REGULARIZED STRUCTURED LOW-RANK APPROXIMATION WITH APPLICATIONS*

MARIYA ISHTEVA†, KONSTANTIN USEVICH†, AND IVAN MARKOVSKY†

Abstract. We consider the problem of approximating an affinely structured matrix, for example a Hankel matrix, by a low-rank matrix with the same structure. This problem occurs in system identification, signal processing and computer algebra, among others. We impose the low-rank by modeling the approximation as a product of two factors with reduced dimension. The structure of the low-rank model is enforced by introducing a regularization term in the objective function. The proposed local optimization algorithm is able to solve the weighted structured low-rank approximation problem, as well as to deal with the cases of missing or fixed elements. In contrast to approaches based on kernel representations (in linear algebraic sense), the proposed algorithm is designed to address the case of small targeted rank. We compare it to existing approaches on numerical examples of system identification, approximate greatest common divisor problem, and symmetric tensor decomposition and demonstrate its consistently good performance.

Key words. low-rank approximation, affine structure, regularization, missing data, system identification, approximate greatest common divisor, symmetric tensor decomposition

AMS subject classifications. 15A23, 15A83, 65F99, 93B30, 37M10, 37N30, 11A05, 15A69

1. Introduction.

Low-rank approximations are widely used in data mining, machine learning, and signal processing, as a tool for dimensionality reduction, feature extraction, and classification. In system identification, signal processing, and computer algebra, in addition to having low rank, the matrices are often structured, e.g., they have (block) Hankel, (block) Toeplitz, or (block) Sylvester structure. Motivated by this fact, in this paper, we consider the problem of approximating a given structured matrix $D \in \mathbb{R}^{m\times n}$ by a matrix $\widehat{D} \in \mathbb{R}^{m\times n}$ with the same structure and with a pre-specified reduced rank ($\operatorname{rank}(\widehat{D}) \leq r < \min(m, n)$).

Existing algorithms solve this problem i) by local optimization, ii) by using relaxations, or iii) using heuristics, such as the widely used Cadzow method [4]. Relaxation methods include subspace-based methods [32, 15] and, more recently, nuclear norm based methods [17, 16, 9]. Local optimization algorithms use kernel or input/output (I/O) (also known as the structured total least squares (STLS) problem) representations of the rank constraint, as described in Table 1.1.

TABLE 1.1
Existing local optimization approaches for the structured low-rank approximation problem

Representation	Summary		References
Kernel	$R\widehat{D} = 0,$	$R \in \mathbb{R}^{(m-r)\times m}$	e.g., [18, 19, 30, 2]
I/O	$\begin{bmatrix} X & I \end{bmatrix} \widehat{D} = 0,$	$X \in \mathbb{R}^{(m-r)\times r}$	e.g., [7, 22, 26, 28, 24]
Image	$\widehat{D} = PL,$	$P \in \mathbb{R}^{m\times r}$, $L \in \mathbb{R}^{r\times n}$	[5]

* The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC Grant agreement number 258581 "Structured low-rank approximation: Theory, algorithms, and applications", the Research Foundation Flanders (FWO-Vlaanderen), the Flemish Government (Methusalem Fund, METH1), and the Belgian Federal Government (Interuniversity Attraction Poles programme VII, Dynamical Systems, Control, and Optimization).

† Dept. ELEC, Vrije Universiteit Brussel, Building K, Pleinlaan 2, B-1050 Brussels, Belgium ({mariya.ishteva, konstantin.usevich, ivan.markovsky}@vub.ac.be).

In this paper, we consider an underrepresented point of view, namely the image representation of the rank constraint:

$$\operatorname{rank}(\widehat{D}) \leq r \iff \widehat{D} = PL \text{ for some } P \in \mathbb{R}^{m\times r},\ L \in \mathbb{R}^{r\times n}.$$

The image representation is widely used in the machine learning community, however, there the matrices being approximated are unstructured. Imposing structure with the image representation is a nontrivial problem [5]. As shown in this paper, however, this problem can be resolved using regularization methods and an alternating projections (block coordinate descent) algorithm. We apply the proposed algorithm on practically relevant and nontrivial simulation examples from system identification, computer algebra (finding a common divisor of polynomials with noisy coefficients), and symmetric tensor decomposition, and demonstrate its consistently good performance.

The main competitors of the proposed local optimization approach are the kernel-based algorithms, which aim at solving the same problem. In contrast to kernel-based approaches which are meant for large rank r (small rank reduction $m-r$), the proposed approach is more efficient for problems with small r. Moreover, for general affine structures, existing kernel approaches have restrictions on the possible values of the reduced rank r. With the new approach we can overcome this limitation.

Another advantage of the proposed algorithm is its simplicity. As we show in Section 5.1, the proposed algorithm reduces to solving a sequence of least squares problems with closed form solutions. Last but not least, we are able to solve the weighted structured low-rank approximation problem, as well as to deal with the cases of missing elements in the data matrix or fixed elements in the structure. These "features" have great impact on the applicability of the proposed approach.

The rest of the paper is organized as follows. In Section 2, we discuss structured matrices and how to obtain the closest structured matrix to a given unstructured matrix (orthogonal projection on the space of structured matrices). In Section 3, the main optimization problem and its extensions are presented. Our two reformulations using regularization are proposed in Section 4. The main algorithm and its properties are discussed in Section 5. In Section 6, it is compared with existing approaches on numerical examples. In Section 7, we draw our final conclusions.

2. Structured matrices. Commonly used structures include Hankel, block Hankel, Toeplitz, block Toeplitz, Sylvester, block Sylvester, banded matrices with fixed bandwidth, and sparse matrices (with fixed sparsity pattern). These matrices have a pattern for the position of their elements. For example, in a Hankel matrix $D \in \mathbb{R}^{m\times n}$, the elements along any anti-diagonal are the same, i.e.,

$$D = \mathcal{H}_m(p) = \begin{bmatrix} p_1 & p_2 & p_3 & \cdots & p_n \\ p_2 & p_3 & & \iddots & \\ p_3 & \iddots & & & \vdots \\ \vdots & \iddots & \iddots & & \\ p_m & p_{m+1} & p_{m+2} & \cdots & p_{n_p} \end{bmatrix}$$

for some vector $p \in \mathbb{R}^{n_p}$, $n_p = m + n - 1$, called *structure parameter vector* for the matrix D. Note that any $m\times n$ (unstructured) matrix can be considered as structured matrix with $n_p = mn$ structure parameters.

In this section, we first formally introduce the affine structures and then discuss the orthogonal projection on the space of structured matrices.

2.1. Affine structures. Formally, affine matrix structures are defined as

$$\mathcal{S}(p) = S_0 + \sum_{k=1}^{n_p} S_k p_k, \tag{2.1}$$

where $S_0, S_1, \ldots, S_{n_p} \in \mathbb{R}^{m\times n}$, $p \in \mathbb{R}^{n_p}$ and $n_p \in \mathbb{N}$ is the number of structure parameters. We require n_p to be minimal in the sense that

$$\text{image}(\mathcal{S}) := \{\mathcal{S}(p) \,|\, p \in \mathbb{R}^{n_p}\}$$

cannot be represented with less than n_p parameters. It is convenient to define the following matrix

$$\mathbf{S} = \begin{bmatrix}\text{vec}(S_1) & \cdots & \text{vec}(S_{n_p})\end{bmatrix} \in \mathbb{R}^{mn\times n_p}, \tag{2.2}$$

where $\text{vec}(X)$ denotes the vectorized matrix X. The minimality of n_p is equivalent to $\mathbf{S}$ having full column rank. For simplicity, we assume that

(A) the matrix $\mathbf{S}$ consists of only zeros and ones and that there is at most one nonzero element in each row of the matrix $\begin{bmatrix}\text{vec}(S_0) & \mathbf{S}\end{bmatrix}$, i.e., every element of the structured matrix corresponds to (at most) one element of p.

This assumption is satisfied for the common structures mentioned earlier ((block) Hankel, (block) Toeplitz, etc.) Assumption **(A)** implies that $n_p \leq mn$ and $\mathbf{S}^\top \text{vec}(S_0) = 0$.

2.2. Orthogonal projection on $\text{image}(\mathcal{S})$**.** Next we discuss the orthogonal projection of an unstructured matrix on the space of structured matrices $\text{image}(\mathcal{S})$. This projection is used in the optimization algorithm of Section 5. After presenting the general formula, we discuss the simple intuitive explanation of the orthogonal projection, namely that it is equivalent to averaging elements corresponding to the same structure parameter p_k.

LEMMA 2.1. *For a structure $\mathcal{S}$ satisfying assumption* **(A)***, the orthogonal projection $P_{\mathcal{S}}(X)$ of a matrix X on* $\text{image}(\mathcal{S})$ *is given by*

$$P_{\mathcal{S}}(X) := \mathcal{S}(\Pi_{\mathbf{S}} \,\text{vec}\,(X)), \quad \textit{where} \quad \Pi_{\mathbf{S}} := (\mathbf{S}^\top \mathbf{S})^{-1}\mathbf{S}^\top. \tag{2.3}$$

The proof is given in the appendix. With some modifications, this lemma also holds for any affine $\mathcal{S}$.

The effect of applying $\Pi_{\mathbf{S}}$ on a vectorized $m\times n$ matrix X is producing a structure parameter vector by averaging the elements of X, corresponding to the same S_k. Indeed, the product $\mathbf{S}^\top \text{vec}(X)$ results in a vector containing the sums of the elements corresponding to each S_k. By assumption **(A)**, $\mathbf{S}^\top \mathbf{S}$ is a diagonal matrix, with elements on the diagonal equal to the number of nonzero elements in each S_k, i.e.,

$$\mathbf{S}^\top \mathbf{S} = \begin{bmatrix}\|S_1\|_F^2 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \|S_{n_p}\|_F^2\end{bmatrix} = \begin{bmatrix}\text{nnz}(S_1) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \text{nnz}(S_{n_p})\end{bmatrix},$$

where nnz stands for the number of nonzero elements. Therefore multiplying by $(\mathbf{S}^\top \mathbf{S})^{-1}\mathbf{S}^\top$ corresponds to averaging.

In particular, applying $\Pi_{\mathbf{S}}$ on a (vectorized) structured matrix extracts its structure parameter vector, since

$$\Pi_{\mathbf{S}} \,\text{vec}(\mathcal{S}(p)) = (\mathbf{S}^\top \mathbf{S})^{-1}\mathbf{S}^\top \mathbf{S}\, p = p.$$

For future reference, using (2.1), (2.2), and (2.3), we also have the following equality

$$\text{vec}(P_{\mathcal{S}}(X)) = \text{vec}(S_0) + \mathbf{S}\,\Pi_{\mathbf{S}} \,\text{vec}(X). \tag{2.4}$$

3. Weighted structured low-rank approximation. In this section, after presenting the main optimization problem, we discuss its extensions and its equivalent matrix representation.

The weighted structured low-rank approximation problem is formulated as follows

$$\min_{\hat{p}} \|p - \hat{p}\|_{\overline{W}} \quad \text{subject to } \operatorname{rank}(\mathcal{S}(\hat{p})) \leq r, \tag{3.1}$$

where $\overline{W} \in \mathbb{R}^{n_p \times n_p}$ is a symmetric positive semidefinite matrix of weights and

$$\|x\|^2_{\overline{W}} := x^\top \overline{W}\, x.$$

If $\overline{W}$ is the identity matrix, then the problem reduces to its unweighted counterpart, i.e., $\|\cdot\|_{\overline{W}} = \|\cdot\|_2$.

3.1. Extensions. Next we discuss three (hidden) extensions, which have a significant impact on the applicability of the structured low-rank approximation problem.

Weights. Being able to deal with weights in (3.1) has a number of advantages in practice. If prior knowledge is available about the importance or the correctness of each (noisy) structure parameter, this knowledge can be encoded in the problem using a diagonal weight matrix $\overline{W}$, where each diagonal element relates to the importance of the corresponding parameter or encodes how much we trust the corresponding parameter. In addition, finding the closest structured matrix to a given structured matrix with respect to the Frobenius norm, can also be encoded with a diagonal weight matrix, as further discussed in Section 6.2. Finally, 0/1 weights are used when dealing with missing elements, as we discuss below.

Missing values. Due to sensor failure, malfunctioning of a communication channel, or simply due to unseen events, real-world data can have unknown (missing) elements. If repeating the experiments until all data are collected is not an option, for example because of high price of the experiments or high computational time, the missing data have to be approximated as well. The problem of estimating missing data is also known as the matrix completion problem and is well-studied in the case of unstructured matrices. In the case of structured matrices, however, this problem has few solutions [21].

One way to deal with missing elements is to introduce zeros in the weight matrix $\overline{W}$ at the positions corresponding to the missing elements. In the most common case when $\overline{W}$ is a diagonal matrix with weights corresponding to the importance of each of the structure parameters p_i, $i = 1, \ldots, n_p$, having a missing parameter p_j is dealt with by taking its corresponding weight to be zero, i.e., $\overline{W}(j, j) = 0$.

Fixed elements. By having a matrix S_0 in (2.1), we allow the considered affine structure to have fixed elements. In practice, the fixed elements are often all zeros ($S_0 \equiv \mathbf{0}$), for example in the case of sparse matrices with a fixed sparsity pattern. However, we aim at dealing with the more general case of arbitrary fixed elements.

3.2. Problem reformulation using matrix representation. Consider the following reformulation of (3.1). For a given matrix $D = \mathcal{S}(p)$,

$$\min_{\hat{D}} \|D - \hat{D}\|^2_W \quad \text{subject to} \quad \operatorname{rank}(\hat{D}) \leq r \text{ and } \hat{D} \in \operatorname{image}(\mathcal{S}). \tag{3.2}$$

In (3.2), $\|\cdot\|^2_W$ is a semi-norm on the space of matrices $\mathbb{R}^{m \times n}$, induced by a positive semidefinite matrix $W \in \mathbb{R}^{mn \times mn}$ as

$$\|D\|^2_W := (\operatorname{vec}(D))^\top W \operatorname{vec}(D).$$

Note that for a given W in (3.2) and $\overline{W}$ defined as

$$\overline{W} = \mathbf{S}^\top W \mathbf{S} \tag{3.3}$$

we have that

$$\|\mathcal{S}(p)\|_W^2 = \mathrm{vec}(\mathcal{S}(p))^\top W \mathrm{vec}(\mathcal{S}(p)) = (\mathbf{S}p)^\top W \mathbf{S}p = p^\top \overline{W} p = \|p\|_{\overline{W}}^2.$$

Therefore, for $\overline{W}$ and W related by (3.3), problems (3.1) and (3.2) are equivalent.

4. Regularized structured low-rank approximation. Each of the constraints in (3.2) can easily be handled separately. Approximating a matrix by a structured matrix without imposing low-rank can be performed by orthogonally projecting the matrix on the space of structured matrices (see Section 2.2). Unweighted low-rank approximation without imposing structure can be done using truncated singular value decomposition SVD [11]. However, imposing both low-rank and fixed structure on the approximation is nontrivial even in the unweighted case (when $\overline{W} = I_{n_p}$). Likewise, the weighted low-rank approximation problem is difficult already in the unstructured case (when $\mathbf{S} = I$) [29].

We approach the weighted structured low-rank approximation problem from a new point of view, namely by a regularization technique. We propose two novel reformulations and discuss their relation to the original problem.

The main idea is to have one of the requirements satisfied at each step whereas the other one is gradually better satisfied at each successive iteration. Upon convergence both constraints should be satisfied. We have the following two choices (see Figure 4.1):

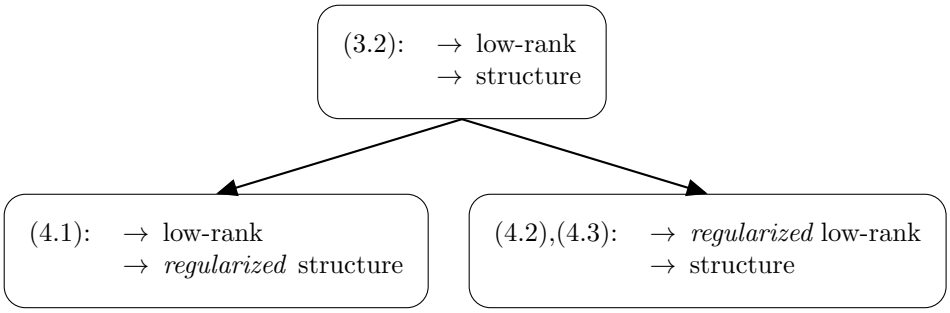


FIG. 4.1. *Optimization problems*

- regularize the structure constraint

$$\min_{P,\,L} \|D - PL\|_W^2 + \lambda \|PL - P_{\mathcal{S}}(PL)\|_F^2, \tag{4.1}$$

  where λ is a regularization parameter (discussed in Section 5.1.1), $\|\cdot\|_F$ stands for the Frobenius norm and $P_{\mathcal{S}}(PL)$ is defined in (2.3), or
- regularize the rank constraint

$$\min_{P,\,L} \|D - P_{\mathcal{S}}(PL)\|_W^2 + \lambda \|PL - P_{\mathcal{S}}(PL)\|_F^2. \tag{4.2}$$

Note that for $\lambda = \infty$ the term $\|PL - P_{\mathcal{S}}(PL)\|$ has to be zero and the three problems (3.2), (4.1) and (4.2) are equivalent. The interpretations of (4.1) and (4.2)

are however different. In (4.2), the structure is satisfied at each step and the low rank is 'secondary'. In (4.1), it is the other way around, although in both cases both constraints are satisfied at the solution. The choice of λ is further discussed in Section 5.1.1.

Given $\overline{W}$ in problem (3.1), there are many possibilities to choose W satisfying (3.3), so that problems (3.2) and (3.1) are equivalent. However, the following holds true.

REMARK 1. *Problem (4.2) is independent of the choice of W and can be formulated using $\overline{W}$ in the following way*

$$\boxed{\min_{P,\,L} \|p - \Pi_{\mathbf{S}} \mathrm{vec}(PL)\|^2_{\overline{W}} + \lambda \|PL - P_{\mathcal{S}}(PL)\|_F^2.} \tag{4.3}$$

Because of this reason, we will focus on problem formulation (4.2) and its equivalent representation (4.3). For a particular choice $W_* = \Pi_{\mathbf{S}}^\top \overline{W} \Pi_{\mathbf{S}}$ (which satisfies (3.3)), problem (4.1) is equivalent to (4.3).

5. The proposed algorithm. In this section, we propose an algorithm in the framework of the penalty methods. We first discuss how the minimization problem (4.3) can be solved for a fixed value of the regularization parameter λ and then present the algorithmic and computational details related to the proposed algorithm.

5.1. Description of the algorithm. The main idea is to solve the minimization problem (4.3) by alternatingly improving the approximations of L, for fixed P,

$$\min_{L} \|p - \Pi_{\mathbf{S}} \,\mathrm{vec}(PL)\|^2_{\overline{W}} + \lambda \|PL - P_{\mathcal{S}}(PL)\|_F^2 \tag{5.1}$$

and of P, for fixed L,

$$\min_{P} \|p - \Pi_{\mathbf{S}} \,\mathrm{vec}(PL)\|^2_{\overline{W}} + \lambda \|PL - P_{\mathcal{S}}(PL)\|_F^2. \tag{5.2}$$

Let I_n be the $n \times n$ identity matrix and let '$\otimes$' be the Kronecker product

$$X \otimes Y = \begin{bmatrix} x_{11}Y & \cdots & x_{1n}Y \\ \vdots & \ddots & \vdots \\ x_{m1}Y & \cdots & x_{mn}Y \end{bmatrix}, \quad \text{for } X \in \mathbb{R}^{m \times n}.$$

LEMMA 5.1. *Problems* (5.1) *and* (5.2) *are equivalent to the following least squares problems*

$$\begin{aligned} (5.1) \quad &\Leftrightarrow \quad \min_{L} \left\| \begin{bmatrix} \overline{M}\,\Pi_{\mathbf{S}} \\ \sqrt{\lambda}(I_{mn} - \mathbf{S}\,\Pi_{\mathbf{S}}) \end{bmatrix} (I_n \otimes P)\,\mathrm{vec}(L) - \begin{bmatrix} \overline{M}p \\ \sqrt{\lambda}\mathrm{vec}(S_0) \end{bmatrix} \right\|_2^2, \\ (5.2) \quad &\Leftrightarrow \quad \min_{P} \left\| \begin{bmatrix} \overline{M}\,\Pi_{\mathbf{S}} \\ \sqrt{\lambda}(I_{mn} - \mathbf{S}\,\Pi_{\mathbf{S}}) \end{bmatrix} (L^\top \otimes I_m)\,\mathrm{vec}(P) - \begin{bmatrix} \overline{M}p \\ \sqrt{\lambda}\mathrm{vec}(S_0) \end{bmatrix} \right\|_2^2, \end{aligned} \tag{5.3}$$

where $\overline{W} = \overline{M}^\top \overline{M}$ with $\overline{M} \in \mathbb{R}^{n_p \times n_p}$.

The proof is given in the appendix.

Both reformulations in Lemma 5.1 are least squares problems and can be solved in closed form. For fixed λ, we propose an algorithm in the framework of alternating

least squares and block coordinate descent, namely we alternatingly improve the approximations of L and of P by solving the least squares problems in (5.3). We discuss the update strategy for λ in Section 5.1.1. The choice of initial approximation P_0 for P and the stopping criteria are discussed in Section 5.1.2. The summary of the proposed algorithm is presented in Algorithm 1.

Algorithm 1 Regularized structured low-rank approximation

Input: $p \in \mathbb{R}^{n_p}$, $S_0 \in \mathbb{R}^{m \times n}$, $\mathbf{S} \in \mathbb{R}^{mn \times n_p}$, $\overline{W} = \overline{M}^\top \overline{M} \in \mathbb{R}^{n_p \times n_p}$, $r \in \mathbb{N}$, $P_0 \in \mathbb{R}^{m \times r}$.
Output: Factors $P \in \mathbb{R}^{m \times r}$ and $L \in \mathbb{R}^{r \times n}$, corresponding to a structured low-rank approximation problem (4.3).

```
Set Π_S = (S^⊤ S)^{-1} S^⊤.
Set P = P_0, λ_1 = 1.
for j = 1, 2, ... until a stopping criterion is satisfied do
    for k = 1, 2, ... until a stopping criterion is satisfied do
        Update L from (5.1).
        Update P from (5.2).
    end for
    Set λ_{j+1} such that λ_{j+1} > λ_j.
end for
```

Due to the simplicity of Algorithm 1, dealing with weights, missing elements and structures with fixed elements is straightforward. The weight matrix $\overline{W}$ and the matrix with fixed elements S_0 are readily introduced in (5.3) thus in and Algorithm 1. Dealing with missing elements is realized by introducing zeros in the weight matrix $\overline{W}$ at the positions corresponding to the missing elements. A numerical example with weights, corresponding to using Frobenius norm as a distance measure in (3.2), is given in Section 6.2. Examples with missing values are given in Section 6.2 and in Section 6.4. The examples in Section 6.3 and in Section 6.4 have fixed zero and nonzero elements, respectively.

5.1.1. Parameter λ. In theory, if we fix $\lambda = \infty$, then (4.3) is the exact structured low-rank approximation problem. In practice, we may fix λ to a "large enough" value and the solution PL is only approximately a structured matrix. The higher the value of λ, the better the structure constraint is satisfied; however, too large values may lead to numerical issues. Alternatively, adaptive schemes for updating λ are also possible. We can start from a small value and increase it with each iteration or set of iterations. This way we allow the algorithm to move to a "good region" first and then impose more strictly the structure constraint [25].

Algorithms using such update scheme for λ are reported to converge faster. The following strategy has been proposed in [25, §17.1]: if solving the previous subproblem was expensive, increase λ only modestly, e.g., $\lambda_{j+1} = 1.5\lambda_j$. If solving the previous subproblem was cheap, increase λ more ambitiously, $\lambda_{j+1} = 10\lambda_j$.

5.1.2. Initial Guess and Stopping Criterion. Let $D = U_r \Sigma_r V_r^\top$ be the truncated SVD of the given matrix D ($D = \mathcal{S}(p)$). Since the truncated SVD provides the best (in the unweighted case) low-rank approximation of a given matrix, U_r is a good initial approximation of the matrix P.

Consider the following stopping criteria. For the inner minimization problem, i.e., when λ is fixed to λ_j, stop when the derivatives of the objective function are smaller than τ_j with $\tau_j \to 0$ as $j \to \infty$. In practice, if we do not want to compute derivatives,

we can stop when there is little change in P and L. Note that for small λ we do not need to solve the problem exactly. Thus, we can stop the inner iteration earlier and avoid slow convergence. Only at the end, when λ becomes large, good approximation is required. For the outer minimization (main problem) we stop when λ is "large enough", e.g, 10^{14}.

We declare that PL is a structured matrix if

$$\frac{\|PL - P_{\mathcal{S}}(PL)\|_F^2}{\|PL\|_F^2} < \varepsilon$$

for a small ε, e.g., $\varepsilon = 10^{-12}$.

5.2. Properties of the algorithm.

5.2.1. Computational Complexity. The main computational cost is due to solving the least squares problems in (5.3), which is equivalent to solving two systems of linear equations. The size of the systems' matrices are $(n_p + mn) \times rn$ and $(n_p + mn) \times rm$, respectively. Suppose that $m \leq n$ and recall that $n_p \leq mn$. Then, the cost for one inner iteration step of the proposed algorithm is

$$O((rn)^2(n_p + mn)) = O(n^3 m r^2).$$

Note that this estimate does not take into account the available structure and sparsity of the matrices in (5.3). If the structure and sparsity are exploited, faster computation can be performed. Additionally, compared to the kernel approaches, which are fast for large values of r (preferably $r = m - 1$), the proposed approach is designed for small values of r. If the structure is not taken into account, the kernel approaches also have computational cost that is cubic in n, namely $O((m-r)^3 n^3)$, and moreover their cost per iteration is higher in m for small r.

5.2.2. Convergence Properties. Algorithm 1 falls into the setting of the penalty methods for constrained optimization [25, §17.1] whose convergence properties are well understood. For fixed λ, the proposed algorithm is an alternating least squares (or block coordinate descent) algorithm. Since we can solve the least squares problems in (5.3) in closed form, every limit point of the generated sequence is a stationary point [1, 12]. The convergence rate of these methods is linear.

For the convergence properties of the algorithm as $\lambda \to \infty$, we have the following theorem borrowed from the theory of quadratic penalty method [25, Th. 17.2.].

THEOREM 5.2. *For $\tau_j \to 0$ and $\lambda_j \to \infty$, if a limit point $(P, L)^*$ of the sequence $\{(P, L)\}_j$ is infeasible, it is a stationary point of the function $\|PL - P_{\mathcal{S}}(PL)\|_F^2$. On the other hand, if a limit point $(P, L)^*$ is feasible, then $(P, L)^*$ is a KKT (Karush-Kuhn-Tucker) point for problem* (3.1) *and*

$$\lim_j -\lambda_j \|PL - P_{\mathcal{S}}(PL)\|_F^2 = \lambda^*,$$

where λ^* is the multiplier that satisfies the KKT conditions.

6. Numerical experiments. In this section we apply the proposed algorithm on three different problems, namely, system identification, finding a common divisor of polynomials (with noisy coefficients), and symmetric tensor decomposition. Although these problems arise in completely different fields, are essentially different, and require different features (weights, fixed elements, or missing element), we demonstrate the consistently good performance of Algorithm 1.

6.1. Related algorithms. In our MATLAB simulations, we compare Algorithm 1 with Cadzow's algorithm [4] and the kernel-based algorithm `slra` [20, 30, 21], which also aim to solve problem (3.2). The former is popular in signal processing applications due to its simplicity and the latter has been recently extended to work with missing data. We next briefly summarize the main ideas behind these algorithms.

Cadzow's algorithm [4] consists of repeating the following two main steps

- "project" the current structured approximation to a low-rank matrix, e.g., using truncated SVD,
- project the current low-rank matrix to the space of structured matrices.

By performing its projections, this algorithm does not optimize over any optimization variables. The algorithm proposed in this paper, on the other hand, optimizes over the parameter matrices P and L. As shown in [8], Cadzows algorithm lacks convergence properties.

The `slra` algorithm [30, 20] is based on the kernel representation of the rank constraint , i.e.,

$$\operatorname{rank}(\widehat{D}) \leq r \iff R\widehat{D} = 0 \text{ for some full row rank matrix } R \in \mathbb{R}^{(m-r)\times m}$$

and uses the variable projection method [10], i.e., reformulation of the problem as inner and outer optimization, where the inner minimization admits an analytic solution. The outer problem is a nonlinear least squares problem and is solved by standard local optimization methods, e.g., the Levenberg–Marquardt method [23]. Generalization to problems with fixed and missing data is presented in [21]. Efficient implementation of the methods based on the variable projection is available for the case of mosaic-Hankel matrices [20, 30]. With $n > m$, the computational complexity of the cost function and derivative evaluation is $O(m^2 n)$. Therefore, this approach is suitable for applications with $n \gg m$ and small rank reduction $m-r$. In statistical estimation and data modeling—the main application areas of the algorithm—$n \gg m$ corresponds to modeling of large amount of data by a low-complexity model. In contrast, Algorithm 1 is meant for problems with small r. Moreover, for general affine structures, `slra` has a restriction on the possible values of the reduced rank r. Algorithm 1 overcomes this limitation.

6.2. Autonomous system identification. In this section, we will use the fact that the proposed algorithm can work with *weighted* norms and with matrices having *missing* elements.

Background. In system theory, an autonomous linear time-invariant dynamical model [19] can be defined by a difference equation

$$\theta_0\, y(t) + \theta_1\, y(t+1) + \cdots + \theta_\ell\, y(t+\ell) = 0, \quad \text{for } t = 1, \ldots, T-\ell, \tag{6.1}$$

where ℓ is the order of the system. The problem of system identification is: estimate the model parameter $\theta = \begin{bmatrix}\theta_0 & \theta_1 & \cdots & \theta_\ell\end{bmatrix} \in \mathbb{R}^{\ell+1}$, given a response $y = [y(1), \ldots, y(T)] \in \mathbb{R}^T$ of the system.

The difference equation (6.1) can equivalently be represented as

$$\begin{bmatrix}\theta_0 & \theta_1 & \cdots & \theta_\ell\end{bmatrix} \underbrace{\begin{bmatrix} y(1) & y(2) & y(3) & \dots & y(T-\ell) \\ y(2) & y(3) & y(4) & \iddots & \\ y(3) & \iddots & & & \vdots \\ \vdots & \iddots & \iddots & & \\ y(\ell+1) & y(\ell+2) & y(\ell+3) & \dots & y(T) \end{bmatrix}}_{\mathcal{H}_{\ell+1}(y)} = 0. \tag{6.2}$$

It follows from (6.2) that the Hankel matrix $\mathcal{H}_{\ell+1}(y)$ is rank deficient, or in other words, $\operatorname{rank}(\mathcal{H}_{\ell+1}(y)) \leq \ell$.

In the more realistic noisy case however, (6.1) and (6.2) hold only approximately. The problem of identifying the system then reduces to finding a rank-ℓ approximation of $\mathcal{H}_{\ell+1}(y)$. The parameter θ can then be easily computed from the null space of the obtained approximation.

If enough samples are provided, which is usually the case, another possible reformulation of (6.1) is the following

$$\begin{bmatrix}\theta_0 & \theta_1 & \cdots & \theta_\ell & 0 \\ 0 & \theta_0 & \theta_1 & \cdots & \theta_\ell\end{bmatrix} \underbrace{\begin{bmatrix} y(1) & y(2) & y(3) & \dots & y(T-\ell-1) \\ y(2) & y(3) & y(4) & \iddots & \\ y(3) & \iddots & & & \vdots \\ \vdots & \iddots & \iddots & & \\ y(\ell+2) & y(\ell+3) & y(\ell+4) & \dots & y(T) \end{bmatrix}}_{\mathcal{H}_{\ell+2}(y)} = 0. \tag{6.3}$$

Compared to (6.2), the matrix $\mathcal{H}_{\ell+2}(y)$ in (6.3) has one more row and one column less but its rank is still ℓ. The rank is however now reduced by 2 since we have $\ell+2$ rows. (This can also be concluded from the fact that there are now 2 linearly independent vectors in the null space of the matrix.) We can continue reshaping by adding rows and removing columns, e.g., until we get an (almost) square matrix. This could be useful since there are indications that truncated SVD of a square Hankel matrix relates to a better (initial) noise reduction. Note that if T is large, the square matrix would have low rank compared to its dimensions.

Example: System identification in Frobenius norm. In this example, we will use the fact that the proposed algorithm can work with *weighted* norms. In particular, in order to approximate a structured matrix $\mathcal{S}(p)$ with a low-rank structured matrix $\mathcal{S}(\hat{p})$ in Frobenius norm, i.e.,

$$\min_{\hat{p}} \|\mathcal{S}(p) - \mathcal{S}(\hat{p})\|_F^2 \quad \text{subject to} \quad \operatorname{rank}(\mathcal{S}(\hat{p})) \leq r$$

we need to take $\overline{W}$ from (3.1) and (4.3) to be a diagonal matrix with weights equal to the number of occurrences of each structure parameter p_i, $i = 1, \dots, n_p$. We consider the Frobenius norm as a distance measure between the data matrix and the approximation to facilitate the comparison with Cadzow's algorithm. In the next example, we illustrate the performance of the proposed algorithm with respect to the 2-norm $\|p - \hat{p}\|_2^2$.

The considered true (noiseless) signal y_0 is the sum of the following two exponentially damped cosines

$$\begin{aligned} y_0(t) &= y_{0,1}(t) + y_{0,2}(t), \\ y_{0,1}(t) &= 0.9^t \cos(\tfrac{\pi}{5}\, t), \\ y_{0,2}(t) &= \tfrac{1}{5}\, 1.05^t \cos(\tfrac{\pi}{12}\, t + \tfrac{\pi}{4}), \end{aligned} \tag{6.4}$$

$t = 1, \ldots, 50$, shown in Figure 6.1. The rank of the corresponding Hankel matrix is

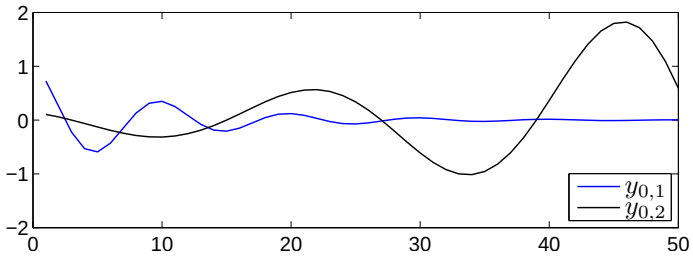


FIG. 6.1. *True components in the system identification examples.*

4, i.e., $\operatorname{rank}(\mathcal{H}_m(y_0)) = 4$, for $m = 5, 6, \ldots, 46$. We added noise in the following way

$$y(t) = y_0(t) + 0.2\, \frac{e(t)}{\|e\|}\, \|y_0(t)\|, \tag{6.5}$$

where $e(t)$ were drawn independently from the normal distribution with zero mean and unit standard deviation. The added noise increases the rank of $\mathcal{H}_m(y_0)$, so rank-4 approximation has to be computed.

We ran Algorithm 1, `slra` and Cadzow's algorithm. To facilitate the comparison with `slra`, we set $m = r + 1$ (i.e., $m = 5$). The initial approximation was obtained using the truncated SVD. Since `slra` is sensitive to the initial approximation, in addition to its default initialization, we ran `slra` starting from the solution obtained by Kung's method [15] (row "Kung → `slra`" in Table 6.1), which is a heuristic method for Hankel structured low-rank approximation.

After 1000 iterations, Cadzow's algorithm still did not converge and the rank of its structured approximation was 5 instead of 4, with smallest singular value of the approximation 0.0027. The result obtained by `slra` when initialized with Kung's method and the result obtained by Algorithm 1 were similar to each other. The computed trajectories (for one run) are presented in Figure 6.2. The numerical

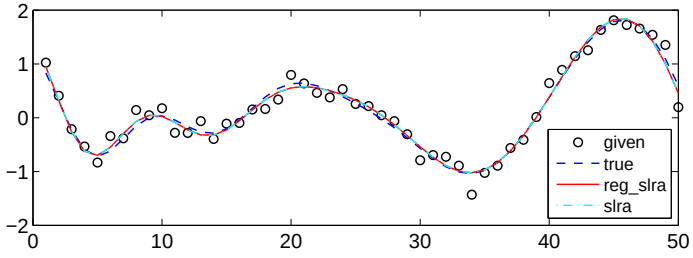


FIG. 6.2. *Noisy data y, true data y_0 and the trajectories obtained from Algorithm 1 and* `slra`*, for the example* (6.4)–(6.5)*. The computations are with respect to the Frobenius norm.*

TABLE 6.1
Numerical errors of the initial approximation (by SVD), Cadzow's algorithm, `slra`*, Kung's heuristic algorithm,* `slra` *initialized with Kung's algorithm's solution, and the proposed algorithm (Algorithm 1), for the example* (6.4)–(6.5).

Algorithm	$\|\mathcal{S}(y) - \mathcal{S}(\hat{y})\|_F^2$	$\|\mathcal{S}(y_0) - \mathcal{S}(\hat{y})\|_F^2$	Remarks
`init.approx.`	37.7107	36.1934	
Cadzow	(4.4916)	(1.0740)	**incorrect rank**
`slra`	11.3892	7.0164	
`Kung` [15]	7.1655	3.6906	heuristic
`Kung` → slra	**4.6106**	0.4915	
Algorithm 1	4.6124	**0.4794**	$\frac{\|PL - P_{\mathcal{S}}(PL)\|_F^2}{\|PL\|_F^2} = 1.6 \cdot 10^{-25}$

errors are presented in Table 6.1. The error of the initial approximation reported in Table 6.1 is computed by finding the closest structured matrix having the same kernel as the truncated SVD approximation.

Different realizations of the example lead to slightly different numerical results. However, the main conclusions of this example stay the same.

Example: System identification with missing data. In this example, we illustrate the fact that the proposed algorithm can work with matrices having *missing* elements. We continue with the above example but since Cadzow's algorithm cannot be applied to the missing data case directly, this time the objective function is with respect to the more standard parameter norm $\|p - \hat{p}\|_2^2$.

Since the rank of the true Hankel matrix is 4, standard algorithms need at least 5 $(5 = 4 + 1)$ consecutive data points for identification. However, in the example below, we removed every 5th data point, so these algorithms cannot be applied directly. Algorithm 1 and `slra`, however, can identify the system, as shown in Figure 6.3. The initial approximation for the missing values was provided by averaging their two

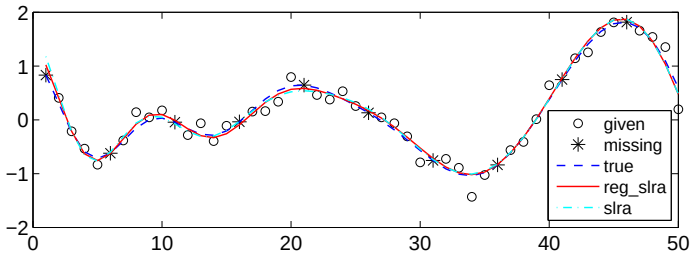


FIG. 6.3. *Noisy data (subset of y), missing data (subset of y_0), true data y_0 and the trajectories obtained from Algorithm 1 and* `slra` *for the example* (6.4)–(6.5) *with missing data.*

neighboring data points, after which the initial approximation for the algorithms was obtained by the truncated SVD. In addition, as before, `slra` was also initialized with the solution of Kung's algorithm. The obtained trajectories from Algorithm 1 and `slra` (initialized with the solution of Kung's algorithm) almost coincide with the true (unobserved) signal. The numerical errors are presented in Table 6.2.

Different realizations of the example lead to slightly different numerical results. We also observed that for smaller noise variance, `slra` and Algorithm 1 compute the same solution, but for higher values of the noise Algorithm 1 is generally more robust.

TABLE 6.2
Numerical errors of the initial approximation (by SVD), `slra`*, Kung's heuristic algorithm,* `slra` *initialized with Kung's algorithm's solution, and the proposed algorithm (Algorithm 1), for the example* (6.4)–(6.5) *with missing data.*

Algorithm	$\|y-\hat{y}\|^2_{given}$	$\|y_0-\hat{y}\|^2_{missing}$	$\|y_0-\hat{y}\|^2_{all}$	Remarks
`init.approx.`	8.4295	1.9481	10.2588	
`slra`	8.4123	1.9447	10.2386	
`Kung` [15]	1.2480	0.2722	0.6935	heuristic
`Kung` → `slra`	0.9228	0.1404	0.2750	
Algorithm 1	**0.9026**	**0.0511**	**0.1540**	$\frac{\|PL-P_{\mathcal{S}}(PL)\|_F^2}{\|PL\|_F^2} = 4.9 \cdot 10^{-27}$

6.3. Approximate common divisor. Another application of structured low-rank approximation and thus of Algorithm 1 is finding approximate common divisors of a set of polynomials. Existence of a nontrivial common divisor is a nongeneric property. Given a noisy observation of the polynomials' coefficients (or due to round-off errors in storing the exact polynomials coefficients in a finite precision arithmetic), the polynomials have a nontrivial common divisor with probability zero. Assuming that the noise free polynomials have a common divisor of a known degree, our aim in the approximate common divisor problem is to estimate the common divisor from the noisy data. The problem can be formulated and solved as Sylvester structured low-rank approximation problems, see [31]. Since the Sylvester matrix has fixed zero elements, in this example, we use the feature of Algorithm 1 to work with structured matrices having *fixed elements*.

Background. For a polynomial

$$a(z) \;=\; a_0 + a_1\, z + \cdots + a_n\, z^n,$$

define the multiplication matrix

$$S_k(a) = \begin{bmatrix} a_0 & a_1 & \cdots & a_n & & \mathbf{0} \\ & \ddots & \ddots & & \ddots & \\ \mathbf{0} & & a_0 & a_1 & \cdots & a_n \end{bmatrix} \Bigg\} k \quad .$$

We consider three polynomials a, b, and c and for simplicity let they be of the same degree n. A basic result in computer algebra, see, e.g., [14], is that a, b, and c have a nontrivial common divisor if and only if the generalized Sylvester matrix

$$\begin{bmatrix} S_n(b) & S_n(c) \\ S_n(a) & \mathbf{0} \\ \mathbf{0} & S_n(a) \end{bmatrix} \tag{6.6}$$

has rank at most $3n-1$. Alternatively, one can consider another type of generalized Sylvester matrix [13]

$$\begin{bmatrix} S_n(a) \\ S_n(b) \\ S_n(c) \end{bmatrix}, \tag{6.7}$$

whose rank deficiency is equal to the degree of the greatest common divisor of a, b, and c. Formulation (6.7) is more compact than the one in (6.6), especially if the

number of polynomials is large. These results are generalizable for arbitrary number of polynomials $(2, 3, \ldots)$ of possibly different degrees and arbitrary order of the required common divisor.

In the case of inexact coefficients, the generalized Sylvester matrices are generically full rank. The problem of finding the approximate common divisor is then transformed to the problem of approximating the matrix in (6.6) or in (6.7) by low-rank matrices with the same structure. This can be done with Algorithm 1 and with the alternative method `slra`. For simplicity, in our example we take three polynomials of degree 2 and desired (greatest) common divisor or order one (one common root). However, Algorithm 1 is applicable in the general case as well.

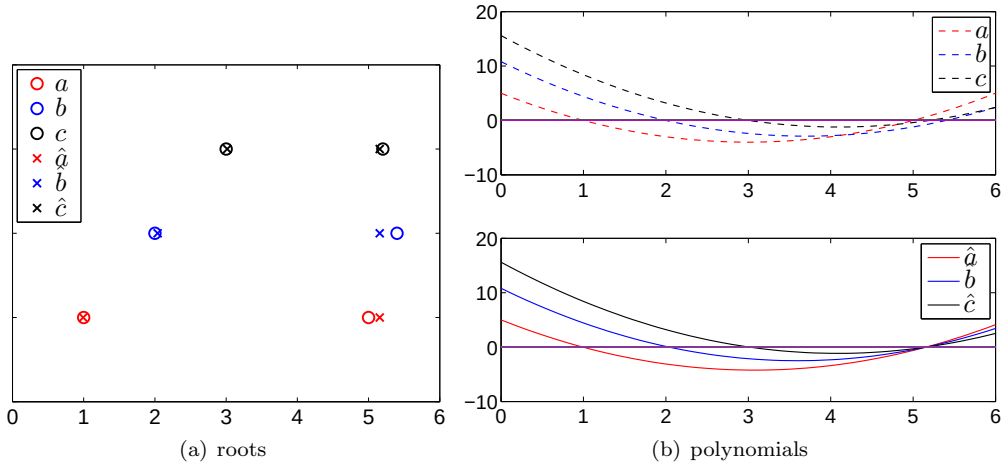
Example. Let

$$\begin{array}{rcrcl} a(z) & = & 5 - 6z + z^2 & = & (1 - z)\,(5 - z), \\ b(z) & = & 10.8 - 7.4z + z^2 & = & (2 - z)\,(5.4 - z), \\ c(z) & = & 15.6 - 8.2z + z^2 & = & (3 - z)\,(5.2 - z). \end{array}$$

Aiming at a common divisor of degree one, we approximate (6.6) and (6.7) with, respectively, rank-5 and rank-3 matrices. The obtained solution with Algorithm 1, applied on (6.7) is

$$\begin{array}{rcrcl} \hat{a}(z) & = & 4.9991 - 6.0046\,z + 0.9764\,z^2 & = & 0.9764\ (0.9928 - z)\,(5.1572 - z), \\ \hat{b}(z) & = & 10.8010 - 7.3946\,z + 1.0277\,z^2 & = & 1.0277\ (2.0378 - z)\,(5.1572 - z), \\ \hat{c}(z) & = & 15.6001 - 8.1994\,z + 1.0033\,z^2 & = & 1.0033\ (3.0149 - z)\,(5.1572 - z), \end{array}$$

with a common root 5.1572. The roots of the original and approximating polynomials, as well as the polynomials themselves are plotted in Figure 6.4. The same results



(a) roots (b) polynomials

FIG. 6.4. *Results for the example of approximate common divisor of three polynomials.*

were obtained with `slra`, applied on (6.6). Due to the non-convexity of the problem, Algorithm 1 applied on (6.6) computed a slightly different solution with comparable accuracy. Applying `slra` on (6.7) resulted in computing a common divisor of degree 2, i.e., $\hat{a} = \hat{b} = \hat{c}$.

6.4. Symmetric tensor approximation. Structured low-rank approximation can be applied to decompose and approximate complex symmetric tensors into a sum of symmetric rank-one terms. With this application we also demonstrate how the proposed algorithm can deal with *missing* and *fixed* elements.

Background. A complex tensor of dimension n and order d, $\mathcal{T} \in \mathbb{C}^{\overbrace{n \times \cdots \times n}^{d}}$ is called symmetric if it is invariant under any permutation of the modes. A symmetric tensor $\mathcal{T}$ admits a symmetric decomposition of rank r if it can be represented as a sum of r rank-one symmetric terms:

$$\mathcal{T} = \sum_{k=1}^{r} \overbrace{\mathbf{v}_k \otimes \cdots \otimes \mathbf{v}_k}^{d}, \tag{6.8}$$

where $\mathbf{v}_k \in \mathbb{C}^n$. The problem of symmetric tensor decomposition is to find a (minimal) decomposition (6.8).

In [3], it was shown that $\mathcal{T}$ has a decomposition of rank r if and only if (excluding some nongeneric cases)

$$\operatorname{rank}(\mathcal{S}(\mathcal{T}, h)) \leq r,$$

where $\mathcal{S}(\mathcal{T}, h)$ is a quasi-Hankel structured matrix constructed from the tensor and the vector h has unknown entries (latent variables). Therefore, the tensor decomposition problem is a low-rank matrix completion of the structured matrix $\mathcal{S}(\mathcal{T}, h)$. Moreover, symmetric low-rank tensor approximation is equivalent to structured low-rank approximation with missing data. The main difficulty for this reformulation is that filling in missing data is nontrivial, and gives rise to multivariate polynomial systems of equations [3].

Symmetric tensors are often represented as homogeneous polynomials [6] using

$$\mathcal{T} \longleftrightarrow \mathcal{T}(\mathbf{x}) := \mathcal{T} \times_1 \mathbf{x} \times_2 \mathbf{x} \cdots \times_d \mathbf{x},$$

where $\mathbf{x} \in \mathbb{C}^n$ and '$\times_i$' is the tensor-vector product with respect to the ith mode of the tensor. With this representation, symmetric tensor decomposition is equivalent to the Waring problem [3] of decomposing a homogeneous polynomial $\mathcal{T}(\mathbf{x})$ of degree d into a sum of d-th powers of linear forms

$$\mathcal{T}(\mathbf{x}) := \sum_{k=1}^{r} (\mathbf{v}_k^\top \mathbf{x})^d.$$

We will represent our results in the polynomial form.

Example. Consider the example from [3, §5.2], which is decomposition of a polynomial (a tensor of dimension 3 and order 4), with $\mathbf{x} = \begin{bmatrix} x_0 & x_1 & x_2 \end{bmatrix}^\top$,

$$\mathcal{T}(\mathbf{x}) = 79x_0x_1^3 + 56x_0^2x_2^2 + 49x_1^2x_2^2 + 4x_0x_1x_2^2 + 57x_1x_0^3,$$

into a sum of 6 symmetric rank-one terms. (For dimension 3 and degree 4 complex symmetric tensors, the generic rank is 6 [6]). In this case, in order to compute the decomposition with the theory of [3] it is crucial to fill in the missing data.

We considered the 10×10 submatrix of the matrix in [3, p.14]. We fixed the non-missing data (by setting them in S_0), and computed the missing elements with

Algorithm 1. The error on the deviation from the structure was around machine precision ($4.5 \cdot 10^{-31}$). The computed tensor decomposition was

$$
\begin{aligned}
\mathcal{T}(\mathbf{x}) \approx \quad & 6.94\,(x_0 + 0.895\,x_1 + 0.604\,x_2)^4 \\
& +6.94\,(x_0 + 0.895\,x_1 - 0.604\,x_2)^4 \\
& -4.94\,(x_0 - 0.982\,x_1 + 0.657\,i\,x_2)^4 \\
& +4.94\,(x_0 - 0.982\,x_1 - 0.657\,i\,x_2)^4 \\
& -(1.99 - 11.9\,i)\,(x_0 + (0.128 + 0.308\,i)\,x_1)^4 \\
& -(1.99 + 11.9\,i)\,(x_0 + (0.128 - 0.308\,i)\,x_1)^4,
\end{aligned} \tag{6.9}
$$

(where we have removed coefficients smaller than 10^{-12}). This is a different expansion from the one reported in [3, p.15]. This can be expected, because for generic ranks the tensor decompositions are usually nonunique [6]. The approximation error of (6.9) on the normalized polynomial coefficients is $1.7421 \cdot 10^{-13}$.

REMARK 2. *Instead of the method used in [3, p.15], we computed the vectors* $\mathbf{v}_k$ *by joint diagonalization of matrices* $\mathbb{M}_{x_1}$ *and* $\mathbb{M}_{x_2}$ *of the quotient algebra (see [3] for the definition of these matrices). For joint diagonalization we used the method [27] from signal processing, where approximate joint eigenvectors are computed by taking the eigenvectors of a linear combination of* $\mathbb{M}_{x_1}$ *and* $\mathbb{M}_{x_2}$*. This was needed because in the case of multiple eigenvalues of* $\mathbb{M}_{x_1}$ *(which is the case of our computed decomposition), the method in [3, p.15] does not work correctly.*

In this example, the data were exact and the goal was to compute exact rank-6 decomposition. However, Algorithm 1 can be used to solve the more general problem of tensor low-rank approximation as well.

7. Conclusions. In this paper, we introduced a novel approach for solving the structure-preserving low-rank approximation problem. We used the image representation to deal with the low-rank constraint, and a regularization technique, to impose the structure on the approximation. The original problem has been reduced to solving a series of simple least squares problems with exact solutions. We have discussed the properties of the proposed local optimization algorithm and ensured that it can solve the weighted problem and deal with the cases of missing or fixed elements. The proposed algorithm was tested on a set of numerical examples from system identification, computer algebra and symmetric tensor decomposition and compared favorably to existing algorithms.

The regularized structured low-rank approximation algorithm proposed in this paper is an attractive alternative to the kernel approach: it is more robust to the initial approximation (Section 6.2), allows us to use a simpler Sylvester matrix (6.7) in the GCD setting, and can be used for symmetric tensor decompositions, where the alternative `slra` method experiences difficulties. In contrast to algorithms based on the kernel representation, the proposed regularized structured low-rank approximation is designed for the problems requiring low ranks (small r). It is also worth noting that there are no restrictions on the values of the rank r. An efficient implementation of the algorithm and more detailed analysis of its applications in case of missing data and GCD computation are a topic of future research.

Appendix.

Proof. [Lemma 2.1.] The closest matrix in image($\mathcal{S}$) to an unstructured matrix X is the solution of the minimization problem

$$\underset{\hat{X}\,\in\,\mathrm{image}(\mathcal{S})}{\arg\min} \ \|X-\hat{X}\|_F^2,$$

where $\|\cdot\|_F$ stands for the Frobenius norm. Equivalently, we need to solve

$$\min_{p\in\mathbb{R}^{n_p}} \|X-\mathcal{S}(p)\|_F^2,$$

which can be written as

$$\min_{p\in\mathbb{R}^{n_p}} \|\mathrm{vec}(X)-\mathrm{vec}(S_0)-\mathbf{S}p\|_2^2. \tag{A.1}$$

Since, by assumption **(A)**, $\mathbf{S}$ has full column rank and $\mathbf{S}^\top\mathrm{vec}(S_0)=0$, (A.1) is a least squares problem with unique solution

$$p^*=(\mathbf{S}^\top\,\mathbf{S})^{-1}\mathbf{S}^\top\mathrm{vec}(X-S_0)=(\mathbf{S}^\top\,\mathbf{S})^{-1}\mathbf{S}^\top\mathrm{vec}(X).$$

Thus,

$$P_{\mathcal{S}}(X)=\mathcal{S}((\mathbf{S}^\top\,\mathbf{S})^{-1}\mathbf{S}^\top\mathrm{vec}(X)),$$

which completes the proof. ∎

Proof. [Lemma 5.1.] Using the following well-known equality

$$\mathrm{vec}(XYZ)=(Z^\top\otimes X)\,\mathrm{vec}(Y),$$

we have

$$\mathrm{vec}(PL)=(I_n\otimes P)\,\mathrm{vec}(L)=(L^\top\otimes I_m)\,\mathrm{vec}(P). \tag{A.2}$$

Consider first problem (5.1). Problem (5.2) can be solved in a similar way. Using (2.4) and (A.2), (5.1) can be reformulated as

$$\begin{aligned}
&\min_L \|p-\Pi_{\mathbf{S}}\,\mathrm{vec}(PL)\|_{\overline{W}}^2+\lambda\|PL-P_{\mathcal{S}}(PL)\|_F^2\\
\iff\quad &\min_L \|\overline{M}(p-\Pi_{\mathbf{S}}\,\mathrm{vec}(PL))\|_2^2+\lambda\|\mathrm{vec}(PL)-\mathrm{vec}(P_{\mathcal{S}}(PL))\|_2^2,\\
\iff\quad &\min_L \|\overline{M}(p-\Pi_{\mathbf{S}}\,\mathrm{vec}(PL))\|_2^2+\lambda\|\mathrm{vec}(PL)-\mathrm{vec}(S_0)-\mathbf{S}\,\Pi_{\mathbf{S}}\,\mathrm{vec}(PL)\|_2^2,\\
\iff\quad &\min_L \|\overline{M}\Pi_{\mathbf{S}}\,\mathrm{vec}(PL))-\overline{M}p\|_2^2+\|\sqrt{\lambda}(I_{mn}-\mathbf{S}\,\Pi_{\mathbf{S}})\,\mathrm{vec}(PL)-\sqrt{\lambda}\mathrm{vec}(S_0)\|_2^2,\\
\iff\quad &\min_L \left\|\begin{bmatrix}\overline{M}\,\Pi_{\mathbf{S}}\\ \sqrt{\lambda}(I_{mn}-\mathbf{S}\,\Pi_{\mathbf{S}})\end{bmatrix}\mathrm{vec}(PL)-\begin{bmatrix}\overline{M}p\\ \sqrt{\lambda}\mathrm{vec}(S_0)\end{bmatrix}\right\|_2^2,\\
\iff\quad &\min_L \left\|\begin{bmatrix}\overline{M}\,\Pi_{\mathbf{S}}\\ \sqrt{\lambda}(I_{mn}-\mathbf{S}\,\Pi_{\mathbf{S}})\end{bmatrix}(I_n\otimes P)\,\mathrm{vec}(L)-\begin{bmatrix}\overline{M}p\\ \sqrt{\lambda}\mathrm{vec}(S_0)\end{bmatrix}\right\|_2^2.
\end{aligned}$$

The derivation for P is analogous. ∎

REFERENCES

[1] D. P. Bertsekas, *Nonlinear programming*, Athena Scientific, 1999.

[2] R. Borsdorf, *Structured matrix nearness problems: Theory and algorithms*, PhD thesis, The University of Manchester, 2012.

[3] J. Brachat, P. Comon, B. Mourrain, and E. Tsigaridas, *Symmetric tensor decomposition*, Linear Algebra and its Applications, 433 (2010), pp. 1851–1872.

[4] J. A. Cadzow and D. M. Wilkes, *Signal enhancement and the SVD*, in Proceedings of the 2nd International Workshop on SVD and Signal Processing, University of Rhode Island, Kingston, RI, 1990, pp. 144–151.

[5] M. T. Chu, R. E. Funderlic, and R. J. Plemmons, *Structured low rank approximation*, Linear algebra and its applications, 366 (2003), pp. 157–172.

[6] P. Comon, *Tensors versus matrices usefulness and unexpected properties*, in IEEE/SP 15th Workshop on Statistical Signal Processing, 2009. SSP '09., 2009, pp. 781–788.

[7] B. De Moor, *Structured total least squares and L_2 approximation problems*, Linear Algebra Appl., 188–189 (1993), pp. 163–207.

[8] ———, *Total least squares for affinely structured matrices and the noisy realization problem*, IEEE Trans. Signal Proc., 42 (1994), pp. 3104–3113.

[9] M. Fazel, TK Pong, D. Sun, and P. Tseng, *Hankel matrix rank minimization with applications in system identification and realization*, SIAM J. Matrix Anal. Appl., 2 (2012), pp. 123–144.

[10] G. Golub and V. Pereyra, *Separable nonlinear least squares: the variable projection method and its applications*, Inverse Problems, 19 (2003), pp. R1–R26.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 3rd ed., 1996.

[12] L. Grippo and M. Sciandrone, *On the convergence of the block nonlinear Gauss-Seidel method under convex constraints*, Operations Research Letters, 26 (2000), pp. 127–136.

[13] N. Karcanias, S. Fatouros, M. Mitrouli, and G.H. Halikias, *Approximate greatest common divisor of many polynomials, generalised resultants, and strength of approximation*, Computers & Mathematics with Applications, 51 (2006), pp. 1817–1830.

[14] N. Karmarkar and Y. Lakshman, *On approximate GCDs of univariate polynomials*, in J. Symbolic Comput., S. Watt and H. Stetter, eds., vol. 26, 1998, pp. 653–666. Special issue on Symbolic Numeric Algebra for Polynomials.

[15] S. Kung, *A new identification method and model reduction algorithm via singular value decomposition*, in Proc. 12th Asilomar Conf. Circuits, Systems, Computers, Pacific Grove, 1978, pp. 705–714.

[16] Z. Liu, A. Hansson, and L. Vandenberghe, *Nuclear norm system identification with missing inputs and outputs*, 62 (2013), pp. 605–612.

[17] Z. Liu and L. Vandenberghe, *Interior-Point method for nuclear norm approximation with application to system identification*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 1235–1256.

[18] I. Markovsky, *Structured low-rank approximation and its applications*, Automatica, 44 (2008), pp. 891–909.

[19] ———, *Low Rank Approximation: Algorithms, Implementation, Applications*, Springer, 2012.

[20] I. Markovsky and K. Usevich, *Software for weighted structured low-rank approximation*, J. Comput. Appl. Math., (2013).

[21] ———, *Structured low-rank approximation with missing data*, SIAM J. Matrix Anal. Appl., 34 (2013), pp. 814–830.

[22] I. Markovsky, S. Van Huffel, and R. Pintelon, *Block-Toeplitz/Hankel structured total least squares*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 1083–1099.

[23] D. Marquardt, *An algorithm for least-squares estimation of nonlinear parameters*, SIAM J. Appl. Math., 11 (1963), pp. 431–441.

[24] N. Mastronardi, P. Lemmerling, and S. Van Huffel, *Fast structured total least squares algorithm for solving the basic deconvolution problem*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 533–553.

[25] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer Series in Operations Research, Springer, 2nd ed., 2006.

[26] H. Park, L. Zhang, and J. B. Rosen, *Low rank approximation of a hankel matrix by structured total least norm*, BIT Numerical Mathematics, 39 (1999), pp. 757–779.

[27] S. Rouquette and M. Najim, *Estimation of frequencies and damping factors by two-dimensional esprit type methods*, IEEE Transactions on Signal Processing, 49 (2001), pp. 237–245.

[28] M. Schuermans, P. Lemmerling, and S. Van Huffel, *Structured weighted low rank approx-*

imation, Numerical linear algebra with applications, 11 (2004), pp. 609–618.

[29] N. Srebro and T. Jaakkola, *Weighted low-rank approximations.*, in ICML, Tom Fawcett and Nina Mishra, eds., AAAI Press, 2003, pp. 720–727.

[30] K. Usevich and I. Markovsky, *Variable projection for affinely structured low-rank approximation in weighted 2-norms*, J. Comput. Appl. Math., (2013).

[31] ———, *Variable projection methods for approximate (greatest) common divisor computations*, tech. report, Vrije Univ. Brussel, 2013. Available from `http://arxiv.org/abs/1304.6962`, Submitted to *Journal of Symbolic Computation*.

[32] P. Van Overschee and B. De Moor, *Subspace identification for linear systems: Theory, implementation, applications*, Boston, 1996.