Authors' response to the referees' reports on "Structured low-rank approximation with missing data"

Ivan Markovsky and Konstantin Usevich

March 7, 2013

We thank the associate editor and the referees for the fast second review round of the paper. As in the first answer letter, comments/questions from the report are quoted in **bold face**. Our replies follow in ordinary print. In blue, we quote passages from the revised manuscript.

Referee #1

- **All the tables should be made easier to read. Use latex tables.**

Done.

Referee #2

- **In the numerical experiments, authors mention that their approach cannot be used when m (the number of rows) is not small (m cannot exceed 10 for the Matlab implementation).**

The statement that 'm cannot exceed 10 for the Matlab implementation' is overly pessimistic. Although we did not attempt to make the implementation efficient, we verified that the method can be used for a 100×100 matrix and rank reduction by 1. The statement is revised as follows:

The implementation [MU12b] of the methods in the paper is applicable for relatively small size problems (say, $m < n < 100$, and $m - r < 3$).

- **I think it would be good to give a little more details to see how this comes together. For example, ...**

The overall computational cost of the algorithm depends on the number of iterations and the cost per iteration. The number of iterations is problem dependent and is difficult to predict: it depends on the initial approximation, optimization algorithm, and convergence tolerance. The cost per iteration depends on the cost of the objective function and derivatives evaluation and the cost of the search direction and the step size computation. The latter depends on the optimization method and the number of optimization variables (dm in our setting). Since we use standard local optimization methods (Levenberg-Marquardt, quasi-Newton, etc.), this step is well studied in the literature and we do not discus it. In addition, we are primarily interested in the case $m \ll n$, so that the cost for the objective function and derivatives evaluation is the dominant one.

There are two distinct algorithms for the evaluation of the objective function—the one of [UM12] and the one in the paper. The former has computational cost $O(m^2 n)$ and applies to mosaic-Hankel structure *without missing values*. The latter has cost $O(n^3)$ and applies to general affine structure with missing values. The computational cost is determined by the cost for the Cholesky factorization of the $nd \times nd$ matrix $GG^\top$, the algorithm of [UM12] or the $\bar{G}\bar{G}^\top$ matrix in the paper (see (M)).

We have revised the paragraph about the computational complexity, as follows:

The solution method proposed in the paper has computational complexity $O(n^3)$ per iteration, where n is the number of columns of the data matrix. It is applicable to general affine structured matrices with missing data, however, it is

unsuitable for large scale applications. In special cases, such as unstructured matrices with missing data or mosaic-Hankel structured matrices without missing data, there are efficient $O(n)$ methods. We conjecture that an efficient $O(n)$ implementation of the algorithms in the paper is possible in the case of mosaic-Hankel matrices with missing data.

- **Is the unstructured case the the worst case?**

No, in the unstructured case, the $\bar{G}\bar{G}^\top$ matrix is block-diagonal with $d \times d$ blocks, so that its Cholesky factorization is cheaper to compute than in the corresponding factorization in the general case. (In general, $\bar{G}\bar{G}^\top$ is block-banded with $d \times d$ blocks and bandwidth depending on the structure.) The block-diagonal structure in the case of usntructured matrix, however, is not exploited by our implementation, so that the computational cost is still $O(n^3)$.

- **Concerning the non-convexity ..., authors might refer to: ...**

Thank you for the reference. We have cited it in support for the statement.

- **"In special cases, such as unstructured matrix with missing data and Hankel structured matrix ..." Shouldn't 'matrix' be plural? I would rather use or between the two spacial cases to avoid confusion.**

Corrected as suggested.

- **p.3 indeces.**

Corrected.

- **p.4 definition of w_p in section 1.2.3 has twice the same entry.**

Corrected.

- **p.5 Theorem 2. In case of unstructured low-rank approximations, don't we have $n_m + n_g = mn$ hence condition 2. is always satisfied? (or am I missing something?) If yes, this could be pointed out.**

Yes, in case of unstructured low-rank approximations, $n_p = n_m + n_g = mn$. Condition 2 of Theorem 2 consists of two inequalities:

$$n_m < (m-r)n \qquad \text{and} \qquad (m-r)n \leq n_p.$$

In the case of unstructured low-rank approximations, the second inequality is indeed always satisfied, but the first inequality still imposes a nontrivial constraint. We have added a note pointing this out.

Note 1. In the case of unstructured matrix, condition 2 of Theorem 2 reduces to $n_m < (m-r)n$.

- **p.10 '... `slra-c` is applied by setting the weights of the missing values to 10^{-6}' Why can't authors use zero weights? What value do authors assign to the missing entries (zero?)? Is this a trick to avoid ill-posedness (that is, the infimum might not be attained when some weights are equal to zero, see, e.g., the reference above)? This should be clarified.**

The implementation of `slra-c` does not allow zero weights. The choice of small weights is a compromise between modification of the cost function (*i.e.*, changing the original problem formulation) and ill-conditioning of the $GG^\top$ matrix. (The correct solution of the problem in the case of zero weights is Theorem 2, where $\bar{G}\bar{G}^\top$ is used instead of $GG^\top$.) The missing data are replaced with zeros. This choice is arbitrary, however, it does not affect the approximation "much" due to the corresponding "small" weights.

We have revised as follows:

For larger problems, the efficient C implementation [MU12a] (denoted by `slra-c` below), of the variable projection approach without missing values, can be used by setting the missing values to zeros and the corresponding weights to a small number (10^{-6} in the simulation examples).

- **p.11 As mentioned by the the first reviewer, only 8 missing elements for a 10-by-100 matrix is rather small (especially if one has in mind recommender systems). Does the proposed algorithm perform badly when the number of missing elements is larger? This should be discussed.**

We did not show results with the 10×100 exact matrix completion problem for more than 8 missing values because then *all* methods fail. ("Fail" means that given exact data they do not recover the missing values exactly.) For example, with 102 missing values and no noise

- `m = 10; n = 100; r = 8; nl = 0; eps = 0.9;`

the results are

```
'  '    'slra-m'        'slra-r'        'slra-c'    'wlra'        'optspace'    'lmafit'    'rtrmc'
'eg'    [1.2994e-16]    [1.2994e-16]    [0.0109]    [  0.0191]    [  0.0436]    [0.0096]    [1.4556e-09]
'em'    [    0.2374]    [    0.2374]    [2.1480]    [740.0182]    [  0.7027]    [0.5785]    [    0.2374]
't '    [    0.7933]    [    2.9547]    [0.2435]    [  0.3503]    [  0.3048]    [0.0628]    [    0.4984]
```

Note that the algorithms in the paper and `rtrmc` fit exactly the given data but not the missing data. This proves that the optimization problem has a nonunique global solution. Also, note that all methods, except for `slra-m`, `slra-r`, and `rtrmc` fail to find a global solution.

The results are qualitatively the same in case of 212 missing values:

- `m = 10; n = 100; r = 8; nl = 0; eps = 0.8;`

```
'  '    'slra-m'        'slra-r'        'slra-c'    'wlra'      'optspace'    'lmafit'    'rtrmc'
'eg'    [5.4294e-17]    [5.4294e-17]    [0.0052]    [0.0086]    [  0.0509]    [0.0079]    [3.9468e-10]
'em'    [    0.5374]    [    0.5374]    [1.1876]    [9.5087]    [  0.6133]    [0.7648]    [    0.5374]
't '    [    0.7271]    [    3.1331]    [0.2497]    [0.3473]    [  0.3229]    [0.1185]    [    0.4272]
```

Again with 212 missing values, by adding noise, a nonzero fitting error appears

- `m = 10; n = 100; r = 8; nl = 0.1; eps = 0.8;`

```
'  '    'slra-m'    'slra-r'     'slra-c'    'wlra'      'optspace'    'lmafit'    'rtrmc'
'eg'    [0.0038]    [ 0.0043]    [0.0093]    [0.0102]    [  0.0531]    [0.0099]    [0.0038]
'em'    [0.6934]    [ 0.6973]    [2.9074]    [7.6141]    [  0.6399]    [0.7587]    [0.6934]
't '    [9.0669]    [24.5793]    [0.2442]    [0.3475]    [  0.3150]    [0.0834]    [0.6758]
```

however, `slra-m` and `rtrmc` still find the same approximate solution, which in the example is the best one (in terms of the fitting error w.r.t. the given data).

We have added in the paper the results with 212 missing values and no noise:

As illustrated by the following example, increasing the number of missing values eventually makes the solution of the exact matrix completion problem nonunique:

⟨*AMC, example 3*⟩≡

```
ex = 'amc-ex3'; m = 10; n = 100; r = 8; nl = 0; eps = 0.8; test_amc
```

	slra-m	slra-r	slra-c	wlra	optspace	lmafit	rtrmc
e_g	5×10^{-17}	5×10^{-17}	0.0052	0.0086	0.0509	0.0079	3×10^{-10}
e_m	0.5374	0.5374	1.1876	9.5087	0.6133	0.7648	0.5374
time, sec	0.7271	3.1331	0.2497	0.3473	0.3229	0.1185	0.4272

Note that, again the methods in the paper and `rtrmc` find the same solution, which in the example fits the given data exactly.

- **p.13 'Efficient computation' sounds weird. Do you mean 'Efficient implementations'?**

Yes, 'efficient' refers to the a method, algorithm, or software implementation. We have replaced the statement 'efficient computation' with 'efficient implementation'.

References

[MU12a] 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.

[MU12b] I. Markovsky and K. Usevich. Structured low-rank approximation with missing values. Technical report, Vrije Univ. Brussel, homepages.vub.ac.be/~imarkovs/publications.html, 2012.

[UM12] K. Usevich and I. Markovsky. Variable projection for affinely structured low-rank approximation in weighted 2-norm, 2012. Available from http://arxiv.org/abs/1211.3938.