Structured low-rank approximation with missing data

Ivan Markovsky and Konstantin Usevich

School of Electronics and Computer Science
University of Southampton
{im,ku}@ecs.soton.ac.uk

Abstract

The approach of *SIAM J. Matrix Anal. Appl.*, 26(4):1083–1099 for solving structured total least squares problems is generalized to weighted structured low-rank approximation with missing data. The method proposed is based on elimination of the correction matrix and solution of the resulting nonlinear least squares problem by local optimization methods. The elimination step is a singular linear least-norm problem, which admits an analytic solution. Two approaches are proposed for the nonlinear least-squares minimization: minimization subject to equality constraints and unconstrained minimization with regularized cost function. The method is generalized to weighted low-rank approximation with singular weight matrix and is illustrated on matrix completion, system identification, and data-driven simulation problems. An extended version of the paper is a literate program, implementing the method and reproducing the presented results.

Keywords: low-rank approximation, structured total least squares, variable projections, missing data, system identification.

1 Introduction and notation

The paper describes a solution method for matrix structured low-rank approximation, *i.e.*, approximation of a given matrix by another matrix whose elements satisfy certain predefined relations (matrix structure) and whose rank is less than or equal to a predefined value. The combination of matrix and low-rank structure makes structured low-rank approximation a tool for data modeling. Low-rank property of a matrix is equivalent to existence of an exact low-complexity linear model for the data. Moreover, the rank of the matrix is related to the complexity of the model. The structure, imposed on the approximation, is related to properties of the model. For example, Hankel structure corresponds to time-invariance of a linear dynamical model for the data.

Structured low-rank approximation has been studied in the literature from different viewpoints: numerical algorithm for computing locally optimal or suboptimal solutions, statistical properties of the resulting estimators, and applications. The subject is closely related to the structured total least squares method. The similarities and differences between the low-rank approximation and total least squares paradigms are well documented in the literature [12, 6, 8] and will not be repeated here.

A novel feature of the low-rank approximation problem, considered in this paper, is that elements of the data matrix can be missing (not specified). Missing data may occur in practical applications due to malfunctioning of measurement device, communication channel, or storing device. In such cases, the best strategy is to collect a complete data record by repeating the data collection experiment. In other applications, however, the missing data problem is intrinsic and can not be avoided by repeated experiments. An example of such an application is the prediction of the user ratings of products (recommendation systems), where all users rate some, but rarely all, products and the task is to predict the missing ratings. Methods for solving unstructured low-rank approximation problems with missing data have been proposed in the literature [4, 5, 16, 1, 7], however, to the best of the authors knowledge none of these methods can deal with matrix structure. In this paper, we generalize the method of [13] to structured low-rank approximation with missing data.

Problem formulation

We denote missing data values by the symbol `NaN` ("not a number"). The considered low-rank approximation problem is

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} & \displaystyle\sum_{\{i \mid p_i \neq \texttt{NaN}\}} (p_i - \widehat{p}_i)^2 \\ \text{subject to} & \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r, & \end{array} \qquad \text{(SLRA)}$$

where

$$\mathscr{S} : \mathbb{R}^{n_p} \to \mathbb{R}^{m \times n}, \qquad \text{defined by} \quad \mathscr{S}(\widehat{p}) = S_0 + \sum_{i=1}^{n_p} S_i \widehat{p}_i, \qquad (\mathscr{S})$$

2a ⟨*default* `s0` 2a⟩≡ (4 9b)

```
  if ~exist('s0') || isempty(s0), s0 = zeros(m, n); end
```

is the matrix structure—an affine function from the structure parameter space $\mathbb{R}^{n_p}$ to the set of matrices $\mathbb{R}^{m \times n}$. With $\mathscr{G}$ denoting the vector of indices of the given values $\{ i \mid p_i \neq \texttt{NaN} \}$ (in decreasing order) and $p_{\mathscr{G}}$ denoting the subvector of p with indices in $\mathscr{G}$, the approximation criterion can be written as

$$\sum_{i \in \mathscr{G}} (p_i - \widehat{p}_i)^2 = \| p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}} \|_2^2.$$

Using the kernel representation of the rank constraint

$$\operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r \quad \iff \quad \text{there is } R \in \mathbb{R}^{(m-r) \times m}, \text{ such that } R\mathscr{S}(\widehat{p}) = 0 \text{ and } R \text{ has full row rank},$$

the following equivalent problem to (SLRA) is obtained

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} \text{ and } R \in \mathbb{R}^{(m-r) \times m} & \| p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}} \|_2^2 \\ \text{subject to} & R\mathscr{S}(\widehat{p}) = 0 \quad \text{and} \quad R \text{ has full row rank.} & \end{array} \qquad (\text{SLRA}_R)$$

Problem (SLRA$_R$) is a double minimization over the parameters R and $\widehat{p}$

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r) \times m} \quad M(R) \quad \text{subject to} \quad R \text{ has full row rank}, \qquad (\text{SLRA}'_R)$$

where

$$M(R) := \min_{\widehat{p}} \| p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}} \|_2^2 \quad \text{subject to} \quad R\mathscr{S}(\widehat{p}) = 0. \qquad \text{(INNER)}$$

The evaluation of the cost function M, *i.e.*, solving (INNER) for a given value of R, is refered to as the *inner minimization problem*. This problem is solved analytically in Section 2. The remaining problem of minimizing M over R is refered to as the *outer minimization problem*. It is a nonlinear least-squares problem, which, in general, admits no analytic solution. General purpose local optimization methods are used in Section 4 for its numerical solution. In Section 3, the approach is generalized to weighted 2-norm approximation criteria with singular weight matrix. Numerical examples of solving approximation problems with missing data by the proposed methods are shown in Section 5.

Notation

In the rest of the paper, we use the following notation.

- $A_{\mathscr{I},\mathscr{J}}$ is the submatrix of A with rows in $\mathscr{I}$ and columns in $\mathscr{J}$. The row/column index can be replaces by the symbol ":", in which case all rows/columns are selected.

- $\mathscr{M}$ / $\mathscr{G}$ is the vector of indices of p (in decreasing order) that are missing / given.

2b ⟨*define* $\mathscr{M}$ *and* $\mathscr{G}$ 2b⟩≡ (4 9a)

```
  Im = find(isnan(p)); Ig = setdiff(1:np, Im);
```

- A^{+} is the pseudo inverse of A and $A^{\perp}$ is a matrix which rows form a basis for the left null space of A.

3a ⟨*define* perp 3a⟩≡ (3c)

```
perp = @(a) null(a')';
```

- For given $\mathscr{S}$ and $R \in \mathbb{R}^{(m-r)\times m}$, we define the matrix

$$G := \begin{bmatrix} \operatorname{vec}(RS_1) & \cdots & \operatorname{vec}(RS_{n_p}) \end{bmatrix} \in \mathbb{R}^{(m-r)n\times n_p}, \tag{G}$$

  where $\operatorname{vec}(\cdot)$ is the column-wise vectorization operator.

3b ⟨*define* G 3b⟩≡ (4)

```
g = reshape(R * phi * reshape(bfs, mp, n * np), size(R, 1) * n, np);
```

2 Analytical solution of the inner minimization problem

In this section, we consider the inner minimization problem (INNER).

Problem 1. Given linear structure $\mathscr{S}$, structure parameter vector $p \in \mathbb{R}^{n_p} \cup \mathtt{NaN}$, and a kernel parameter $R \in \mathbb{R}^{(m-r)\times m}$, find the cost function $M(R)$, defined in (INNER), and a value of $\widehat{p}$ that attains the minimum.

Theorem 2. *Under the following assumptions:*

1. $G_{:,\mathscr{M}}$ *is full column rank,*

2. $1 \leq (m-r)n - n_{\mathrm{m}} \leq n_p - n_{\mathrm{m}}$, *and*

3. $\bar{G} := G^{\perp}_{:,\mathscr{M}} G_{:,\mathscr{G}}$ *is full row rank,*

3c ⟨*define* $\bar{G}$ 3c⟩≡ (4)

```
⟨define perp 3a⟩, perp_gm = perp(g(:, Im)); bg = perp_gm * g(:, Ig);
```

Problem 1 has a unique minimum

$$M(R) = s^{\top}\left(\bar{G}\bar{G}^{\top}\right)^{-1} s, \qquad \textit{where} \quad s := \left(\bar{G} p_{\mathscr{G}} - G^{\perp}_{:,\mathscr{M}} \operatorname{vec}(RS_0)\right), \tag{M}$$

attained by

$$\widehat{p}_{\mathscr{G}} = p_{\mathscr{G}} - \bar{G}^{\top}\left(\bar{G}\bar{G}^{\top}\right)^{-1} s \qquad \textit{and} \qquad \widehat{p}_{\mathscr{M}} = -G^{+}_{:,\mathscr{M}} G_{:,\mathscr{G}} \widehat{p}_{\mathscr{G}}. \tag{$\widehat{p}$}$$

3d ⟨*compute* M *and* $\widehat{p}$ 3d⟩≡ (4)

```
dpg = bg' * pinv(bg * bg') * (bg * p(Ig) - perp_gm * vec(R * s0));
M = dpg' * dpg; ph(Ig) = p(Ig) - dpg; ph = ph(:);
ph(Im) = - pinv(g(:, Im)) * g(:, Ig) * ph(Ig);
```

Proof. Defining

$$\Delta p_{\mathscr{G}} := p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}}$$

and using the identity

$$R\mathscr{S}(\widehat{p}) = 0 \qquad \Longleftrightarrow \qquad G\widehat{p} = -\operatorname{vec}(RS_0),$$

we have

$$R\mathscr{S}(\widehat{p}) = -\operatorname{vec}(RS_0) \qquad \Longleftrightarrow \qquad \begin{bmatrix} G_{:,\mathscr{G}} & G_{:,\mathscr{M}} \end{bmatrix} \begin{bmatrix} p_{\mathscr{G}} - \Delta p_{\mathscr{G}} \\ \widehat{p}_{\mathscr{M}} \end{bmatrix} = -\operatorname{vec}(RS_0).$$

Therefore, (INNER) is equivalent to

$$M(R) := \min_{\Delta p_{\mathscr{G}} \in \mathbb{R}^{n_p - n_{\mathrm{m}}},\ \widehat{p}_{\mathscr{M}} \in \mathbb{R}^{n_{\mathrm{m}}}} \|\Delta p_{\mathscr{G}}\|_2^2 \quad \text{subject to} \quad \begin{bmatrix} G_{:,\mathscr{G}} & G_{:,\mathscr{M}} \end{bmatrix} \begin{bmatrix} \Delta p_{\mathscr{G}} \\ -\widehat{p}_{\mathscr{M}} \end{bmatrix} = G_{:,\mathscr{G}} p_{\mathscr{G}} - \operatorname{vec}(RS_0),$$

which is a generalized linear least norm problem. The solution follows from Lemma 3. □

```
4   ⟨misfit_ext 4⟩≡
      function [M, ph] = misfit_ext(R, tts, p, w, bfs, phi)
      ⟨S ↦ (m,n,n_p) 16a⟩
      ⟨default phi 16d⟩
      ⟨default s0 2a⟩
      if ~exist('bfs') | isempty(bfs), ⟨S ↦ S 16b⟩, end
      ⟨define ℳ and 𝒢 2b⟩
      ⟨preprocess p and bfs with w 5b⟩
      ⟨define G̲ 3b⟩
      ⟨define G̅ 3c⟩
      ⟨compute M and p̂ 3d⟩
      ⟨postprocess ph with w 6a⟩
```

Generalized least norm problem

Lemma 3. *Consider the generalized linear least norm problem*

$$f = \min_{x,y} \quad \|x\|_2^2 \quad \textit{subject to} \quad Ax + By = c, \tag{GLN}$$

with $A \in \mathbb{R}^{m \times n_x}$, $B \in \mathbb{R}^{m \times n_y}$, and $c \in \mathbb{R}^m$. Under the following assumptions:

1. *B is full column rank,*
2. *$1 \leq m - n_y \leq n_x$, and*
3. *$\bar{A} := B^{\perp}A$ is full row rank,*

problem (GLN) has a unique solution

$$\begin{aligned} &f = c^{\top}(B^{\perp})^{\top}\left(\bar{A}\bar{A}^{\top}\right)^{-1}B^{\perp}c, \\ &x = A^{\top}(B^{\perp})^{\top}\left(\bar{A}\bar{A}^{\top}\right)^{-1}B^{\perp}c \qquad \textit{and} \qquad y = B^{+}(c - Ax). \end{aligned} \tag{SOL}$$

Proof. Under assumption 1, B has a nontrivial left kernel of dimension $m - n_y$. Therefore for the nonsingular matrix $T = \begin{bmatrix} B^+ \\ B^{\perp} \end{bmatrix} \in \mathbb{R}^{m \times m}$

$$TB = \begin{bmatrix} B^+ \\ B^{\perp} \end{bmatrix} B = \begin{bmatrix} T^+B \\ T^{\perp}B \end{bmatrix} = \begin{bmatrix} I_{n_y} \\ 0 \end{bmatrix}.$$

Pre-multiplying both sides of the constraint of (GLN) by T, we have the following equivalent constraint

$$\begin{bmatrix} B^+Ax \\ B^{\perp}Ax \end{bmatrix} + \begin{bmatrix} y \\ 0 \end{bmatrix} = \begin{bmatrix} B^+c \\ B^{\perp}c \end{bmatrix}.$$

The first equation

$$y = B^+(c - Ax)$$

uniquely determines y, given x. The second equation

$$B^{\perp}Ax = B^{\perp}c \tag{$*$}$$

defines a linear constraint for x only. By assumption 2, it is an underdetermined system of linear equations. Therefore, (GLN) is equivalent to the following standard least norm problem

$$f = \min_{x} \ \|x\|_2^2 \quad \text{subject to} \quad B^{\perp}Ax = B^{\perp}c. \tag{GLN'}$$

By assumption 3 the solution is unique and is given by (SOL). □

Note 4 (About assumptions 1–3). Assumption 1 is a necessary condition for uniqueness of the solution. Relaxing assumptions 1 implies that any vector in the affine space

$$\mathscr{Y} = B^{+}(c - Ax) + \operatorname{null}(B)$$

is a solution to (GLN). Assumption 2 ensures that the problem is a least norm problem and has a nontrivial solution. In the case $m = n_y$, the problem has a trivial solution $f = 0$. In the case $m - n_y > n_x$, the problem generically has no solution because the constraint $(*)$ is an overdetermined system of equations. Assumption 3 is also required for uniqueness of the solution. It can also be relaxed, making y nonunique.

Note 5 (Link to weigted least norm problems with singular weight matrix). Consider the weighted least norm problem

$$\min_{z}\; z^{\top} W z \quad \text{subject to} \quad Dz = c,$$

with singular positive semidefinite weight matrix W. Using a change of variables $\bar{z} = T^{-1}z$, where T is an nonsingular matrix, we obtain the equivalent problem

$$\min_{z}\; \bar{z}^{\top} T^{\top} W T \bar{z} \quad \text{subject to} \quad DT\bar{z} = c.$$

There exists an nonsingular matrix T, such that

$$T^{\top} W T = \begin{bmatrix} I_{n_x} & \\ & 0 \end{bmatrix}.$$

Partitioning $\bar{z}$ and $\bar{D} := DT^{-1}$ conformably as

$$\bar{z} = \begin{bmatrix} x \\ y \end{bmatrix} \qquad \text{and} \qquad \bar{D} = \begin{bmatrix} A & B \end{bmatrix}$$

we obtained problem (GLN).

3 Weighted approximation

Problem (SLRA) is generalized in this section to the weighted structured low-rank approximation problem

$$\begin{array}{lll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} & (p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}})^{\top} W_{\mathrm{g}} (p_{\mathscr{G}} - \widehat{p}_{\mathscr{G}}) \\ \text{subject to} & \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r, & \end{array} \tag{WSLRA}$$

where W_{g} is a positive definite matrix. The change of variables

$$p'_{\mathscr{G}} = \sqrt{W_{\mathrm{g}}}\, p_{\mathscr{G}} \qquad \text{and} \qquad \widehat{p}'_{\mathscr{G}} = \sqrt{W_{\mathrm{g}}}\, \widehat{p}_{\mathscr{G}} \tag{$p \mapsto p'$}$$

reduces Problem (WSLRA) to an equivalent unweighted problem (SLRA). We have

$$\mathscr{S}(\widehat{p}) = S_0 + \operatorname{vec}^{-1}(\mathbf{S}\widehat{p}), \qquad \text{where} \quad \mathbf{S} := \begin{bmatrix} \operatorname{vec}(S_1) & \cdots & \operatorname{vec}(S_{n_p}) \end{bmatrix} \in \mathbb{R}^{mn \times n_p}. \tag{$\mathbf{S}$}$$

5a ⟨$(\mathbf{S}, \widehat{p}) \mapsto \widehat{D} = \mathscr{S}(\widehat{p})$ 5a⟩≡

```
  dh = phi * reshape(bfs * ph, mp, n);
```

The structure $\mathscr{S}'$ of the equivalent problem is defined by the matrices S_0 and $\mathbf{S}' = \begin{bmatrix} \operatorname{vec}(S'_1) & \cdots & \operatorname{vec}(S'_{n_p}) \end{bmatrix}$, where

$$\mathbf{S}'_{:,\mathscr{G}} = \mathbf{S}_{:,\mathscr{G}} \sqrt{W_{\mathrm{g}}^{-1}} \qquad \text{and} \qquad \mathbf{S}'_{:,\mathscr{M}} = \mathbf{S}_{:,\mathscr{M}}. \tag{$\mathscr{S} \mapsto \mathscr{S}'$}$$

We showed that problem (WSLRA) is solved by:

1. preprocessing the data p and the structure $\mathscr{S}$, as in $(p \mapsto p')$ and $(\mathscr{S} \mapsto \mathscr{S}')$,

5b ⟨*preprocess* `p` *and* `bfs` *with* `w` 5b⟩≡ (4 9a)

```
  if exist('w') & ~isempty(w)
    sqrt_w = sqrtm(w); inv_sqrt_w = pinv(sqrt_w); bfs = double(bfs);
    p(Ig) = sqrt_w * p(Ig); bfs(:, Ig) = bfs(:, Ig) * inv_sqrt_w;
  end
```

2. solving the equivalent unweighted problem with structure parameter vector p', structure specification $\mathscr{S}'$, and rank specification r, and

3. postprocessing the solution $\widehat{p}'$, obtained in step 2, in order to obtain the solution $\widehat{p}_{\mathscr{G}} = \sqrt{W_{\mathrm{g}}^{-1}}\,\widehat{p}'_{\mathscr{G}}$ of the original problem.

6a ⟨*postprocess* ph *with* w 6a⟩≡ (4 9a)

```
  if exist('w') & ~isempty(w), ph(Ig) = inv_sqrt_w * ph(Ig); end
```

Using the transformation $(p \mapsto p')$, $(\mathscr{S} \mapsto \mathscr{S}')$ and the solution (M) of (SLRA), we obtain the following explicit expression for the cost function of (WSLRA)

$$M(R) = \big(\bar{G}p_{\mathscr{G}} - G^{\perp}_{:,\mathscr{M}}\operatorname{vec}(RS_0)\big)^{\top} W_{\mathrm{g}}^{-1}\bar{G}^{\top}\big(\bar{G}W_{\mathrm{g}}^{-1}\bar{G}^{\top}\big)^{-1}\bar{G}W_{\mathrm{g}}^{-1}\big(\bar{G}p_{\mathscr{G}} - G^{\perp}_{:,\mathscr{M}}\operatorname{vec}(RS_0)\big), \qquad (\mathrm{M}_W)$$

where $\bar{G} = G^{\perp}_{:,\mathscr{M}}G_{:,\mathscr{G}}$ and G is defined in (G).

Note 6 (Weighted structured low-rank approximation with a singular weight matrix). A more general formulation of problem (WSLRA) is

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{p} \in \mathbb{R}^{n_p} \quad (p - \widehat{p})^{\top} W (p - \widehat{p}) \\ \text{subject to} & \operatorname{rank}\big(\mathscr{S}(\widehat{p})\big) \leq r, \end{array} \qquad \text{(WSLRA')}$$

with positive semidefinite weight matrix W. Problem (WSLRA') can be reduced to an equivalent unweighted low-rank approximation problem with missing data (SLRA). There exists a nonsingular matrix T, such that

$$T^{\top}WT = \begin{bmatrix} I_{n_{\mathrm{g}}} & \\ & 0 \end{bmatrix}.$$

Defining structure $\mathscr{S}'$, specified by the matrices S_0 and $\mathbf{S}' = \mathbf{S}T$, and changing the variables $p' = T^{-1}p$, $\widehat{p}' = T^{-1}\widehat{p}$, we obtain an equivalent problem in the form (SLRA) with last $n_p - \operatorname{rank}(W)$ values missing.

Note 7 (Solving (SLRA) as weighted unstructured problem). Consider an instance of problem (SLRA), refer to as problem P1, with structure $\mathscr{S} = \mathscr{S}_1$ and an instance of problem (WSLRA), refer to as problem P2, with unstructured correction and weight matrix

$$W = \mathbf{S_1}\mathbf{S_1}^{\top}. \qquad (\mathscr{S} \mapsto W)$$

6b ⟨s2w 6b⟩≡

```
  function w = s2w(s)
  tts = s2s(s);
  ⟨S ↦ (m,n,n_p) 16a⟩
  ⟨S ↦ S 16b⟩
  w = double(bfs) * double(bfs)';
```

It can be verified by inspection that the cost functions (M) and (M_W) of problems P1 and P2, respectively, coincide. The weight matrix $W \in \mathbb{R}^{mn \times mn}$, defined in $(\mathscr{S} \mapsto W)$, however is singular ($\operatorname{rank}(W)$ is equal to the number of structure parameters of problem P1, which is less than mn). In the derivation of the cost function (M_W) it is assumed that W_{g} is positive definite, so that minimization of (M_W) is not equivalent to problem P2. Using pseudo-inverse instead of inverse in $(p \mapsto p')$ and $(\mathscr{S} \mapsto \mathscr{S}')$ and observing that $(W^{+})^{+} = W$, minimization of (M_W) for problem P2 yields and equivalent problem to problem P1.

6c ⟨*test equivalence of structure and weights* 6c⟩≡

```
  % Hankel
  s1.m = 2; s1.n = 5; r = 1;
  np1 = s2np(s1); p1 = 0.8 .^ (1:np1)' + 0.01 * randn(np1, 1);
  opt.solver = 'm';
  [ph1, info1] = slra(p1, s1, r, opt);
  Dh1 = blkhank(ph1, s1.m);
  D = blkhank(p1, s1.m); norm(D - Dh1, 'fro')
```

```
% Weighed
s2.m = ones(s1.m, 1); s2.n = ones(s1.n, 1); s2.w = pinv(s2w(s1));
p2 = vec(D);
[ph2, info2] = slra(p2, s2, r, opt);
Dh2 = reshape(ph2, s1.m, s1.n); norm(D - Dh2, 'fro')
%norm(p1 - [Dh2(1:s1.m, 1); Dh2(end, 2:s1.n)'])

vec(D - Dh2)' * s2.w * vec(D - Dh2)

addpath ~/mfiles/wtls
[R, P] = lra(Dh2, r);
opt.p0 = P; opt.MaxIter = 1000; opt.TolFun = 1e-10; opt.Display = 'off';
[Ph, M, Dh3, info3] = wtlsap(D, r, s2.w, opt)

vec(D - Dh3)' * s2.w * vec(D - Dh3)
```

4 Outer minimization problem

The outer minimization problem (SLRA$_R$) is a nonlinear least-squares problem, which we solve by general purpose local optimization methods. In order to apply standard optimization methods, however, we need first to replace the rank constraint with equivalent equality or inequality constraints.

The kernel parameter R is constrained to have a specified structure

$$\mathscr{R} : \mathbb{R}^{n_\theta} \to \mathbb{R}^{(m-r)\times m},$$

i.e., $R = \mathscr{R}(\theta)$, for some $\theta \in \mathbb{R}^{n_\theta}$. An example of a kernel structure $\mathscr{R}$ is a linear function

$$R = \mathscr{R}(\theta) := \operatorname{vec}^{-1}_{m-r}(\theta\Psi), \qquad (\theta \mapsto R)$$

defined by a matrix $\Psi \in \mathbb{R}^{n_\theta \times (m-r)m}$.

7a $\langle$*default* th2R 7a$\rangle \equiv$ (9b)

```
if ~exist('th2R'), th2R = @(th) reshape(th * psi, m - r, m); end
```

7b $\langle$*default* psi 7b$\rangle \equiv$ (9b)

```
if ~exist('psi', 'var') | isempty(psi), psi = eye(m * (m - r)); end
```

The full row rank constraint on R is equivalent to and can be enforced in the parameter optimization method by the equality constraint

$$RR^\top = I_{m-r}. \qquad (\text{f.r.r. } R)$$

$$\mathscr{C}(\theta) := \mathscr{R}(\theta)\mathscr{R}^\top(\theta) - I_{m-r} = 0. \qquad (\text{f.r.r. } R)$$

7c $\langle$*default* $\mathscr{C}$ 7c$\rangle \equiv$ (9b)

```
if ~exist('C'), C = @(th) th2R(th) * th2R(th)' - eye(m - r); end
```

Then the outer minimization problem becomes a constrained nonlinear least squares problems

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) \quad \text{subject to} \quad RR^\top - I_{m-r} = 0, \qquad (\text{SLRA}'_R)$$

which can be solved by general purpose constrained optimization methods [14]. Another approach of solving the outer minimization problem is to reformulate it as a regularized unconstrained nonlinear least squares problem by adding the regularization term $\gamma\|RR^\top - I_{m-r}\|_\text{F}^2$ to the cost function, *i.e.*,

$$\text{minimize} \quad \text{over } R \in \mathbb{R}^{(m-r)\times m} \quad M(R) + \gamma\|RR^\top - I_{m-r}\|_\text{F}^2. \qquad (\text{SLRA}''_R)$$

The parameter γ should be chosen "large enough" in order to enforce the constraint (f.r.r. R). A corollary of the following theorem shows that $\gamma = \|p_\mathscr{G}\|_2^2$ is sufficiently large.

7d $\langle$*define* γ 7d$\rangle \equiv$ (9a)

```
if ~exist('opt') || ~isfield(opt, 'g') || isempty(opt.g), opt.g = norm(p(Ig)) ^ 2; end
```

Theorem 8. *Let $M : \mathbb{R}^{(m-r)\times m} \to \mathbb{R}_+$ be a homogeneous function,* i.e., *$M(R) = M(TR)$, for any R and a nonsingular $m\times m$ matrix T. The optimal solutions of problem (SLRA$''_R$) with $\gamma = \max_R M(R)$ coincide with the optimal solutions of (SLRA$'_R$).*

Proof. Let R be an solution to (SLRA$''_R$). We will show that

$$\|RR^\top - I_{m-r}\|_\text{F}^2 = m - r - \operatorname{rank}(R). \tag{$*$}$$

There exists an orthogonal matrix U diagonalizing $RR^\top$. We have

$$\begin{aligned}\|RR^\top - I_{m-r}\|_\text{F}^2 &= \|URR^\top U^\top - I_{m-r}\|_\text{F}^2 \\ &= \|\operatorname{diag}(a_1,\dots,a_{\operatorname{rank}(R)},0,\dots,0) - I_{m-r}\|_\text{F}^2, \qquad \text{where } a_i > 0 \\ &= \sum_{i=1}^{\operatorname{rank}(R)} (a_i - 1)^2 + m - r - \operatorname{rank}(R).\end{aligned}$$

Suppose that $a_i \neq 1$ for some i. The matrix

$$R' = \operatorname{diag}(1,\dots,1,1/\sqrt{a_i},1,\dots,1)R$$

has the same kernel and rank as R, so that by the homogeneity property of M, $M(R) = M(R')$. However, we have

$$\|RR^\top - I_{m-r}\|_\text{F}^2 > \|R'R'^\top - I_{m-r}\|_\text{F}^2,$$

so that R' achieves smaller value of the cost function of (SLRA$''_R$) than R. This is a contradiction. Therefore, $a_i = 1$ for all i. This concludes the proof of ($*$).

So far we showed that the cost function of (SLRA$''_R$) is

$$M(R) + \gamma\big(m - r - \operatorname{rank}(R)\big). \tag{M''}$$

Denote by M_r^* the optimal value of (SLRA$'_R$) (the index in the subscript is the upper bound for the rank) and note that the optimal value of (SLRA$''_R$) is equal to M_r^* provided that the solution R of (SLRA$''_R$) is full row rank. Therefore, in order to prove the theorem it is sufficient to show that the cost function of (SLRA$''_R$) achieves its minimum for a full rank R, i.e.,

$$M_r^* < M_{r-i}^* + \gamma i \quad \iff \quad \gamma > \frac{1}{i}(M_r^* - M_{r-i}^*), \qquad \text{for } i = 1,2,\dots,m-r. \tag{$**$}$$

Since $M_r^* > 0$, $\gamma = \max_R M(R)$ is a sufficient condition for ($**$). □

8a ⟨*set optimization solver and options* 8a⟩≡ (9a)

```
prob = optimset();
reg = exist('opt') && isfield(opt, 'method') && strcmp(opt.method, 'reg');
if reg
  prob.solver = 'fminunc';
else
  prob.solver = 'fmincon';
end
prob.options = optimset('disp', 'off');
prob.x0 = R2th(Rini, phi * p(tts), psi);
```

8b ⟨*call optimization solver* 8b⟩≡ (9a)

```
if reg
  [x, fval, flag, info] = fminunc(prob);
else
  [x, fval, flag, info] = fmincon(prob);
end
info.fmin = fval;
```

9a ⟨*nonlinear optimization over R* 9a⟩≡ (9b)

```
⟨set optimization solver and options 8a⟩
⟨define ℳ and 𝒢 2b⟩
⟨preprocess p and bfs with w 5b⟩
if reg
  ⟨define γ 7d⟩
  prob.objective = @(th) misfit_ext(th2R(th), tts, p, [], bfs, phi) ...
                         + opt.g * norm(C(th), 'fro') ^ 2;
else
  prob.objective = @(th) misfit_ext(th2R(th), tts, p, [], bfs, phi);
  prob.nonlcon = @(th) deal([], C(th));
end
⟨call optimization solver 8b⟩, info.Rh = th2R(x);
[M, ph] = misfit_ext(info.Rh, tts, p, [], bfs, phi);
⟨postprocess ph with w 6a⟩
```

The resulting function is:

9b ⟨*Structured low-rank approximation* 9b⟩≡

```
function [ph, info] = slra_ext(tts, p, r, w, Rini, phi, psi, opt, th2R, C)
⟨S ↦ (m, n, n_p) 16a⟩
⟨S ↦ S 16b⟩
⟨default phi 16d⟩
⟨default s0 2a⟩
⟨default psi 7b⟩
⟨default th2R 7a⟩
⟨default 𝒞 7c⟩
⟨default initial approximation 9c⟩
⟨nonlinear optimization over R 9a⟩
⟨define lra 9d⟩
⟨define R2th 10⟩
```

Note 9 (Initial approximation). Solving the outer minimization problem by either constrained or requilarized local minimization requires an initial approximation for the parameter R, *i.e.*, a suboptimal solution of the structured low-rank approximation problem. Such a solution can be computed from a heuristic that ignores the data matrix structure $\mathscr{S}$ and fills in zeros for the missing values. The resulting unstructured low-rank approximation problem can then be solved analytically in terms of the singular value decomposition.

9c ⟨*default initial approximation* 9c⟩≡ (9b)

```
if ~exist('Rini') | isempty(Rini), Rini = lra(phi * p(tts), r); end
```

9d ⟨*define* `lra` 9d⟩≡ (9b)

```
function [R, P, dh] = lra(d, r)
d(find(isnan(d))) = 0;
[u, s, v] = svd(d); R = u(:, (r + 1):end)'; P = u(:, 1:r);
if nargout > 2, dh = u(:, 1:r) * s(1:r, 1:r) * v(:, 1:r)'; end
```

The computed or user supplied initial approximation R_{ini} may not satisfy the constraint ($\theta \mapsto R$). In the case when Ψ is square and nonsingular, for any R_{ini}, there is corresponding θ_{ini} parameter:

$$\theta_{\text{ini}} := \text{vec}^{\top}(R_{\text{ini}})\Psi^{-1}.$$

In the general case of rectangular Ψ matrix, an approximation is needed in order to obtain a θ_{ini} parameter, such that $\mathscr{R}(\theta_{\text{ini}})$ is in some sense close to R_{ini}. Let $\widehat{D}_{\text{ini}}$ be the best unstructured approximation with image is equal to $\ker(R_{\text{ini}})$ of the data matrix.

9e ⟨$(R,D) \mapsto \widehat{D}$ 9e⟩≡ (10)

```
P = null(R); dh = P * (P \ d);
```

Since, $R_{\text{ini}}\widehat{D}_{\text{ini}} = 0$, the closeness between R_{ini} and $\mathscr{R}(\theta_{\text{ini}})$ can be measured by the Frobenius norm of the residual $\mathscr{R}(\theta_{\text{ini}})\widehat{D}_{\text{ini}}$. Imposing the normalization constraint $\|\theta_{\text{ini}}\| = 1$, the resulting approximation problem is

$$\text{minimize} \quad \text{over } \theta \quad \|\mathscr{R}(\theta)\widehat{D}_{\text{ini}}\|_{\text{F}} \quad \text{subject to} \quad \|\theta\| = 1, \tag{INI}$$

which is equivalent to unstructured approximation of the matrix $\Psi(D \otimes I_{m-r})$ by a rank $n_\theta - 1$ matrix.

10 ⟨*define* R2th 10⟩≡ (9b)

```
  function th = R2th(R, d, psi)
  if size(psi, 1) == size(psi, 2)
    th = R(:)' / psi;
  else
    ⟨(R,D) ↦ D̂ 9e⟩
    th = lra(psi * kron(dh, eye(size(R, 1))), size(psi, 1) - 1);
  end
```

Note 10 (Efficient computation and software implementation). Efficient evaluation of the cost function and its derivatives in the special case of mosaic-Hankel matrix structure is presented in a companion paper [17]. The method, presented in this paper (general linear structure) and the efficient methods of [17] are implemented in Matlab (using Optimization Toolbox) and C++, respectively. Description of the software and overview of its applications is given in [10].

5 Applications

As an illustration of how the developed methods can be used in practice and as a verification of their effectiveness, we present in this section three sample applications:

- unstructured noisy matrix completion,
- scalar autonomous system identification with missing data, and
- data-driven simulation.

Numerical examples comparing the methods developed in the paper with alternative methods, specifically developed for these applications, are shown. All simulations are done in Matlab and are reproducible in the sense of [2]. An extended version [11] of this paper is a literate program (in noweb format [15]), implementing the methods in the paper and generating the presented numerical results. The necessary m-files can be downloaded from

`http://eprints.soton.ac.uk/340718`.

5.1 Unstructured matrix with missing data

In the case of unstructured data matrix, the results obtained by the methods in the paper are compared with the results of alternative methods

- the alternating projections method of [7] and
- the singular value thresholding method of [3].

The alternating projections method for weighted low-rank approximation uses an image representation PL, where P is $m \times r$ and L is $r \times n$, of the $m \times n$ rank-r matrix $\mathscr{S}(\widehat{p})$. The algorithm iteratively minimizes the cost function over P with fixed L from the previous iteration step and over L with fixed P to its previously computed values. Both problems—minimization over P and minimization over L—are weighted linear least-squares problems, so that they can be solved globally and reliably. The cost function value is monotonically non-increasing over the iterations of the alternating projections method. The method is adapted in [7] to the case of missing data and is effective in solving large scale noisy matrix completion problems. A Matlab implementation is available from `http://eprints.ecs.soton.ac.uk/18296/`.

Singular value thresholding is a method for low-rank matrix completion, *i.e.*, a low-rank approximation with missing and exact values only. Although singular value thresholding is initially designed for the exact data case, it is shown to handle noisy data as well. Therefore, it solve low-rank approximation problems with missing data. The method is based on convex relaxation of the rank constraint and does not require an initial approximation. A Matlab implementation is available from `http://svt.caltech.edu/`

The results of a numerical example with a data matrix with the following pattern of missing values

$$\begin{bmatrix} \texttt{NaN} & \times & \times & \texttt{NaN} & \times & \times & \texttt{NaN} & \times & \times & \times \\ \times & \texttt{NaN} & \times & \times & \texttt{NaN} & \times & \times & \texttt{NaN} & \times & \times \\ \times & \times & \texttt{NaN} & \times & \times & \texttt{NaN} & \times & \times & \texttt{NaN} & \times \end{bmatrix}$$

and rank one specification are shown in Tables 1 and 2. Tables 1 shows the approximation errors $M_r^* = \|p_{\mathscr{M}} - \widehat{p}_{\mathscr{M}}\|$ achieved by the algorithms upon convergence and Tables 2 shows the number of iterations performed by the algorithms.

# of missing values	1	2	3	4	5	6	7	8	9
(SLRA_R')	0.8859	0.8642	0.8598	0.7900	0.7570	0.7568	0.6659	0.6031	0.6022
(SLRA_R'')	0.8859	0.8639	0.8593	0.7900	0.7570	0.7568	0.6658	0.6031	0.6022
alternating projections	0.8859	0.8639	0.8593	0.7900	0.7570	0.7568	0.6658	0.6031	0.6022
singular value thresholding	0.8892	0.8668	1.0539	1.0393	1.0467	1.0468	1.0310	0.9890	3.8635

Table 1: Approximation error $M_1^* \times 10^{-3}$ for the compared methods on problems with $1,\dots,9$ missing values.

# of missing values	1	2	3	4	5	6	7	8	9
(SLRA_R')	4	3	4	3	4	4	4	3	10
(SLRA_R'')	3	4	5	3	5	5	5	6	14
alternating projections	3	3	4	4	4	4	4	4	7

Table 2: Number of iterations for convergence of the compared methods on the problems in Table 1.

The results show that the proposed methods based on quadratic equality constraint (SLRA_R') and regularization (SLRA_R'') achieve the same approximation error as the alternating projections method and require similar number of iterations (starting from the same suboptimal initial approximation, see Note 9). The approximation error achieved by the singular value thresholding method increases with the increase of the number of missing values.

The simulation parameters are matrix size `m`, `n`, rank `r`, noise standard deviation `nl`, indices of fixed `If`, and missing values `Im`.

11a ⟨*unstructured matrix with missing data example* 11a⟩≡ 11b▷

```
clear all, randn('seed', 0), rand('seed', 0)
m = 3; n = 10; r = 1; nl = 0.03;
If = []; IM = [1 5 9 10 14 18 19 23 27];
```

The following lines are pointing Matlab to the alternating projections and singular value thresholding methods.

11b ⟨*unstructured matrix with missing data example* 11a⟩+≡ ◁11a 11c▷

```
addpath ~/mfiles/missing-data
addpath ~/mfiles/candes/matrix-completion/
addpath ~/mfiles/candes/matrix-completion/PROPACK_simple
```

The true data `p0` is generated as the product of $m \times r$ and $r \times n$ random factors.

11c ⟨*unstructured matrix with missing data example* 11a⟩+≡ ◁11b 11d▷

```
p0 = rand(m, r) * rand(r, n);
e  = randn(m, n); pf = p0 + nl * e / norm(e, 'fro') * norm(p0, 'fro');
```

The approximation methods are applied on data with $1,2,\dots$ missing data points.

11d ⟨*unstructured matrix with missing data example* 11a⟩+≡ ◁11c

```
np = m * n; tts = reshape(1:np, m, n); opt.method = 'reg'; % slra_ext calling parameters
tau = 5 * sqrt(m * n); % SVT calling parameters
```

```
  for i = 1:length(IM)
    Im = IM(1:i); Ig = setdiff(1:np, Im);
    w = ones(np, 1); w(Im) = 0; w(If) = inf;
    p = pf; p(If) = p0(If); p(Im) = NaN;

    [ph, info] = slra_ext(tts, p(:), r, diag(w(Ig)), [], [], []);
    opt.g = norm(p(Ig));
    [ph_, info_] = slra_ext(tts, p(:), r, diag(w(Ig)), [], [], [], opt);
    [Ph, Lh, info_wlra] = wlra(p, r, sqrt(reshape(w, m , n)));
    delta = 1.2 / (length(Ig) / m / n);
    [U, S, V, iter_svt] = SVT([m n], Ig, p(Ig), tau, delta, 10000);
    ph_svt  = U(:, 1:r) * S(1:r, 1:r) * V(:,1:r)’;
    e = vec(p - ph_svt); fmin_svt = norm(sqrt(w(Ig)) .* e(Ig)) ^ 2;

    res_fmin(:, i) = [
    info.fmin
    info_.fmin
    info_wlra.err
    fmin_svt
    ];

    res_iter(:, i) = [
    info.iterations
    info_.iterations
    info_wlra.iter
    iter_svt
    ];
  end
  res_fmin * 1e3, res_iter
```

5.2 System identification with missing data

Identification of a linear time-invariant system from a noisy trajectory of the system is a mosaic-Hankel structured low-rank approximation problem [17]. In the simplest case of a scalar autonomous system, the structure is Hankel

$$\mathscr{H}_{m,n}(p) := \begin{bmatrix} p_1 & p_2 & p_3 & \cdots & p_n \\ p_2 & p_3 & \iddots & & p_{n+1} \\ p_3 & \iddots & & & \vdots \\ \vdots & & & & \\ p_m & p_{m+1} & \cdots & & p_{m+n-1} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

and the identification problem is

$$\begin{array}{ll} \text{minimize} & \text{over } \widehat{y} \in \mathbb{R}^T \quad \|y_\text{d} - \widehat{y}\|_2^2 \\ \text{subject to} & \operatorname{rank}\big(\mathscr{H}_{\ell+1,T-\ell}(\widehat{y})\big) \leq \ell, \end{array}$$

where ℓ is the system’s lag (assumed known) and

$$y_\text{d} = \big(y_\text{d}(1), \dots, y_\text{d}(T)\big) \in \mathbb{R}^T$$

is the given trajectory. As an application of the algorithms developed in the paper, we consider autonomous system identification when samples of y_d are missing at arbitrary locations.

Figure 1 shows the results of a simulation example with the system defined by the difference equation

$$y(t) = 1.456y(t-1) - 0.81y(t-2).$$

The data is a trajectory of the system perturbed by additive noise with $T = 50$ samples. (For details on the simulation setup, see [11]) The missing values are distributed periodically with a period of 3 samples, so that standard system identification methods are not applicable. In this experiment the default initial approximation results in convergence to a poor local minimum and is replaced by $\begin{bmatrix}0.8 & -1.5 & 1\end{bmatrix}$.

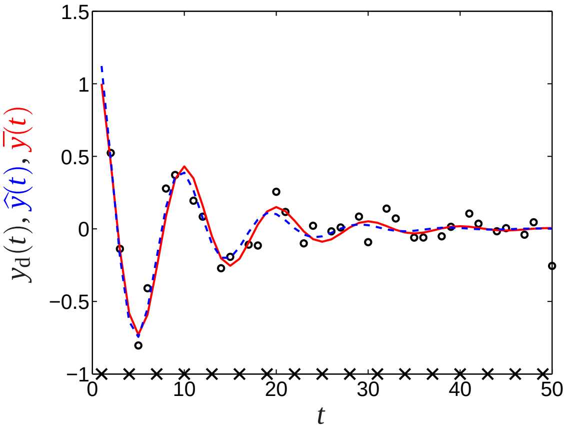

Figure 1: System identification with periodically missing data (crosses on the t-axis). Noisy samples (circles), optimal approximation (dashed blue), and true trajectory (solid red).

The simulation parameters are the noise standard deviation `nl` and the distribution of the missing values.

13a ⟨*system identification with missing data example* 13a⟩≡ 13b▷

```
clear all, randn('seed', 0), rand('seed', 0)
ell = 2; T = 50; Im = 1:(ell + 1):T; nl = 0.4;
```

The true trajectory is the impulse response of a second order system.

13b ⟨*system identification with missing data example* 13a⟩+≡ ◁13a 13c▷

```
sys0 = zpk(1, 0.9 * [exp(i * pi / 5) exp(-i * pi / 5)], 1, 1);
y0 = impulse(sys0, T); y0 = y0(2:end);
e = randn(T, 1); y = y0 + nl * e / norm(e) * norm(y0); y(Im) = NaN;
```

Apply the method and plot the results.

13c ⟨*system identification with missing data example* 13a⟩+≡ ◁13b

```
tts = blkhank(1:T, ell + 1); Rini = [0.8 -1.5 1];
[yh, info] = slra_ext(tts, y, ell, [], Rini);
plot(y, 'ok'), hold on,
plot(y0(1:T), 'r-'), plot(yh, 'b-')
ax = axis;
plot(Im, ax(3) * ones(size(Im)), 'Xk', 'markersize', 15)
axis(ax), print_fig('slra-ext-f1')
```

5.3 Data-driven simulation

Data-driven simulation problems [9] are special cases of missing data low-rank approximation. The to-be-simulated system is assumed to be linear time-invariant with a known upper bound ℓ of the lag. The system is implicitly specified by a trajectory $w_{\mathrm{d}}^{(1)} = (u_{\mathrm{d}}^{(2)}, y_{\mathrm{d}}^{(2)}) \in (\mathbb{R}^q)^{T_1}$. The to-be-simulated trajectory $w_{\mathrm{d}}^{(2)} = (u_{\mathrm{d}}^{(2)}, y_{\mathrm{d}}^{(2)}) \in (\mathbb{R}^q)^{T_2}$ is specified by the initial conditions $w_{\mathrm{ini}} = \big(w_{\mathrm{d}}^{(2)}(1), \dots, w_{\mathrm{d}}^{(2)}(\ell)\big)$ and the input $\big(u_{\mathrm{d}}^{(2)}(\ell+1), \dots, u_{\mathrm{d}}^{(2)}(T_2)\big)$. The data-driven simulation problem is a mosaic-Hankel structured low-rank approximation problem

$$\begin{aligned} &\text{minimize} \quad \text{over } \widehat{w} \quad \|w_{\mathrm{d}} - \widehat{w}\|_2^2 \\ &\text{subject to} \quad \operatorname{rank}\left(\begin{bmatrix} \mathscr{H}_{\ell+1}(u_{\mathrm{d}}^{(1)}) & \mathscr{H}_{\ell+1}(u_{\mathrm{d}}^{(2)}) \\ \mathscr{H}_{\ell+1}(y_{\mathrm{d}}^{(1)}) & \mathscr{H}_{\ell+1}(y_{\mathrm{d}}^{(2)}) \end{bmatrix}\right) \leq 2\ell + 1, \end{aligned}$$

with missing data being the to-be-simulated response $\big(y_{\mathrm{d}}^{(2)}(\ell+1), \dots, y_{\mathrm{d}}^{(2)}(T_2)\big)$.

Consider a simulation example with the second order single-input single-output system, defined by the difference equation

$$y(t) = 1.456y(t-1) - 0.81y(t-2) + u(t) - u(t-1).$$

The data $w_{\mathrm{d}}^{(1)}$ is a trajectory of the system generated from random input and additive noise. The to-be-simulated trajectory $w_{\mathrm{d}}^{(2)}$ is the impulse response $\bar{h}$ of the system, *i.e.*, the response under zero initial conditions and pulse input:

$$u_{\mathrm{d}}^{(2)} = (\underbrace{0,\dots,0}_{\ell},\underbrace{1,0,\dots,0}_{\text{pulse input}}), \qquad \text{and} \qquad y_{\mathrm{d}}^{(2)} = (\underbrace{0,\dots,0}_{\ell},\underbrace{\widehat{h}(0),\widehat{h}(1),\dots,\widehat{h}(T_2-\ell-1)}_{\text{impulse response}}).$$

Figure 2 shows the true and estimated by the proposed low-rank approximation method impulse responses.

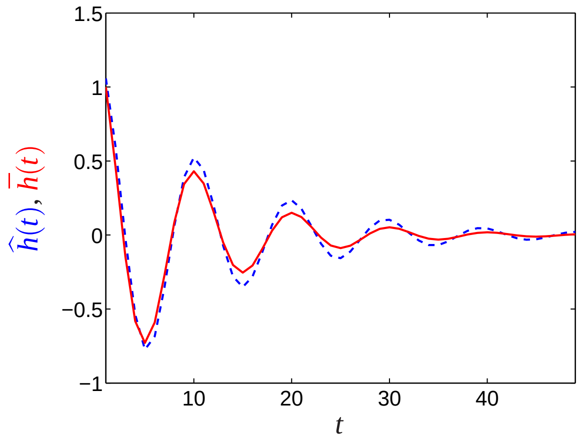

Figure 2: Data-driven simulation of impulse response: true (red solid line), optimal approximation (dashed blue).

14 ⟨*data-driven simulation example* 14⟩≡

```
clear all, close all, randn('seed', 0), rand('seed', 0)
T1 = 30; T2 = 52; nl = 0.4;

sys0 = zpk(1, 0.9 * [exp(i * pi / 5) exp(-i * pi / 5)], 1, 1); ell = 2;
u1 = rand(T1, 1); y1 = lsim(sys0, u1);
E = rand(T1, 2); w1 = [u1 y1] + nl * E / norm(E, 'fro') * norm([u1 y1], 'fro');

u2 = [zeros(ell, 1); 1; zeros(T2 - ell - 1, 1)];
y2 = [zeros(ell, 1); NaN * ones(T2 - ell, 1)]; w2 = [u2 y2];

tts = [blkhank(reshape(1:(T1 * 2), 2, T1), ell + 1) ...
       blkhank(reshape(T1 * 2 + (1:(T2 * 2)), 2 , T2), ell + 1)];
[wh, info] = slra_ext(tts, vec([w1' w2']), 2 * ell + 1);

y1h = wh(2:2:(2 * T1));

hh = wh(((T1 + ell) * 2 + 2):2:end);
h0 = impulse(sys0, T2 - ell - 1);

plot(hh(2:end), '-b'), hold on, plot(h0(2:end), '-r')
ax = axis; axis([1, T2 - ell - 1, ax(3:4)]), print_fig('slra-ext-f2')
```

6 Conclusions

A variable-projection-like approach for structured low-rank approximation with missing data was developed. The approach was furthermore generalized to weighted structured low-rank approximation with singular weight matrix. Two optimization strategies were proposed for the nonlinear least-squares optimization: optimization subject to quadratic equality constraints and regularized unconstrained optimization. The problem and solution methods developed have applications in matrix completion (unstructured problems), system identification with missing data, and data-driven simulation and control (mosaic-Hankel structured problems). The performance of the methods was illustrated on simulation examples from these applications and was compared with the performance of problem specific methods.

Acknowledgements

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".

References

[1] M. Brand. Fast low-rank modifications of the thin singular value decomposition. *Lin. Algebra and its Appl.*, 415(1):20–30, 2006.

[2] J. Buckheit and D. Donoho. *Wavelets and statistics*, chapter "Wavelab and reproducible research". Springer-Verlag, Berlin, New York, 1995.

[3] J.-F. Cai, E. Candés, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, January 2010.

[4] K. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics*, 21:489–498, 1979.

[5] B. Grung and R. Manne. Missing values in principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 42:125–139, 1998.

[6] I. Markovsky. Structured low-rank approximation and its applications. *Automatica*, 44(4):891–909, 2008.

[7] I. Markovsky. *Algorithms and literate programs for weighted low-rank approximation with missing data*, volume 3 of *Springer Proc. Mathematics*, pages 255–273. Springer, 2011.

[8] I. Markovsky. *Low Rank Approximation: Algorithms, Implementation, Applications*. Springer, 2012.

[9] I. Markovsky and P. Rapisarda. Data-driven simulation and control. *Int. J. Control*, 81(12):1946–1959, 2008.

[10] I. Markovsky and K. Usevich. Software for weighted structured low-rank approximation. Technical Report 339974, Univ. of Southampton, `http://eprints.soton.ac.uk/339974`, 2012.

[11] I. Markovsky and K. Usevich. Structured low-rank approximation with missing values. Technical Report 340718, Univ. of Southampton, `http://eprints.soton.ac.uk/340718`, 2012.

[12] I. Markovsky and S. Van Huffel. Overview of total least squares methods. *Signal Proc.*, 87:2283–2302, 2007.

[13] I. Markovsky, S. Van Huffel, and R. Pintelon. Block-Toeplitz/Hankel structured total least squares. *SIAM J. Matrix Anal. Appl.*, 26(4):1083–1099, 2005.

[14] J. Nocedal and S. Wright. *Numerical optimization*. Springer-Verlag, 1999.

[15] N. Ramsey. Literate programming simplified. *IEEE Software*, 11:97–105, 1994.

[16] N. Srebro. *Learning with matrix factorizations*. PhD thesis, MIT, 2004.

[17] K. Usevich and I. Markovsky. Efficient algorithms for mosaic-Hankel weighted low-rank approximation. Technical report, Univ. of Southampton, `http://eprints.soton.ac.uk/`, 2012.

A Special case of affine structure

A commonly encountered special case of the affine structure is

$$\left[\mathscr{S}(\widehat{p})\right]_{ij} = S_0 \widehat{p}_{\mathtt{S}_{ij}} \qquad \text{for some} \qquad \mathtt{S}_{ij} \in \{1,\dots,n_p\}^{m\times n}. \tag{S}$$

In (S), each element of the structured matrix $\mathscr{S}(p)$ is equal to the sum of the $\mathtt{S}_{ij}$th element of the parameter vector p and the constant $S_{0,ij}$. The structure is then specified by the matrices S_0 and S. Although (S) is a special case of the general linear structure ($\mathscr{S}$), it covers many linear modeling problems and will therefore be used in the implementation of the solution method.

In the implementation of the algorithm, the matrix S corresponds to a variable `tts`. Given the matrix S, specifying the structure, and a structure parameter vector $\widehat{p}$, the structured matrix $\mathscr{S}(\widehat{p})$ is constructed by `dh = s0 + ph(tts)`. The matrix dimensions `m`, `n`, and the number of parameters `np` are obtained from S as follows:

16a ⟨$\mathtt{S} \mapsto (m,n,n_p)$ 16a⟩≡ (4 6b 9b)

```
[mp, n] = size(tts); np = max(max(tts));
if exist('phi', 'var') && ~isempty(phi), m = size(phi, 1); else m = mp; end
```

The transition from the specification of (S) to the specification in the general linear case ($\mathscr{S}$) is done by

16b ⟨$\mathtt{S} \mapsto \mathbf{S}$ 16b⟩≡ (4 6b 9b)

```
vec_tts = tts(:); NP = 1:np;
bfs = vec_tts(:, ones(1, np)) == NP(ones(mp * n, 1), :);
```

Conversely, for a linear structure of the type (S), defined by **S** (and m, n), the matrix S is constructed by

16c ⟨$\mathbf{S} \mapsto \mathtt{S}$ 16c⟩≡

```
tts = reshape(bfs * (1:np)', mp, n);
```

For compatibility with the software package for mosaic-Hankel matrices [17], we consider structures of the form $\Phi\mathscr{S}$, where Φ is a full row rank matrix and $\mathscr{S}$ is an affine structure. The default value for Φ is the identity matrix I_m.

16d ⟨*default* `phi` 16d⟩≡ (4 9b)

```
if ~exist('phi', 'var') | isempty(phi), phi = eye(size(tts, 1)); end
```