Fast Multilevel Algorithms for Compressive Principle Component Pursuit

Vahan Hovhannisyan* Yannis Panagakis† Panos Parpas†
Stefanos Zafeiriou†

April 19, 2018

Abstract

Recovering a low-rank matrix from highly corrupted measurements arises in compressed sensing of structured high-dimensional signals (e.g., videos and hyperspectral images among others). Robust principal component analysis (RPCA), solved via principal component pursuit (PCP), recovers a low-rank matrix from sparse corruptions that are of unknown value and support by decomposing the observation matrix into two terms: a low-rank matrix and a sparse one, accounting for sparse noise and outliers. In the more general setting, where only a fraction of the data matrix has been observed, low-rank matrix recovery is achieved by solving the compressive principle component pursuit (CPCP). Both PCP and CPCP are well-studied convex programs, and numerous iterative algorithms have been proposed for their optimisation. Nevertheless, these algorithms involve singular value decomposition (SVD) at each iteration, which renders their applicability challenging in the case of massive data. In this paper, we propose a multilevel approach for the solution of PCP and CPCP problems. The core principle behind our algorithm is to apply SVD in models of lower-dimensionality than the original one and then lift its solution to the original problem dimension. We show that the proposed algorithms are easy to implement, converge at the same rate but with much lower iteration cost. Numerical experiments on numerous synthetic and real problems indicate that the proposed multilevel algorithms are several times faster than their original counterparts, namely PCP and CPCP.

1 Introduction

Low-rank matrix recovery is a cornerstone in data analysis and dimensionality reduction, with the principal component analysis (PCA) [12] being the most widely employed method for this task. However, PCA is fragile to the presence of gross, non-Gaussian, noise and outliers, and the estimated low-rank subspace may be arbitrarily away from the true one; even when a small fraction of the data is corrupted

*Department of Computing, Imperial College London, 180 Queen's Gate, SW7 2AZ London, UK (vh13@imperial.ac.uk, i.panagakis@imperial.ac.uk, p.parpas@imperial.ac.uk, s.zafeiriou@imperial.ac.uk).

[14]. To alleviate this drawback, robust PCA (RPCA) models have been proposed [6]. RPCA aims to recover a low-rank matrix from sparse corruptions that are of unknown value and support by decomposing the observation matrix ($\mathbf{D}$) into two parts, namely $\mathbf{D} = \mathbf{L} + \mathbf{S}$. The first part is a low-rank matrix ($\mathbf{L}$) and the second part is a sparse matrix ($\mathbf{S}$) that accounts for sparse noise and outliers. In case of partially observed data, the RPCA model is extended to consider the following decomposition [34]: $\mathbf{D} \doteq \mathcal{P}_{\mathcal{Q}}[\mathbf{M}] = \mathcal{P}_{\mathcal{Q}}[\mathbf{L} + \mathbf{S}]$, where $\mathcal{Q} \subseteq \mathbb{R}^{m \times n}$ is a linear subspace and $\mathcal{P}_{\mathcal{Q}}$ denotes the projection operator onto that subspace. The aforementioned low-rank matrix recovery models have profound impact in visual data analysis and computer vision applications such as image denoising [6], background subtraction, image alignment [28], texture recovery [36], deformable models [29], face frontalization [30], and structure from motion [2], to mention but a few examples.

A natural approach to estimate the low-rank and sparse components in the above mentioned models is to minimise the rank of $\mathbf{L}$ and the number on non-zero entries of $\mathbf{S}$, measured by the ℓ_0 quasi norm [6]. Unfortunately, both rank and ℓ_0-norm minimisation are NP-hard [32, 23]. The nuclear- and the ℓ_1- norms are typically adopted as convex surrogates to rank and ℓ_0- norm, respectively yielding the convex *principle component pursuit* (PCP) [6] and *compressive principle component pursuit* (CPCP) programs [34].

Common solvers for the convex PCP and CPCP models include: Iterative Thresholding (IT) [7], Accelerated Proximal Gradient (APG) [26], Augmented Lagrange Multipliers (ALM) [17] and Linearized Augmented Lagrangian method [35]. However, all these solvers exhibit significant computational drawbacks. In particular, at each iteration, they require computing several (not necessarily all) singular values and vectors of a large matrix, which is computationally expensive.

There have been several attempts to reduce the computational cost of large nuclear-norm regularised optimisation problems. Concretely, [18] proposed to reduce the dimensions of the problem by factorising the low-rank matrix as the product of two smaller matrices, resulting in a non-convex problem which is solved by employing the augmented Lagrangian alternating direction method. Another very popular approach for reducing the dimensions of large-scale problems is to create smaller sub-problems by applying randomised techniques [1, 8, 19, 21, 25]. More recently, the Frank-Wolfe (FW) algorithm has regained popularity for solving large-scale problems due to its extremely low iteration complexity. Specifically, the Frank-Wolfe Thresholding (FW-T) method proposed in [20] is arguably the most efficient method for solving large (C)PCP problems. Nevertheless, FW type of methods require significantly more iterations to converge, and hence they can be impractically time consuming.

In this paper, motivated by the recent advances in multilevel optimisation algorithms [11, 13, 15, 22, 5, 27], we propose a simple, yet generic and very effective multilevel approach for significantly reducing computational costs for many problems that require solving nuclear norm based oracles, including RPCA models such as the PCP and CPCP. The core of our proposed methodology is to construct and solve lower dimensional (coarse) models for each optimisation oracle and then lift its solution to the original problem dimension. We show that using appropriately chosen restriction and prolongation operators result in algorithms that converge to an (approximate) solution of the original problem. We apply the proposed multilevel approach on

two state-of-the-art algorithms, namely the Inexact Augmented Lagrange Multiplier Method (IALM) for the PCP problem [17], and the Frank-Wolfe Thresholding method for the more general CPCP model [20]. In particular, our main contributions are:

- In section 3.2 we show that the proposed multilevel IALM algorithm converges to an approximate solution, and in sections 4.2 and 4.3 we show that in practice it is several times faster than the standard IALM.
- The first provably convergent variant of IALM with approximate updates (Theorem 5).
- In section 3.4 we show that for the FW-T method we prove that its multilevel variant converges in function value with the same worst-case iteration complexity (Theorem 6). However, in sections 4.2 and 4.3 we show that in practice each iteration of the multilevel algorithm is up to two times cheaper.
- Numerical tests in both synthetic and real data indicate that the proposed multilevel variants solve large-scale problems two times faster than their standard counterparts (Section 4).

Notation. Throughout the paper, scalars are denoted by lower-case letters, vectors (matrices) are denoted by lowercase (upper-case) boldface letters, e.g. $\mathbf{x}$ ($\mathbf{X}$). $\mathbf{I}$ denotes the identity matrix with appropriate dimension. The ℓ_1 and ℓ_2 norms of a vector $\mathbf{x}$ are defined as $\|\mathbf{x}\|_1 = \sum_i |x_i|$, where $|\cdot|$ denotes the absolute value operator, and $\|\mathbf{x}\|_2 = \sqrt{\sum_i x_i^2}$, respectively. The matrix ℓ_1 norm is defined as $\|\mathbf{X}\|_1 = \sum_i \sum_j |x_{ij}|$. The Frobenius norm is defined as $\|\mathbf{X}\|_F = \sqrt{\sum_i \sum_j x_{ij}^2}$, and the nuclear norm of $\mathbf{X}$ (i.e., the sum of singular values of a matrix) is denoted by $\|\mathbf{X}\|_*$. The l-th largest singular value of matrix $\mathbf{X}$ is denoted as $\sigma_l(\mathbf{X})$. In algorithm pseudocodes we use $\mathbf{X}^{(k)}$ (u_k) to denote the value of matrix $\mathbf{X}$ (scalar u) at iteration k.

2 Compressive Principle Component Pursuit and Robust PCA

In this section, we give a formal presentation of the CPCP [34] model. Let $\mathcal{Q} \subseteq \mathbb{R}^{m\times n}$ be a linear subspace spanned by the set of sensing matrices, and $\mathcal{P}_\mathcal{Q}$ denote the projection operator onto that subspace. In many applications $\mathcal{Q}$ is the subset of observed values of data $\mathbf{D}$. Then the problem is to find a low rank matrix $\mathbf{L}^\star$ and a sparse matrix $\mathbf{S}^\star$ such that the $\mathcal{P}_\mathcal{Q}[\mathbf{L}^\star + \mathbf{S}^\star] = \mathcal{P}_\mathcal{Q}[\mathbf{D}]$. The problem can be written as a convex unconstrained minimisation problem as follows:

$$\min_{\mathbf{L},\mathbf{S}} \frac{1}{2}\|\mathcal{P}_\mathcal{Q}[\mathbf{L}+\mathbf{S}-\mathbf{D}]\|_F^2 + \lambda_L\|\mathbf{L}\|_* + \lambda_S\|\mathbf{S}\|_1,$$

where λ_L and λ_S are positive penalising coefficients.

An important special case of (2) is the well known PCP problem robust principle component analysis (RPCA), where $\mathcal{Q}$ is the entire space $\mathbb{R}^{m\times n}$ and $\mathcal{P}_\mathcal{Q}$ is the identity,

i.e. all values of $\mathbf{D}$ have been observed. The problem in this case can be formulated to represent the input data matrix $\mathbf{D} \in \mathbb{R}^{m \times n}$ as a sum of a low rank matrix $\mathbf{L}^\star$ and a sparse matrix $\mathbf{S}^\star$. This can be exactly solved via the following convex constrained optimisation problem:

$$\min_{\mathbf{L},\mathbf{S}} \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1, \quad \textit{subject to} \quad \mathbf{D} = \mathbf{L} + \mathbf{S},$$

where $\lambda > 0$ is a weighting parameter.

Before proceeding in the presentation of the proposed multilevel algorithms for the PCP and CPCP problems, we will provide an overview of the most widely adopted solvers for these problems

2.1 Inexact ALM for Robust PCA

We start with the simpler PCP problem for RPCA. A classical approach for solving (2) is by minimising its augmented Lagrangian defined as

$$\mathcal{L}(\mathbf{L}, \mathbf{S}, \mathbf{Y}, \mu) = \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1 + \langle \mathbf{Y}, \mathbf{D} - \mathbf{L} - \mathbf{S} \rangle + \frac{\mu}{2} \|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F^2,$$

where $\mathbf{Y} \in \mathbb{R}^{m \times n}$ is the Lagrangian variable and $\mu > 0$ is a penalty parameter. The convex optimization model in (2.1) can be solved via alternating directions method. The latter method first solves the problem for each primal variable $\mathbf{L}$ and $\mathbf{S}$ separately for a fixed $\mathbf{Y}$. The dual variable $\mathbf{Y}$ is updated according to a linear rule and μ_k is chosen as an increasing sequence at each iteration [17]. The method is computationally attractive because each resulting subproblem has a closed form solution. The resulting procedure was dubbed Inexact ALM (IALM) in [17] and is formally given here in Algorithm 1.

Algorithm 1 Inexact ALM (IALM)

Input: $\mathbf{D}, \mathbf{S}^{(0)}, \mathbf{Y}^{(0)} \in \mathbb{R}^{m \times n}$; $\mu_0 > 0$

for $k \leftarrow 1$ to ... **do**
  // Solve $\mathbf{L}^{(k+1)} = \arg\min_{\mathbf{L}} \mathcal{L}(\mathbf{L}, \mathbf{S}^{(k)}, \mathbf{Y}^{(k)}, \mu_k)$
  $\mathbf{M}^{(k)} \leftarrow \mathbf{D} - \mathbf{S}^{(k)} + \mu_k^{-1} \mathbf{Y}^{(k)}$
  $(\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}) \leftarrow \text{SVD}(\mathbf{M}^{(k)})$
  $\mathbf{L}^{(k+1)} \leftarrow \mathbf{U} \mathcal{S}_{\mu_k^{-1}}[\mathbf{\Sigma}] \mathbf{V}^\top$
  // Solve $\mathbf{S}^{(k+1)} = \arg\min_{\mathbf{S}} \mathcal{L}(\mathbf{L}^{(k+1)}, \mathbf{S}, \mathbf{Y}^{(k)}, \mu_k)$
  $\mathbf{S}^{(k+1)} \leftarrow \mathcal{S}_{\lambda \mu_k^{-1}}[\mathbf{D} - \mathbf{L}^{(k+1)} + \mu_k^{-1} \mathbf{Y}^{(k)}]$
  // Update the Lagrangian variable
  $\mathbf{Y}^{(k+1)} \leftarrow \mathbf{Y}^{(k)} + \mu_k (\mathbf{D} - \mathbf{L}^{(k+1)} - \mathbf{S}^{(k+1)})$
  Update $\mu_k \leftarrow \mu_{k+1}$
end for
return $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+1)})$

Minimising (2.1) over $\mathbf{L}$ requires computing singular values of a large $m \times n$ matrix. It is well known that computing the k largest singular values has a computational complexity of $\mathcal{O}(kmn)$. Thus for practical efficiency it is important to compute only a few singular values [17]. However, the SVD in step 4 remains the computational bottleneck of Algorithm 1. Theorem 1 [17] gives an asymptotic convergence result for Algorithm 1.

Theorem 1. *For Algorithm 1, if μ_k is non-decreasing and $\sum_{k=1}^{+\infty} \mu_k^{-1} = +\infty$, then $(\mathbf{L}^{(k)}, \mathbf{S}^{(k)})$ asymptotically converges to an optimal solution of the RPCA problem.*

2.2 Frank-Wolfe Method

In this section, we review the more general CPCP problem and the well studied Frank-Wolfe (FR) method [9], also known as the conditional gradient method [16] and its associated convergence result. Since the method minimises any convex smooth function f over a bounded convex set $\mathcal{D} \subseteq \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space endowed with an inner product $\langle \cdot, \cdot \rangle$, we study the more general optimisation problem:

$$\min f(\mathbf{x}) \quad \text{s.t} \quad \mathbf{x} \in \mathcal{D},$$

where f has a L-Lipschitz continues gradient:

$$\forall \mathbf{x}, \mathbf{y} \in \mathcal{D}, \quad \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|.$$

Throughout, we let $D = \max_{\mathbf{x},\mathbf{y} \in \mathcal{D}} \|\mathbf{x} - \mathbf{y}\|$ denote the diameter of the feasible set $\mathcal{D}$. The Frank-Wolfe method, as well as all its variants and extensions studied in this work will assume that the feasibility set $\mathcal{D}$ is bounded, i.e. $D < \infty$. The classical Frank-Wolfe method has many variants with different update rules, but in the most general form it can be written as in Algorithm 2 [20].

Algorithm 2 Frank-Wolfe (FW)

Input: $\mathbf{x}^{(0)} \in \mathcal{D}$
1: **for** $k \leftarrow 1$ to ... **do**
2: $\quad \mathbf{v}^{(k)} \in \arg\min_{\mathbf{v} \in \mathcal{D}} \langle \mathbf{v}, \nabla f(\mathbf{x}^{(k)}) \rangle$;
3: $\quad \gamma = \frac{2}{k+2}$
4: $\quad$ Update $\mathbf{x}^{(k+1)}$ to a point in $\mathcal{D}$ so that $f(\mathbf{x}^{(k+1)}) \leq f(\mathbf{x}^{(k)} + \gamma(\mathbf{v}^{(k)} - \mathbf{x}^{(k)}))$;
5: **end for**
6: **return** $\mathbf{x}^{(k+1)}$

The two most common updating rules for $\mathbf{x}^{(k+1)}$ are the simple

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \gamma(\mathbf{v}^{(k)} - \mathbf{x}^{(k)}),$$

and the following slightly more sophisticated one

$$\mathbf{x}^{(k+1)} \in \arg\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{x} \in \text{conv}\{\mathbf{x}^{(k)}, \mathbf{v}^{(v)}\}.$$

In this paper we will use the more advanced update rule (2.2) for its better practical performance. Using standard techniques it can be shown that the FW method converges at a rate of $\mathcal{O}(1/k)$ in function values.

Theorem 2. *Let $\mathbf{x}^\star$ be an optimal solution of (2.2). For $\{\mathbf{x}^{(k)}\}$ generated by Algorithm 2, we have for $k = 0, 1, 2, ..$*

$$f(\mathbf{x}^{(k)}) - f(\mathbf{x}^\star) \leq \frac{2LD^2}{k+2}.$$

Proof. The proof can be found, for example, in [20]. □

2.3 Frank-Wolfe Thresholding Method for CPCP

In this section we discuss the application of the Frank-Wolfe method to problem (2). Since the FW algorithm can be applied only for smooth and constrained convex optimisation problems with a bounded feasible set, we reformulate (2) into a such problem. In [20] the reformulation was done by first performing an epigraph reformulation on (2) obtaining,

$$\begin{array}{ll} \min & f(\mathbf{L}, \mathbf{S}, t_L, t_S) := \frac{1}{2}\|\mathcal{P}_\mathcal{Q}[\mathbf{L} + \mathbf{S} - \mathbf{D}]\|_F^2 + \lambda_L t_L + \lambda_S t_S \\ s.t. & \|\mathbf{L}\|_* \leq t_L, \|\mathbf{S}\|_1 \leq t_S. \end{array}$$

Now the objective function has a 2-Lipschitz gradient ∇f with partial derivatives given as follows:

$$\nabla_L f(\mathbf{L}, \mathbf{S}, t_L, t_S) = \nabla_S f(\mathbf{L}, \mathbf{S}, t_L, t_S) = \mathcal{P}_\mathcal{Q}[\mathbf{L} + \mathbf{S} - \mathbf{D}],$$

$$\nabla_{t_L} f(\mathbf{L}, \mathbf{S}, t_L, t_S) = \lambda_L, \nabla_{t_S} f(\mathbf{L}, \mathbf{S}, t_L, t_S) = \lambda_S.$$

Then to make the feasible region bounded we introduce upper bounds U_L and U_S for t_L and t_S, respectively. It can be shown [20] that we can choose

$$U_L = \tfrac{1}{2\lambda_L}\|\mathcal{P}_\mathcal{Q}[\mathbf{D}]\|_F^2, \quad U_S = \tfrac{1}{2\lambda_S}\|\mathcal{P}_\mathcal{Q}[\mathbf{D}]\|_F^2,$$

and the resulting feasible set has a bounded diameter: $D \leq \sqrt{5} \cdot \sqrt{U_L^2 + U_S^2}$.

Now we can apply the FW algorithm on

$$\begin{array}{ll} \min & f(\mathbf{L}, \mathbf{S}, t_L, t_S) := \frac{1}{2}\|\mathcal{P}_\mathcal{Q}[\mathbf{L} + \mathbf{S} - \mathbf{D}]\|_F^2 + \lambda_L t_L + \lambda_S t_S \\ s.t. & \|\mathbf{L}\|_* \leq t_L \leq U_L, \|\mathbf{S}\|_1 \leq t_S \leq U_S. \end{array}$$

Setting $\mathbf{x} = (\mathbf{L}, \mathbf{S}, \lambda_L, \lambda_S)$ and using the gradient expressions (2.3)-(2.3) we can derive the linear optimisation oracle in step 2 of Algorithm 2 as two independent linear optimisation problems:

$$(\mathbf{V}_L^{(k)}, V_{t_L}^{(k)}) \in \underset{\|\mathbf{V}_L\|_* \leq V_{t_L} \leq U_L}{\arg\min} \langle \mathcal{P}_\mathcal{Q}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{V}_L \rangle + \lambda_L V_{t_L},$$

$$(\mathbf{V}_S^{(k)}, V_{t_S}^{(k)}) \in \underset{\|\mathbf{V}_S\|_* \le V_{t_S} \le U_S}{\arg\min} \langle \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{V}_S \rangle + \lambda_S V_{t_S}.$$

Both (2.3) and (2.3) are separable and can be solved in closed form using the leading singular values and the largest in magnitude elements of $\mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}]$ [20]. This is given in steps $3 - 16$ in Algorithm 3. Finally, we use the update rule (2.2) for problem (2.3) resulting to step 17 of Algorithm 3.

A major drawback of the FW method is that for the ℓ_1-norm, at each iteration it projects a linear function on a ℓ_1 ball, thus updating only one entry of a very large matrix. To resolve this problem [20] suggested to add a thresholding operation to the FW algorithm, calling it Frank-Wolfe Thresholding (FW-T). This is performed in steps $18 - 19$ of Algorithm 3. Finally, as suggested in [20], in steps $20 - 21$ we update the bounds U_L and U_S tightening the feasibility set.

We present the complete Frank-Wolfe Thresholding method in Algorithm 3. Both primal and dual convergence of the FW-T algorithm was established in [20]. Since the thresholding (and more generally proximal) operator decreases the function value more than the Frank-Wolfe update, the Frank-Wolfe Thresholding algorithm can be seen as a special case of Algorithm 2 [20]. Therefore FW-T converges to the solution $(\mathbf{L}^\star, \mathbf{S}^\star)$ in function value with the same $\mathcal{O}(1/k)$ rate. Although the FW-T method requires only computing the largest singular value of a $m \times n$ matrix at each iteration, SVD computations still remain the computational bottleneck.

3 Multilevel Algorithms

We propose multilevel variants of the two classical methods discussed above. IALM and FW-T have the same computational bottleneck of computing one or more singular values at each iteration. Our methods will build lower dimensional, so-called *coarse*, models for each problem and use their singular values for iteration updates. We will show that both algorithms converge to an (approximate) solution of the original problem with the same worst case iteration complexity of their standard counterparts. However, the per iteration cost of multilevel methods is much smaller, since SVDs are performed on much smaller coarse models [1].

3.1 The Coarse Model

In this section we will introduce a generic lower dimensional model for (2) and (2). It uses the so-called restriction operator $\mathbf{R} \in \mathbb{R}^{n \times n_H}$, for some $n_H \le n$, where n_H is the dimension of the coarse model. Throughout the paper we will make the following assumption about the restriction operator $\mathbf{R}$.

Assumption 1. *The restriction operator $\mathbf{R}$ has linearly independent columns. Therefore, $\mathbf{R}$ has a left inverse $\mathbf{R}^\dagger \in \mathbb{R}^{n_H \times n}$ so that $\mathbf{R}^\dagger \mathbf{R} = \mathbf{I}_{n_H}$ (in general, $\mathbf{R}\mathbf{R}^\dagger \neq \mathbf{I}_n$ and $\mathbf{R}$ may not have a right inverse).*

[1] In our experiments we observed 2-10 times cheaper per iteration complexities

Algorithm 3 Frank-Wolfe Thresholding (FW-T) for (2)

Input: $\mathbf{D} \in \mathbb{R}^{m\times n}; \lambda_L, \lambda_S > 0$

Set $\mathbf{L}^{(0)} = \mathbf{S}^{(0)} = \mathbf{0}; t_L^{(0)} = t_S^{(0)} = 0; U_L^{(0)} = f(\mathbf{L}^{(0)}, \mathbf{S}^{(0)}, t_L^{(0)}, t_S^{(0)})/\lambda_L; U_S^{(0)} = f(\mathbf{L}^{(0)}, \mathbf{S}^{(0)}, t_L^{(0)}, t_S^{(0)})/\lambda_S$.

for $k \leftarrow 1$ to ... **do**

$\mathbf{M}_L^{(k)} \in \underset{\|\mathbf{M}_L\|_* \le 1}{\arg\min} \langle \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{M}_L \rangle;$

$\mathbf{M}_S^{(k)} \in \underset{\|\mathbf{M}_S\|_1 \le 1}{\arg\min} \langle \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{M}_S \rangle;$

if $\lambda_L \ge -\langle \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{M}_L^{(k)} \rangle$ **then**

$\mathbf{V}_L^{(k)} = \mathbf{0}; V_{t_L}^{(k)} = 0$

else

$\mathbf{V}_L^{(k)} = U_L \mathbf{M}_L^{(k)}; V_{t_L}^{(k)} = U_L$

end if

if $\lambda_S \ge -\langle \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}], \mathbf{M}_S^{(k)} \rangle$ **then**

$\mathbf{V}_S^{(k)} = \mathbf{0}; V_{t_S}^{(k)} = 0$

else

$\mathbf{V}_S^{(k)} = U_S \mathbf{M}_S^{(k)}; V_{t_S}^{(k)} = U_S$

end if

Compute $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+\frac{1}{2})}, t_L^{(k+1)}, t_S^{(k+\frac{1}{2})})$ as a minimiser of

$$\begin{aligned} \min_{\mathbf{L},\mathbf{S},t_L,t_S} \quad & \tfrac{1}{2}\|\mathcal{P}_{\mathcal{Q}}[\mathbf{L}+\mathbf{S}-\mathbf{D}]\|_F^2 + \lambda_L t_L + \lambda_S t_S \\ s.t. \quad & \begin{pmatrix}\mathbf{L}\\ t_L\end{pmatrix} \in \operatorname{conv}\left\{ \begin{pmatrix}\mathbf{L}^{(k)}\\ t_L^{(k)}\end{pmatrix}, \begin{pmatrix}\mathbf{V}_L^{(k)}\\ V_{t_L}^{(k)}\end{pmatrix} \right\}, \\ & \begin{pmatrix}\mathbf{S}\\ t_S\end{pmatrix} \in \operatorname{conv}\left\{ \begin{pmatrix}\mathbf{S}^{(k)}\\ t_S^{(k)}\end{pmatrix}, \begin{pmatrix}\mathbf{V}_S^{(k)}\\ V_{t_S}^{(k)}\end{pmatrix} \right\}; \end{aligned}$$

$\mathbf{S}^{(k+1)} \leftarrow \mathcal{S}_{\lambda_S}[\mathbf{S}^{(k+\frac{1}{2})} - \mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k+\frac{1}{2})} + \mathbf{S}^{(k+\frac{1}{2})} - \mathbf{D}]]$

$t_S^{(k+1)} \leftarrow \|\mathbf{S}^{(k+1)}\|_1$

$U_L^{(k+1)} \leftarrow g(\mathbf{L}^{(K+1)}, \mathbf{S}^{(k+1)}, t_L^{(k+1)}), t_S^{(k+1)}/\lambda_L$

$U_S^{(k+1)} \leftarrow g(\mathbf{L}^{(K+1)}, \mathbf{S}^{(k+1)}, t_L^{(k+1)}), t_S^{(k+1)}/\lambda_S$

end for

return $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+1)})$

Assumption 1 is a very generic and natural assumption about the restriction operator and it is satisfied for all restriction operators used in this work. Indeed, there is no practical advantage of having redundant columns in $\mathbf{R}$, and we can always remove the redundant columns thus creating lower dimensional coarse models. We make two generic assumptions about the coarse model.

Assumption 2.

$$\operatorname{rank}(\mathbf{L}^\star) \leq n_H \leq \frac{m+1}{2}.$$

The first inequality of (2) holds whenever $n < m$, which is the case in all practical problems we consider in the paper. The second inequality, on the other hand, has to be explicitly enforced using an approximate guess on the rank of $\mathbf{L}^\star$, which is well known for most applications. For example, for the video background extraction problem $\operatorname{rank}(\mathbf{L}^\star) \approx 1$ and for the facial shadow removal problem it is ≈ 9. In all experiments we use this prior information to set the number of levels so that (2) is satisfied.

Assumption 3. *The low-rank component $\mathbf{L}^\star$ can be represented as $\mathbf{L}^\star = \mathbf{L}_H^\star \mathbf{R}^\top$ for some coarse $\mathbf{L}_H^\star \in \mathbb{R}^{m \times n_H}$.*

Assumption 3 is not restrictive for the problems considered here. We demonstrate this fact by using a simple example. Let $\mathbf{L}^\star$ be a simple white rank one background extracted from a video. This means that each of its columns is a white frame stacked as a column vector. Thus it can be written as a matrix of ones. Also, as it is a standard practice in multigrid literature [4], assume $\mathbf{R}$ is the normalised interpolation operator (3.1):

$$\mathbf{R}_n = \frac{1}{2}\begin{bmatrix} 2 & 0 & 0 & 0 & \dots & 0 & 0 \\ 2 & 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 2 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 1 & 0 & \dots & 0 & 0 \\ & & & & \dots & & \\ 0 & 0 & 0 & 0 & \dots & 1 & 2 \end{bmatrix} \in \mathbb{R}^{n \times \frac{n}{2}}.$$

Then setting $\mathbf{L_H}^\star$ as a matrix of ones we get:

$$\mathbf{L}_H^\star \mathbf{R}^\top = \begin{bmatrix} 1 & 1 & 1 \\ & \cdots & \\ 1 & 1 & 1 \end{bmatrix} \cdot \frac{1}{2} \begin{bmatrix} 2 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ & & & \cdots & & \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \mathbf{L}^\star.$$

As can be seen from this example, the interpolation restriction operator is suitable for rank one matrices. Indeed, consider the case where $\mathbf{L}_H^\star$ is constructed by repeating a column vector next to each other (this will obviously give a rank one matrix). In fact, it is easy to notice that multiplying a matrix $\mathbf{L}_H^\star$ by this $\mathbf{R}^\top$ from right, effectively adds linear combinations of columns of $\mathbf{L}_H^\star$ without altering the existing columns. And since $\mathbf{R}$ is normalised, the resulted higher dimensional matrix will have exactly

the same columns as $\mathbf{L}^\star$. Having this in mind, given a rank one $\mathbf{L}^\star$ with normalised columns, we can always construct a $\mathbf{L}_H^\star$ simply by removing a certain number of its columns.

Assumption 3 is one of the key assumptions of this paper. It essentially says that the solution can be represented with varying degrees of fidelity, which is what we observe in many applications, including those studied here. It is important to notice, that while we assume the existence of such $\mathbf{L}_H^\star$, we do *not* need to know its value. Nowhere in our multilevel algorithms we use the value of $\mathbf{L}_H^\star$.

In the multilevel literature the standard choice for restriction operator is the interpolation operator (3.1) and we will also use it in our experiments. Often in practice we use more than 2 levels of coarse models. Specifically, we use a restriction operator $\mathbf{R} = \mathbf{R}_n \cdot \mathbf{R}_{\frac{n}{2}} \cdot \ldots \cdot \mathbf{R}_{n_H} \in \mathbb{R}^{n \times n_H}$, where $\mathbf{R}_k \in \mathbb{R}^{k \times \frac{k}{2}}$ is the interpolation operator of appropriate dimensions. For all experiments we use up to the deepest possible levels, so that $n_H > \max\{\mathrm{rank}(\mathbf{L}^\star), r\}$, where r is the number of singular values required by the overlying algorithm. Clearly, this $\mathbf{R}$ has linearly independent columns and thus is full rank.

3.2 Multilevel IALM

In this section we present a computationally efficient multilevel variant of the Inexact ALM algorithm. First, we make an additional assumption for this subsection:

Assumption 4. $\mathbf{R}$ *is normalised so that* $\|\mathbf{R}\|_2 \leq \|\mathbf{R}\|_\star \leq 1$.

Before we proceed, we shall present an important inequality about singular values that we will use in the proofs. The proof can be found for instance in [33].

Theorem 3. *Let* $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$ *be matrices and* $m \geq n$. *Then for any* $k \in \{1, \ldots, n\}$

$$\sum_{i=1}^{k} \sigma_i(\mathbf{A})\sigma_{n-i+1}(\mathbf{B}) \leq \sum_{i=1}^{k} \sigma_i(\mathbf{AB}) \leq \sum_{i=1}^{k} \sigma_i(\mathbf{A})\sigma_i(\mathbf{B}).$$

Now we show that the restriction operator approximately preserves the nuclear norm.

Theorem 4. *For any* $\mathbf{L}_H \in \mathbb{R}^{m \times n_H}$ *and* $\mathbf{R} \in \mathbb{R}^{n \times n_H}$ *with* $n_H \leq n \leq m$, *the following inequalities hold:*

1. $$\|\mathbf{L}_H \mathbf{R}^\top\|_* \geq \|\mathbf{L}_H\|_* - \epsilon,$$
   with $\epsilon = \sum_{k=1}^{r_H} \sigma_k(\mathbf{L}_H)(1 - \sigma_{n_H - k + 1}(\mathbf{R}))$, *where* $r_H = \mathrm{rank}(\mathbf{L}_H)$; *and*
2. *if* $\|\mathbf{R}\|_2 \leq 1$, *then also*
   $$\|\mathbf{L}_H\|_* \geq \|\mathbf{L}_H \mathbf{R}^\top\|_*.$$

Proof. Since $n \leq n_H$, then using (3) we have

$$\begin{aligned}\|\mathbf{L}_H\mathbf{R}^\top\|_* &\geq \sum_{k=1}^{n_H} \sigma_k(\mathbf{L}_H\mathbf{R}^\top) \\ &\geq \sum_{k=1}^{n_H} \sigma_k(\mathbf{L}_H)\sigma_{n_H-k+1}(\mathbf{R}^\top) \\ &= \|\mathbf{L}_H\|_* - \sum_{k=1}^{r_H} \sigma_k(\mathbf{L}_H)(1-\sigma_{n_H-k+1}(\mathbf{R})).\end{aligned}$$

The second part can be shown similarly. From Assumption 1 we have

$$\begin{aligned}\|\mathbf{L}_H\|_* &= \|\mathbf{L}_H\mathbf{R}^\top{\mathbf{R}^\dagger}^\top\|_* \\ &\geq \sum_{k=1}^{n_H} \sigma_k(\mathbf{L}_H\mathbf{R}^\top{\mathbf{R}^\dagger}^\top) \\ &\geq \sum_{k=1}^{n_H} \sigma_k(\mathbf{L}_H\mathbf{R}^\top)\sigma_{n_H-k+1}({\mathbf{R}^\dagger}^\top) \\ &= \|\mathbf{L}_H\mathbf{R}^\top\|_* - \sum_{k=1}^{r_H} \sigma_k(\mathbf{L})(1-\sigma_{n_H-k+1}(\mathbf{R}^\dagger)) \\ &= \|\mathbf{L}_H\mathbf{R}^\top\|_* - \sum_{k=1}^{r_H} \sigma_k(\mathbf{L})(1-\sigma^{-1}_{n_H-k+1}(\mathbf{R})).\end{aligned}$$

Finally, if $\sigma_k(\mathbf{R}) \leq 1$ for all $k = n_H - r_H + 1, \ldots, n_H$, then $\|\mathbf{L}_H\mathbf{R}^\top\|_* \leq \|\mathbf{L}_H\|_\star$. □

As it can been seen from Theorem 4, in general having $\epsilon = 0$ requires an orthogonal $\mathbf{R}$, which may not be sparse, thus making each iteration of the algorithm computationally expensive. However, if $\mathbf{L}^\star$ is low rank with quickly decreasing singular values and $\sigma_i(\mathbf{R})$ drop slowly, then ϵ will be small. This is indeed the case for many computer vision applications such as video background extraction and facial shadow removal discussed in the paper, where $\mathbf{L}^\star$ is not only low rank, but also has quadratically decreasing singular values. Moreover, it is easy to check that the singular values of the interpolation restriction operator (3.1) satisfy $\sigma_1(\mathbf{R})/\sigma_{n_H}(\mathbf{R}) \leq 2$.

Now we proceed to define a coarse model for the Augmented Lagrangian function (3.2):

$$\mathcal{L}(\mathbf{L}, \mathbf{S}, \mathbf{Y}, \mu) = \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 + \langle \mathbf{Y}, \mathbf{D} - \mathbf{L} - \mathbf{S}\rangle + \frac{\mu}{2}\|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F^2,$$

For any fixed $\mathbf{S}$, $\mathbf{Y}$ and μ, we define the coarse augmented Lagrangian function of (2) as

$$\mathcal{L}_H(\mathbf{L}) = \|\mathbf{L}\|_* + \lambda\|\mathbf{S}\|_1 + \langle \mathbf{Y}, [(\mathbf{D} - \mathbf{S})\mathbf{R} - \mathbf{L}]\mathbf{R}^\top\rangle + \frac{\mu}{2}\|(\mathbf{D} - \mathbf{S})\mathbf{R} - \mathbf{L}\|_F^2.$$

We will use the minimiser of $\mathcal{L}_H$ to approximately minimise the true Augmented Lagrangian function (3.2) over $\mathbf{L}$. The minimiser of $\mathcal{L}_H$ is given by the singular value thresholding operator as

$$\mathbf{L}_H = \mathbf{U}_H \mathcal{S}_{\mu^{-1}}[\mathbf{\Sigma}_H]\mathbf{V}_H^\top,$$

where $\mathbf{U}_H\mathbf{\Sigma}_H\mathbf{V}_H^\top = (\mathbf{D} - \mathbf{S} + \mu^{-1}\mathbf{Y})\mathbf{R} = \mathbf{M}\mathbf{R}$. Then we use the prolongation operator $\mathbf{R}^\top$ to lift the minimiser of (3.2) to the fine dimension. The Multilevel IALM algorithm becomes as stated in Algorithm 4.

Algorithm 4 Multilevel Inexact ALM (ML-IALM)

Input: $\mathbf{D}, \mathbf{S}^{(0)}, \mathbf{Y}^{(0)} \in \mathbb{R}^{m\times n}$; $\mu_0 > 0$

for $k \leftarrow 1$ to ... **do**
  // Solve $\mathbf{L}_H^{(k+1)} = \arg\min_{\mathbf{L}} \mathcal{L}_H(\mathbf{L}, \mathbf{S}^{(k)}, \mathbf{Y}^{(k)}, \mu_k)$
  $\mathbf{M}_H^{(k)} \leftarrow (\mathbf{D} - \mathbf{S}^{(k)} + \mu_k^{-1}\mathbf{Y}^{(k)})\mathbf{R}$
  $(\mathbf{U}_H, \mathbf{\Sigma}_H, \mathbf{V}_H) \leftarrow \text{SVD}(\mathbf{M}_H^{(k)})$
  $\mathbf{L}_H^{(k+1)} \leftarrow \mathbf{U}_H \mathcal{S}_{\mu^{-1}}[\mathbf{\Sigma}_H]\mathbf{V}_H^\top$
  $\mathbf{L}^{(k+1)} \leftarrow \mathbf{L}_H^{(k+1)}\mathbf{R}^\top$
  // Continue as in Algorithm 1
  $\mathbf{S}^{(k+1)} \leftarrow \mathcal{S}_{\lambda\mu_k^{-1}}[\mathbf{D} - \mathbf{L}^{(k+1)} + \mu_k^{-1}\mathbf{Y}^{(k)}]$
  $\mathbf{Y}^{(k+1)} \leftarrow \mathbf{Y}^{(k)} + \mu_k(\mathbf{D} - \mathbf{L}^{(k+1)} - \mathbf{S}^{(k+1)})$
  Update $\mu_k \leftarrow \mu_{k+1}$
end for
return $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+1)})$

Now we proceed to study the convergence of Algorithm 4. First, we define the sequence $\widehat{\mathbf{Y}}^{(k)}$, which we will use throughout this section. Let

$$\widehat{\mathbf{Y}}^{(k+1)} = \mathbf{Y}^{(k)} + \mu_k[\mathbf{D} - \mathbf{S}^{(k)} - \mathbf{L}^{(k+1)}].$$

We start the convergence proof of Algorithm 4 with the multilevel versions of some lemmas of [17]. We begin with Lemma 1 from [17] and show that Multilevel IALM also produces bounded sequences.

Lemma 1. *Let $\widehat{\mathbf{Y}}^{(k+1)}$ be defined as in (3.2) and $\mathbf{Y}^{(k)}$ be defined as in Algorithm 4. Then the sequence $\{\mathbf{Y}^{(k)}\}$ is bounded and the following hold:*

1.

$$\widehat{\mathbf{Y}}^{(k+1)}\mathbf{R} + \mu_k\mathbf{L}_H^{(k+1)}(\mathbf{R}^\top\mathbf{R} - \mathbf{I}) \in \partial\|\mathbf{L}_H^{(k+1)}\|_*,$$

for any $\mathbf{L}_H^{(k+1)} \in \mathbb{R}^{m\times n_H}$.

2.

$$\mathbf{Y}^{(k+1)} \in \partial\|\mathbf{S}^{(k+1)}\|_*.$$

Proof. To show (1) we use the optimality condition of $\mathcal{L}_H$ and the construction $\mathbf{L}^{(k+1)} = \mathbf{L}_H^{(k+1)}\mathbf{R}^\top$. We have

$$\begin{aligned}\mathbf{0} \in & \partial_{\mathbf{L}_H}\mathcal{L}(\mathbf{L}_H^{(k+1)}, \mathbf{S}^{(k)}, \mathbf{Y}^{(k)}, \mu_k) \\ = & \partial\|\mathbf{L}_H^{(k+1)}\|_* - \mathbf{Y}^{(k)}\mathbf{R} - \mu_k[(\mathbf{D} - \mathbf{S}^{(k)})\mathbf{R} - \mathbf{L}_H^{(k+1)}] \\ = & \partial\|\mathbf{L}_H^{(k+1)}\|_* - \widehat{\mathbf{Y}}^{(k+1)}\mathbf{R} - \mu_k(\mathbf{L}_H^{(k+1)}\mathbf{R}^\top\mathbf{R} - \mathbf{L}_H^{(k+1)}).\end{aligned}$$

Similarly, using the optimality condition for updating $\mathbf{S}^{(k+1)}$ we can show (2):

$$\mathbf{Y}^{(k+1)} \in \partial\|\mathbf{S}^{(k+1)}\|_*.$$

□

Lemma 2 ([17], Lemma 2)**.**

$$\begin{aligned}&\|\mathbf{S}^{(k+1)} - \mathbf{S}^\star\|_F^2 + \mu_k^{-2}\|\mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\|_F^2 \\ =&\|\mathbf{S}^{(k)} - \mathbf{S}^\star\|_F^2 + \mu_k^{-2}\|\mathbf{Y}^{(k)} - \mathbf{Y}^\star\|_F^2 - \|\mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}\|_F^2 - \mu_k^{-2}\|\mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}\|_F^2 \\ &- 2\mu_k^{-1}(\langle\mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}, \mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}\rangle + \langle\mathbf{L}^{(k+1)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k+1)} - \mathbf{Y}^\star\rangle + \langle\mathbf{S}^{(k+1)} - \mathbf{S}^\star, \mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\rangle).\end{aligned}$$

Proof. Since the proof of Lemma 2 in [17] relies only on the optimality of $\mathbf{L}^\star$ and $\mathbf{S}^\star$ and the update formula for $\mathbf{Y}^{(k+1)}$, but not on the update rule for $\mathbf{L}^{(k+1)}$ (the only multilevel update part of ML-IALM), then its proof can be exactly repeated for Algorithm 4. □

Lemma 3. *If $\|\mathbf{R}\|_2 \leq 1$, then under Assumption 3*

$$\langle\mathbf{L}^{(k+1)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k+1)} - \mathbf{Y}^\star\rangle \geq -\epsilon - \mu_k\Delta_{k+1},$$

for ϵ defined as in Theorem 4 and $\Delta_{k+1} = \langle\mathbf{L}_H^{(k+1)}(\mathbf{R}^\top\mathbf{R} - \mathbf{I}), \mathbf{L}_H^{(k+1)} - \mathbf{L}_H^\star\rangle$.

Proof. Since $\mathbf{Y}^\star \in \partial\|\mathbf{L}^\star\|_*$, we have

$$\|\mathbf{L}_H^{(k+1)}\mathbf{R}^\top\|_* - \|\mathbf{L}_H^\star\mathbf{R}^\top\|_* \geq \langle\mathbf{Y}^\star, \mathbf{L}^{(k+1)} - \mathbf{L}^\star\rangle.$$

Then applying Theorem 4 with $\mathbf{L}_H = \mathbf{L}_H^\star$ we derive:

$$\|\mathbf{L}_H^{(k+1)}\mathbf{R}^\top\|_* - \|\mathbf{L}_H^\star\|_* \geq \langle\mathbf{Y}^\star, \mathbf{L}^{(k+1)} - \mathbf{L}^\star\rangle - \epsilon.$$

On the other hand, from (1) we have

$$\|\mathbf{L}_H^\star\|_* - \|\mathbf{L}_H^{(k+1)}\|_* \geq \langle\widehat{\mathbf{Y}}^{(k+1)}\mathbf{R} + \mu_k\mathbf{L}_H^{(k+1)}(\mathbf{R}^\top\mathbf{R} - \mathbf{I}), \mathbf{L}_H^\star - \mathbf{L}_H^{(k+1)}\rangle.$$

Then adding (3.2) and (3.2) and using Assumption 3 we get

$$\|\mathbf{L}_H^{(k+1)}\mathbf{R}^\top\|_* - \|\mathbf{L}_H^{(k+1)}\|_* \geq \langle \mathbf{Y}^\star - \widehat{\mathbf{Y}}^{(k+1)}, \mathbf{L}^{(k+1)} - \mathbf{L}^\star\rangle - \epsilon - \mu_k \langle \mathbf{L}_H^{(k+1)}(\mathbf{R}^\top\mathbf{R} - \mathbf{I}), \mathbf{L}_H^{(k+1)} - \mathbf{L}_H^\star\rangle.$$

We finish the proof applying the construction $\mathbf{L}^{(k+1)} = \mathbf{L}_H^{(k+1)}\mathbf{R}^\top$ and denoting $\Delta_{k+1} := \langle \mathbf{L}_H^{(k+1)}(\mathbf{R}^\top\mathbf{R} - \mathbf{I}), \mathbf{L}_H^{(k+1)} - \mathbf{L}_H^\star\rangle$.

□

Lemma 4. *Let Δ_k be defined as in Lemma 3 and define*

$$c_k := \mu_k^{-1}(\langle \mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}, \mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}\rangle + \langle \mathbf{L}^{(k+1)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k+1)} - \mathbf{Y}^\star\rangle + \langle \mathbf{S}^{(k+1)} - \mathbf{S}^\star, \mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\rangle).$$

Then if μ_k is non-decreasing, then

- $c_k \geq -\epsilon\mu_k^{-1} - \Delta_{k+1}$, *and*
- $\sum_{k=1}^{k=+\infty} c_k < +\infty$.

Proof. Let $(\mathbf{L}^\star, \mathbf{S}^\star, \mathbf{Y}^\star)$ be a saddle point of the Lagrangian of (2). So we have

$$\mathbf{Y}^\star \in \partial\|\mathbf{L}^\star\|_*, \qquad \mathbf{Y}^\star \in \partial\|\lambda\mathbf{S}^\star\|_1.$$

Then from Lemma 3 of [17] and $\mathbf{Y}^{(k+1)} \in \partial\|\lambda\mathbf{S}^{(k+1)}\|_1$ we have

$$\langle \mathbf{S}^{(k+1)} - \mathbf{S}^\star, \mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\rangle \geq 0,$$
$$\langle \mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}, \mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}\rangle \geq 0.$$

Adding (3) and the two inequalities of (3.2) we get

$$\begin{aligned}&-\epsilon - \mu_k\Delta_{k+1}\\ \leq&\langle \mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}, \mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}\rangle + \langle \mathbf{L}^{(k+1)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k+1)} - \mathbf{Y}^\star\rangle + \langle \mathbf{S}^{(k+1)} - \mathbf{S}^\star, \mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\rangle.\end{aligned}$$

Showing that $c_k \geq -\epsilon\mu_k^{-1} - \Delta_{k+1}$.

Show part 2, we apply Lemma 2 and use $\mu_{k+1} \geq \mu_k$ to get

$$\|\mathbf{S}^{(k+1)} - \mathbf{S}^\star\|_F^2 + \mu_{k+1}^{-2}\|\mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\|_F^2 \leq \|\mathbf{S}^{(k)} - \mathbf{S}^\star\|_F^2 + \mu_k^{-2}\|\mathbf{Y}^{(k)} - \mathbf{Y}^\star\|_F^2 + 2\epsilon\mu_k^{-1} + 2\Delta_{k+1}.$$

Therefore from Lemma 2 we conclude that

$$\begin{aligned}&2\mu_k^{-1}(\langle \mathbf{Y}^{(k+1)} - \mathbf{Y}^{(k)}, \mathbf{S}^{(k+1)} - \mathbf{S}^{(k)}\rangle + \langle \mathbf{L}^{(k+1)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k+1)} - \mathbf{Y}^\star\rangle\\ &+ \langle \mathbf{S}^{(k+1)} - \mathbf{S}^\star, \mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\rangle)\\ \leq&(\|\mathbf{S}^{(k)} - \mathbf{S}^\star\|_F^2 + \mu_k^{-2}\|\mathbf{Y}^{(k)} - \mathbf{Y}^\star\|_F^2) - (\|\mathbf{S}^{(k+1)} - \mathbf{S}^\star\|_F^2 + \mu_{k+1}^{-2}\|\mathbf{Y}^{(k+1)} - \mathbf{Y}^\star\|_F^2).\end{aligned}$$

□

Theorem 5 (Convergence of Multilevel IALM)**.** *For Algorithm 4, if* $\{\mu_k\}$ *is non-decreasing,* $\sum_{k=1}^{+\infty}\mu_k^{-1}=+\infty$ *and* $\sum_{k=1}^{+\infty}\mu_k^{-2}<+\infty$*, then* $(\mathbf{L}^{(k)},\mathbf{S}^{(k)})$ *asymptotically converges to an approximate solution of (2).*

Proof. Similarly to the proof of Lemma 4 we have that

$$\begin{aligned}
&\mu_k^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^{(k)}\|_F^2-2\epsilon\mu_k^{-1}-2[\Delta_{k+1}]_+\\
\leq&\mu_k^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^{(k)}\|_F^2-2\epsilon\mu_k^{-1}-2\Delta_{k+1}\\
\leq&(\|\mathbf{S}^{(k)}-\mathbf{S}^\star\|_F^2+\mu_k^{-2}\|\mathbf{Y}^{(k)}-\mathbf{Y}^\star\|_F^2)-(\|\mathbf{S}^{(k+1)}-\mathbf{S}^\star\|_F^2+\mu_{k+1}^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^\star\|_F^2)\\
&-2\epsilon\mu_k^{-1}-2\Delta_{k+1}-2\mu_k^{-1}(\langle\mathbf{Y}^{(k+1)}-\mathbf{Y}^{(k)},\mathbf{S}^{(k+1)}-\mathbf{S}^{(k)}\rangle+\langle\mathbf{L}^{(k+1)}-\mathbf{L}^\star,\widehat{\mathbf{Y}}^{(k+1)}-\mathbf{Y}^\star\rangle\\
&+\langle\mathbf{S}^{(k+1)}-\mathbf{S}^\star,\mathbf{Y}^{(k+1)}-\mathbf{Y}^\star\rangle)\\
\leq&(\|\mathbf{S}^{(k)}-\mathbf{S}^\star\|_F^2+\mu_k^{-2}\|\mathbf{Y}^{(k)}-\mathbf{Y}^\star\|_F^2)-(\|\mathbf{S}^{(k+1)}-\mathbf{S}^\star\|_F^2+\mu_{k+1}^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^\star\|_F^2).
\end{aligned}$$

Therefore,

$$\sum_{k=1}^{+\infty}(\mu_k^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^{(k)}\|_F^2-2\epsilon\mu_k^{-1}-2[\Delta_{k+1}]_+)<+\infty.$$

Thus, $\mu_k^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^{(k)}\|_F^2-2\epsilon\mu_k^{-1}-2[\Delta_{k+1}]_+\to 0$, and since $2\epsilon\mu_k^{-1}\to 0$ we see that

$$\|\mathbf{D}-\mathbf{L}^{(k)}-\mathbf{S}^{(k)}\|_F=\mu_k^{-1}\|\mathbf{Y}^{(k)}-\mathbf{Y}^{(k-1)}\|_F\to(2[\Delta_{k+1}]_+)^{1/2},$$

Moreover, since we showed in Lemma 1 that $\{\mathbf{Y}^{(k+1)}\}$ is a bounded sequence, it follows that so is $[\Delta_k]_+$. Therefore, denoting $\max\{\Delta_k\}:=\delta$ we show that any accumulation point of $(\mathbf{L}^{(k)},\mathbf{S}^{(k)})$ is a δ-feasible solution.

On the other hand, denote the optimal objective value of problem (2) by f^*. As $\widehat{\mathbf{Y}}^{(k)}\mathbf{R}\in\partial\|\mathbf{L}_H^{(k)}\|_*$ and $\mathbf{Y}^{(k)}\in\partial(\lambda\|\mathbf{S}^{(k)}\|_1)$, under Assumption 4 we have

$$\begin{aligned}
&\|\mathbf{L}^{(k)}\|_*+\lambda\|\mathbf{S}^{(k)}\|_1\\
\leq&\|\mathbf{L}_H^{(k)}\|_*+\lambda\|\mathbf{S}^{(k)}\|_1\\
\leq&\|\mathbf{L}_H^\star\|_*+\lambda\|\mathbf{S}^\star\|_1-\langle\widehat{\mathbf{Y}}^{(k)}\mathbf{R},\mathbf{L}_H^\star-\mathbf{L}_H^{(k)}\rangle-\langle\mathbf{Y}^{(k)},\mathbf{S}^\star-\mathbf{S}^{(k)}\rangle\\
\leq&\|\mathbf{L}^\star\|_*+\epsilon+\lambda\|\mathbf{S}^\star\|_1-\langle\widehat{\mathbf{Y}}^{(k)},\mathbf{L}^\star-\mathbf{L}^{(k)}\rangle-\langle\mathbf{Y}^{(k)},\mathbf{S}^\star-\mathbf{S}^{(k)}\rangle\\
=&f^\star+\langle\mathbf{Y}^\star-\widehat{\mathbf{Y}}^{(k)},\mathbf{L}^\star-\mathbf{L}^{(k)}\rangle+\langle\mathbf{Y}^\star-\mathbf{Y}^{(k)},\mathbf{S}^\star-\mathbf{S}^{(k)}\rangle-\langle\mathbf{Y}^\star,\mathbf{L}^\star-\mathbf{L}^{(k)}+\mathbf{S}^\star-\mathbf{S}^{(k)}\rangle+\epsilon\\
=&f^\star+\langle\mathbf{Y}^\star-\widehat{\mathbf{Y}}^{(k)},\mathbf{L}^\star-\mathbf{L}^{(k)}\rangle+\langle\mathbf{Y}^\star-\mathbf{Y}^{(k)},\mathbf{S}^\star-\mathbf{S}^{(k)}\rangle-\langle\mathbf{Y}^\star,\mathbf{D}-\mathbf{L}^{(k)}-\mathbf{S}^{(k)}\rangle+\epsilon.
\end{aligned}$$

Similarly to Lemma 4 we notice that from Lemma 2

$$\begin{aligned}
&\mu_k^{-1}(\langle\mathbf{L}^{(k)}-\mathbf{L}^\star,\widehat{\mathbf{Y}}^{(k)}-\mathbf{Y}^\star\rangle+\langle\mathbf{S}^{(k)}-\mathbf{S}^\star,\mathbf{Y}^{(k)}-\mathbf{Y}^\star\rangle)\\
\leq&(\|\mathbf{S}^{(k)}-\mathbf{S}^\star\|_F^2+\mu_k^{-2}\|\mathbf{Y}^{(k)}-\mathbf{Y}^\star\|_F^2)-(\|\mathbf{S}^{(k+1)}-\mathbf{S}^\star\|_F^2+\mu_{k+1}^{-2}\|\mathbf{Y}^{(k+1)}-\mathbf{Y}^\star\|_F^2).
\end{aligned}$$

And therefore,

$$\sum_{k=1}^{k=+\infty} \mu_k^{-1}(\langle \mathbf{L}^{(k)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k)} - \mathbf{Y}^\star \rangle + \langle \mathbf{S}^{(k)} - \mathbf{S}^\star, \mathbf{Y}^{(k)} - \mathbf{Y}^\star \rangle) < +\infty.$$

As $\sum_{k=1}^{+\infty} \mu_k^{-1} = +\infty$, there must exist a subsequence $(\mathbf{L}^{(k_j)}, \mathbf{S}^{(k_j)})$ such that

$$\langle \mathbf{L}^{(k_j)} - \mathbf{L}^\star, \widehat{\mathbf{Y}}^{(k_j)} - \mathbf{Y}^\star \rangle + \langle \mathbf{S}^{(k_j)} - \mathbf{S}^\star, \mathbf{Y}^{(k_j)} - \mathbf{Y}^\star \rangle \to 0.$$

Then since $\|\mathbf{D} - \mathbf{L}^{(k)} - \mathbf{S}^{(k)}\|_F \leq \delta$, we have that

$$\lim_{j \to +\infty} \|\mathbf{L}^{(k_j)}\|_* + \lambda \|\mathbf{S}^{(k_j)}\|_1 \leq f^\star + \epsilon + \delta.$$

So $(\mathbf{L}^{(k_j)}, \mathbf{S}^{(k_j)})$ approaches to an $(\epsilon + \delta)$-approximate solution of problem (2).

□

Notice that Theorem 5 gives a similar convergence results as Theorem 1, meaning that one should expect a similar number of iterations for IALM and ML-IALM methods. This is indeed the case, as observed from empirical studies. However, since ML-IALM performs SVDs on much smaller dimensional matrices, each iteration is significantly cheaper.Of course, as opposed to the original IALM algorithm, here we only showed an approximate convergence. However, as several numerical experiments will demonstrate, the approximation error is practically negligible.

3.3 Multilevel Frank-Wolfe

In this section we use operator notation for the linear restriction operator, i.e. $\mathcal{R} : \mathcal{H} \to \mathcal{H}_H$ is a linear operator from the original space $\mathcal{H}$ to the *coarse space* $\mathcal{H}_H$. $\mathcal{H}$ and $\mathcal{H}_H$ are both Hilbert spaces endowed with inner products and $\mathcal{H}_H$ has lower dimension. In the next subsection we will see what $\mathcal{H}$, $\mathcal{H}_H$ and $\mathcal{R}$ are for the CPCP model.

First we create a coarse model for the gradient by applying the restriction operator $\mathcal{R}$, then solve the linear optimisation oracle over the coarse gradient, then lift the solution back to the original dimension applying the transpose of the restriction operator. For the algorithm we use a convex set $\mathcal{D}_H \subseteq \mathcal{H}_H$ such that for every $\mathbf{x}_H \in \mathcal{D}_H$ it holds that $\mathcal{R}^\top(\mathbf{x}_H) \in \mathcal{D}$. The method is given in Algorithm 5.

Using techniques similar to the original proof [20], it can be shown that the ML-FW method converges to a point obtained from the coarse level at a $\mathcal{O}(1/k)$ rate in function values.

Theorem 6. *For any $\mathbf{x}_H^\star \in \mathcal{D}_H$ and for $\{\mathbf{x}^{(k)}\}$ generated by Algorithm 5, we have for any $\mathbf{x}_H^\star \in \mathcal{H}_H$ and $k = 0, 1, 2, ..$*

$$f(\mathbf{x}^{(k)}) - f(\mathcal{R}^\top(\mathbf{x}_H^\star)) \leq \frac{2LD^2}{k+2}.$$

Proof. For $k = 0, 1, 2, \ldots$ we have

Algorithm 5 Multilevel Frank-Wolfe (ML-FW)

Input: $\mathbf{x}_H^{(0)} \in \mathcal{D}_H$

1: $\mathbf{x}^{(0)} \leftarrow \mathcal{R}^\top(\mathbf{x}_H^{(0)})$
2: **for** $k \leftarrow 0, 1, \text{ to } \ldots$ **do**
3: $\quad \mathbf{v}_H^{(k)} \in \arg\min_{\mathbf{v} \in \mathcal{D}_H} \langle \mathbf{v}, \mathcal{R}(\nabla f(\mathbf{x}^{(k)})) \rangle$
4: $\quad \mathbf{v}^{(k)} \leftarrow \mathcal{R}^\top(\mathbf{v}_H^{(k)})$
5: $\quad \gamma \leftarrow \frac{2}{k+2}$
6: $\quad$ Set $\mathbf{x}^{(k+1)} \in \mathcal{D}$ so that $f(\mathbf{x}^{(k+1)}) \leq f(\mathbf{x}^{(k)} + \gamma(\mathbf{v}^{(k)} - \mathbf{x}^{(k)}))$
7: **end for**
8: **return** $\mathbf{x}^{(k+1)}$

$$
\begin{aligned}
f(\mathbf{x}^{(k+1)}) &\leq f(\mathbf{x}^{(k)} + \gamma(\mathbf{v}^{(k)} - \mathbf{x}^{(k)})) \\
&\leq f(\mathbf{x}^{(k)}) + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathbf{v}^{(k)} - \mathbf{x}^{(k)} \rangle + \frac{L\gamma^2}{2}\|\mathbf{v}^{(k)} - \mathbf{x}^{(k)}\|^2 \\
&\leq f(\mathbf{x}^{(k)}) + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathcal{R}^\top(\mathbf{v}_H^{(k)}) - \mathcal{R}^\top(\mathbf{x}_H^{(k)}) \rangle + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathcal{R}^\top(\mathbf{x}_H^{(k)}) - \mathbf{x}^{(k)} \rangle \\
&\quad + \frac{\gamma^2 LD^2}{2} \\
&= f(\mathbf{x}^{(k)}) + \gamma\langle \mathcal{R}(\nabla f(\mathbf{x}^{(k)})), \mathbf{v}_H^{(k)} - \mathbf{x}_H^{(k)} \rangle + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathcal{R}^\top(\mathbf{x}_H^{(k)}) - \mathbf{x}^{(k)} \rangle + \frac{\gamma^2 LD^2}{2} \\
&\leq f(\mathbf{x}^{(k)}) + \gamma\langle \mathcal{R}(\nabla f(\mathbf{x}^{(k)})), \mathbf{x}_H^\star - \mathbf{x}_H^{(k)} \rangle + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathcal{R}^\top(\mathbf{x}_H^{(k)}) - \mathbf{x}^{(k)} \rangle + \frac{\gamma^2 LD^2}{2} \\
&= f(\mathbf{x}^{(k)}) + \gamma\langle \nabla f(\mathbf{x}^{(k)}), \mathcal{R}^\top(\mathbf{x}_H^\star) - \mathbf{x}^{(k)} \rangle + \frac{\gamma^2 LD^2}{2} \\
&\leq f(\mathbf{x}^{(k)}) + \gamma(f(\mathcal{R}^\top(\mathbf{x}_H^\star)) - f(\mathbf{x}^{(k)})) + \frac{\gamma^2 LD^2}{2}.
\end{aligned}
$$

Here for the first line we used the updating rule in Algorithm 5; for the second line we used the Lipschitz continuity of f; for the third line - the definitions of $\mathbf{v}_H^{(k)}$ and D, and we added and subtracted $\mathcal{R}^\top(\mathbf{x}_H^{(k)})$; for the fourth line - the property of inner product; for the fifth line - the optimality of $\mathbf{v}_H^{(k)}$; for the sixth line - the property of inner product and the definition of $\mathbf{x}^{(k)}$; and for the last line - the convexity of f. Then rearranging the terms we get

$$
f(\mathbf{x}^{(k+1)}) - f(\mathcal{R}^\top(\mathbf{x}_H^\star)) \leq (1-\gamma)(f(\mathbf{x}^{(k)}) - f(\mathcal{R}^\top(\mathbf{x}_H^\star))) + \frac{\gamma^2 LD^2}{2}.
$$

Therefore, by mathematical induction, it can be verified that

$$
f(\mathbf{x}^{(k)}) - f(\mathcal{R}^\top(\mathbf{x}_H^\star)) \leq \tfrac{2LD^2}{k+2} \quad \text{for} \quad k = 1, 2, 3, \ldots.
$$

□

Theorem 6 tells us, that if the minimiser $\mathbf{x}^\star$ can be accurately represented in terms of a coarse variable, then ML-FWT is a good and efficient method for that particular problem. As we will see in the next subsection, this is indeed the case for the CPCP model.

3.4 Multilevel Frank-Wolfe Thresholding for CPCP

In this subsection we will modify the Multilevel Frank-Wolfe method similarly to the Frank-Wolfe Thresholding method introducing the Multilevel Frank-Wolfe Thresholding method and apply it for the CPCP problem (2). In this case as well we will apply the multilevel update only on nuclear ball projections.

We begin with defining the fine and coarse spaces, and the restriction operator for the CPCP problem. In this setting our variables becomes $\mathbf{x} = (\mathbf{L}, \mathbf{S}, t_L, t_S)$ and the space is $\mathcal{H} = \mathbb{R}^{m\times n} \times \mathbb{R}^{m\times n} \times \mathbb{R} \times \mathbb{R}$.

Since we are applying the multilevel steps only for updating $\mathbf{L}^{(k)}$, we will use the following restriction operator:

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_x & & & \\ & \mathbf{I}_{n\times n} & & \\ & & 1 & \\ & & & 1 \end{pmatrix},$$

so that

$$\begin{aligned} &\mathcal{R}(\mathbf{L}, \mathbf{S}, \lambda_L, \lambda_S) = (\mathbf{L}\mathbf{R}_x, \mathbf{S}, \lambda_L, \lambda_S) \\ &\mathcal{R}^\top(\nabla_{\mathbf{L}} f, \nabla_{\mathbf{S}} f, \nabla_{\lambda_L} f, \nabla_{\lambda_S} f) = (\nabla_{\mathbf{L}} f \mathbf{R}_x, \nabla_{\mathbf{S}} f, \nabla_{\lambda_L} f, \nabla_{\lambda_S} f), \end{aligned}$$

and thus $\mathcal{R}(\nabla f(\mathbf{x}))$ only affects $\nabla_{\mathbf{L}} f(\mathbf{L}, \mathbf{S}, t_L, t_S)$ and correspondingly, $\mathcal{R}^\top(\mathbf{v}_H)$ only affects $\nabla f_{\mathbf{L}}$. Here $\mathbf{R}_x = \mathbf{R}_n \cdot \ldots \cdot \mathbf{R}_{n_H}$ is the restriction operator as defined in Section 3.1. Therefore, the coarse space becomes $\mathcal{H}_H = \mathbb{R}^{m\times n_H} \times \mathbb{R}^{m\times n} \times \mathbb{R} \times \mathbb{R}$. Now we can define the coarse feasibility set $\mathcal{D}_H$ for the CPCP problem as follows:

$$\|\mathbf{M}_{L,H}\|_* \leq 1/\|\mathcal{R}\|_*,$$

so that for each $k = 0, 1, \ldots$

$$\|\mathbf{M}_L^{(k)}\|_* = \|\mathcal{R}^T(\mathbf{M}_{L,H}^{(k)})\|_* \leq \|\mathcal{R}\|_* \|\mathbf{M}_{L,H}^{(k)}\|_* \leq 1,$$

is a feasible point of the fine problem, where $\mathbf{M}_{L,H}$ is the coarse variable. We call the new algorithm Multilevel Frank-Wolfe Thresholding (ML-FWT) (Algorithm (6)).

Thus the multilevel update step for $\mathbf{M}_L^{(k)}$ requires calculating only the largest singular value with the corresponding singular vectors for much lower dimensional matrices. The next theorem gives convergence guarantees for the ML-FWT method.

Theorem 7. *Let $f(\mathbf{L}, \mathbf{S}, t_L, t_S)$ be defined as in (2.3) and $\operatorname{rank}(\mathbf{L}^\star) \leq n_H$. Then under Assumptions 1 and 3, for $\mathbf{x}^{(k)}$, $k = 1, 2, \ldots$ defined as in Algorithm 6 the following holds*

$$f(\mathbf{x}^{(k)}) - f(\mathbf{x}^\star) \leq \frac{2LD^2}{k+2}.$$

Algorithm 6 Multilevel Frank-Wolfe Thresholing (ML-FWT)

Input: $\mathbf{D} \in \mathbb{R}^{m\times n}; \lambda_L, \lambda_S > 0$

Initialize as in Algorithm 3
for $k \leftarrow 1$ to ... **do**
    $\mathbf{M}_{L,H}^{(k)} \in \underset{\|\mathbf{M}_{L,H}\|_* \leq 1/\|\mathcal{R}\|_*}{\arg\min} \langle \mathcal{R}(\mathcal{P}_{\mathcal{Q}}[\mathbf{L}^{(k)} + \mathbf{S}^{(k)} - \mathbf{D}]), \mathbf{M}_{L,H}\rangle$
    $\mathbf{M}_L^{(k)} \leftarrow \mathcal{R}^\top(\mathbf{M}_{L,H}^{(k)})$
    // Continue as in Algorithm 3
end for
return $(\mathbf{L}^{(k+1)}, \mathbf{S}^{(k+1)})$

Proof. From assumption 3 we derive

$$\begin{aligned} f(\mathcal{R}^\top(\mathbf{L}_H^\star, \mathbf{S}^\star, t_L^\star, t_S^\star)) =& \frac{1}{2}\|\mathcal{P}_{\mathcal{Q}}[\mathbf{D} - \mathbf{L}_H^\star \mathbf{R}^\top - \mathbf{S}^\star]\|_F^2 + \lambda_L t_L^\star + \lambda_S t_S^\star \\ =& \frac{1}{2}\|\mathcal{P}_{\mathcal{Q}}[\mathbf{D} - \mathbf{L}^\star - \mathbf{S}^\star]\|_F^2 + \lambda_L t_L^\star + \lambda_S t_S^\star \\ =& f(\mathbf{L}^\star, \mathbf{S}^\star, t_L^\star, t_S^\star). \end{aligned}$$

Therefore, the claim follows from Theorem 6.

□

Finally, note that since FWT and ML-FWT have the same convergence rate and multiplying by the sparse matrix $\mathbf{R}$ (and its powers) is much cheaper than computing one singular value (although both have the same $\mathcal{O}(mn)$ worst case complexity), ML-FWT has a lower overall complexity.

4 Experiments

To test the practical efficiency of the proposed methods we compare them with the standard Inexact ALM [17] and Frank-Wolfe Thresholding [24] algorithms on several synthetically generated problems, as well as real life video background extraction and facial shadow removal problems. For the standard Inexact ALM and Frank-Wolfe Thresholding algorithms we used the provided Matlab code. Then for each multilevel variant we replaced the standard singular value thresholding parts of respective algorithms with corresponding multilevel singular value thresholding code, keeping the rest of the algorithms unchanged. Particularly, we used the same optimality criteria, so that the comparisons are fair. All methods were tested in Matlab R2015a on a standard Ubuntu 16.4 machine with Intel Core i7 processor and 32GB RAM. The code is available online at https://github.com/vahanhov/ml-rpca.

problem			IALM			ML-IALM		
dimensions	rank	sec	$f^\star$	error($\mathbf{L}^\star$)	error($\mathbf{S}^\star$)	$f^\star$	error($\mathbf{L}^\star$)	error($\mathbf{S}^\star$)
5000×100	2	5	19	7	0.1	10	1	0.02
5000×100	5	5	18	6	0.1	7.5	1	0.02
5000×1000	2	10	64	42	0.8	7	1	0.01
5000×1000	5	10	64	43	0.8	8	1	0.01

Table 1: Achieved objective function values and relative errors from ground truth after running IALM and ML-IALM on RPCA problems with synthetic data for a fixed time.

4.1 Synthetic Data

First we test the multilevel algorithms on synthetically generated data matrix $\mathbf{D} = \hat{\mathbf{L}} + \hat{\mathbf{S}} \in \mathbb{R}^{m \times n}$, where $\hat{\mathbf{L}}$ has a fixed low rank r and $\hat{\mathbf{S}}$ is η-sparse (i.e. has at most $\eta \cdot mn$ non-zero entries). We generate the synthetic data so that the singular values of the low rank component follow $1/k^2$, where k indicates the k-th largest singular value.

We run two sets of experiments. The first one compares the results achieved after running IALM and ML-IALM on RPCA problems for a fixed time. We run two pairs of experiments: with smaller and larger data, each with lower and higher rank of the low-rank component. The experiments are described in Table 1, with each row corresponding to one experimental setting. The first three columns describe the particular setting and the number of seconds dedicated to solve the problem. Then each triplet of columns gives the results achieved by IALM and ML-IALM algorithms correspondingly. The reported results are $f^\star$ - achieved objective value, error($\mathbf{S}^\star$) and error($\mathbf{L}^\star$) - achieved relative errors from corresponding ground truths. It is evident that ML-IALM accurately solves all four problems, while both objective values and relative errors from IALM are several times larger than those of ML-IALM. This effectively means that ML-IALM can solve large problems in reasonable time that may require impractically long times for IALM.

Then we synthetically generate similar data, but this time with partial observations. This setting is modelled as a CPCP problem and is then solved using FWT and ML-FWT methods. The experimental settings are given in Table 2. Here as well we have two pairs of problems: larger and smaller with larger and smaller ranks of the low-rank component. Here we run both problems until 10^{-3} tolerance as suggested in [20]. Here both algorithms achieve relatively small objective values and relative errors from the ground truth, with ML-FWT being slightly better, however ML-FWT takes significantly less time to do so. In fact, it is more than twice faster for the smaller rank settings.

4.2 Video Background Extraction

Now we test the algorithms on real surveillance videos. Assume we are given a surveillance video from a fixed camera and the task is to separate the constant background from moving objects. This problem can be modelled as a RPCA problem [3]. We first stack each frame of the video as a column vector creating a data matrix $\mathbf{D}$. Then, since

problem		FWT				ML-FWT			
dimensions	rank	sec	$f^\star$	err($\mathbf{L}^\star$)	err($\mathbf{S}^\star$)	sec	$f^\star$	err($\mathbf{L}^\star$)	err($\mathbf{S}^\star$)
10000×500	2	14.5	$8 \cdot 10^{-4}$	1.3	88	9	$2 \cdot 10^{-4}$	1.1	89.4
10000×5000	2	209	0.02	1.3	125	91	$7 \cdot 10^{-4}$	1	125
10000×500	5	14.4	$8 \cdot 10^{-4}$	1.3	89	12.5	$2 \cdot 10^{-4}$	1.1	89
10000×5000	5	203	0.002	1.3	125	90	$7 \cdot 10^{-4}$	1.1	125

Table 2: CPU times (in seconds), achieved objective function values and relative errors from ground truth after running FWT and ML-FWT on CPCP problems with synthetic data until 10^{-3} convergence error.

the fixed background remains (approximately) constant in each frame and the moving objects take a relatively small portion of each frame, they can respectively represent the low rank and sparse components of the RPCA decomposition. We tested all algorithms on several surveillance videos described below.

- **highway**: $48 \times 64 \times 400$; run 5 seconds
- **copy machine**: $48 \times 72 \times 3400$; run 50 seconds [2]
- **walk:** $240 \times 320 \times 794$; run 30 seconds [31]
- **gates:** $240 \times 320 \times 1895$; run 200 seconds [31]

First we test the IALM and ML-IALM methods. Here we run both methods for a fixed amount of time until a reasonably small error from ground truth has been achieved. The running times for each problem are indicated above. We then compare the results, which are reported in Figure 1. Each row represents a tested video. The first column contains sample frames from each corresponding video, then each of the following column triplets contains corresponding low rank and sparse components as returned from IALM and ML-IALM algorithms. Below each frame we also report the corresponding achieved rank and the feasibility gap (FG) i.e. $\|\mathbf{D} - \mathbf{L}^\star - \mathbf{S}^\star\|_F / \|\mathbf{D}\|_F$.

As the results indicate, both algorithms produce similar results for all videos, except the larger **copymachine** and **gates** examples, for which ML-IALM produces significantly clearer separation of background than IALM.

In order to further investigate the convergence properties of the ML-IALM algorithm compared to the standard IALM, we measure the relative error of the current iterates compared to the ground truth $(\mathbf{L}_0, \mathbf{S}_0)$ and FGs during the iterations of both algorithms through the same time interval. We report those relative errors against CPU time (seconds) and iteration numbers in Figure 2. The plots suggest that ML-IALM performs only slightly faster than IALM on the smaller **highway** example, however, as expected it is significantly faster on the larger **copy machine** problem. As we could anticipate from the theory, at each iteration ML-IALM achieves a very good approximation as measured by the reconstruction error, and since its iterations are significantly cheaper, it performs more iterations during the same time interval than IALM.

[2] `http://wordpress-jodoin.dmi.usherb.ca/dataset2012/`

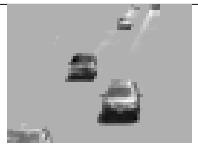
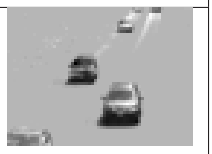
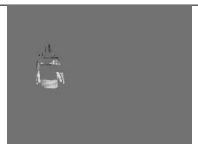
Original	Low Rank		Sparse	
	IALM	ML-IALM	IALM	ML-IALM
				
highway	rank =5	rank=3	FG = 0.0176	FG=0.01
				
copy machine	rank = 7	rank=4	FG = 0.0364	FG=0.0031
				
walk	rank = 2	rank=1	FG = 0.02	FG=0.0231
				
gates	rank = 3	rank=3	FG = 0.05	FG=0.04

Figure 1: Examples from solving video background extraction problems via IALM and ML-IALM methods. Both IALM and ML-IALM run for a fixed CPU seconds. Each row corresponds respectively to **highway** ($48 \times 64 \times 400$), **copy machine** ($48 \times 72 \times 3,400$), **walk** ($240 \times 320 \times 794$) and **gates** ($240 \times 320 \times 1,895$) videos from top to bottom. With each frame we also report the respective rank of the low rank component and the feasibility gap (FG): $\|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F / \|\mathbf{D}\|_F$.

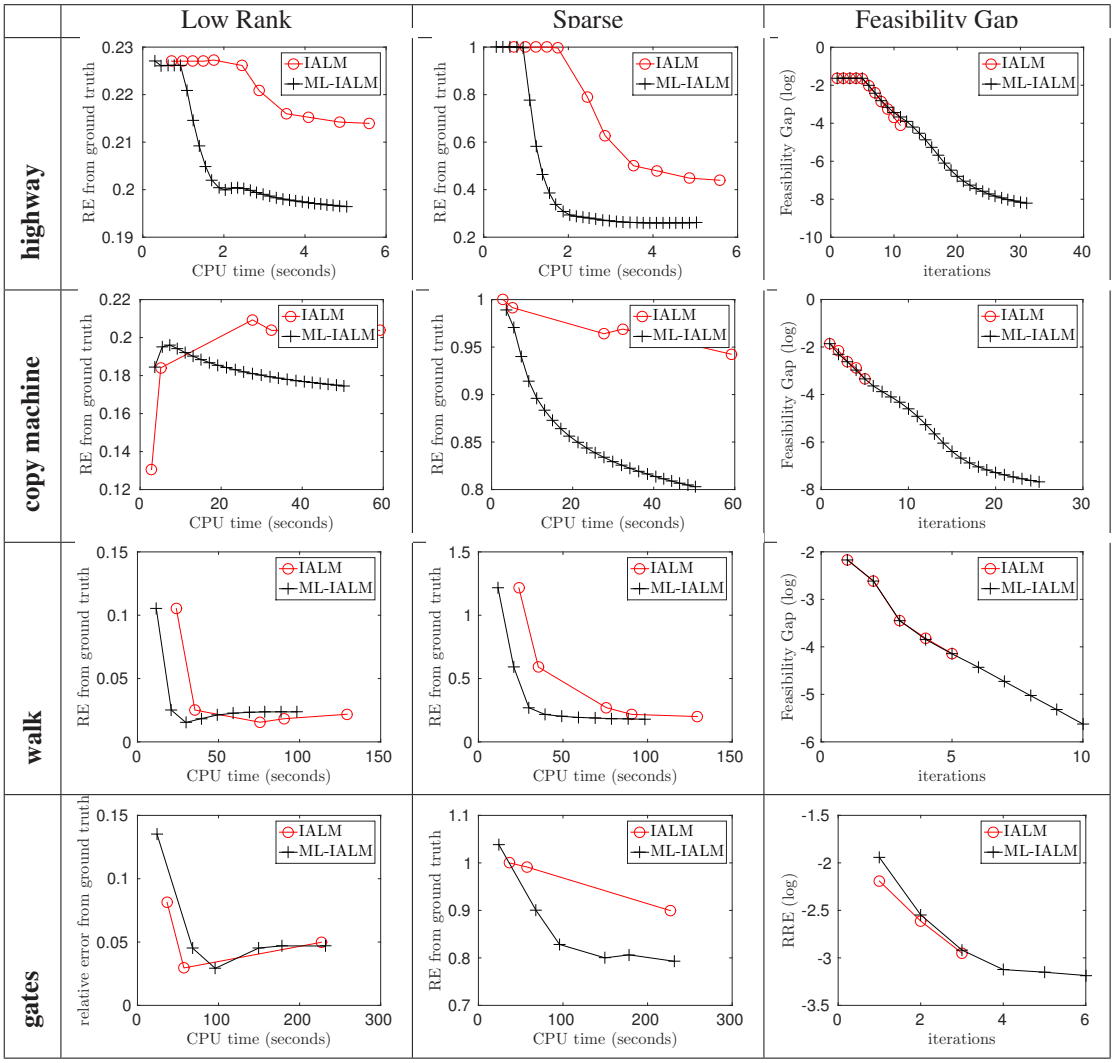


Figure 2: Comparing the relative errors during IALM and ML-IALM iterations. The first two columns give relative errors (RE) compared to the ground truth $(\mathbf{L}_0, \mathbf{S}_0)$, and the third column gives feasibility gaps (FG) during iterations. Each row corresponds respectively to **highway** $(48 \times 64 \times 400)$, **copy machine** $(48 \times 72 \times 3,400)$, **walk** $(240 \times 320 \times 794)$ and **gates** $(240 \times 320 \times 1,895)$ videos from top to bottom.

problem (dimensions)	FWT				ML-FWT			
	sec	$f^\star$	$\text{rank}(\mathbf{L}^\star)$	$\text{sp}(\mathbf{S}^\star)$	sec	$f^\star$	$\text{rank}(\mathbf{L}^\star)$	$\text{sp}(\mathbf{S}^\star)$
highway (3072×400)	14	0.001	39	0.22	10	0.001	36	0.22
hall (25344×200)	50	0.001	38	0.47	37	0.001	12	0.47
copym. (3456×3400)	373	0.001	104	0.11	160	0.001	26	0.17
mall (81920×300)	337	0.001	32	0.42	195	0.001	9	0.43
lobby (20480×1000)	487	0.001	111	0.05	385	0.001	31	0.06

Table 3: CPU time (in seconds), achieved objective value, rank and sparsity after solving the resulting PCP problem for noisy video background extraction up to tolerance 10^{-3} using the standard Frank-Wolfe Thresholding (FWT) and its multilevel variant ML-FWT.

In all experiments we used 4 levels of coarse models for all four problems. In this case as well, the multilevel variant largely outperforms the original algorithm. In fact, the larger the original problem, the bigger relative speed up can be achieved using the multilevel approach, since for larger n we can use deeper levels.

Next we test the performance of our ML-FWT algorithm against the standard FWT. In this case we will also add 75% random noise to the original video. Here we run both algorithms until convergence with 10^{-3} accuracy as suggested in [20]. Consequently, here we will report the running times and the achieved results (objective value, rank of the low-rank component and sparsity of the sparse component) of each algorithm in Table 3. As the numbers indicate both algorithms achieve very similar objective values and sparsity of the sparse component. However that ranks of the low rank components is better for ML-FWT. In fact for the largest problems FWT returns values with very large ranks and thus fails to solve the problem, while ML-FWT performs equally well on all problems. Furthermore, ML-FWT is much faster, especially on larger problems.

4.3 Shadow removal from facial images

Here we have a set of facial images from one individual under various illuminations and the task is to remove shadow/light noises from images. This problem can also be modelled as RPCA by stacking the facial images as column vectors and then putting them together to form the data matrix. Then since aligned frontal facial images span a low dimensional subspace, we can represent the clear images as the low-rank component of the data matrix and the shadow will become the sparse component.

We used images of individuals from the Yale B facial extended database [10]. It contains (96×84) dimensional facial images of 39 subjects taken under various poses and illuminations each, with total $2,414$ images. For this setting as well we ran the IALM and ML-IALM algorithms for a fixed 5 second and compare the returned results, which are reported in Figure 3. Here as well each row represents a particular problem setting (individual). The first column contains sample frames from each corresponding facial database, then each of the following four columns contains correspondingly low rank and sparse components as returned from IALM, and ML-IALM algorithms. With each image we also report the corresponding achieved rank of the low rank component

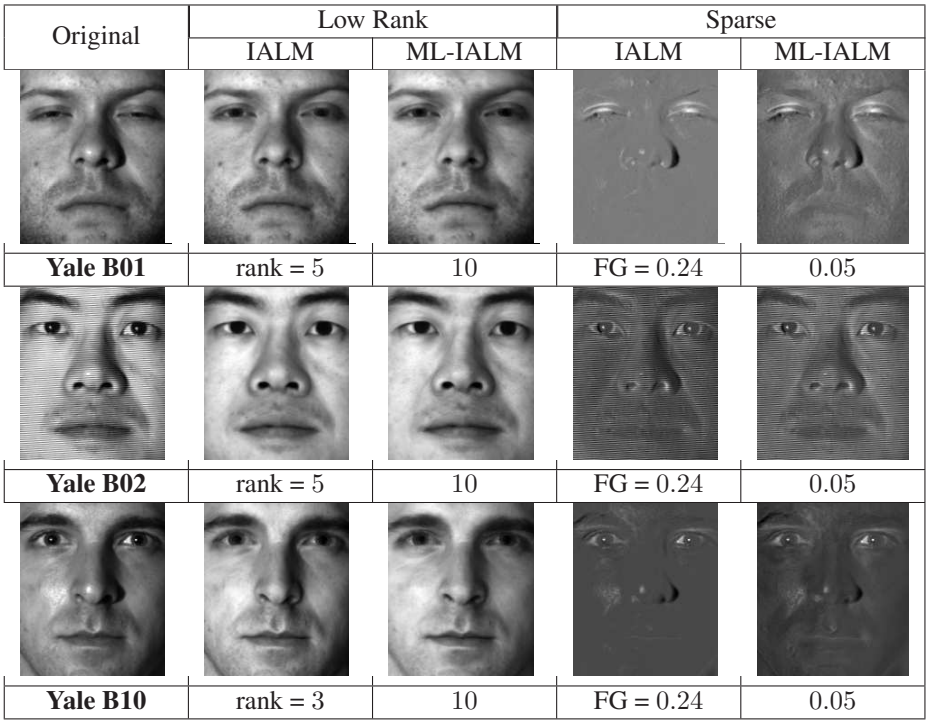


Figure 3: Examples from solving facial shadow removal problems via IALM and ML-IALM algorithms on cropped **Yale B** database ($96 \times 84 \times 2414$). We run both IALM and ML-IALM for fixed five seconds. With each image we also report the respective rank of the low rank component and the feasibility gap (FG): $\|\mathbf{D} - \mathbf{L} - \mathbf{S}\|_F / \|\mathbf{D}\|_F$.

and the feasibility gap.

A brief examination of the Table 3 reveals that ML-IALM produces much better separation for each subject, with a more accurate rank 10 of the low-rank component and a five times smaller feasibility gap.

For the shadow removal problem as well, we test the Frank-Wolfe methods on the noisy data with 75% contaminated entries. Both FWT and ML-FWT run until convergence with 10^{-3} tolerance and record CPU times (seconds) and the achieved rank of the low-rank component and sparsity of the sparse component. The results of all 35 subjects are reported in Table 4. In all experiments we used up to 4 levels of coarse models. Here in all experiments both methods achieved similar objective values and sparsity values, as expected. However ML-FWT is not only twice faster, but it also achieves a much better rank of the low-rank component. In fact, FWT fails to solve the problem, since the low-rank component is essentially full rank.

problem	FWT				ML-FWT			
	sec	$f^\star$	$\mathrm{rank}(\mathbf{L}^\star)$	$\mathrm{sp}(\mathbf{S}^\star)$	sec	$f^\star$	$\mathrm{rank}(\mathbf{L}^\star)$	$\mathrm{sp}(\mathbf{S}^\star)$
yaleB01	83	0.001	65	0.71	41	0.001	8	0.73
yaleB02	71	0.00098	65	0.71	39	0.00098	8	0.74
yaleB03	75	0.00099	65	0.71	40	0.00099	8	0.73
yaleB04	72	0.001	65	0.72	42	0.001	8	0.73
yaleB05	61	0.00096	65	0.7	33	0.00094	8	0.73
yaleB06	77	0.001	65	0.72	48	0.001	8	0.73
yaleB07	78	0.001	65	0.72	42	0.00099	8	0.73
yaleB08	75	0.00099	65	0.71	44	0.00098	8	0.73
yaleB09	84	0.001	65	0.71	48	0.00097	8	0.73
yaleB10	67	0.00099	65	0.72	45	0.001	8	0.73
yaleB11	75	0.00098	60	0.71	41	0.00097	7	0.73
yaleB12	79	0.00099	59	0.71	41	0.00097	7	0.73
yaleB13	60	0.00096	60	0.71	35	0.00097	7	0.74
yaleB15	85	0.001	63	0.71	43	0.00097	8	0.73
yaleB16	77	0.00098	62	0.7	47	0.001	7	0.73
yaleB17	66	0.00098	63	0.71	42	0.00099	8	0.73
yaleB18	85	0.001	63	0.71	45	0.00099	8	0.73
yaleB19	77	0.001	64	0.71	48	0.001	8	0.73
yaleB20	73	0.00099	64	0.7	43	0.001	8	0.73
yaleB21	75	0.00098	64	0.7	43	0.00099	8	0.73
yaleB22	88	0.00098	64	0.69	44	0.00093	8	0.72
yaleB23	63	0.00098	64	0.71	45	0.001	8	0.73
yaleB24	74	0.00099	64	0.71	49	0.001	8	0.73
yaleB25	75	0.00099	64	0.7	48	0.001	8	0.73
yaleB26	77	0.001	64	0.71	45	0.00099	8	0.73
yaleB27	61	0.00097	64	0.69	40	0.00096	8	0.72
yaleB28	61	0.00096	64	0.69	40	0.00097	8	0.73
yaleB29	71	0.00099	64	0.7	44	0.00099	8	0.73
yaleB30	72	0.00098	64	0.71	43	0.00099	8	0.73
yaleB31	78	0.001	64	0.71	48	0.001	8	0.73
yaleB32	60	0.00096	64	0.7	37	0.00097	8	0.72
yaleB33	79	0.00099	64	0.7	44	0.00099	8	0.73
yaleB34	78	0.00099	64	0.7	37	0.00097	8	0.73
yaleB35	75	0.00099	64	0.7	45	0.00099	8	0.73
yaleB36	72	0.00099	64	0.71	42	0.00098	8	0.73

Table 4: CPU times (in seconds) after solving shadow removal problems up to a fixed tolerance using the standard Frank-Wolfe Thresholding algorithm and its multilevel variant. For all experiments we used 2 levels for the multilevel algorithm.

5 Conclusion

In this paper, we presented two multilevel algorithms for solving problems modelled as robust principle component analysis or compressive principle component pursuit optimisation problems. The first algorithm is a multilevel variant of the well-known inexact augmented Lagrange method (or more generally, ADMM), called ML-IALM. We proved that ML-IALM converges to an approximate solution, with approximation error being small for many computer vision problems, including those studied here. To the best of our knowledge this is the first time when an ADMM with approximate steps was proven to converge. Our second algorithm is a multilevel variant of the well known Frank-Wolfe method modified to be most efficient for CPCP problems. We showed that this multilevel algorithm also converges to the solution of the CPCP problem with the same rate as its standard counterpart, while having much lower per iteration complexity. We tested both methods on various synthetic and real life problems. The results clearly show that the multilevel algorithms are not only several times faster (especially on larger problems), but also can often solve problems that their standard counterparts cannot.

References

[1] Zeyuan Allen-Zhu and Yuanzhi Li. Even faster svd decomposition yet without agonizing pain. In *Advances in Neural Information Processing Systems*, pages 974–982, 2016.

[2] Roland Angst, Christopher Zach, and Marc Pollefeys. The generalized trace-norm and its application to structure-from-motion problems. In *2011 International Conference on Computer Vision*, pages 2502–2509. IEEE, 2011.

[3] Thierry Bouwmans, Andrews Sobral, Sajid Javed, Soon Ki Jung, and El-Hadi Zahzah. Decomposition into low-rank plus additive matrices for background/foreground separation: A review for a comparative evaluation with a large-scale dataset. *Computer Science Review*, 2016.

[4] William L Briggs, Steve F McCormick, et al. *A multigrid tutorial*. Siam, 2000.

[5] Juan S Campos and Panos Parpas. A multigrid approach to sdp relaxations of sparse polynomial optimization problems. *SIAM Journal on Optimization*, 28(1):1–29, 2018.

[6] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.

[7] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

[8] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices ii: Computing a low-rank approximation to a matrix. *SIAM Journal on computing*, 36(1):158–183, 2006.

[9] Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[10] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[11] Chin Pang Ho and Panos Parpas. Multilevel optimization methods: Convergence and problem structure. 2016.

[12] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417, 1933.

[13] Vahan Hovhannisyan, Panos Parpas, and Stefanos Zafeiriou. Magma: Multilevel accelerated gradient mirror descent algorithm for large-scale convex composite minimization. *SIAM Journal on Imaging Sciences*, 9(4):1829–1857, 2016.

[14] Peter J Huber. *Robust statistics*. Springer, 2011.

[15] Ashkan Javaherian and Sean Holman. A multi-grid iterative method for photoacoustic tomography. *IEEE transactions on medical imaging*, 36(3):696–706, 2017.

[16] Evgeny S Levitin and Boris T Polyak. Constrained minimization methods. *USSR Computational mathematics and mathematical physics*, 6(5):1–50, 1966.

[17] Zhouchen Lin, Minming Chen, and Yi Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

[18] Guangcan Liu and Shuicheng Yan. Active subspace: Toward scalable low-rank learning. *Neural computation*, 24(12):3371–3394, 2012.

[19] Risheng Liu, Zhouchen Lin, Zhixun Su, and Junbin Gao. Linear time principal component pursuit and its extensions using l1 filtering. *Neurocomputing*, 142:529–541, 2014.

[20] Cun Mu, Yuqian Zhang, John Wright, and Donald Goldfarb. Scalable robust matrix recovery: Frank–wolfe meets proximal methods. *SIAM Journal on Scientific Computing*, 38(5):A3291–A3317, 2016.

[21] Cameron Musco and Christopher Musco. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Advances in Neural Information Processing Systems*, pages 1396–1404, 2015.

[22] Stephen Nash. A multigrid approach to discretized optimization problems. *Optimization Methods and Software*, 14(1-2):99–116, 2000.

[23] Balas Kausik Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[24] Praneeth Netrapalli, UN Niranjan, Sujay Sanghavi, Animashree Anandkumar, and Prateek Jain. Non-convex robust pca. In *Advances in Neural Information Processing Systems*, pages 1107–1115, 2014.

[25] Tae-Hyun Oh, Yasuyuki Matsushita, Yu-Wing Tai, and In So Kweon. Fast randomized singular value thresholding for nuclear norm minimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4484–4493, 2015.

[26] Neal Parikh and Stephen Boyd. Proximal algorithms. *Foundations and Trends in optimization*, 1(3):123–231, 2013.

[27] Panos Parpas. A multilevel proximal gradient algorithm for a class of composite optimization problems. *SIAM Journal on Scientific Computing*, 39(5):S681–S701, 2017.

[28] Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. RASL: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2233–2246, 2012.

[29] Christos Sagonas, Yannis Panagakis, Stefanos Zafeiriou, and Maja Pantic. Raps: Robust and efficient automatic construction of person-specific deformable models. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1789–1796. IEEE, 2014.

[30] Christos Sagonas, Yannis Panagakis, Stefanos Zafeiriou, and Maja Pantic. Robust statistical frontalization of human and animal faces. *International Journal of Computer Vision*, pages 1–22, 2016.

[31] Antoine Vacavant, Thierry Chateau, Alexis Wilhelm, and Laurent Lequièvre. A benchmark dataset for outdoor foreground/background extraction. In *Asian Conference on Computer Vision*, pages 291–300. Springer, 2012.

[32] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996.

[33] Bo-Ying Wang and Bo-Yan Xi. Some inequalities for singular values of matrix products. *Linear algebra and its applications*, 264:109–115, 1997.

[34] John Wright, Arvind Ganesh, Kerui Min, and Yi Ma. Compressive principal component pursuit. *Information and Inference*, 2(1):32–68, 2013.

[35] Junfeng Yang and Xiaoming Yuan. Linearized augmented lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation*, 82(281):301–329, 2013.

[36] Zhengdong Zhang, Arvind Ganesh, Xiao Liang, and Yi Ma. Tilt: Transform invariant low-rank textures. *International Journal of Computer Vision*, 99(1):1–24, 2012.