Imputing missing groundwater observations

Achiya Dax and Michael Zilberbrand

ABSTRACT

In this paper, we consider the problem of completing missing records of annual water levels. The water levels records are taken once a year, in a group of neighboring wells. The collected records are assembled into a data matrix, where each column refers to a different well and each row refers to a different year. Yet some entries of the matrix are unknown and we want to assign appropriate values to these entries. The need for solving such problems arises in many applications, as many models and programs require a complete set of data. Traditional approaches for handling missing groundwater records are based on statistical techniques for treating missing data. The current paper introduces a new approach for solving this problem. One that is able to take advantage of the 'matrix structure' of annual water levels. This type of 'matrix imputing methods' has been proved successful in many modern areas, but it has not yet been tested in hydrology. Special attention is given to the question of assessing the quality of the imputed water levels. The proposed methods are examined on a number of test cases.

Key words | annual water levels, groundwater, matrix imputing methods, missing records

Achiya Dax (corresponding author)
Michael Zilberbrand
Hydrological Service,
P.O.B. 36118,
Jerusalem 91360,
Israel
E-mail: *dax20@water.gov.il*

INTRODUCTION

The paper considers a network of observation wells which are used for monitoring the hydrologic situation in a certain aquifer. The water levels which are measured in the wells form the basis of regular reports, numerical modeling, water resources assessments, and various research projects. Many of these applications are based on annual records of data. That is, data records which are collected once a year, at about the same time. In Israel, for example, each year is composed of two seasons, a rainy one and a dry one, and the annual records of water levels are measured in October, at the end of the dry season. The term water level refers to the height of the water table at a well. Assume that annual records of water levels were taken during m years, in a group of n neighboring wells. Then the collected data form an m x n matrix, in which each row refers to a different year, and each column refers to a different well. Let A denote the resulting data matrix, and let a_{ij} denote the (i,j) entry of A. Then a_{ij} gives the water level measured at the ith year in the jth well, $i = 1, \ldots, m$, $j = 1, \ldots, n$ (see Table 1). Most reports and models require a complete data matrix as input. That is, a matrix without missing entries. Thus, even one missing entry may prevent us from using the collected data. However, as explained below, in practice it is quite common that some entries of the matrix are missing. In this case it is desired to assign 'appropriate values' to the missing entries. Once all the missing water levels are imputed, the data matrix can be used for any purpose.

The task of imputing missing values is, therefore, to design an algorithm that substitutes appropriate values into the missing entries of a given m x n data matrix, $A = (a_{ij})$. Other common names of this task are 'missing values estimation', 'completing missing entries', 'matrix completion', 'recovering', 'reconstructing', etc. The need for completing missing entries of a data matrix is a well known problem that arises in many applications. Many ingenious algorithms have been proposed and there is vast literature on this issue. In particular, the statistical treatment of missing data is a relatively matured area which preceded

doi: 10.2166/nh.2017.220

most of the other fields, e.g., Afifi & Elashoff (1966) or Dempster *et al.* (1977). Other so-called 'traditional' fields include business reports, psychometrica, biology and meteorology. Modern applications arise in machine learning, data mining, DNA microarray data, computer vision, recommender systems, and collaborative filtering. See Dax (2014) for a recent comprehensive survey of imputing methods. Most of the methods consider the imputing problem within the context of a specific application. The current research concentrates on imputing methods which are able to take advantage of the special features that characterize annual records of water levels.

There are several reasons for having missing records of water levels. One comes from human factors such as shortage of manpower, illness, strikes, typing errors, etc. A second stems from (unexpected) technical difficulties, such as technical faults in the measuring equipment, blocked roads, cars breakdown, etc. Another difficulty comes from the fact that water levels should be measured in a rest situation. This means that pumping from the well should be stopped a number of hours before taking the measurement. Yet in practice this is not always possible. Otherwise, when the stop is too short, the measured water level provides erroneous information. Hence, the recorded water levels are always examined and searched for outliers and other types of suspicious records. Once we find an erroneous data we delete it. Then it is necessary to substitute a 'correct' value instead of the wrong one.

The requirement for a complete set of data is typical to several applications. Consider, for example, the plotting of a contour map of the water levels surface. Recall that contour maps provide a useful tool for studying the behavior of an aquifer. In particular, a contour map enables us to draw stream lines, to compute gradients, and to detect the main flow directions. A comparison of two consecutive contour maps enables us to see how the groundwater surface changes from one year to another, and allows us to compute the relevant changes in the storage volume of the aquifer. Nowadays contour mapping is often done by a computerized plotting system whose input is a list of triplets, where each triplet contains the coordinates and the water level of one well. If one well is located far from the other wells then the water level in this well has a great effect on the shape of the computed water table surface. Deleting the well from the input list is likely, therefore, to cause a severe distortion to the shape of the computed surface. Similar distortions may occur when the missing water level is considerably higher (or lower) than the other water levels.

A further example is the case when the data are needed for the use of numerical models such as finite difference or finite element models. Here the observed water levels are used in the calibration process, to test the quality of the proposed model, and for setting initial and boundary conditions. In this case it is often necessary to transfer the data collected on a network of observation wells into a mesh of cells or elements which constitute the model. Usually each cell (or element) is associated with a small number of neighboring wells, and the water level at the cell is defined as a certain weighted average of the water levels in these wells. This averaging operation is often done by some computer program which does not take into account the effect of missing observations. So missing data are likely to cause either a breakdown or serious malformation to the output.

Missing water level observations are expected to cause similar difficulties in almost any monitoring system that is based on a permanent network of observation wells. The number of observation wells that constitute such a network can be quite large. For example, the Hydrological Service of Israel stores and updates records of over 6000 wells, which are used for monitoring the hydrological regime in several aquifers. In the past the majority of these wells were measured monthly. But twenty years ago the monitoring policy has been changed, and since then in most of the wells water levels are measured only twice a year: Once in the spring and once in the autumn. (However, a small percentage of important wells are still monitored on a monthly basis.) The reasons for this change of policy are two: first, a lack of manpower and budget; second, the belief that reports and models that are based on annual records of data are satisfactory for most purposes. (A similar policy is used in sampling concentrations of chloride and nitrate in wells. These samples are taken once a year, during the dry season.)

Note that there is substantial difference between the imputing of monthly records of water levels and the imputing of annual records. Assume for a moment that we have monthly records of water levels in a certain well, which were measured during m consecutive years, but contain a

small percentage of missing records. Recall that monthly collected water levels are expected to show clear seasonal behavior, i.e. increasing during the rainy season and decreasing during the dry season. The seasonal pattern, and the fact that we have nearly m x 12 records per well, enable us to complete the missing records in a certain well by relying only on its own records (Dax 1985, and references therein). However, when having (at most) one record per year, the information gathered in one well is not sufficient for reliable imputing of the missing records. Consequently we are forced to rely on relevant information from neighboring wells, while the neighboring wells may also suffer from missing records. This raises the need for solving the above matrix completion problem.

The traditional treatment of missing data in hydrology is based on statistical methods. For example, autoregressive techniques for time series, or geostatistical techniques, like kriging, for spatial data. Nowadays statistical computing packages, like SAS, SPSS or R include built-in programs that achieve imputing. (For example, R users can choose between *Amelia*, *Mice*, and *mitools*.) Hence it is always tempting to use an off-the-shelf package. Indeed, in some cases we obtain reasonably good results. Yet, as explained in the next section, the statistical approach has certain limitations.

In this paper we propose a different approach. A 'matrix imputing' approach that takes advantage of matrix properties that can be derived from the matrix at hand. This approach has been proved very successful in several modern fields, such as machine learning, data mining, DNA microarrays data, computer vision, recommender systems, and collaborative filtering. However, as far as we know, it has not yet been tested in hydrology. One aim of this paper is, therefore, to introduce this approach, to explain the main ideas behind the method, and to illustrate how it works. This is achieved by describing the leading algorithms that characterize this method. Numerical experiments demonstrate the usefulness of this approach in the context of missing groundwater data.

The new approach achieves imputing by constructing a low-rank approximation of A. Let k denote the rank of the desired approximation, and let

$$\mathbf{B_k} = \{B | B \in \mathbf{R}^{m\,x\,n}, \text{rank } (B) \leq k\}$$

denote the set of all real m x n matrices, $B = (b_{ij})$, whose rank is at most k. (The nature of such matrices is clarified in the coming sections.) Then the constructed rank-k approximation is often aimed at solving the least-squares problem

$$\textbf{minimize } F(B) = \Sigma(a_{ij} - b_{ij})^2 \tag{1a}$$

$$\textbf{subject to } B \in \mathbf{B_k}, \tag{1b}$$

where the sum in (1b) is restricted to the set of known entries

$$\Omega = \{(i, j) | a_{ij} \text{ is known}\}. \tag{2}$$

Let the matrix B_k denote a computed solution of Problem (1). Then, once B_k is computed, the missing entries of A are replaced with the corresponding entries of B_k.

The implementation of the above approach raises two issues. First we need an effective algorithm for computing a rank-k approximation. Second, we need to find a suitable value of k. The last problem is closely related to the question of how to determine the number of principle components (principle factors) of a data matrix, e.g., Howard & Gordon (1963), Jackson (1993), Peres-Neto *et al.* (2005) and Ledesma & Valero-Mora (2007). Yet, when A has missing entries we face a further difficulty: As k increases behind a certain threshold, the number of degrees of freedom in Problem (1) exceeds the number of known entries. At this point the solution of Problem (1) loses its uniqueness and the quality of the imputed values starts to deteriorate. A further aim of the paper is to illustrate this phenomenon and to outline an effective solution method.

The solution proposed in this paper is based on three innovations. The first one is an improved building process that generates a sequence of matrices, B_1, B_2, …, B_k, …, where B_k is a rank-k matrix that solves Problem (1). The building process is based on a new minimization procedure that uses B_k as a starting point for the computation of B_{k+1}. This results in a considerable saving of computing efforts. The second innovation regards the construction of a 'virtual objective function' that measures the quality of the imputed entries. This enables us to compute a sequence of positive

numbers, V_1, V_2, …, V_k, …, where V_k reflects the ability of B_k to estimate the missing entries in A. The smaller is V_k, the better imputing we have. The third innovation is an effective rule for choosing an optimal value of k. As we shall see, the virtual objective function decreases in the first steps. Yet, as k increases behind a certain point, the values of V_k starts to increase. The optimal imputing is obtained, therefore, for the smallest value of V_k.

The plan of the paper is as follows. The Methodology section is divided into three parts. The first part explains the main differences between the traditional statistical approach and the current matrix approach. The second part outlines some of the leading matrix algorithms. There are several ways to compute a rank-k approximation. This can be done by direct solution of Problem (1), or by solving a closely related problem. In our review the focus is on methods that are most suitable for groundwater data. Starting from averaging methods we describe some basic imputing algorithms, including Iterative Column Regression (ICR), k – Nearest Neighbor (KNN) impute, and iterative Singular Value Decomposition (SVD) impute. Then we move to consider recently proposed methods such as rank minimization, and nuclear norm minimization. The third part of the Methodology section introduces a new matrix method that combines the three innovations that we mentioned above. The basic building block of the new method is a simple algorithm that computes a rank-one approximation of matrix. The usefulness of the proposed scheme is illustrated on a number of typical test cases.

METHODOLOGY

Statistical imputing of missing values

The analysis of sample surveys is a major task in statistics. The problem of nonresponse in surveys is an inherent difficulty that hinders users from achieving accurate analysis. The statistical treatment of missing data is motivated, therefore, by the need to solve this problem, which leads to several differences between the statistical approach and the proposed matrix approach.

One feature that characterizes statistical imputing is that the methods rely on statistical assumptions on the origin of the data and the nature of the missing entries. For example, in maximum likelihood expectation-maximization (EM) methods, and in multiple imputing methods, it is common to assume that the entries of the matrix are generated by some known probabilistic distribution function (such as multivariate normal distribution). In contrast, matrix methods make no assumptions of this kind. Moreover, annual water levels are not expected to obey such assumptions.

Another type of assumption regards the missingness mechanism that determines the locations of the missing entries. Here it is common to distinguish between 'not missing at random', 'missing at random', and 'missing completely at random' (Dempster *et al.* 1977; Little & Rubin 1987; Allison 2009). Again, in matrix methods there is no need in such characterization, while missing water levels are not expected to obey these classifications.

The main difference between the statistical approach and the matrix approach lies in their goals.

Matrix methods have one aim: To provide 'suitable' values for the missing entries. In contrast, the ultimate aim of statistical analysis is to produce reliable estimates of statistical parameters that characterize the data, such as means, variances, correlations, covariances, principle components, and so on. If this task can be satisfied without assigning values to missing entries, then the analysis is carried out without achieving imputing. Consequently, 'listwise deletion' is the default option in most of the statistical software packages. In matrix terminology, listwise deletion means that we simply delete any row with missing entries. Clearly, this is not a reasonable solution for our problem. Yet the simplicity of this method makes it a popular option in many statistical applications. For example, as reported in Tsikriktsis (2005), an inspection of 103 research articles shows that listwise deletion is heavily used in operations management surveys. However, listwise deletion may lead to inaccurate estimates of statistical parameters, especially when the data fail to satisfy the 'missing completely at random' assumption. Thus, from the statistical point of view, the reason for conducting imputation is to gain improved estimates of the desired parameters. Moreover, in many methods the computed values are not guaranteed to help in other applications. For example, Allison (2009) gives the following warning about the use of imputed

values that are obtained in the maximum likelihood method:

> '*The principle output from this algorithm is the set of maximum likelihood estimates of the means, variances, and covariances. Although imputed values are generated as part of the estimation process, it is not recommended that these values be used in any other analysis*.'

In statistical terminology, methods like mean substitution, hot-deck imputation, and EM, are classified as single imputation methods, since here each missing entry is assigned a single value. These methods have certain advantages over listwise deletion, but are liable to suffer from certain drawbacks. For example, mean substitution is likely to produce smaller estimates of the variance. The multiple-imputing approach is designed to avoid biased estimates (Little & Rubin 1987; Rubin 1987). In this method the values of the missing entries are randomly drawn from a given probabilistic distribution. Usually a multivariate normal distribution which is supplied by the user. Once all the missing entries in the matrix attain values, the algorithm computes the desired set of statistical parameters. This process of random draws and analysis is repeated ℓ times, generating ℓ matrices with ℓ sets of parameters. Typically ℓ is a small integer between 3 and 10. Then the final estimates of the parameters are obtained by taking means over the ℓ sets. This process generates an improved set of statistical parameters, as well as some indication on the accuracy of these parameters. Similarly, values for the missing entries can be obtained by averaging the corresponding entries in the ℓ matrices. However, as noted above, the assumption that water levels behave like random draws is somewhat doubtful.

A review of existing matrix imputing methods

Below we describe some popular algorithms and explain the main ideas behind these methods. The review is not extensive and many good methods have been left unmentioned. It is aimed to serve as a short introduction into the concept of matrix-imputing methods, with a focus on methods that are suitable for handling annual records of water levels.

All the methods share the same aim: To complete the missing entries of a given m x n data matrix, $A = (a_{ij})$. The description is based on standard terms and techniques of numerical computing. No assumptions are made on the statistical nature of the observed data.

Averaging methods

Let α denote the average value of the known entries in A. The simplest way of imputing is, perhaps, substituting each missing entry of A by α. This mean-imputing method is useful in handling random matrices, whose entries are considered as random numbers that obey the same probabilistic distribution. However, the mean-imputing method is not suitable for groundwater data. Assume, for example, that a certain row of A refers to a rainy year. Then the (missing) water levels in this row are expected to exceed α. Similarly, consider the case when certain column of A refers to a well whose water levels are considerably lower than those of the other wells. Then the (missing) water levels in this column are expected to stay below the overall average.

The above examples suggest that the basic mean imputing method can be modified in a number of ways. In 'row-averaging' the missing entries in a certain row are replaced by the mean value of the known entries in this row. Similarly, in 'column-averaging' the missing entries in a certain column are replaced by the mean value of the known entries in this column. Combining the two methods gives an 'additive model' of the form

$$a_{ij} = y_i + w_j + \varepsilon_{ij}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n, \tag{3}$$

where w_j denotes the average water level at the jth well, y_i reflects the changes during the ith year, and ε_{ij} denotes the model error. The parameters of the additive model are easily found by solving the least-squares problem

$$\textbf{minimize } f(y_1, \ldots, y_m, w_1, \ldots, w_n) = \Sigma(a_{ij} - y_i - w_j)^2, \tag{4}$$

where the sum is restricted to known entries of A. See Dax (1985) for details. The additive model is often modified in the form

$$a_{ij} = \alpha + y_i + w_j + \varepsilon_{ij}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n, \tag{5}$$

where α denotes the overall average.

In some cases, a 'multiplicative' model is preferred. In this approach

$$a_{ij} = y_i w_j + \varepsilon_{ij}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n, \tag{6}$$

where here y_i can be interpreted as the intensity of rain during the ith year. The parameters are found by solving the related least-squares problem

$$\textbf{minimize } f(y_1, \ldots, y_m, w_1, \ldots, w_n) = \Sigma(a_{ij} - y_i w_j)^2, \tag{7}$$

where the sum is restricted to known entries of A. An algorithm for solving (7) is described later in this paper. Once the model parameters are computed, the unknown entries of A attain the predicted values.

Let $\mathbf{y} = (y_1, \ldots, y_m)^{\mathbf{T}}$ and $\mathbf{w} = (w_1, \ldots, w_n)^{\mathbf{T}}$ denote the corresponding vectors of unknowns. Let $\mathbf{e} = (1, 1, \ldots, 1)^{\mathrm{T}}$ denote a vector of ones whose length is either m or n, depending on the context. Recall that $\mathbf{w}$ denotes a column vector, $\mathbf{w}^{\mathbf{T}} = (w_1, \ldots, w_n)$ denotes a row vector, and the product $\mathbf{y}\ \mathbf{w}^{\mathbf{T}}$ defines an m x n rank-one matrix whose (i,j) entry equals $y_i\, w_j$, $i = 1, \ldots, m$, $j = 1, \ldots, n$. With these notations at hand the six averaging schemes mentioned above are essentially low-rank approximations of A regarding the matrices $\alpha\ \mathbf{e}\ \mathbf{e}^{\mathbf{T}}$, $\mathbf{y}\ \mathbf{e}^{\mathbf{T}}$, $\mathbf{e}\ \mathbf{w}^{\mathbf{T}}$, $\mathbf{y}\ \mathbf{e}^{\mathbf{T}} + \mathbf{e}\ \mathbf{w}^{\mathbf{T}}$, $\mathbf{y}\ \mathbf{e}^{\mathbf{T}} + \mathbf{e}\ \mathbf{w}^{\mathbf{T}} + \alpha\ \mathbf{e}\ \mathbf{e}^{\mathbf{T}}$, and $\mathbf{y}\ \mathbf{w}^{\mathbf{T}}$, respectively. These observations suggest that higher rank approximations are likely to produce smaller model errors. Yet, averaging schemes are often used to compute starting values for the more sophisticated methods that are described below.

Iterative column regression (ICR)

This method is most suitable to cases where the number of years, m, is considerably larger than the number of wells, n. As its name says, ICR is an iterative algorithm. It starts by setting initial trial values to the missing entries of A. Then these values are updated during the iterative process. The basic iteration is composed of n steps, where the jth step updates the values of the missing entries in the jth column, $j = 1, \ldots, n$. This is achieved by regressing the jth column against the other columns of A. The details are given below.

The jth step of the basic iteration, $j = 1, \ldots, n$.

(a) Let $\tilde{A}$ denote the current m x n matrix at the beginning of the jth step.

(b) Let the m x (n – 1) matrix A_j be obtained from $\tilde{A}$ by deleting the jth column of $\tilde{A}$.

(c) Let the m–vector $\mathbf{b}$ denote the jth column of $\tilde{A}$.

(d) Compute $\mathbf{x}^*$, the minimum norm solution of the linear least-squares problem

$$\textbf{minimize } ||A_j\, \mathbf{x} - \mathbf{b}||_2^2.$$

(e) Compute the m-vector $\mathbf{d} = A_j\, \mathbf{x}^*$.

(f) Set new trial values to the missing entries in the jth column of A. The new values are those of the corresponding entries in $\mathbf{d}$.

In statistician's eyes the above algorithm can be viewed as descendant of the statistical EM approach. Yet it is not clear whether the iterative process always converges, nor what properties a limit point has. Another related question is how the initial trial values of the unknown entries affect the limit point. The experiments in Hastie *et al.* (1999) reveal a slow rate of convergence (as is typical of the EM), but the imputed values are satisfactory. A similar behavior is observed in our experiments.

The ICR algorithm is based on the assumption that we have a small group of neighboring wells whose water levels behave in a similar way. So regressing one well against the others is likely to provide good results. However, if we have a large group of wells, from various places in the aquifer, there is no point in regressing one well against all the others. In this case the regression should be restricted to a smaller group of wells. This idea motivates the next method.

K Nearest Neighbors (KNN) imputing

The KNN algorithm is a simple method that is able to handle a large number of wells. It relies on the assumption that for each well with missing entries we can find a small group of neighboring wells. The method is not iterative. It achieves only one sweep over the columns of A, considering one column at a time.

The treatment of the jth column, $j = 1, \ldots, n$.

(a) Let the m-vector $\mathbf{b}$ denote the jth column of A.

If $\mathbf{b}$ has no missing entries skip to the next column.

(b) Select k other columns of A which are most 'similar' to $\mathbf{b}$.

These columns are used to construct an m x k matrix, A_j.

(c) Compute a k-vector **d** that estimates **b**. (Usually **d** is a weighted average of the k nearest neighbors.)
(d) Set the missing entries in **b** to be the corresponding entries of **d**.

The above four steps are repeated for each column of A. Thus for each column we construct a different matrix of 'neighboring' columns, and a different simulating vector, **d**. Usually the number of neighbors, k, is determined in advance. A typical value of k lies between 4 and 10. It is also helpful to assign initial trial values to the missing entries. This simplifies the implementation of steps (b) and (c). For further discussions of this method see Hastie *et al.* (1999), Troyanskaya *et al.* (2001), and Kim *et al.* (2005).

Iterative SVD imputing

The iterative SVD method generates a sequence of admissible matrices, A_ℓ, $\ell = 1,2, \ldots$, where $A_{\ell+1}$ is obtained by computing a SVD of A_ℓ. The term 'admissible' refers to any m x n matrix, $\tilde{A} = (\tilde{a}_{ij})$, which is obtained from A by substituting some trial values into the missing entries of A. That is, $\tilde{a}_{ij} = a_{ij}$ when a_{ij} is known. Let

$$A_\ell = U\,S\,V^T$$

denote a computed SVD of A_ℓ. Then the matrices $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$ have orthonormal columns, and $S = \mathrm{diag}\{\sigma_1, \ldots, \sigma_n\}$ is a diagonal matrix. The diagonal entries of S are called singular values and assumed to satisfy

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n \geq 0.$$

Using the SVD, we are able to construct a rank-k Truncated SVD of A_ℓ. Let the matrices $U_k = [u_1, \ldots, u_k]$, $V_k = [v_1, \ldots, v_k]$ and $S_k = \mathrm{diag}\{\sigma_1, \ldots, \sigma_k\}$ be obtained from the first columns of U, V, and S, respectively. Then the rank-k matrix

$$T_k(A_\ell) = U_k S_k V_k^T = \sigma_1 \mathbf{u_1 v_1^T} + \sigma_2 \mathbf{u_2 v_2^T} + \ldots + \sigma_k \mathbf{u_k v_k^T}$$

is called a Truncated SVD of A_ℓ , and this matrix solves the minimum norm problem

$$\textbf{minimize } \rho(B) = ||A_\ell - B||_F^2$$

$$\textbf{subject to } B \in \mathbf{B_k},$$

where $||\,.\,||_F$ denotes Frobenius matrix norm. As in Problem (1), $\mathbf{B_k}$ denotes the set of all m x n matrices whose rank is at most k. Recall that the Frobenius norm of a matrix $G = (g_{ij})$ satisfies $\|G\|_F^2 = \Sigma(g_{ij})^2$, where the sum is over all the entries of the matrix. For detailed discussion of the SVD and its properties, see Golub & Van Loan (1983).

The basic iteration:

(a) Given an admissible matrix, A_ℓ, compute $T_k(A_\ell)$.
(b) Compute a new admissible matrix $A_{\ell+1}$. The entries of $A_{\ell+1}$ which refer to missing entries in A obtain the values of the corresponding entries in $T_k(A_\ell)$.

The iterative SVD algorithm is another example of a method that descends from the EM framework. It alternates between the imputing process (expectation) and the SVD computation (maximization). The success of the method depends on proper choices of k and the starting matrix. A common way to compute A_1 is by applying an averaging scheme. It is shown in Dax (2014) that the related sequence of objective functions values is monotonic decreasing, which suggests that the sequence $T_k(A_\ell)$ may converge toward a solution of Equation (1.1). For further discussions of this method see Hastie *et al.* (1999), Troyanskaya *et al.* (2001), Friedland *et al.* (2006a, 2006b), and Kurucz *et al.* (2007).

Rank minimization

This method achieves imputing by solving the rank minimization problem

$$\textbf{minimize} \quad \mathrm{rank}(B) \tag{8a}$$

$$\textbf{subject to} \quad B \in \mathbf{A}, \tag{8b}$$

where **A** denotes the set of all admissible matrices. The rank function, rank(B), provides the rank of B. (If B is a full rank matrix rank(B) = n, if B is a rank-one matrix rank(B) = 1, and so forth.) The motivation behind this approach comes from imputing problems in which the original data constitute a large low-rank matrix with a small percentage of known entries. The Netflix prize competition (www.netflixprize.com) is, perhaps, the most celebrated problem of this kind. Direct algorithms for solving (8) are difficult to implement, due to the combinatorial nature of the rank function. This difficulty

motivates the nuclear norm minimization approach, which is considered in the next section. A different way to solve (8) stems from the following observation whose proof is given in Dax (2014). For $k = 1, \ldots, n$, let the rank-k matrix B_k solve Problem (1). Then there exists a rank index, k*, that satisfies $F(B_k) > 0$ for $k = 1, \ldots, k^* - 1$, and $F(B_k) = 0$ for $k = k^*, \ldots, n$. That is, k* is the first rank index for which $F(B_k) = 0$. Furthermore, the matrix B_{k^*} solves (8). It follows, therefore, that the rank minimization problem (8) can be solved by computing the sequence $B_1, \ldots, B_{k^*}$. Methods for building this sequence are considered in the following sections.

Nuclear norm minimization

This approach is also aimed at solving large imputing problems in which the data matrix is known to be a low-rank matrix. The method achieves imputing by computing an admissible matrix that solves the minimum norm problem

$$\textbf{minimize} \quad ||B||_{\nu} \tag{9a}$$

$$\textbf{subject to} \quad B \in \mathbf{A}, \tag{9b}$$

where $||B||_{\nu}$ denotes the nuclear norm of B, which is the sum of its singular values. The replacement of (8) with (9) is advocated by Candes & Recht (2009), who suggest solving (9) via semidefinite programming. Recent nuclear norm methods replace (9) with 'regularized' versions of this problem, which enables the use of more efficient solution methods, e.g., Ma *et al.* (2008) or Mazumder *et al.* (2009).

A new matrix imputing method

We shall finish the discussion of matrix methods by presenting a new method of this kind. As before, the matrix B_k denotes a computed solution of (1). The idea is to build the sequence

$$B_1, B_2, \ldots, B_{k-1}, B_k, \ldots \tag{10}$$

by using B_{k-1} to compute B_k. The building process stops at an optimal value for k. The basic building block is a simple iterative algorithm that computes a rank-one approximation of a matrix with missing entries. This algorithm is used in the solution of (1) and enables efficient construction of (10).

Computing a rank-one approximation

Recall that a rank-one m x n matrix has the form $\mathbf{u}\,\mathbf{v}^T$, where $\mathbf{u} = (u_1, u_2, \ldots, u_m)^T$ and $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$. Observe that the (i, j) entry of this matrix equals $u_i v_j$. In this case, when $k = 1$, the least-squares problem (1) is reduced to

$$\textbf{minimize } f(\mathbf{u}, \mathbf{v}) = \Sigma(a_{ij} - u_i v_j)^2, \tag{11}$$

where the sum is over the known entries of A. Below we describe a simple iterative algorithm for solving this problem.

Let $\mathbf{u}_{\ell-1}$ and $\mathbf{v}_{\ell-1}$ denote the current estimate of the solution at the beginning of the ℓ-th iteration. Then the ℓth iteration, $\ell = 1, 2, \ldots$, is composed of the following two steps.

Step 1 – Given $\mathbf{v}_{\ell-1} = (\nu_1, \nu_2, \ldots, \nu_n)^T$ the vector $\mathbf{u}_{\ell} = (\mu_1, \mu_2, \ldots, \mu_m)^T$ is obtained by solving the least-squares problem

$$\textbf{minimize } \phi(\mu_1, \mu_2, \ldots, \mu_m) = \Sigma(a_{ij} - \mu_i v_j)^2,$$

where the sum is over all the known entries of A. That is

$$\mu_i = \frac{\Sigma a_{ij} v_j}{\Sigma (v_j)^2}, \quad i = 1, \ldots, m, \tag{12}$$

where the two sums in (12) are over j indices of known entries in the ith row.

Step 2 – Given $\mathbf{u}_{\ell} = (\mu_1, \mu_2, \ldots, \mu_m)^T$, the vector $\mathbf{v}_{\ell} = (\nu_1, \nu_2, \ldots, \nu_n)^T$ is obtained by solving the least-squares problem

$$\textbf{minimize } \psi(\nu_1, \nu_2, \ldots, \nu_n) = \Sigma(a_{ij} - \mu_i \nu_j)^2,$$

where the sum is over all the known entries of A. That is

$$\nu_j = \frac{\Sigma a_{ij} \mu_i}{\Sigma (\mu_i)^2}, \quad j = 1, \ldots, n, \tag{13}$$

where the two sums in (13) are over i indices of known entries in the jth column.

Observe that minimizing $f(\mathbf{u},\mathbf{v})$ by changing one variable at a time, results in the same basic iteration. Define $f_\ell = f\ (\mathbf{u}_\ell,\mathbf{v}_\ell)$ then, clearly, $f_\ell \geq f_{\ell+1} \geq 0$ for $\ell = 1, 2, \ldots$, and the sequence $\{f_\ell\}$ converges.

Computing a rank-k approximation

In this section we describe a simple iterative algorithm for calculating a matrix B_k that solves (1). The lth iteration, $\ell = 1, 2, \ldots$, starts with an m x n matrix

$$H_\ell = \mathbf{u_1}\mathbf{v_1^T} + \mathbf{u_2}\mathbf{v_2^T} + \ldots + \mathbf{u_k}\mathbf{v_k^T} \tag{14}$$

and ends with the matrix $H_{\ell+1}$ that has a similar form. Note that (14) is the general form of an m x n matrix whose rank is, at most, k. Hence the actual objective function that we minimize is obtained from (1) by writing B as the sum (14). The basic iteration is composed of k steps. At the jth step, $j = 1, \ldots, k$, only the jth rank-one matrix $\mathbf{u_j}\mathbf{v_j^T}$ is changed, while the other rank-one matrices are kept fixed. The change in $\mathbf{u_j}\mathbf{v_j^T}$ attempts to minimize the objective function with respect to this matrix. This minimization is carried out by using the algorithm for solving (11) with the matrix $A - H_\ell + \mathbf{u_j}\mathbf{v_j^T}$ instead of A. The iteration (12)–(13) starts with the current values of $\mathbf{u_j}$ and $\mathbf{v_j}$. This ensures that the objective function is successively decreasing from step to step and from iteration to iteration.

The above algorithm is called Successive Rank-One Modifications (SROM). It is, perhaps, the simplest way to solve (1). Other methods for solving this problem include the Alternating Least Squares (ALS) algorithm, and the methods of Newton, Gauss-Newton, and Wiberg, e.g. Dax (2014, 2017) and Chen & Suter (2004). A major factor that effects the efficiency of the algorithm is the choice of the starting point. This issue is resolved below.

An improved building process

We now describe a useful modification of the computational process that builds the sequence (10). The first matrix, B_1, is obtained by solving (11). Then, for $k = 2, 3, \ldots$, the matrix B_k is obtained from B_{k-1} in the following way. The matrix B_{k-1} is presented as the sum of $k-1$ rank-one matrices,

$$B_{k-1} = \mathbf{y_1}\mathbf{w_1^T} + \mathbf{y_2}\mathbf{w_2^T} + \ldots + \mathbf{y_{k-1}}\mathbf{w_{k-1}^T}. \tag{15}$$

The first step is to compute a rank-one matrix, $\mathbf{y_k}\mathbf{w_k^T}$. This task is carried out by solving the rank-one problem (11), using the matrix $A - B_{k-1}$ instead of A. Once (11) is solved we build the rank-k matrix

$$\begin{aligned} H_1 &= B_{k-1} + \mathbf{y_k}\mathbf{w_k^T} \\ &= \mathbf{y_1}\mathbf{w_1^T} + \mathbf{y_2}\mathbf{w_2^T} + \ldots + \mathbf{y_{k-1}}\mathbf{w_{k-1}^T} + \mathbf{y_k}\mathbf{w_k^T}. \end{aligned} \tag{16}$$

Then (1) is solved by the SROM algorithm (or any other method) with H_1 as its starting point. In this way the matrix H_1 provides a good starting point for the iterative algorithm that solves (1).

A second advantage of this process lies in the information contained in the related sequence of '**formal objective function**' values

$$F_k = \left(\frac{F(B_k)}{\nu}\right)^{1/2},\ k = 1, 2, \ldots, \tag{17}$$

where ν denotes the number of known entries in A. Note that F_k is the root mean squared error of the least-squares problem (1). Hence the value of F_k provides the average difference between known entries of A and the corresponding entries of B_k. Furthermore, the definition of this sequence implies that

$$F_1 \geq F_2 \geq \ldots \geq F_k \geq \ldots \geq F_n = 0. \tag{18}$$

Below we derive a similar technique that enables us to assess the quality of the imputed entries.

Assessing the quality of the imputed entries: the use of a 'virtual objective function'

We have seen that F_k reflects the distance between known entries of A and the corresponding entries of B_k. It is tempting to think that F_k is also reflecting the ability of B_k to assess the unknown part of A. However, a small value of F_k does not guarantee that the entries of B_k provide a good estimate for the unknown entries of A. Moreover, as k increases behind a certain value, the number of degrees of freedom in (1) exceeds the number of known entries. At this point the solution of (1) loses its uniqueness, and the quality of the imputed values starts to deteriorate. (The number of degrees of freedom is close to $k(m+n)$, which is the number of unknown variables in the presentation

(16) of a rank-k matrix.) Yet, in spite of this deterioration, the values of F_k continue to decrease. This difficulty raises the need for another objective function that is able to assess the quality of the imputed entries.

For this purpose we introduce the concept of a 'virtual objective function'. In order to explain the main idea behind this concept we assume for a moment that the 'true' values of the missing entries in A are known. With this assumption at hand, the virtual objective function is defined as

$$V(B) = \Sigma(a_{ij} - b_{ij})^2, \tag{19}$$

where the last sum is restricted to the set of missing entries. Then, in analogy to (17), we define the related sequence of virtual objective function values

$$V_k = \left(\frac{V(B_k)}{\mu}\right)^{1/2}, \quad k = 1, 2, \ldots, \tag{20}$$

where μ denotes the number of missing entries in A. Thus V_k measures the average deviation between the missing entries of A and the corresponding entries of B_k. The smaller V_k is, the better imputing we have. The sequence (20) enables us to see how the quality of the imputed entries changes with k. Usually the quality is expected to improve in the first steps, but the value of V_k stops to decrease after a number of steps (for the reasons mentioned above). See Tables 3–5 in the next section. Let k* denote a value of k at which V_k attains its smallest value. Then B_{k^*} achieves the best imputing.

However, in real life applications, the 'true' values of the missing entries are never known, so the definition of the virtual objective function needs some amendments. The modified definition is based on a small test set of entries which are chosen from the known entries of A. These entries are removed from the list of known entries and added to the list of unknown entries. Then the value of F_k in (17) is computed according to the new list. Similarly, V_k is computed by using (19) with the test set, while the value of μ in (20) gives the number of entries in the test set.

The idea of using formal and virtual objective functions is borrowed from the statistical Cross-Validation method. However, it seems to be a new idea in the context of groundwater imputing. The theory behind this method attempts to answer the questions of how many entries to include in the test set, and how to choose the test entries from among the known ones. In our experiments the entries of the test set are randomly chosen from the known entries of A.

Determining an optimal rank

Below we outline a simple rule for determining a value of k for which B_k achieves the best imputing. Once we decide on the value of k, the unknown entries of A are replaced by the corresponding entries of B_k. The basic tools for achieving this goal are the related sequences of formal and virtual objective functions values.

We have seen that the sequence of formal objective function values is always monotonic decreasing. In contrast, the virtual objective function values, V_1, V_2, …, stop to decrease after a few iterations. The 'optimal' value of k is that for which V_k attains its smallest value. Usually a brief inspection of the two sequences is sufficient for determining a suitable value of k.

NUMERICAL EXPERIMENTS

In this section we present the results of some experiments with the proposed imputing methods. The algorithms were programmed in Fortran, using double precision arithmetic with round off unit about 10^{-16}. The water level matrices are taken from the archive of the Hydrological Service of Israel and represent typical cases of missing records. As before, each data matrix, $A = (a_{ij})$, has m rows and n columns, where a_{ij} denotes the measured water level (if available) at the end of the ith hydrological year in the jth well. Thus each matrix contains the measured data during m years in a group of n neighboring wells. The names of the water level matrices come from the area in which the wells are located. The first three groups of wells, Rehovot, Shiflat Lod and Rishon, are located in the coastal aquifer of Israel. The other three groups, Ein Kerem, Yartan 4, and Yartan 5, are located in the Yarkon-Taninim ('Yartan') mountain aquifer of Israel.

The details of the data matrices are summarized in Table 2. The reading of this table is quite straightforward. Consider for example the third column, which considers the 'Rishon' matrix. Then this matrix has $n = 6$ columns, and $m = 46$ rows, while the number of missing entries, μ,

Table 1 | A typical water-level matrix with missing entries (the Ein Kerem matrix)

		Water level (m)					
Row	**Year**	**Well 1**	**Well 2**	**Well 3**	**Well 4**	**Well 5**	**Well 6**
1	1960	484.35			485.15		
2	1961	448.88		448.88	486.25		
3	1962	446.48		445.53	484.10		
4	1963	400.06	400.06	410.74	478.68	491.24	
5	1964	405.11	404.70	443.99	482.26	494.46	
6	1965	433.15	406.48			490.90	
7	1966	413.23	400.96	442.06	476.02	496.31	471.17
8	1967	425.09	407.25	440.98	482.28	506.08	473.31
9	1968	443.76		449.89	483.99	492.37	474.50
10	1969	439.32		450.35	481.25	489.17	471.48
11	1970	421.91	403.64	442.06	474.69	488.56	463.43
12	1971	402.45	402.03	438.63	472.61	488.60	469.54
13	1972	403.66	403.19	437.29	473.21	499.13	467.34
14	1973	418.28	401.33	436.49	469.40	497.08	466.08
15	1974	436.18	410.54			498.62	474.85
16	1975	426.25	407.06	442.40	468.92	498.75	470.43
17	1976	412.70	401.61	434.76	461.34	505.58	463.60
18	1977	412.67	400.12	435.51	464.18	503.88	465.04
19	1978	423.67	399.04	432.27	460.23	506.27	459.95
20	1979	398.58	398.33	429.79	451.04	500.23	463.10
21	1980	427.71	406.89	441.83	467.04	494.76	468.29
22	1981	420.53	404.72	439.96	459.25	501.68	473.08
23	1982	414.50	401.71	436.22	459.84	501.39	466.23
24	1983	436.57	411.73	452.51	475.89	497.89	476.70
25	1984	422.53	404.33	439.86	463.84	495.20	471.46
26	1985	416.22	402.51	437.71	467.40	502.16	468.77
27	1986	426.45	400.11	432.46	460.67		461.40
28	1987	414.89	402.00	436.64	469.26	491.70	462.79
29	1988	422.58	406.08	434.85	478.96	505.79	474.99
30	1989	419.28	405.19	438.24	470.56	488.63	469.57
31	1990	415.23	403.53	436.55	465.87	487.86	466.15
32	1991	413.02	401.74	436.18	462.17	487.32	462.51
33	1992	444.03	417.26	457.50	485.75	502.82	480.75
34	1993	443.17	423.96	454.76	484.81	513.12	480.25
35	1994	427.78	397.73	452.28	475.85	489.40	477.23
36	1995	425.36	398.45	447.08	476.23	488.83	471.45
37	1996	421.95	406.48	442.83	475.30	492.14	471.59
38	1997	422.56	405.78	442.26	476.83	493.93	471.30
39	1998	419.13	403.83	439.94	476.25	488.60	472.74
40	1999	410.46	402.25	436.31	470.97		467.52
41	2000	407.18	399.49	432.32	466.65	489.18	465.81
42	2001	406.51	399.43	432.98	468.98	488.91	467.07
43	2002	428.90	403.08	440.47	481.17	488.84	471.67
44	2003	435.94	409.53	449.82	485.54	510.70	479.46
45	2004	422.58	405.56	441.78	482.02	489.05	472.81
46	2005	424.97	404.73	433.76	475.3		473.74

equals 17. Note that the matrices Yartan 4 and Yartan 5 contain no missing entries.

The imputing methods that we have tested include three methods: the additive model, the ICR method, and the new algorithm, the Improved Building Process. The last method builds a sequence B_1, B_2, …, where B_k is a rank-k matrix that solves (1). (We have not tested the KNN imputing because it is mainly aimed at handling larger sets of wells.)

The data in Tables 3–5 provide the corresponding values of formal and virtual objective functions. The virtual objective function is defined by constructing a test set made of τ entries, which are randomly chosen from the known entries of A. The actual number of missing entries is, therefore, $\mu + \tau$. The other $m_*n - (\mu + \tau)$ entries are used for constructing the formal objective function.

The matrix B_0 denotes the null matrix, a matrix with zero entries. The formal and the virtual values obtained for this matrix enable us to assess the improvement made by the tested methods. In additive model imputing the related formal and virtual values are computed by using the resulting rank-2 matrix, $\mathbf{y}\ \mathbf{e}^{\mathbf{T}} + \mathbf{e}\ \mathbf{w}^{\mathbf{T}} + \boldsymbol{\alpha}\ \mathbf{e}\ \mathbf{e}^{\mathbf{T}}$, instead of B_k. The ICR method does not produce an

Table 2 | The water levels matrices

Matrix name	Rehovot	Shiflat Lod	Rishon	Ein Kerem	Yartan 4	Yartan 5
Number of wells n	5	7	6	6	4	5
Number of years m	45	46	46	46	48	33
Number of missing records μ	10	26	17	22	0	0
Number of test entries τ	5	7	6	6	10, 18, 24	10, 16, 24
First year of records	1961	1960	1960	1960	1962	1977
Last year of records	2005	2005	2005	2005	2009	2009

Table 3 | Formal and virtual objective function values (m)

		Matrix			
Method	Objective function	Rehovot m = 45, n = 5, μ = 10, τ = 5.	Shiflat Lod m = 46, n = 7, μ = 26, τ = 7.	Rishon m = 46, n = 6, μ = 17, τ = 6.	Ein Kerem m = 46, n = 6, μ = 22, τ = 6.
B_0	Formal	2.06	15.90	14.17	451.62
	Virtual	3.19	18.62	16.44	456.04
Additive model	Formal	1.05	0.40	1.21	5.82
	Virtual	1.61	0.25	1.36	7.01
B_1	Formal	0.88	0.40	1.79	6.00
	Virtual	1.48	0.19	1.56	6.96
B_2	Formal	0.60	0.26	0.42	3.71
	Virtual	1.89	0.18*	0.95	7.18
B_3	Formal	0.25	0.17	0.31	2.33
	Virtual	0.31	0.42	0.63*	5.41*
B_4	Formal	0.12	0.11	0.23	1.25
	Virtual	0.19*	0.51	0.87	7.52
B_5	Formal	0.00	0.07	0.13	0.70
	Virtual	0.19*	0.53	0.68	11.15
B_6	Formal		0.03	0.00	0.00
	Virtual		0.28	0.68	11.03
B_7	Formal		0.00		
	Virtual		0.27		
ICR	Virtual	0.35	0.21	0.73	6.96

Table 4 | Formal and virtual objective function values (m) for the Yartan 4 matrix, m = 48, $n = 4$, $\mu = 0$

		Number of entries in the test set		
Method	**Objective function**	$\tau = 10$	$\tau = 18$	$\tau = 24$
B_0	Formal	13.97	13.87	13.93
	Virtual	13.27	14.53	13.94
Additive model	Formal	0.41	0.41	0.40
	Virtual	0.67	0.59	0.58
B_1	Formal	0.38	0.38	0.39
	Virtual	0.62	0.55	0.49
B_2	Formal	0.10	0.10	0.08
	Virtual	0.28*	0.23*	0.30*
B_3	Formal	0.06	0.06	0.04
	Virtual	0.60	0.53	0.39
B_4	Formal	0.00	0.00	0.00
	Virtual	0.60	0.53	0.38
ICR	Virtual	0.28*	0.23*	0.30*

Table 5 | Formal and virtual objective function values (m) for the Yartan 5 matrix, m = 33, $n = 5$, $\mu = 0$

		Number of entries in the test set		
Method	**Objective function**	$\tau = 10$	$\tau = 16$	$\tau = 24$
B_0	Formal	14.28	14.25	14.20
	Virtual	14.30	14.52	14.74
Additive model	Formal	0.53	0.52	0.44
	Virtual	0.59	0.67	1.00
B_1	Formal	0.61	0.63	0.53
	Virtual	0.79	0.63	1.16
B_2	Formal	0.17	0.17	0.15
	Virtual	0.29*	0.34*	0.75*
B_3	Formal	0.09	0.09	0.07
	Virtual	0.29*	0.47	1.05
B_4	Formal	0.05	0.05	0.02
	Virtual	0.89	0.64	1.19
B_5	Formal	0.00	0.00	0.00
	Virtual	0.88	0.64	1.19
ICR	Virtual	0.47	0.37	0.75*

approximating matrix and, therefore, only the related virtual values are computed. ***The starred figures denote minimal values of the virtual objective function.*** The smallest value of the formal objective function is always obtained for B_n.

The experiments with the first four test matrices are summarized in Table 3. Consider, for example, the experiments with the Rishon matrix. Running the additive model method on this matrix results in a formal objective value of 1.21, while the related value of the virtual objective function is 1.36. Similarly, when using the new method on the Rishon matrix, the matrix B_3 results in a formal objective function of 0.31 and a virtual objective function value of 0.63. The last value is the smallest virtual value obtained for the Rishon matrix. The ICR method gave a virtual value of 0.73, which is slightly above that value. The experiments in Table 3 clearly illustrate the usefulness of the proposed method for determining a value of k for which B_k achieves the best imputing.

The experiments in Tables 4 and 5 provide a further demonstration of the above advantage of the new method. In addition, these experiments enable us to see how the number of missing entries affects the imputing process. For this purpose we have used the matrices Yartan 4 and Yartan 5 with three different values of τ. (Recall that τ denotes the number of entries in the test set.) In these matrices $\mu = 0$, so the overall number of missing entries equals τ. Let us consider, for example, the experiments on the Yartan 4 matrix when $\tau = 18$. In this case the matrix B_2 results in a formal objective function value of 0.10 and a virtual objective function value of 0.23. The last value is a minimal virtual value, which is shared with the ICR algorithm. Similarly, let us consider the Yartan 5 matrix with $\tau = 16$. In this case the minimal virtual value, 0.34, is obtained for B_2.

Graphical presentation of the imputed water levels are displayed in Figures 1 and 2. In these figures each curve refers to a different well, and the imputed values are marked with bold characters. The first figure considers the Ein-Kerem matrix, using B_3. (The measured water levels of the wells in Figure 1 are displayed in Table 1.) The second figure considers the Yartan 4 matrix with $\tau = 18$, using B_2. These figures clearly illustrate the ability of the new method to achieve satisfactory imputing.

The current experiments use typical records of annual water levels, which were taken from the archive of the Hydrological Service of Israel. The statistical approach suggests further tests, with various scenarios of missing data. Such as 'missing at random' and 'not missing at random'. However, this interesting issue is left for future research.

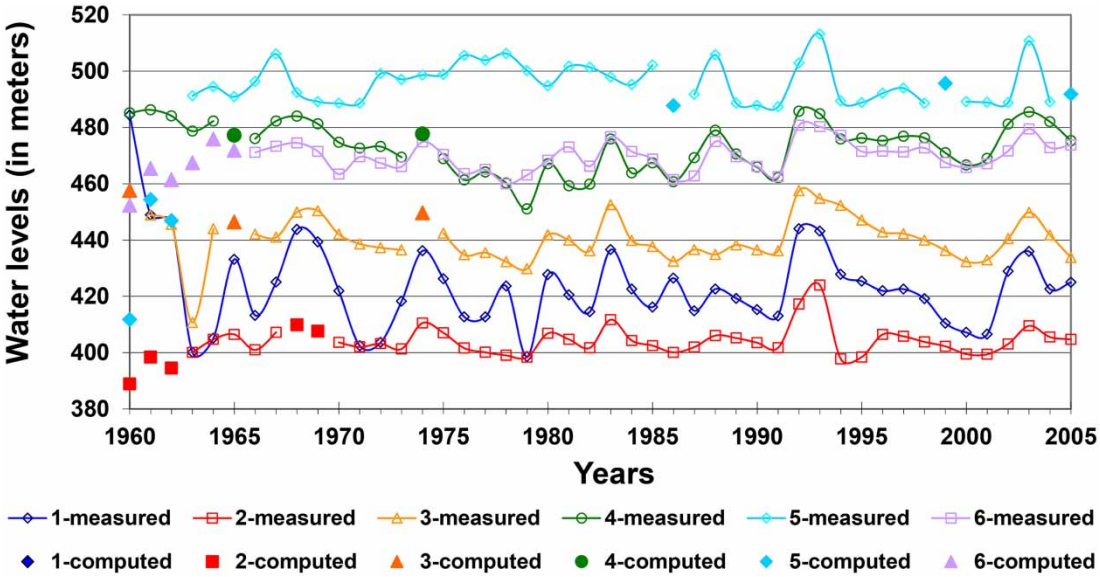

Figure 1 | Imputing the Ein_Kerm matrix. Each curve refers to a different well, with imputed values marked in bold.

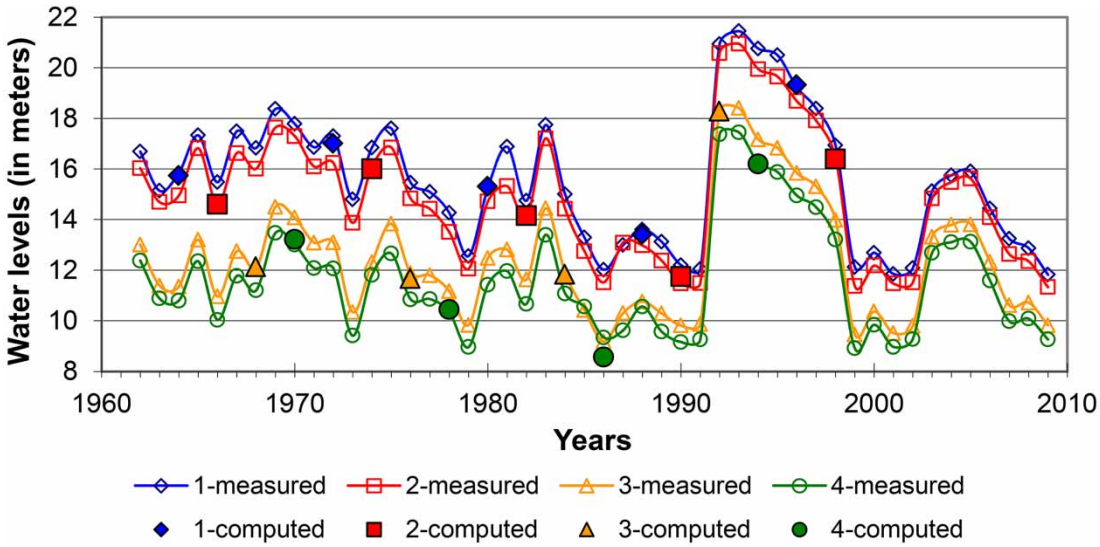

Figure 2 | Imputing the Yartan-4 matrix. Each curve refers to a different well, with imputed values marked in bold.

CONCLUDING REMARKS

The need for imputing missing records of annual groundwater data arises in many applications. Traditional approaches for handling this problem are based on statistical techniques for treating missing data. In this paper we introduce a different approach, one that is based on matrix properties of the data matrix. The new approach has been found successful in many modern areas, but it has not yet been used in hydrology. Hence, one aim of this paper is to introduce the matrix approach and to demonstrate its usefulness. Indeed our experiments illustrate the ability of the new method to achieve satisfactory imputation of missing groundwater observations.

The method proposed in this paper is based on three important innovations. The first is a gradual rank increasing

process that efficiently builds a sequence of matrices, B_k, $k = 1, 2, 3, \ldots$, where B_k is a rank-k matrix that solves (1). The second innovation regards the construction of a 'virtual objective function' that measures the quality of the imputed entries. This enables us to build a related sequence, V_k, $k = 1, 2, 3, \ldots$, where V_k reflects the ability of B_k to estimate the missing entries in A. The third innovation is an effective rule for choosing an optimal value of k, for which B_k achieves the best imputing. As we have seen, the virtual values, V_1, V_2, …, stop decreasing after a few iterations, and the optimal value of k is that for which V_k attains its smallest value. Another benefit of using a 'virtual objective function' is that it enables us to compare the performance of different methods. This point is illustrated in our experiments, which compare the additive model, the ICR algorithm, and the new method.

ACKNOWLEDGEMENTS

The authors are greatly indebted to three anonymous referees for several helpful comments.

REFERENCES

Afifi, A. A. & Elashoff, R. M. 1966 Missing observations in multivariate statistics. I: review of the literature. *J. Amer. Statist. Ass.* **61**, 595–605.

Allison, P. D. 2009 Missing data pp. 72–89 in the *SAGE Handbook of Quantitative Methods in Psychology* (R. E. Milesap & A. Maydeu-Olivers, eds). Thousand Oaks, CA.

Candes, E. J. & Recht, B. 2009 Exact matrix completion via convex optimization. *Foundations of Computational Mathematics* **9**, 717–772.

Chen, P. & Suter, D. 2004 Recovering the missing components in a large noisy low-rank matrix: application to SFM. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **26**, 1051–1063.

Dax, A. 1985 Completing missing groundwater observations by interpolation. *Journal of Hydrology* **81**, 375–399.

Dax, A. 2014 Imputing missing entries of a data matrix: a review. *Journal of Advanced Computing* **3**, 98–222.

Dax, A. 2017 A gradual rank increasing process for matrix completion. *Numerical Algorithms*. doi:10.1007/s11075-017-0292-2.

Dempster, A. P., Laird, N. M. & Rubin, D. B. 1977 Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* **39**, 1–38.

Friedland, S., Niknejad, A. & Chihara, L. 2006a A simultaneous reconstruction of missing data in DNA microarrays. *Linear Algebra and its Applications* **416**, 8–28.

Friedland, S., Kaveh, M., Niknejad, A. & Zara, H. 2006b An algorithm for missing value estimation for DNA microarray data. In: *IEEE Proceedings of ICASSP 2006*, 14–19 May, Toulouse, France, Volume II, pp. 1092–1095.

Golub, G. H. & Van Loan, C. F. 1983 *Matrix Computations*. Johns Hopkins Univ. Press, MD, USA.

Hastie, T., Tibshirani, R., Sherlock, G., Eisen, M., Brown, P. & Botstein, D. 1999 *Imputing Missing Data for Gene Expression Arrays*. Technical Report, Division of Biostatistics, Stanford University, CA.

Howard, K. I. & Gordon, R. A. 1963 Empirical note on the 'number of factors' problem in factor analysis. *Psychol. Rep.* **12**, 247–250.

Jackson, D. A. 1993 Stopping rules in principal component analysis: a comparison of heuristical and statistical approaches. *Ecology* **74**, 2204–2214.

Kim, H., Golub, G. H. & Park, H. 2005 Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics* **21**, 187–198.

Kurucz, M., Benczur, A. A. & Csalogany, K. 2007 Methods for large scale SVD with missing values. In: *KDD Cup and Workshop in Conjunction with KDD 2007*, 12–15 August, San Jose, California.

Ledesma, R. D. & Valero-Mora, P. 2007 Determining the number of factors to retain in EFA: an easy-to-use computer program for carrying out parallel analysis. *Practical Assessment Research & Evaluation* **12**. http://pareonline.net/getvn.asp?v=12&n=2.

Little, R. J. A. & Rubin, D. B. 1987 *Statistical Analysis with Missing Data*. John Wiley and Sons, New York.

Ma, S., Goldfarb, D. & Chen, L. 2008 *Fixed Point and Bregman Iterative Methods for Matrix Rank Minimization*. Tech. Rep., Department of IEOR, Columbia University, New York.

Mazumder, R., Hastie, T. & Tibshirani, R. 2009 *Regularization Methods for Learning Incomplete Matrices*. Technical Report, arXiv:0906.2034, June 2009.

Peres-Neto, P. R., Jackson, D. A. & Somers, K. M. 2005 How many principal components? Stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis* **49**, 974–997.

Rubin, D. B. 1987 *Multiple Imputation for Nonresponse in Surveys*. John Wiley and Sons, New York.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D. & Altman, R. 2001 Missing value estimation for DNA microarrays. *Bioinformatics* **17**, 520–525.

Tsikriktsis, N. 2005 A review of techniques for treating missing data in OM survey research. *Journal of Operations Management* **24**, 53–62.

First received 28 August 2016; accepted in revised form 6 May 2017. Available online 27 July 2017