition) system because it was becoming obsolete. AVSA is the enterprise responsible for the water supply of the city of Vic. The SCADA is a tool for the technicians of the Water Purification Plant (WPP), where the Ter river water is purified, and for the operators of the Water Distribution System (WDS). The old system is usefully to receive information from the sensors and to make decisions, but not to remotely configure and control the devices. For example, in the case of a pumping system, it is possible to see the pumps' configuration. Still, if it is necessary to reduce the pumped water flow, the operator must go where the pumps are located and do it manually.

In the SCADA renewal process, great importance was given to preserving the data collected by the equipment to be replaced. To avoid the loss of information accumulated by the old SCADA system during the last four years, the data have to be imported from the old database to the new one. So, during this procedure, the historical data series were verified with the aim to not import unusable data, and some problems related to missing data were detected. This case of study comes from the data collected by the level sensor of the deposit located in the main water reservoir of the city of Vic. It is crucial to preserve these data since they can be used to find consumption patterns in the city of Vic and then to build models that allow detecting anomalies in the operation of the water distribution network. As drinking water is a scarce resource in many regions of the planet, the efficient management of water becomes strategic. So, it is necessary to know the behaviour of distribution networks to improve their operation and to build reliable models that support predictive maintenance and early fault detection functionalities. This is why modern distribution networks monitor the status of hundreds of variables through SCADA equipment. Technological improvements such as those we are proposing will be implemented progressively and we hope that improvements in management will also be incremental. At this point, and before exploiting this huge amount of data, its quality should be guaranteed. In this sense, one of the problems encountered is the occasional loss of data due to sensors failures, sensor re-calibrations or communication failures that take time to be repaired and therefore causes the loss of entire bursts of data. Ensuring data integrity and being able to fill as credibly as possible data gaps is a critical step to the later exploitation of these data and it is the first step in the processing chain so that the errors made in this stage could be easily propagated towards the following ones. The aim of this paper is to present an effective method of data completion.

Different estimators based on interpolation or in linear prediction techniques were used to restore lost samples [5–7] with acceptable results, but in the case of large amounts of consecutive lost samples, the performance of these estimators falls dramatically. However, this type of data loss is frequent as it is caused by a communication failure between the Programmable Logic Controller (PLC) where the sensor is connected, and the central SCADA server where the data is stored. The reason seems to be that classical data estimation methods hardly exploit patterns that occur on a combination of time scales, such as daily and weekly scales, as occurs in our case of study. In contrast, methods based on tensor decomposition are capable of revealing these patterns when the data are properly ordered in a multidimensional way [8–13].

In a previous study [14], an *ad hoc* method that combines a tensor decomposition and linear prediction techniques was implemented. The method was specially adapted to work with the water deposit level signals and to deal with long bursts of lost samples. That approach was compared with other reconstruction methods based on tensor techniques found in the literature, providing better results for these specific conditions. The work in [14] also presented a *continuity correction method* that guaranteed the continuity of the signal of the data recovered. The method presented in this paper is reminiscent of [14] in some aspects, but gives some novelties that significantly impact the performance, improving it very considerably.

This method starts by filling the lost burst values to avoid missing elements before to *tensorizing* the data. The main significant differences and contributions are introduced: (1), to fill empty values the most straightforward interpolation is chosen, which we called *ramp method*, discarding other computational more intense strategies as [5–7] or as the one used in [14], (2), a new way to organize the tensor, called *burst centered tensorization*, is introduced, and (3), there is applied a two-step reconstruction process by concatenating two tensor decompositions of different tensor cores. Each time the data is reconstructed, the *continuity correction* method is carried out. The first tensor decomposition has a very small-dimension core and obtains a rough approximation which is refined thought a second reconstruction done by applying a tensor decomposition with a bigger core. Note that, although several tensor decompositions exist, the two most extended and well-known are the Tucker [15–17] and the CANDECOMP/PARAFAC (CP) [18,19] which are the two decompositions considered in this work as well as in [14]. At this point, we could combine different types of tensor decompositions,

the results obtained employing CP and Tucker are very close so, in order to maintain the text simplicity, this study only considers these two. References [11–13,20] can provide the reader a quality introduction to the tensor algebra.

For the type of signals treated, when data losses are distributed uniformly or even in short bursts of less than 30–40 samples, all methods work more or less likewise. Above that length, tensor-based methods, like the one we propose in this work, show better performance. In practice, it is observed that the length of the bursts of data lost on a SCADA system communication cutting off can be much longer of 40 samples. The present work obeys the need to improve the performance of the data completion methods currently used. The main contribution of this research is to improve the data reconstruction methodology developed in [14], whose results are taken as a reference since they were better in comparison with the proven tensor methods that already exist in the literature.

We have focused on a type of signal obtained from a drinking water distribution network because we have a large amount of data to quantify the quality of the solution provided and compare it with other reference algorithms. Besides, the proposed algorithm can be used to complete data of other types, but above all, to those data that are directly or indirectly related with the human activity and its hourly, daily and weekly patterns. In addition, the sensor techniques allow to reveal and express in a very compact way complicated relationships between modes of different dimensions, so that we think that they can be also very useful for the development of models for early detection of failures in water transport systems.

Henceforth, the work is organized as follows. In the *Related previous work* section we briefly expose the algorithm developed in [14] on the objective that the reader can appreciate the differences with the presented method. In the *Materials and methods* section the details for reproducing results are explained. Aspects related to the database and its pre-processing are treated briefly because they are the same as those carried out in [14]. The same is applied for tensor concepts or for the explanation about the algorithm evaluation and their performance quantification. In order that the article can be read in a simple way, in this section all the steps to complete the algorithm are explained, whether they are contributions of this work, such as the *burst centered tensorization*, the *signal smoothing* and the *double tensor decomposition approach*, whether they are contributions of [14] such as the *continuity correction*. In the *Results* section, since the proposed method uses a combination of two tensor decompositions, a study of the combination of decomposition sizes that provide better results for both CP and Tucker models is performed. Then the algorithm is tested by parts applying each of the proposed improvements, one by one, and then all together, in order to quantify their impact in the whole algorithm. Finally, the most remarkable aspects will be summarized in the *Conclusions* section.

2. Related Previous Work

This section presents a summary of some previous work closely related to this approach. That includes the explanation of a data-completion method that was designed to recover bursts of lost data.

Data-Completion Method Based on a Tensor Decomposition to Recover Bursts of Lost Data

The tensor method proposed in [14] to restore burst of lost data can be divided in four main operations as it is shown in the diagram of the Figure 1. The first one consist on filling the lost data because the next operations cannot work with empty data. To fill in the empty data is necessary to use a value imputation method that can be more or less complicated. In [14] different options were tested. The best performance was found with a linear predictor filter technique whose coefficients were calculated according to Wiener's theory of optimal filtering. This involved calculating several autocorrelation coefficients and determining the size of a pair of FIR (Finite Impulse Response) filters. A straightforward interpolation method had also been evaluated, which produced lower but acceptable results which, in contrast, was computationally much easier. In [14] this imputation method is called the *ramp method*, and as it is also used in the presented approach, it is described in the Section 3.3.2.

The second operation is the *tensorization* of the data, which is the process of packaging lower-dimensional data into a container, the tensor, with more dimensions than the original one. This allows to find the relations between dimensions, which are difficult to perceive in more simple structures. Because of a visual inspection of the data seems to reveal patterns on a daily and weekly scale, in order to take advantage of these regularities, a 3-dimensional tensor was composed. The proposal was to build a three-dimensional tensor $\chi \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ in which, the first index indicates the 5-min day-intervals fixed by the sample frequency of the SCADA system, the second index indicates the day of the week and, the third index depends on the number of weeks n_w included in the tensor. According to this, $\chi \in \mathbb{R}^{288 \times 7 \times n_w}$. The proposed organization uses past and future data with respect to the burst location in order to contribute with past and future information. The week where the burst is located is placed in the central week index, and some weeks before and some weeks after are taken, depending on the tensor size n_w. Note that to always have a central week in the tensor n_w must be an odd value (3, 5, 7, ...). We refer to this *tensorization* as *week-centered tensorization*. As we can see, it collects data corresponding to a whole number of weeks. The third operation consists of constructing a simplified version of the tensor, named $\chi_{(1)}^{288 \times 7 \times n_w}$ in Figure 1 by using a tensor decomposition. In Figure 1a it was represented the Tucker decomposition in Figure 1b the CP. This operation reduces the irregularities introduced by the linear estimator used in the first operation. The last operation is the continuity correction, Section 3.3.4, used to guarantee the continuity of the signal in the extremes of the restored burst. In Section 3.3.2 there is explained the *ramp method*. An overview of the main tensor decompositions are presented in Sections 3.2 and 3.3.4 for the continuity correction process.

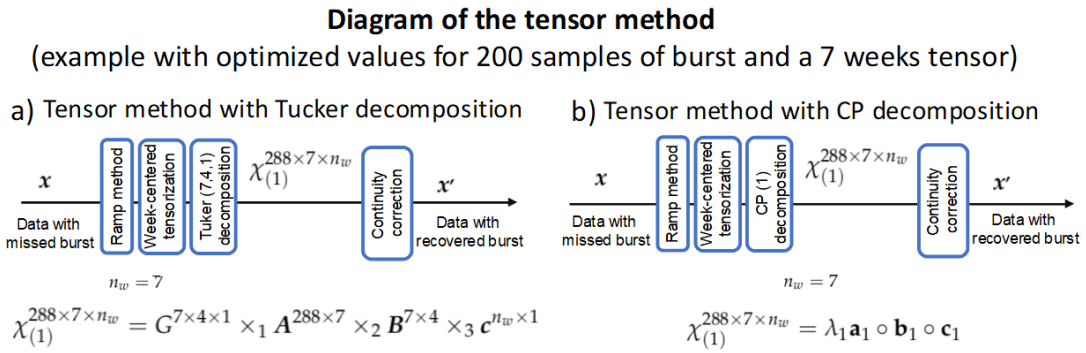

Figure 1. Graphic representation of the tensor method operations with the optimised configuration for our database in order to recover a burst of 200 lost samples when employing a tensor size of $288 \times 7 \times 7$. (**a**) The Tucker model. (**b**) The CANonical DECOMPosition/PARAllel FACtorization (CANDECOMP/PARAFAC or CP) model.

3. Materials and Methods

3.1. Used Database

The historical data used to perform the simulations are provided by Aigues de Vic S.A. (AVSA). Their Supervisory Control And Data Acquisition (SCADA) system collect approximately 1300 different signals. Specifically, the data used on the simulations is provided by a water level sensor located in the deposit of Castell d'en Planes, which is the water reserve of the city of Vic. Data from this sensor were collected from 1 October 2015 onwards. The data used for the simulations were verified, discarding the weeks in which there is an excess of lost data because, in these weeks, the results of the algorithms cannot be quantified by comparing them with the actual data.

3.2. Tucker and CP Tensor Decompositions

A tensor is a container that can arrange data in N-ways or dimensions. An N-way tensor of real elements is denoted as $\chi \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and its elements as: $x_{i_1,i_2,\ldots,i_N}$. According this, an $N \times 1$ vector $\mathbf{x}$ is considered a tensor of order one, and an $N \times M$ matrix $\mathbf{X}$ (or $\mathbf{X}^{N \times M}$), a tensor of order two.

The procedure of reshaping a lower-dimensional original data organized in a vector or in a matrix into a tensor is referred to as *tensorization*. The procedure of reshaping the elements of a tensor into matrices or vectors is named *matrization* and *vectorization*, respectively.

Tensor decompositions are a very useful tool for revealing patterns between the dimensions in which the data are organized. In particular, low order tensor decompositions provide a simplified version of the data while making the relation between dimensions explicit.

In the case of the 3-dimensional tensor $\chi^{I\times J\times K}$ the approximations are given in the form of a smaller tensor core $G^{L\times M\times N}$, (where usually $I > L$, $J > M$, and $K > N$) and the L, J and K eigenvectors of modes -1, -2, and -3. The eigenvectors of each mode are organized as column vectors in their respective matrices $\boldsymbol{A}^{I\times L}$, $\boldsymbol{B}^{J\times M}$, and $\boldsymbol{C}^{K\times N}$. The size ($L$, M, N) of the core determines the level of the decomposition.

Many known tensor decompositions exist but overall of them the most widely used are the Tucker [15] and the CP [18] ones. These two are briefly presented below for the three-dimensional case.

In the 3-way Tucker model, the core is defined by parameters L, M, N, relative to the size of the core $G^{L\times M\times N}$ of the decomposition and, the decomposition, is expressed as Tucker(L, M, N) according to:

$$\chi^{I\times J\times K} \approx G^{L\times M\times N} \times_1 \boldsymbol{A}^{I\times L} \times_2 \boldsymbol{B}^{J\times M} \times_3 \boldsymbol{C}^{K\times N}, \tag{1}$$

where the symbol $\times_i$ is the i-way product of a tensor by a matrix; such a tensor operation defined, for instance, in [21]. The matrices of Equation (1) in terms of the column eigenvectors are: $\boldsymbol{A}^{I\times L} = [\mathbf{a}_1 \cdots \mathbf{a}_L]$, $\boldsymbol{B}^{J\times M} = [\mathbf{b}_1 \cdots \mathbf{b}_M]$ and $\boldsymbol{C}^{K\times N} = [\mathbf{c}_1 \cdots \mathbf{c}_N]$.

The 3-way CANDECOMP/PARAFAC (from CANonical DECOMPosition/PARAllel FACtorization) model is commonly known as CP and can be seen as particular case of the Tucker decomposition when the core $G^{L\times M\times N}$ is diagonal $G^{D\times D\times D}$ ($L = M = N = D$). Taking this observation into account the CP decomposition can be written in the same terms as in the case of Tucker decomposition, as follows:

$$\chi^{I\times J\times K} \approx G^{D\times D\times D} \times_1 \boldsymbol{A}^{I\times D} \times_2 \boldsymbol{B}^{J\times D} \times_3 \boldsymbol{C}^{K\times D} \tag{2}$$

In the case of decomposition CP, all dimensions have the same number D of eigenvectors and it depends on the only parameter D, so that it is referenced as CD(D). It is frequent to see the CD(D) decomposition written in function of the elements λ_i of the $G^{D\times D\times D}$ diagonal such as:

$$\chi^{I\times J\times K} \approx \sum_{i=1}^{D} \lambda_i \mathbf{a}_i \circ \mathbf{b}_i \circ \mathbf{c}_i, \tag{3}$$

where the symbol $\circ$ stands for the outer product and the column vectors $\mathbf{a}_i$, $\mathbf{b}_i$ and $\mathbf{c}_i$ which are related with the matrices of Equation (2) according to: $\boldsymbol{A}^{I\times D} = [\mathbf{a}_1 \cdots \mathbf{a}_D]$, $\boldsymbol{B}^{J\times D} = [\mathbf{b}_1 \cdots \mathbf{b}_D]$ and $\boldsymbol{C}^{K\times D} = [\mathbf{c}_1 \cdots \mathbf{c}_D]$.

The algebra of tensors is explained in detail and often with the support of graphical illustrations in [10–12,20]. Figure 2 shows a unified representation of both 3D tensor decompositions.

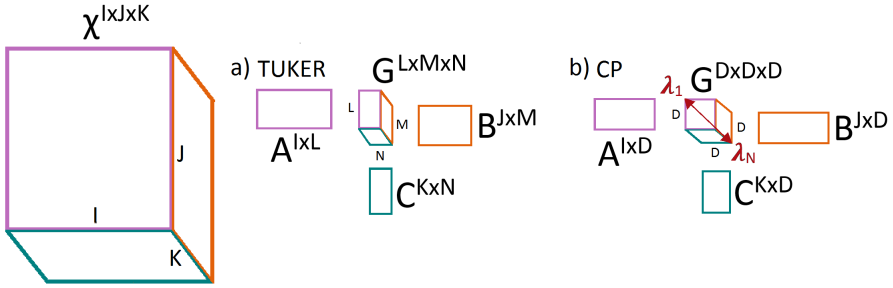

Figure 2. Diagram of the tensor decompositions used. (**a**) Tucker (**b**) CANDECOMP/PARAFAC (CP).

3.3. Double Decomposition Approach

In this section the proposed data completion method that achieves better results on the data reconstruction that the previous one developed in [14]. Figure 3 is used to clarify the explanation by showing the different parts as blocks. As was already mentioned, only the two more widely known tensor decomposition models have been considered, these being the Tucker and the CP. Thus the configuration of both decomposition algorithms have been analyzed with the aim of taking the biggest advantage possible from each one.

The first step consists of applying the *Smoothing process* described in the Section 3.3.1, which contributes to make the tensor decomposition a bit more effective. The tensor decomposition produces a continuous response. The sensor, however, measures the level as a percentage with resolution of 1%, providing a discrete signal of the deposit capacity. When the signal levels oscillate around the point of quantification, oscillations occur between adjacent discrete values. The *Smoothing process* is applied on the received data and corrects this effect.

The second step consists of applying an imputation method to fill the missing data. The *ramp method* is the selected one, which draws a line to join the known extreme values that delimit the burst as explained in Section 3.3.2. It is a very rough approximation, but does not need any configuration, which brings simplicity to the algorithm.

Once the data have no empty values, and the positions of lost burst values have been saved, the *burst centered tensorization* explained in Section 3.3.3 is applied. That is a new *tensorization* that places the original burst positions just in the central positions of the tensor. Note that this new *tensorization* does not modify the dimensions of the resulting tensor that remain $288 \times 7 \times n_w$ because they have relation with the hourly, daily and weekly patterns. Figure 3 considers the best algorithm configuration for $n_w = 7$ when the tensor decompositions are (a) Tucker and (b) CP. Once the data in $\boldsymbol{x}$ has no empty values, and the positions of lost burst values have been saved, the *burst centered tensorization* explained in Section 3.3.3 is applied. The tensor obtained is $\chi^{288\times7\times n_w}$. To remember, its first dimension indexes the SCADA measurements taken in 24 h every 5-min, its second dimension indexes the 7 days to complete a week, and its third dimension indexes the number n_w of weeks considered (which is an odd number: 3 or 7 in the tests realized). Figure 3 shows the best algorithm configuration for the particular case of $n_w = 7$ when the tensor decomposition considered are: (a) Tucker and (b) CP. In both cases, after been applied the first three blocks corresponding to the operations: *smoothing*, *ramp method* and *burst centered tensorization*, we have the tensor $\chi^{288\times7\times n_w}$.

The next step takes $\chi^{288\times7\times n_w}$ as input and gives the reconstruction $\chi_{(1)}^{288\times7\times n_w}$ as output. That reconstruction is performed following the decomposition of Tucker(4,6,1) in (a) and CP(1) in (b).

At that point, the samples of $\chi_{(1)}^{288\times7\times n_w}$ occupying the positions of the lost data burst are extracted, and a *continuity correction* performed with the original data is applied according to the details of Section 3.3.4. Then we put the corrected data in original tensor $\chi^{288\times7\times n_w}$ in substitution of the values provided by the ramp method. The resulting tensor after this step is $\chi'^{288\times7\times n_w}_{(1)}$.

Following the processing chain, the tensor $\chi'^{288\times7\times n_w}_{(1)}$ is taken and modeled using a decomposition of Tucker(4,7,7) in (a) and, CP(15) in (b), to obtain $\chi_{(2)}^{288\times7\times n_w}$. Again, the set of samples of $\chi_{(2)}^{288\times7\times n_w}$ located in the positions of the lost burst are taken to apply the *continuity correction* with the original data. The data obtained is used to complete the burst of missing data.

The Tucker(4,6,1) and Tucker(4,7,7) decomposition shown in Figure 3a correspond to the values that optimize in our database the recovery of a burst of 200 lost samples when employing the *tensorization* of size $288 \times 7 \times 7$ and the Tucker decomposition is used. The decomposition CP(1) and CP(15) shown in Figure 3b optimize the recovery of a burst of 200 lost samples using the same *tensorization* of size $288 \times 7 \times 7$ and the CP decomposition. The results are a statistical measure obtained after running 1000 simulations.

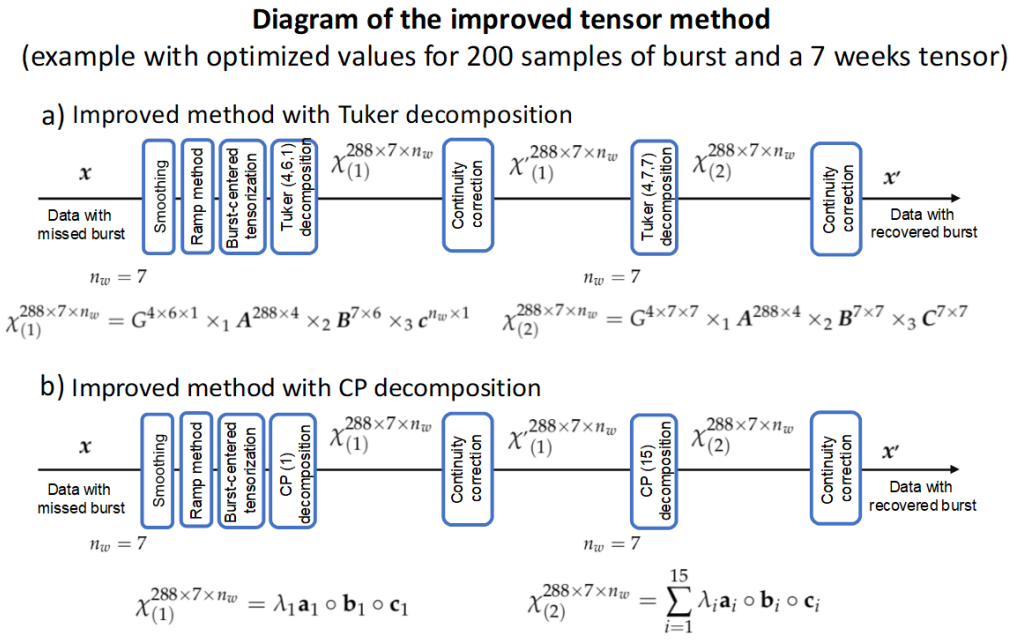

Figure 3. Graphic representation of the operations that compound the double tensor decomposition algorithm with the size of the decompositions optimised to recover bursts of 200 lost samples when employing a tensor size of $288 \times 7 \times 7$. (**a**) when using the Tucker model. (**b**) when using the CP model.

3.3.1. Signal Smoothing

The *ad hoc* smoothing algorithm adopted is developed considering the sensor way of working. The samples are processed in groups with the same integer value, and taking into account whether the signal is increasing, decreasing or is in a relative minimum or maximum. The blocks of samples of identical integer value A are processed keeping in mind the values of the contiguous blocks. The procedure is effortless. There are more elaborate filtering methods but those introduce delays in the signal, and thus of that this straightforward solution has been chosen instead. If the block corresponds to a signal increment, a line with a positive slope is built with extreme values $A - 0.5$ and $A + 0.5$. If the block corresponds to a signal decrement, a line with a negative slope is built similarly. If it is detected that it is a local maximum or minimum, the block is replaced by a triangular shape with the corresponding orientation. Figure 4 shows the smoothing performed through an example.

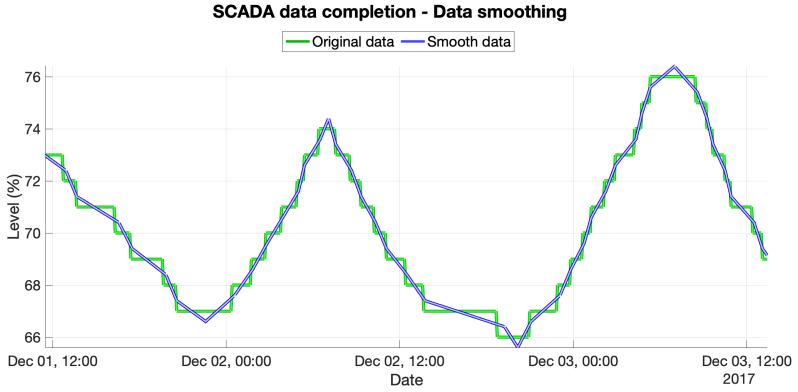

Figure 4. Smoothing process applied to the level sensor signal before the data *tensorization*.

3.3.2. The Ramp Method

The *ramp method* consists of filling the lost data by drawing a line between the last known sample before the lost burst starting, x_n, and the first known sample after the lost burst ending, x_{n+B+1}, where B is the length of the data burst lost in number of samples. So that, considering a lost burst of B samples and the index i going from 1 to B, to use a constant increment (or decrement), m, and fill the entire lost burst, x_{n+i} must be:

$$x_{n+i} = x_n + m \cdot i \quad \text{for} \quad m = \frac{x_n - x_{n+B+1}}{B+1}. \tag{4}$$

Figure 5 shows the performance of the *ramp method*.

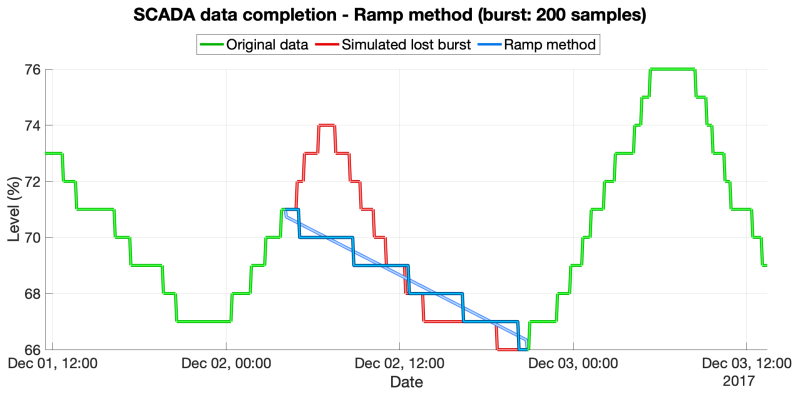

Figure 5. The *ramp method*. The red line shows the burst of the simulated lost data. The blue line corresponds to the data reconstruction of the linear method called the *ramp method*. The soft blue line shows the linear method result and the strong blue line shows the final signal reconstruction, which is adapted to the sensor resolution of 1%.

3.3.3. Burst Centered Tensorization

In the data completion methods that use tensor techniques, as far as we know, the way to organize the data into a tensor does not take into account the position of the lost data. In this work, a *burst centered tensorization method* is proposed, where the data selected to fill the tensor depends on the burst location. Figure 6 shows this process. In Figure 6a, the dark blue window shows the selection of data as was proposed in [14], in a typical way and with the data presented in a uni-dimensional view. Through the dark blue window it can be seen how the burst is not exactly located in the center of the selected data, which would mean being in the center of the dark blue window, even if the week where the burst is located is selected as the central week. This happens because the burst is hardly located in the middle of a week, which only happens if the burst is located exactly in the center of Thursday. The *burst centered tensorization* forces the burst to be located in the center of the data selected. The cyan window in Figure 6a shows this new data selection, where the burst is placed exactly on the center of the window. Figure 6b–d show the *tensorized* data by the typical way. The Figure 6e–g show the *tensorized* data from the *burst centered tensorization* method, which placed the burst in the core of the tensor (in the center of the central day of the week, which is located in the middle of the tensor). As explained in [14], lost data bursts never exceed the day, meaning that their length, B, is always less than 288 samples and that the burst can be located in the center of a day. Thus, given a B burst in a tensor $\chi^{I \times J \times K}$, the burst samples are placed at $J = 4$, $K = 0.5(W+1)$ with initial position $I_i = 0.5(288 - B)$ according to and indexation $\chi^{I_i:I_i+B-1 \times 4 \times 0.5(W+1)}$. Note that the daily cycles of the *burst centered tensorization* rarely start at 00:00 and the weeks do not start on Mondays, as occurred in [14], however there are always the same number of samples before and after the lost burst, which hardly happens with the previous *tensorization* method.

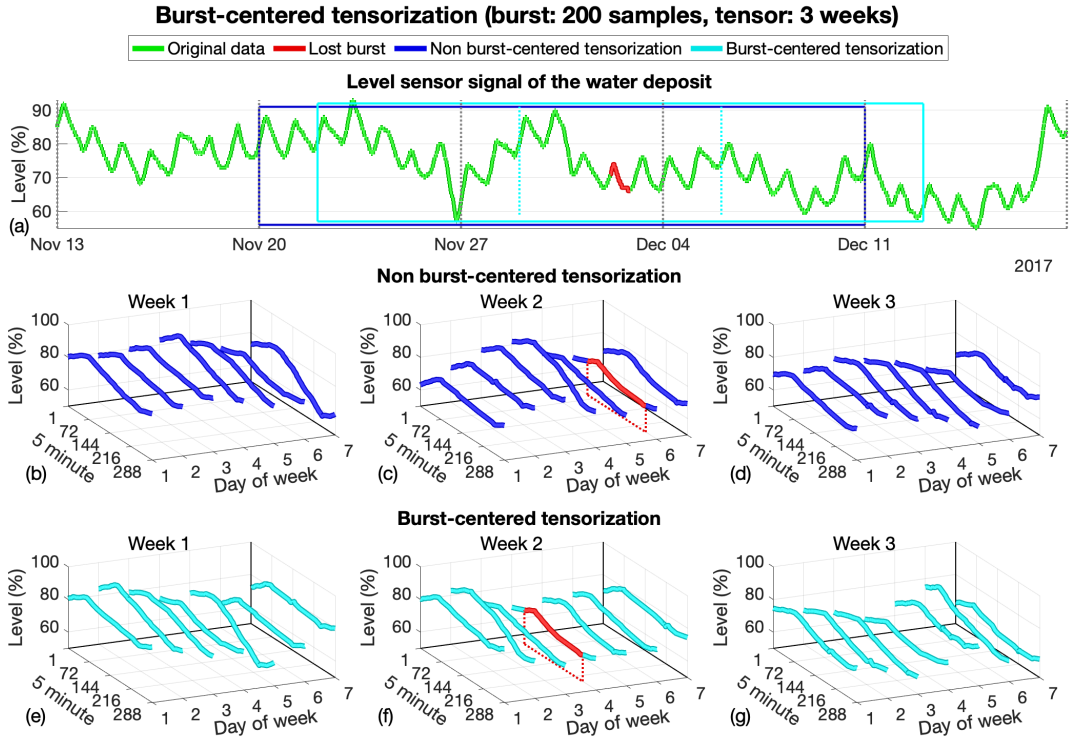

Figure 6. Example of the data *tensorization* of a 3 week tensor with 200 samples of data burst lost. In figure (**a**) the green line shows the original data and the red line shows the lost burst. The strong blue window shows the data introduced in the *non burst-centered* tensor and the soft blue one shows the data introduced in *burst-centered* tensor, which forces the burst to be on the center of the window. Figures (**b**–**d**) show the three weeks of the *non burst-centered* tensor and the location of the burst, which is located on the central week but not on the center of this week. The figures (**e**–**g**) show the 3 weeks of the *burst-centered* tensor and the new location of the burst in the core of the tensor, which is in the middle of the central week.

3.3.4. The Continuity Correction

This procedure was developed in order to maintain the continuity of the estimate provided by a tensor decomposition in its vector form $\hat{\mathbf{x}}$ and the known values of $\mathbf{x}$ at the edges of the burst. As consistently observed previously, the samples in the burst positions after a low-rank tensor reconstruction follow the original signal pretty well but with significant discontinuities in the extremes. Considering x_0 to be the last original known sample before the burst and $\hat{x}_0$ the sample from the tensor reconstruction in that position, we define the initial burst offset as $O_0 = x_0 - \hat{x}_0$. Similarly, for a lost burst of length B, the final burst offset can be defined as: $O_{B+1} = x_{B+1} - \hat{x}_{B+1}$. The corrected offset estimates $\tilde{x}_i$ are computed as follows:

$$\tilde{x}_i = \hat{x}_i + \frac{(B-i)O_0 + (i-1)O_{B+1}}{B-1} \qquad i = 1, \cdots, B. \tag{5}$$

Figure 7 shows graphically the *continuity correction* applied.

3.4. Algorithm Performance Evaluation

To test all the methods on the same conditions, firstly one thousand different starting positions are randomly selected from the 77 weeks of historical data previously verified. These set of starting positions determine the groups of consecutive samples which are deleted to simulate the burst of missing data. The strategy, the data set, the block of 77 consecutive weeks, and the burst lengths B were the same as used in [14] in order to compare the evolution of the algorithm performances.

When an algorithm replenishes the missing burst, the Mean Square Error (MSE) per sample with the original data is computed. The same algorithm processes those one thousand different randomly selected cases and the MSE per sample is taken as the parameter to evaluate its performance. Before calculating the MSE, the reconstructed signal is adapted to the sensor resolution of 1% by rounding the values with decimals to the nearest integer the values with decimals. Then, considering $\hat{x}_i$ to be the samples provided by a completion algorithm and x_i to be the true values that had been eliminated in the verified data set to simulate a lost burst of length B, the MSE per sample is computed as:

$$MSE = \frac{1}{B}\sum_{i=1}^{B}\sqrt{(x_i - \hat{x}_i)^2} \tag{6}$$

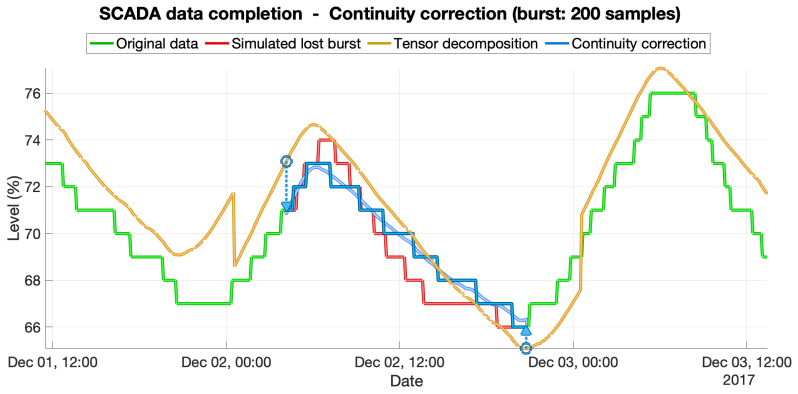

Figure 7. Un example of the continuity correction method. The green line is the original data, x_i. The red line shows the 200 samples burst of simulated lost data. The Tucker decomposition (1,1,1) is used. The orange line is the result of the tensor process with this configuration, $\hat{x}_i$. The blue arrows indicate the initial and the final offsets, O_0 and O_{B+1}. The soft blue line shows the effect of the *continuity correction*, $\tilde{x}_i$ and the strong blue line indicates the final signal reconstruction, which is adapted to the sensor resolution of 1%.

4. Results

In the first part of this section, the study conducted to find the orders of the two decompositions that optimise the MSE per sample is shown. This exploration is performed by completing bursts of known length that have been randomly deleted from the reference database. A test of 1000 simulations is done with 100, and 200 lost samples and using a three and a seven weeks tensors. The results are given in terms of the MSE per sample, according to the exposed methodology.

4.1. Optimal Tensor Decompositions

The same procedure is followed for both CP and Tucker decomposition. The dimensions of the first core (corresponding to the first decomposition) are optimized by executing the first five steps of the algorithm with different sizes of cores and selecting the size that provides the best results in terms of MSE per sample. The complete algorithm is then executed by fixing the first decomposition (with the dimensions found in the first phase of the experiment) and testing different sizes of cores for the second decomposition, thus selecting the one that provides the best results.

4.1.1. CP Case

Since the CP decomposition only depends on one parameter the optimization is simpler than for the case of the Tucker decomposition that depends on three parameters. Still, due to the computational cost of the statistical experiments, the process of searching for the best combination of decomposition size has been approximated in two steps.

The Figure 8 shows the results for the optimal CP decompositions configuration when the length of the bursts are 100 and 200 and for *tensorizations* of 3 and 7 weeks of data. Four cases result from the combination of burst lengths (B) and tensor sizes (w_n).

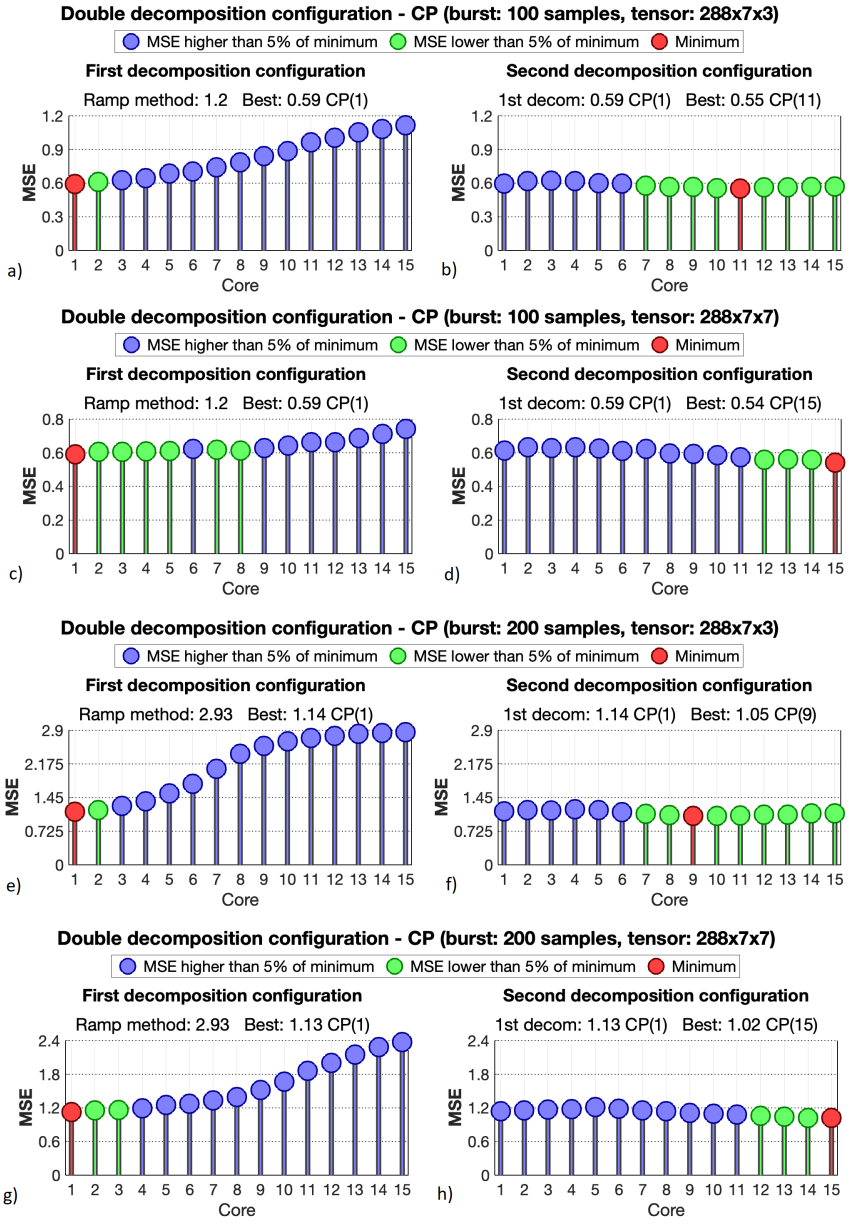

Figure 8. Test of *double decomposition* configuration for the CP model. Each pair of graphics show the result for different conditions of burst length and size of the tensor. (**a**,**c**,**e**,**g**) show the MSE obtained applying only the first decomposition procedure until the bloc number five of the whole algorithm, for different core configurations. (**b**,**d**,**f**,**h**) show the MSE of the *double decomposition method*, the complete algorithm, for different core configurations on the second decomposition, and using the best core configuration obtained for the first decomposition which is CP(1).

To correctly interpret the figure we must bear in mind that the subplots aligned in the left column of Figure 8b show the result of processing only the firsts five blocs of the algorithm described in Figure 3 after computing the MSE of the burst estimated and the parameter D, of CP(D), is swept in a range of values going from 1 to 15. So that, the values obtained in the positions of the burst to be estimated -in this algorithm step- are compared with those of the original burst (which has been eliminated for the experiment) by computing the MSE per sample. The subplots aligned in

the right column of Figure 8 are obtained by applying all the steps of the algorithm described in Figure 3b when the first decomposition is CP(1) and a swept of D going from 1 to 15 is explored for the second decomposition. In all the subplots the minimum MSE obtained is highlighted in red and the configurations that generate values very close to the minimum are highlighted in green.

As important aspects to emphasize, notice that in all the tested conditions of B and w_n, the best decomposition core for the first decomposition is the lowest one, CP(1). Notice also that, in all cases, the second decomposition improves the performance of the algorithms and further reduces the MSE error per sample. If we look at the second decomposition dimensions we see that the performance of the algorithm usually improves when D is higher although the MSE tend to stabilize. In this case, there is no better value for all B and w_n combinations but choosing a high value, for example, D = 15, we will always get results very close to the optimum. Therefore, for the CP method, the choice of the first decomposition core is very robust, and it has to be the lowest one. Then, for the second stage a higher decomposition core has to be selected, taking into account that there is a wide margin of acceptable configurations (values highlighted in green) because when the minimum is reached, the choice of an even higher decomposition core gives a very similar MSE.

4.1.2. Tucker Case

Determining the size of the two decompositions that minimize the MSE per sample, when using Tucker decomposition, is computationally expensive and difficult to visualize. This is because there are more parameters than in the CP model to configure the decomposition core. The number of parameters depends on the tensor structure, and in the 3-dimensions *tensorization* of our problem, implies having three parameters.

In this case, we proceed in a similar way to the previous case; first, we adjust the first decomposition. We evaluate the MSE obtained in the first five steps (blocks) of the algorithm in Figure 3a for a given range of i, j, k corresponding to the decomposition TK(i,j,k). We will obtain a three-dimensional tensor with the MSE values. The i, j, k values that produce the lowest MSE determine the dimension of the first decomposition. Once determined the order of the first decomposition, we recalculate the MSE, now with the complete algorithm, sweeping the same range of values i, j, k for the second decomposition. We will obtain another three-dimensional tensor with the MSE values and we select the i, j, k that produce the minimum MSE.

The experiments carried out include the cases of the burst length 100 and 200 and the *tensorizations* $\chi^{288\times7\times3}$ and $\chi^{288\times7\times7}$, like in the CP case. Figures 9 and 10 show the results for the bursts of 100 and 200 missing samples respectively. Each of these figures shows a graphical representation of the MSE values (which are ordered in a 3-way tensor) for the first and the second Tucker decompositions. For each figure, the first column of graphs represents the results corresponding to the first decomposition. The graphs of the second column show the MSE corresponding to the second decomposition after selecting the combination that gives the lowest value for the first one. In all cases, the red dot represents the configuration with the lowest MSE value and the green points are those configurations with values very close to the minimum. It is noted that the red dots are within the clusters of green points which represent quasi-optimal solutions. This means that there is a whole set of different solutions that behave in a very similar way to the optimal set.

In general, the best option is to select the minimum possible value on the parameter related to the number of weeks for the first decomposition and the maximum for the second one. The other two parameters seem to have more variability, but in general, the parameter relative to the weekday must be high, near the maximum, and the parameter relative to the day hour must be a little lower than it.

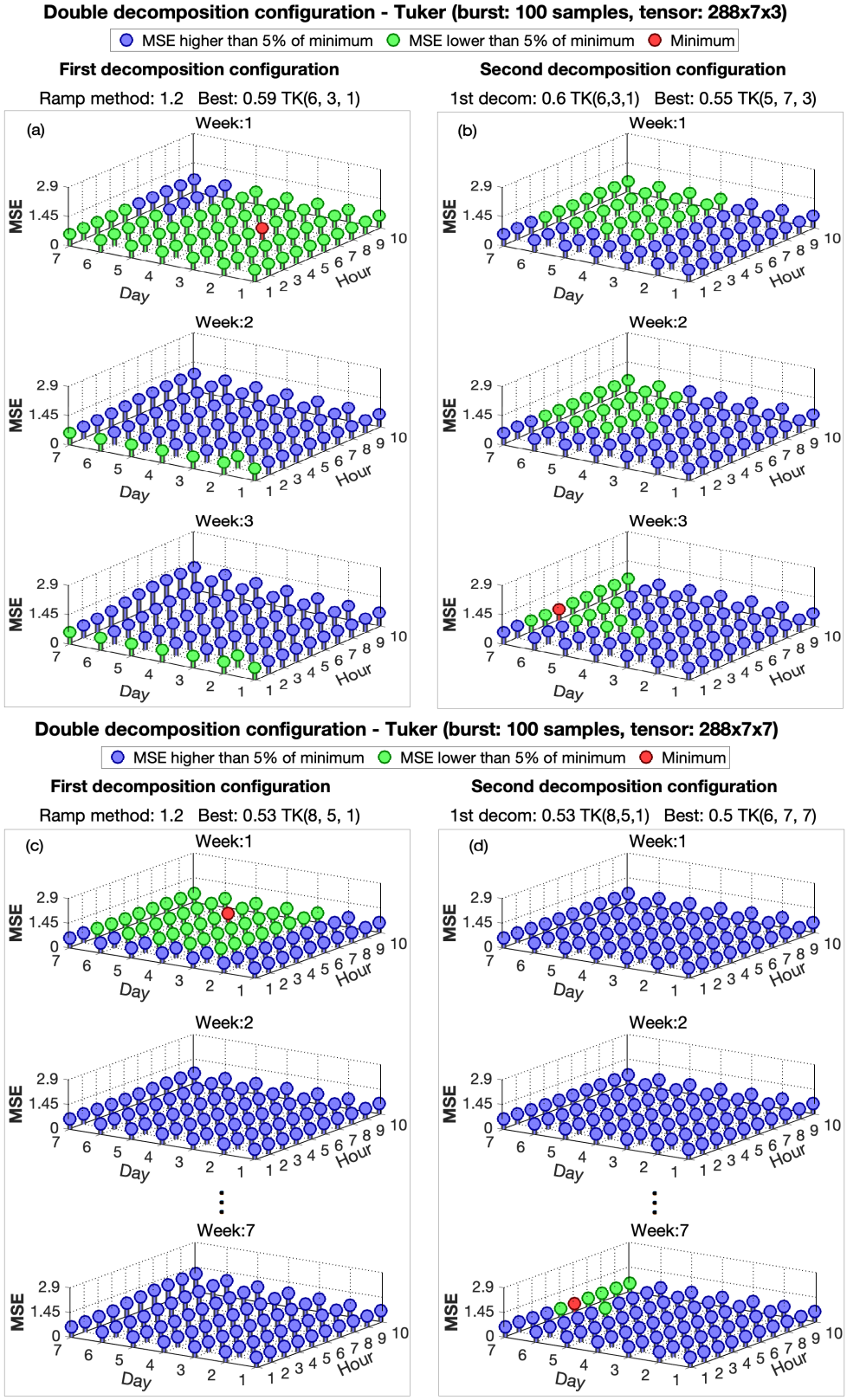

Figure 9. Test of the *double decomposition* configuration with the Tucker(TK) model and a burst of 100 samples. (**a**,**c**) show the MSE of the first decomposition procedure for different core configurations. (**b**,**d**) show the MSE of the second decomposition for different core configurations and using the best core configuration obtained on (**a**) or (**c**) in the first decomposition, respectively.

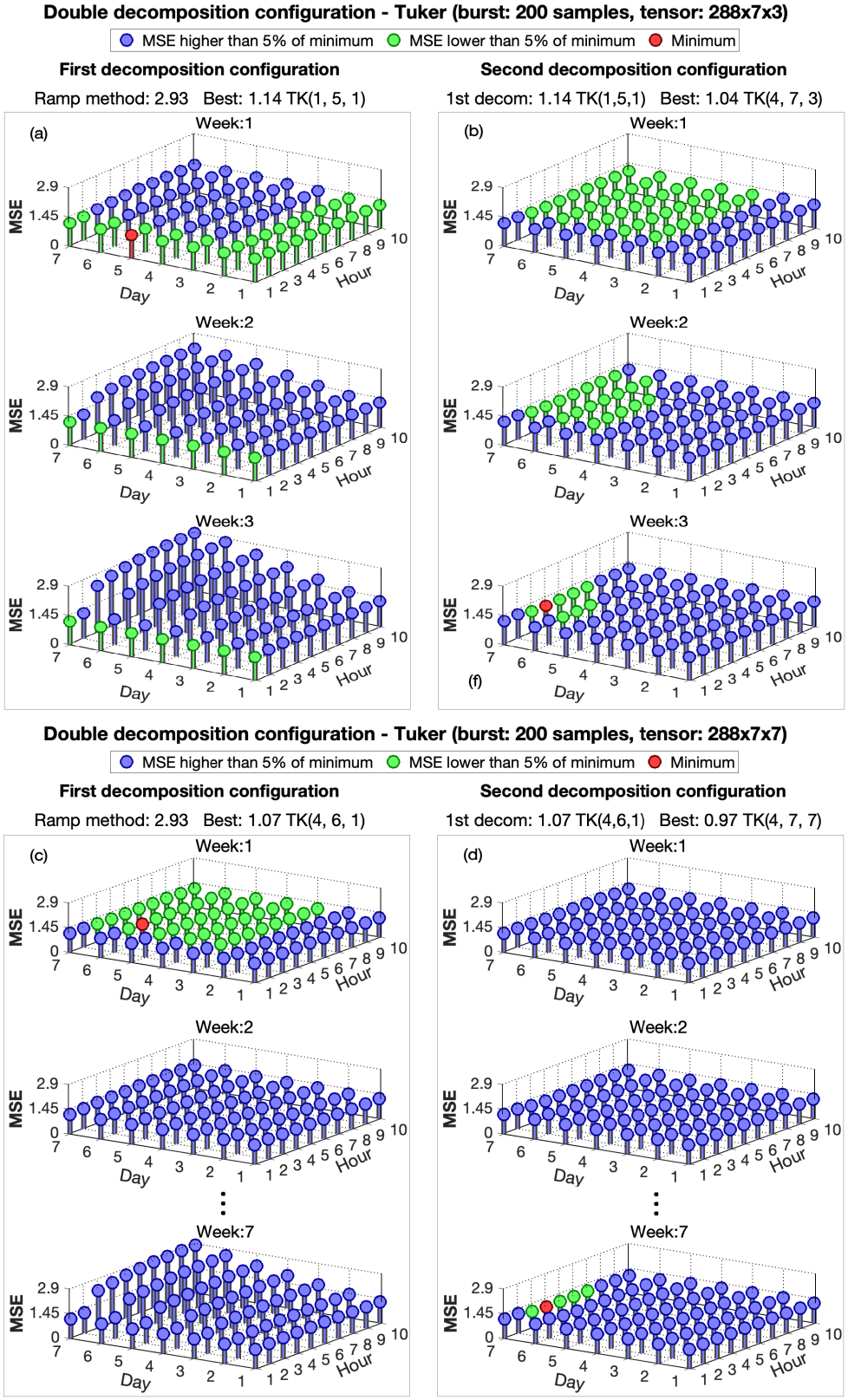

Figure 10. Test of the *double decomposition* configuration with the Tucker model and a burst of 200 samples. (**a**,**c**) show the MSE of the first decomposition procedure for different core configurations. (**b**,**d**) show the MSE of the second decomposition for different core configurations and using the best core configuration obtained on (**a**) or (**c**) in the first decomposition, respectively.

4.2. Quantification of the Different Innovations in the Performance of the Double Decomposition Algorithm

The data completion algorithms proposed in this article have been represented in Figure 3 through a set of blocks. Each of these blocks provides an improvement in the final performance of the algorithm. In this section we compare the effect that a given block has on the final performance and, above all, with the result obtained in [14] that already improved the methods found in the literature when those methods were applied in the problem of recovering data lost in bursts for the context of the problem we are dealing with.

So that, to quantify the effect of the different blocks, the MSE provided by the best configuration of the algorithm proposed in [14], which could be compared with a single tensor decomposition block of our algorithm, is taken as a reference. Therefore, on the base algorithm, different blocks (different improvements) are incorporated, first separately and then combined. The same 1000 simulations are done to see the effect of each block combination. The results are shown in in Figure 11. The best results seem to be achieved with the rearrangement of the tensor using the *burst centered tensorization*, which is the best improvement if it is applied alone. Applying only the *smoothing* process provides a little improvement on any case, not very high but constant for all the tensor and burst sizes checked.

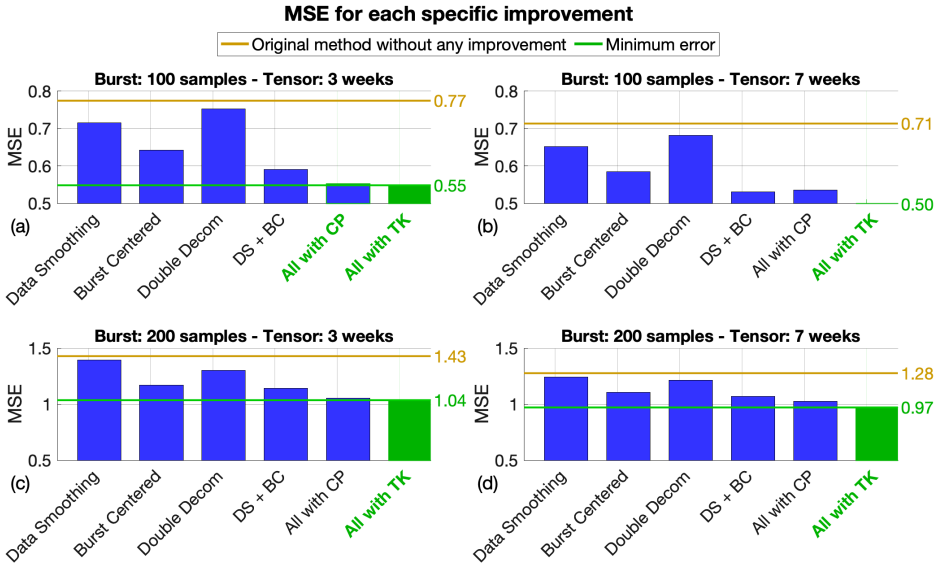

Figure 11. MSE of the proposed improvements. All of them are tested with 100 and 200 samples of data burst lost and with 3 and 7 weeks of *tensorized* data. The orange line indicates the best result obtained in [14]. The *Smooth* is the result of applying only the smooth process to the signal before *tensorizing* it. The *burst centered tensorization* is the result of the rearrangement of the tensor according to the burst location. The *Double decom* is the result of applying the decomposition two times, with $G^{4\times6\times1}$ and $G^{4\times7\times7}$ for Tucker, and $G^{1\times1\times1}$ and $G^{15\times15\times15}$ for CP. The *DS-Bc* is the result of combining the data smoothing and the *burst centered tensorization* without using the double decomposition. The *All with CP* and the *All with TK* are the results of applying all the proposed algorithm with the CP and the Tucker models respectively.

A curious result of the *double decomposition* is that it seems to have, proportionally, a more positive effect when it is used in combination with the other options. This can be seen by comparing the MSE reduction obtained by applying only the *double decomposition* compared to using a combination of smoothing and *burst centered tensorization* or using all the improvements, especially with the Tucker model results. With any size of tensor and burst, the effect of each option is complementary to the others, which means that applying all of the improvements together provides a considerable positive impact in comparison to not using any of them in all the cases. Note that using different tensor sizes or to restoring bursts of different lengths results in a different optimal configuration of the decomposition core, although with similar characteristics (See Figures 9 and 10).

The study carried out on the optimization of the size of decompositions allows us to find the configuration that produces the minimum MSE value that our algorithm will make on our database for a certain combination of B and w_n values. The results remain very similar when parameters that have been optimal in one configuration of B and w_n are used in another. Table 1 also shows the consistency of algorithm results with changes in the size of the decomposition, even when combining the CP and Tucker models as well. We verify that the MSE obtained are very close to the expected optimal ones.

We emphasise that when the algorithm uses Tucker models, better results are obtained than with CD models. However, the measurable advantage in terms of MSE is minimal so that the use of CD is still reasonable because it is easier to interpret and robust to configure.

In this table, one can get an idea of the differences that the proposed algorithm obtain in terms of MSE when the sizes and types of tensor decompositions are combined and/or different tensorizations sizes are used. Above all, it is interesting to compare the difference achieved with respect to the best values obtained in [14], which in all cases are very significant.

Table 1. MSE of the different tested methods. The results of 100 and 200 lost samples, B, are shown working with a 3 and 7 weeks of tensor size, n_w. "The best configurations in [14]" show the minimum MSE obtained for the algorithm presented in [14], with the CP and the Tucker models. "The best configurations for the proposed algorithm" show the minimum MSE obtained for different pairs of decompositions. In these cases, only the core of the first decomposition is fixed, and it is shown the minimum MSE obtained with the best core for the second decomposition.

MSE per Sample	$B = 100$ $n_w = 3$	$B = 100$ $n_w = 7$	$B = 200$ $n_w = 3$	$B = 200$ $n_w = 7$
The best configurations in [14]				
optimal CP	0.87	0.80	1.70	1.58
optimal TK	0.77	0.71	1.43	1.28
The best configurations for the proposed algorithm				
1st decom: CP(1), 2nd decom: optimal CP	0.55	0.54	1.05	1.03
1st decom: TK(6,3,1), 2nd decom: optimal CP	0.57	0.52	1.14	1.02
1st decom: TK(8,5,1), 2nd decom: optimal CP	0.57	0.53	1.14	1.03
1st decom: TK(1,5,1), 2nd decom: optimal CP	0.55	0.53	1.06	1.02
1st decom: TK(4,6,1), 2nd decom: optimal CP	0.56	0.53	1.06	1.02
1st decom: CP(1), 2nd decom: optimal TK	0.54	0.52	1.04	1.00
1st decom: TK(6,3,1), 2nd decom: optimal TK	0.55	0.51	1.11	0.98
1st decom: TK(8,5,1), 2nd decom: optimal TK	0.54	0.50	1.11	0.97
1st decom: TK(1,5,1), 2nd decom: optimal TK	0.53	0.52	1.04	1.00
1st decom: TK(4,6,1), 2nd decom: optimal TK	0.55	0.50	1.11	0.97

5. Conclusions

Completing data lost in bursts remains a difficult challenge and is where most data completion methods fail. However, data being lost in bursts is quite common. It is often associated with the failure of a component involved in capturing or transmitting the data. In the contribution of [14] it is presented an *ad hoc* data completion method to recover data lost in bursts that outperforms the methods available in the literature for the proposed application. This work significantly improves the method in [14], which is taken as a reference for the new algorithm evaluation and for comparison purposes. The incorporated novelties are a *smoothing* process, a new *tensorization* we have called *burst centered tensorization*, and the application of two tensor decomposition, one after the other using two different cores.

It is difficult to evaluate a data completion method in a practical framework. To do this in this article an experiment was carried out using only verified sets of samples from the historical data of the level sensor, avoiding to use on the experiments the weeks where there were real lost samples. The set of verified data is used, randomly erasing a burst of data and later applying the data reconstruction (completion) method. The error of the reconstruction method can be calculated because the original

data is known. To compare statistically the tested methods the same 1000 simulated lost bursts are restored with each of them. Then the MSE per sample, which is the mean of the error generated with this 1000 simulations, is calculated for each method. By this way the results of the tested methods can be compared in the same conditions.

It is important to emphasize that comparing the MSEs obtained in the same database and the same distribution and lengths of lost frames, the proposed method reduces the MSE between 24% and 40%, the best results obtained in [14]. Note in Figure 11 that the MSE corresponding to a burst of 100 samples falls from 0.71, the best result obtained in [14], to 0.50, the best result obtained with the new methodology. That means a reduction of the MSE approximately of the 39.5%. And in the case of the burst of 200 samples the MSE falls from 1.28 to 0.97 which is approximately a 24.2% of reduction. Note that the best results in [14] were obtained using an imputation method based on two linear predictors that are difficult to adjust since they require to estimate the length of two FIR filters and, besides, it must be continuously adapted. Introducing a double tensor decomposition in the proposed method gives stability and robustness. From the studies carried out, we see that the first tensor decomposition must have a small core that keeps only the most relevant interactions between modes, while the second decomposition must have a larger core to capture the interactions between modes in greater detail. A signal reconstruction example of this procedure is shown in Figure 12 using $G^{4\times6\times1}$ and $G^{4\times7\times7}$, the best core configurations for the double decomposition according to the tests for the Tucker model in the case of a 200 samples of burst length.

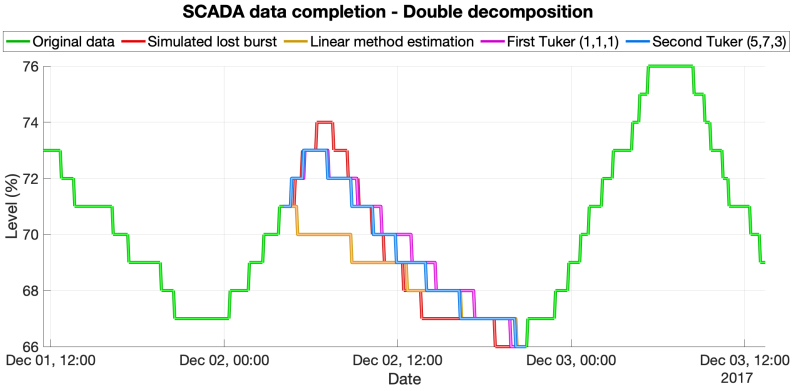

Figure 12. Example of the reconstruction methodology with double decomposition. The green line shows the original data and the red line the burst of lost samples. The orange line is the linear estimation with the *ramp method*. The purple line is the result of the first *tensorization* procedure with Tucker using $G^{4\times6\times1}$. The blue line shows the result of the second *tensorization* procedure with $G^{4\times7\times7}$.

The rest of the results are quite expected. If we use more data for the restoration, i.e., when we use seven-week rather than three-week tensors, we obtain lower MSE in the tests, although the percentage reductions are small and tend to stabilise with the increment of new data. It is also expected that when lost bursts are long, 200 samples in our experiments, the tests will provide higher MSE than when shorter bursts of 100 samples. The experiments also indicate that it makes more sense to use 7-week rather than 3-week *tensorizations* when the lost bursts are long (200 samples) since the differential obtained in terms of MSE is more significant than in the case of shorter bursts (100 samples), which could seem very small.

As also pointed in the introduction, the presented algorithm can be exportable to complete data of other types of problems, especially to those that the data capture patterns related to human activity because in these cases, there are interactions at different time levels, be it timetable, daily, weekly, seasonal, etc. The gas or electricity networks also fall into this category. However, it must keep in mind that when the algorithm will be used in another context it will probably be necessary to calculate the optimal dimensions of the decompositions. Finally, before developing models to monitor plants, enabling predictive maintenance or discovering failures at a very early stage, it is necessary to have

reliable data. Tensor algebra is a helpful tool in completing data and for sure it will also be very useful in the development of predictive maintenance models.

Author Contributions: Conceptualization, P.M.-P. and A.M.-S.; methodology, P.M.-P., M.S.-S. and A.M.-S.; software, P.M.-P. and A.M.-S.; validation, P.M.-P., M.S.-S. and A.M.-S.; formal analysis, P.M.-P., M.S.-S. and A.M.-S.; investigation, P.M.-P. and A.M.-S.; resources, P.M.-P. and M.S.-S.; writing–original draft preparation, P.M.-P. and A.M.-S.; writing–review and editing, P.M.-P. and A.M.-S.; supervision, P.M.-P. and M.S.-S.; funding acquisition, P.M.-P. and M.S.-S. All authors have read and agreeed to the published version of the manuscript.

Funding: This work was partially supported by the Spanish Government project PHENOFISH CTM2015-69126-C2-2-R.

Acknowledgments: We thank the company Aigües de Vic S.A. for giving us access to their databases to perform this research.

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

References

1. Langhammer, J.; Česák, J. Applicability of a Nu-Support Vector Regression Model for the Completion of Missing Data in Hydrological Time Series. *Water* **2016**, *8*, 560. [CrossRef]
2. Ahlheim, M.; Frör, O.; Luo, J.; Pelz, S.; Jiang, T. Towards a Comprehensive Valuation of Water Management Projects When Data Availability Is Incomplete—The Use of Benefit Transfer Techniques. *Water* **2015**, *7*, 2472–2493. [CrossRef]
3. Zhao, Q.; Zhu, Y.; Wan, D.; Yu, Y.; Cheng, X. Research on the Data-Driven Quality Control Method of Hydrological Time Series Data. *Water* **2018**, *10*, 1712. [CrossRef]
4. Ekeu-Wei, I.T.; Blackburn, G.A.; Pedruco, P. Infilling Missing Data in Hydrology: Solutions Using Satellite Radar Altimetry and Multiple Imputation for Data-Sparse Regions. *Water* **2018**, *10*, 1483. [CrossRef]
5. Blanch, J.; Puig, V.; Saludes, J.; Quevedo, J. Arima models for data consistency of flowmeters in water distribution networks. *IFAC Proc. Vol.* **2009**, *42*, 480–485. [CrossRef]
6. Lamrini, B.; Lakhal, E.K.; Le Lann, M.V.; Wehenkel, L. Data validation and missing data reconstruction using self-organizing map for water treatment. *Neural Comput. Appl.* **2011**, *20*, 575–588. [CrossRef]
7. Puig, V.; Ocampo-Martinez, C.; Pérez, R.; Cembrano, G.; Quevedo, J.; Escobet, T. *Real-Time Monitoring and Operational Control of Drinking-Water Systems*; Springer: Berlin, Germany, 2017.
8. Acar, E.; Dunlavy, D.M.; Kolda, T.G.; Mørup, M. Scalable tensor factorizations for incomplete data. *Chemom. Intell. Lab. Syst.* **2011**, *106*, 41–56. [CrossRef]
9. Signoretto, M.; Van de Plas, R.; De Moor, B.; Suykens, J.A. Tensor versus matrix completion: A comparison with application to spectral data. *IEEE Signal Process. Lett.* **2011**, *18*, 403. [CrossRef]
10. Mørup, M. Applications of tensor (multiway array) factorizations and decompositions in data mining. *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.* **2011**, *1*, 24–40. [CrossRef]
11. Kolda, T.G.; Bader, B.W. Tensor decompositions and applications. *SIAM Rev.* **2009**, *51*, 455–500. [CrossRef]
12. Cichocki, A.; Mandic, D.; De Lathauwer, L.; Zhou, G.; Zhao, Q.; Caiafa, C.; Phan, H.A. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag.* **2015**, *32*, 145–163. [CrossRef]
13. Comon, P. Tensors: a brief introduction. *IEEE Signal Process. Mag.* **2014**, *31*, 44–53. [CrossRef]
14. Marti-Puig, P.; Martí-Sarri, A.; Serra-Serra, M. Different Approaches to SCADA Data Completion in Water Networks. *Water* **2019**, *11*, 1023. [CrossRef]
15. Tucker, L.R. Some mathematical notes on three-mode factor analysis. *Psychometrika* **1966**, *31*, 279–311. [CrossRef] [PubMed]
16. Carroll, J.D.; Chang, J.J. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika* **1970**, *35*, 283–319. [CrossRef]
17. De Lathauwer, L.; De Moor, B.; Vandewalle, J. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.* **2000**, *21*, 1253–1278. [CrossRef]
18. Harshman, R.A. *Foundations of the PARAFAC Procedure: Models and Conditions for an "Explanatory" Multimodal fActor Analysis*; UCLA Working Papers in Phonetics; UCLA: Los Angeles, CA, USA, 1970; Volume 16, pp. 1–84.

19. Sørensen, M.; Lathauwer, L.D.; Comon, P.; Icart, S.; Deneire, L. Canonical polyadic decomposition with a columnwise orthonormal factor matrix. *SIAM J. Matrix Anal. Appl.* **2012**, *33*, 1190–1213. [CrossRef]
20. Sidiropoulos, N.D.; De Lathauwer, L.; Fu, X.; Huang, K.; Papalexakis, E.E.; Faloutsos, C. Tensor decomposition for signal processing and machine learning. *IEEE Trans. Signal Process.* **2017**, *65*, 3551–3582. [CrossRef]
21. Kolda, T.G. *Multilinear Operators for Higher-Order Decompositions*; Technical report; Sandia National Laboratories: Albuquerque, NM, USA, 2006.