Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 18, No. 2, pp. 265–280, 2022

AN ACCELERATED GRADIENT METHOD FOR NONCONVEX SPARSE SUBSPACE CLUSTERING PROBLEM*

Hongwu Li, Haibin Zhang and Yunhai Xiao†

Abstract: The sparse subspace clustering problem is to group a set of data into their underlying subspaces and correct the underlying noise simultaneously. It was shown in the recent literature that, the clustering task can be characterized as a block diagonal matrix regularized nonconvex minimization problem. However, this problem is not easy to solve because it contains a nonconvex bilinear function. The earliest method named block diagonal regularization (BDR) only solved a penalized model, but not the original problem itself. The recently algorithm named accelerated block coordinated gradient descent (ABCGD) can solve the original problem efficiently, but its convergence is not given. In this paper, we attempt to use an accelerated gradient method (AGM), and establish its convergence in the sense of converging to a critical point with a certain stepsize policy. We show that closed-form solutions are enjoyed for each subproblem by taking full use of the constraints' structure so that the algorithm is easily implementable. Finally, we do numerical experiments by the using of two real datasets. The numerical results illustrate that the proposed algorithm AGM performs better than BDR and ABCGD evidently.

Key words: *sparse subspace clustering, nonconvex nonsmooth optimization, accelerated gradient method, Hopkins* 155 *real datasets, Extended Yale B database*

Mathematics Subject Classification: *90C26, 90C90*

---

1 Introduction

Many applications in various areas can be captured by finding and exploiting low-dimensional structure in high-dimensional data. Let $\mathcal{S} := \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_l\}$ be a set of independent subspaces and $X := [X_1, X_2, \ldots, X_l]$ be a given sample drawn from $\mathcal{S}$ with the relation $X_i \in \mathcal{S}_i$. Assume that each sample X_i is with size $m \times n_i$ and rank d_i, then it can be represented as a linear combination by itself, i.e., $X_i = X_i Z_i$ where $Z_i \in \mathbb{R}^{n_i \times n_i}$ is expected not to be an identity. Moreover, let $n = \sum_i n_i$, $d = \sum_i d_i$, and $Z = \mathrm{Diag}(Z_1, \ldots, Z_l)$, i.e., a block diagonal matrix with its diagonal block Z_i, then we have

$$X = XZ, \tag{1.1}$$

where $X \in \mathbb{R}^{m\times n}$ is with rank d, and $Z \in \mathbb{R}^{n\times n}$ is a called representation coefficient of X.

Under the assumption of $m < n$, i.e., the dimension of X is less than the number of samples, then there exist infinite number of coefficients Z that satisfing (1.1). Noting that

*This work is supported by the National Natural Science Foundation of China (No. 11971149, 11771003).
†Corresponding author

all the off-diagonal blocks of Z are zeros, and all the diagonal elements are zeros, hence, it is reasonable to find a sparsest one among all the solutions of (1.1), that is

$$\min_{Z\in\mathbb{R}^{n\times n}} \left\{\|Z\|_1, \quad \text{s.t. } X = XZ,\ Z_{jj} = 0,\ j = 1, \dots n\right\}, \tag{1.2}$$

where $\|\cdot\|_1$ is a so-called extension ℓ_1-norm of matrix. It was shown by Elhamifar & Vidal [3] that, if each samples $\{X_i\}_1^l$ are noiseless and drawn from an independent subspaces, then the optimal solution of (1.2) is block diagonal. On other hand, each sample Z_i, and even Z, actually has low rank structure if the d_i is assumed to be $d_i \ll \min\{m, n_i\}$. In such case, it turns to seek the lowest-rank representation of the data samples, that is

$$\min_{Z\in\mathbb{R}^{n\times n}} \left\{\|Z\|_*, \quad \text{s.t. } X = XZ\right\}, \tag{1.3}$$

where $\|\cdot\|_*$ is a nuclear norm which serves as a convex surrogate of a rank function. As shown by Liu et al. [8], the optimal solution to problem (1.3) is also block diagonal under the assumption of noiselessness and subspaces independence. The method NNLRS of Zhuang et al. [16] is to seek a non-negative low-rank and sparse coefficient via solving the following optimization problem

$$\min_{Z\in\mathbb{R}^{n\times n}} \left\{\|Z\|_* + \gamma\|Z\|_1, \quad \text{s.t. } X = XZ,\ Z \geq 0\right\},$$

where $\gamma > 0$ is a weighting parameter to control the balance between low rank and sparsity, and $Z \geq 0$ means that all the entries of Z are nonnegative.

However, in practice, a fraction of the data vectors may be grossly corrupted by noise, and hence the diagonal structure of the optimal solution might be violated. In this case, it is shown by Liu et al. [7] that it is appropriate to find a low-rank representation (LRR) coefficient via solving

$$\min_{Z\in\mathbb{R}^{n\times n}, E\in\mathbb{R}^{m\times n}} \left\{\|Z\|_* + \lambda\|E\|_{\ell_2/\ell_1}, \quad \text{s.t. } X = XZ + E\right\}, \tag{1.4}$$

where $\lambda > 0$ is balance between low rank and error of residuals, and $\|\cdot\|_{\ell_2/\ell_1}$ is defined as the sum of the ℓ_2-norm of each column of matrix. The multi-subspace representation (MSR) [10] combines the idea of NNLRS and LRR, which is formulated as

$$\min_{Z\in\mathbb{R}^{n\times n}} \left\{\|Z\|_* + \gamma\|Z\|_1 + \lambda\|X - XZ\|_{\ell_2/\ell_1}\right\}. \tag{1.5}$$

It should be noted that the optimal solutions obtained by both approaches obey block diagonal structures even when the data is heavily corrupted by noise.

The independent subspaces assumption is essential to guarantee the block diagonal property, but it can be removed by the using of subspace segmentation with quadratic programming (SSQP) [4] in which a regularization $\|Z^\top Z\|_1$ instead of $\|Z\|_1$ is used. Besides, there is another exciting progress to encourage a nonnegative symmetric matrix to be block diagonal. Let "$\mathbf{1}$" be a vector with all entries are one, and define a Laplacian matrix of a given symmetric matrix Z as $L_Z := \mathrm{Diag}\,(Z\mathbf{1}) - Z$. The block diagonal representation (BDR) method for subspace clustering of Lu et al. [9] is formulated as the following optimization

$$\begin{aligned} \min_{Z} \quad & \tfrac{1}{2}\|X - XZ\|_F^2 + \gamma\|L_Z\|_{[l]} \\ \text{s.t.} \quad & Z \geq 0,\ Z = Z^\top,\ Z_{jj} = 0,\ j = 1, \dots, n, \end{aligned} \tag{1.6}$$

where $\|\cdot\|_F$ is a Frobenius norm of a matrix, and $\|Z\|_{[l]}$ is a block diagonal regularization defined as the sum of the smallest l eigenvalues of L_Z. The key challenge for solving (1.6) lies in the regularization term $\|\cdot\|_{[l]}$. To address this issue, Lu et al. [9] approximated $\|L_Z\|_{[l]}$ as convex programming to get the following model

$$\begin{aligned} \min_{Z,W} \quad & \tfrac{1}{2}\|X - XZ\|_F^2 + \gamma\big\langle \mathrm{Diag}(Z\mathbf{1}) - Z, W\big\rangle \\ \text{s.t.} \quad & Z \geq 0,\ Z = Z^\top,\ Z_{jj} = 0,\ j = 1, \ldots, n, \\ & I \succeq W \succeq 0,\ \mathrm{Tr}(W) = l, \end{aligned} \tag{1.7}$$

where $\langle\cdot,\cdot\rangle$ and $\mathrm{Tr}(\cdot)$ are matrix inner product and trace respectively, and $W \succeq 0$ means that W is positive semi-definite. The BDR method of Lu et al. [9] used a block coordinate descent method [15] to solve a penalty variant of (1.7) and its effectiveness and high-efficiency is experimentally demonstrated by the using of several real dataset. Subsequently, the performance of BDR is greatly improved by Kong et al. [6] where a Nesterov's accelerated technique [11] is employed, but the convergence of the accelerated variant is not given. We must emphasize that the algorithms [9] and [6] are both concerned on an approximated model, but not the original (1.7) itself. The accelerated block coordinated gradient descent (ABCGD) [6] has the ability to solve (1.7), but its convergence is still unknown. Therefore, it is necessary to develop an effective algorithm with convergence guarantee to solve the problem (1.7) directly.

It is much difficult and challenging to solve (1.7) due to the nonconvex bilinear term "$\langle\mathrm{Diag}(Z\mathbf{1}) - Z, W\big\rangle$" as well as the constraints on the variables. To address this issue, we focus on the using of the novel accelerated gradient method of Ghadimi & Lan et al.[5] which can be reviewed as a generalized variant of the well-known Nesterov's accelerated gradient method [1, 11] to solve nonconvex and possibly stochastic optimization problems. It has been known that the attractive feature of using this method is that an optimal rate of convergence is exhibited if the composite problem is convex, and the best known rate of convergence is improved if the problem is nonconvex. Nevertheless, the convergence of the algorithm depends on a pair of stepsize which is related with the Lipschitz constant of the gradient to the smooth functions. To tackle this difficulty, we numerically estimate the Lipschitz constant by the using of the structure of the given data. Finally, we do a series of numerical experiments on some real data which demonstrates that the proposed algorithm is highly more efficient than BDR and ABCGD.

The remaining parts of this paper are organized as follows. In Section 2, we quickly review some key ingredients needed for our subsequent developments. In Section 3, we propose an AGM to solve the model (1.7) followed by a convergence theorem. In Section 4, we present some numerical experiments using some real data to show the efficiency of our algorithm. Finally, we conclude our paper with some remarks in Section 5.

2 Preliminaries

In this section, we summarize some basic concepts in convex analysis [12] and quickly review the accelerated gradient method of Ghadimi & Lan [5] used to the subsequent developments. Let $\mathcal{X}$ be a finite-dimensional real Euclidean space with an inner product and associated norm denoted by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|_2$, respectively. For any $z \in \mathcal{E}$, the metric projection of z onto $\mathcal{C}$ denoted by $\Pi_{\mathcal{C}}(z)$ is the optimal solution of the minimization problem $\min_y\{\|y - z\| \mid y \in \mathcal{C}\}$. Let $g : \mathcal{X} \to (-\infty, +\infty]$ be a closed proper convex function. The subdifferential of $g(\cdot)$ is a convex set defined as $\partial g(x) = \{x^* \mid g(z) \geq g(x) + \langle x^*, z - x\rangle,\ \forall z \in \mathcal{X}\}$. Obviously, $\partial g(x)$ is a closed convex set when it is not empty [12]. A necessary but not sufficient condition for $x^* \in \mathbb{R}^n$ to be a minimizer of function $g(\cdot)$ is $0 \in \partial g(x^*)$, where x^* is called a critical point.

The Moreau-Yosida regularization of g at $x \in \mathcal{X}$ with positive scalar $\eta > 0$ is defined by

$$\varphi_g^\eta(x) := \min_{y \in \mathcal{X}} \Big\{ g(y) + \frac{1}{2\eta}\|y - x\|^2 \Big\}. \tag{2.1}$$

For any $x \in \mathcal{X}$, problem (2.1) has an unique optimal solution, which is known as the proximal mapping of x associated with g and simply denoted by $\mathcal{P}_g^\eta(x)$, i.e.,

$$\mathcal{P}_g^\eta(x) := \arg\min_{y \in \mathcal{X}} \Big\{ g(y) + \frac{1}{2\eta}\|y - x\|^2 \Big\}. \tag{2.2}$$

We now briefly review the accelerated gradient method for a class of the following composite problem

$$\min_x \ f(x) + h(x) + g(x), \tag{2.3}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function (possibly nonconvex), $h : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable convex function, and $g : \mathbb{R}^n \to (-\infty, +\infty]$ is a proper closed convex function. Let $\{\alpha_k\}$ be a positive sequence such that $\alpha_k \in (0, 1)$ and $\alpha_0 = 1$. Starting from x^0 and x^0_{ag}, the accelerated gradient method generates an iterate sequence $\{(x^k_{md}, x^k_{ag}, x^k)\}$ via the iterative scheme:

$$\begin{cases} x^k_{md} = (1 - \alpha_k)x^{k-1}_{ag} + \alpha_k x^{k-1}, \\ x^k_{ag} = \mathcal{P}_g^{\beta_k}\Big(x^k_{md} - \beta_k[\nabla f(x^k_{md}) + \nabla h(x^k_{md})]\Big), \\ x^k = \mathcal{P}_g^{\rho_k}\Big(x^{k-1} - \rho_k[\nabla f(x^k_{md}) + \nabla h(x^k_{md})]\Big), \end{cases} \tag{2.4}$$

where $\beta_k > 0$ and $\rho_k > 0$ are the suitable stepsizes. Note that, if $\rho_k = \beta_k$, then we have $x^{k-1}_{ag} = x^{k-1}$ and $x^k_{md} = x^{k-1}$, in this case, this accelerated gradient method reduces to the tranditional proximal gradient method, and then it also reduces to the Nesterov's accelerated gradient method [11] if α_k is chosen properly. For more details on this method, one may refer to [5].

3 Accelerated Gradient Method

3.1 Algorithm's construction

For convenience, we define

$$\mathcal{C}_1 := \big\{ W \in \mathbb{R}^{n\times n} \mid I \succeq W \succeq 0,\ \mathrm{Tr}(W) = l \big\},$$
$$\mathcal{C}_2 := \big\{ Z \in \mathbb{R}^{n\times n} \mid Z \geq 0,\ Z = Z^\top,\ Z_{jj} = 0,\ j = 1, \dots n \big\},$$

which are all convex set. Therefore, the problem (1.7) can be represented equivalently as follows:

$$\min_{W,Z} \ \Big\{ \delta_{\mathcal{C}_1}(W) + \delta_{\mathcal{C}_2}(Z) + \frac{1}{2}\|X - XZ\|^2 + \gamma\big\langle \mathrm{Diag}(Z\mathbf{1}) - Z, W \big\rangle \Big\}, \tag{3.1}$$

where $\delta_{\mathcal{C}_i}(\cdot)$ represents an indicator function over $\mathcal{C}_i$. Obviously, the objective function is nonconvex because W and Z are coupled together in the last term.

For the sake of simplicity, we denote $Y := [Z; W] \in \mathbb{R}^{2n\times n}$ and $\mathcal{C} := \mathcal{C}_1 \times \mathcal{C}_2$. Using these notations, the nonsmooth term $\delta_{\mathcal{C}_1}(W) + \delta_{\mathcal{C}_2}(Z)$ can be rewritten equivalently as $\delta_{\mathcal{C}}(Y)$. Besides, we define a linear operator $\mathcal{A} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ such that $\mathcal{A}(Z) = \mathrm{Diag}(Z\mathbf{1}) - Z$, then the coupled term can be reformulated as $\big\langle \mathcal{A}(Z), W\big\rangle = \big\langle Q_1Y, Q_2Y\big\rangle = \big\langle Q_2^\top Q_1Y, Y\big\rangle = \big\langle QY, Y\big\rangle$ with

$$Q_1 = \begin{pmatrix} \mathcal{A} & 0 \\ 0 & 0 \end{pmatrix},\ Q_2 = \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix},\ \text{and } Q = \begin{pmatrix} 0 & 0 \\ \mathcal{A} & 0 \end{pmatrix}. \tag{3.2}$$

Moreover, denote $\Theta := [X; 0] \in \mathbb{R}^{m\times 2n}$, then the problem (3.1) is transformed into

$$\min_{Y\in\mathbb{R}^{2n\times n}} \Big\{F(Y) := \delta_{\mathcal{C}}(Y) + \frac{1}{2}\|X - \Theta Y\|_F^2 + \gamma\big\langle QY, Y\big\rangle\Big\}. \tag{3.3}$$

Clearly, $F(\cdot)$ is nonsmooth and nonconvex because of the indefiniteness of Q. The problem (3.3) can also be rewritten equivalently as

$$\min_{Y\in\mathbb{R}^{2n\times n}} \ F(Y) := \psi(Y) + \delta_{\mathcal{C}}(Y), \tag{3.4}$$

where $\psi(Y) := h(Y) + f(Y)$ with

$$h(Y) := \frac{1}{2}\|X - \Theta Y\|_F^2, \quad \text{and} \quad f(Y) := \gamma\big\langle QY, Y\big\rangle. \tag{3.5}$$

Obviously, f is a nonconvex continuously differentiable function. We also assume that $\nabla f(\cdot)$ and $\nabla h(\cdot)$ are Lipschitz continuous with the modulus L_f and L_h, respectively, and hence $\nabla\psi(\cdot)$ satisfies Lipschitz continuous with modulus $L_\psi = L_f + L_h$.

We now turn our attention to the solving of problem (3.4). Because $\delta_{\mathcal{C}}(\cdot)$ is an indicator function on convex compact set $\mathcal{C}$, from [5, Lemma 2], it is easy to see that there exists a positive constant M such that $\|\mathcal{P}^{\eta}_{\delta_{\mathcal{C}}}(Y - \eta\nabla\psi(Y))\| \le M$ for any given $\eta > 0$ and $Y \in \mathbb{R}^{2n\times n}$, where $\mathcal{P}^{\eta}_{\delta_{\mathcal{C}}}(\cdot)$ is a proximal mapping. From the definition of $\mathcal{P}^{\eta}_{\delta_{\mathcal{C}}}(Y - \eta\nabla\psi(Y))$, we define an important quantity as follows

$$\mathcal{G}(Y, \nabla\psi(Y), \eta) \triangleq \frac{1}{\eta}\Big(Y - \mathcal{P}^{\eta}_{\delta_{\mathcal{C}}}(Y - \eta\nabla\psi(Y))\Big). \tag{3.6}$$

From [5, Lemma 3], we know that as the size of $\mathcal{G}(Y, \nabla\psi(Y), \eta)$ vanishes, the $\mathcal{P}^{\eta}_{\delta_{\mathcal{C}}}(Y - \eta\nabla\psi(Y))$ approaches to a critical point Y^* of problem (3.4).

We now focus on the practical implementation for (3.3), or equivalently (3.4). The employed algorithm here is based on the AGM of Ghadimi & Lan [5], which starts from the initial point (Y^0_{ag}, Y^0), and generates an iterate sequence $\{(Y^k_{md}, Y^k_{ag}, Y^k)\}$ via the iterative scheme:

$$\begin{cases} Y^k_{md} = (1-\alpha_k)Y^{k-1}_{ag} + \alpha_k Y^{k-1}, \\ Y^k = \mathcal{P}^{\rho_k}_{\delta_{\mathcal{C}}}\big(Y^{k-1} - \rho_k\nabla\psi(Y^k_{md})\big), \\ Y^k_{ag} = \mathcal{P}^{\beta_k}_{\delta_{\mathcal{C}}}\big(Y^k_{md} - \beta_k\nabla\psi(Y^k_{md})\big), \end{cases} \tag{3.7}$$

where $\alpha_k \in (0, 1)$ with $\alpha_1 = 1$, and $\beta_k > 0$, $\rho_k > 0$ are the suitable stepsizes. Note that, if $\rho_k = \beta_k$, then we have $Y^{k-1}_{ag} = Y^{k-1}$ and $Y^k_{md} = Y^{k-1}$, in this case, this accelerated gradient method reduces to the simplest proximal gradient method. From [5, Corollary 2], we can get the main convergence properties of (3.7) as follows provided that the parameters $\{\alpha_k\}$, $\{\beta_k\}$, and $\{\rho_k\}$ are chosen properly. For more details on the choices of these parameters $\{\alpha_k\}$, $\{\beta_k\}$, and $\{\rho_k\}$, one can refer to [5, Lemma 1, Corollary 1].

$\boxed{3.2}$ Subproblems' solving

Notice that $\delta_{\mathcal{C}}(\cdot)$ is very simple so that the subproblems involved in (3.7) are easily computable. From the definition of proximal mapping, we see that the subproblem involved in (3.7) are with the form

$$Y^k = \underset{Y\in\mathbb{R}^{2n\times n}}{\arg\min}\ \delta_{\mathcal{C}}(Y) + \frac{1}{2\rho_k}\left\|Y - \left(Y^{k-1} - \rho_k\nabla\psi(Y^k_{md})\right)\right\|_F^2, \tag{3.8}$$

$$Y^k_{ag} = \underset{Y\in\mathbb{R}^{2n\times n}}{\arg\min}\ \delta_{\mathcal{C}}(Y) + \frac{1}{2\beta_k}\left\|Y - \left(Y^k_{md} - \beta_k\nabla\psi(Y^k_{md})\right)\right\|_F^2. \tag{3.9}$$

Recalling that $Y = [Z; W]$. We notice that the Y-subproblems involved in (3.8) and (3.9) are easily implementable in the sense that it can be partitioned into a couple of lower-dimensional subproblems regarding to Z and W, that is,

$$\begin{aligned}[Z^k; W^k] = \underset{Z,W}{\arg\min}\Big\{\delta_{\mathcal{C}_1}(W) + \delta_{\mathcal{C}_2}(Z) + \frac{1}{2\rho_k}\left\|Z - \left(Z^{k-1} - \rho_k\nabla\psi_Z(Z^k_{md})\right)\right\|_F^2 \\ + \frac{1}{2\rho_k}\left\|W - \left(W^{k-1} - \rho_k\nabla\psi_W(W^k_{md})\right)\right\|_F^2\Big\}.\end{aligned}$$

Noting that Z and W are independent with each other, they can be computed individually, that is,

$$Z^k = \underset{Z\in\mathbb{R}^{n\times n}}{\arg\min}\Big\{\delta_{\mathcal{C}_2}(Z) + \frac{1}{2\rho_k}\|Z - M_{2k}\|_F^2\Big\},\quad W^k = \underset{W\in\mathbb{R}^{n\times n}}{\arg\min}\Big\{\delta_{\mathcal{C}_1}(W) + \frac{1}{2\rho_k}\|W - N_{2k}\|_F^2\Big\}, \tag{3.10}$$

where $M_{2k} := Z^{k-1} - \rho_k\nabla\psi_Z(Z^k_{md})$ and $N_{2k} := W^{k-1} - \rho_k\nabla\psi_W(W^k_{md})$ with

$$\nabla\psi_Z(Z^k_{md}) = X^\top(XZ^k_{md} - X) + \gamma\mathcal{A}^*W^k_{md}, \quad \text{and} \quad \nabla\psi_W(W^k_{md}) = \gamma\mathcal{A}(Z^k_{md}). \tag{3.11}$$

We now show that computing the projection onto $\mathcal{C}_1$ or $\mathcal{C}_2$ is really a trivial task. On the one hand, denote $\tilde{N}_{2k} := (N_{2k} + N_{2k}^\top)/2$, and there exists an orthogonal matrix V^k such that

$$\tilde{N}_{2k} = V^k\Sigma_k(V^k)^\top, \tag{3.12}$$

where Σ_k is a diagonal matrix whose diagonal entries consist of the eigenvalue of $\tilde{N}_{2k}$ in nondecreasing order, and V^k is an orthogonal matrix whose columns are the corresponding eigenvectors. Let $V^k_{\mathcal{L}}$ be the submatrices associated with the l smallest eigenvalues of $\tilde{N}_{2k}$. Then the optimal solution W^k can be explicitly described as

$$W^k = V^k_{\mathcal{L}}{V^k_{\mathcal{L}}}^\top.$$

On the other hand, it is from [9, Proposition 7] that, the new Z^k has an analytical solution with form

$$Z^k = \max\Big\{\frac{\hat{M}_{2k} + \hat{M}_{2k}^\top}{2}, 0\Big\},$$

where $\hat{M}_{2k} = M_{2k} - \mathrm{Diag}(\mathrm{diag}(M_{2k}))$. In a similar way, the variables W^k_{ag} and Z^k_{ag} can be obtained by

$$W^k_{ag} = \underset{W\in\mathbb{R}^{n\times n}}{\arg\min}\Big\{\delta_{\mathcal{C}_1}(W) + \frac{1}{2\beta_k}\|W - N^{ag}_{2k}\|_F^2\Big\},\ Z^k_{ag} = \underset{Z\in\mathbb{R}^{n\times n}}{\arg\min}\Big\{\delta_{\mathcal{C}_2}(Z) + \frac{1}{2\beta_k}\|Z - M^{ag}_{2k}\|_F^2\Big\}, \tag{3.13}$$

where $M_{2k}^{ag} := Z_{md}^k - \beta_k \nabla\psi_Z(Z_{md}^k)$ and $N_{2k}^{ag} := W_{md}^k - \beta_k \nabla\psi_W(W_{md}^k)$. Furthermore, denote $\tilde{N}_{2k}^{ag} := (N_{2k}^{ag} + (N_{2k}^{ag})^\top)/2$, and do eigenvalue decomposition as

$$\tilde{N}_{2k}^{ag} = U^k \Sigma_k (U^k)^\top, \tag{3.14}$$

Besides, denote $U_{\mathcal{L}}^k$ be the submatrices associated with the last l columns of U^k, then we get

$$W_{ag}^k = U_{\mathcal{L}}^k {U_{\mathcal{L}}^k}^\top.$$

and

$$Z_{ag}^k = \max\Big\{\frac{\hat{M}_{2k}^{ag} + (\hat{M}_{2k}^{ag})^\top}{2}, 0\Big\},$$

where $\hat{M}_{2k}^{ag} = M_{2k}^{ag} - \mathrm{Diag}(\mathrm{diag}(M_{2k}^{ag}))$ with $M_{2k}^{ag} := Z_{md}^k - \beta_k \nabla\psi_Z(Z_{md}^k)$.

In light of the above analysis and definition of $\nabla\psi$ in (3.11), we are ready to state the iterative framework of the accelerated gradient method (AGM) to solve the problem (3.4).

Algorithm: AGM

Step 0. Input a Lipschitz constant $L_\psi > 0$; Initialize: $Y_{md}^0 = Y_{ag}^0 = Y^0 \in \mathbb{R}^{2n\times n}$. For $k = 1, 2, \ldots$, do the following operations iteratively:

Step 1. Compute

$$\alpha_k = \frac{2}{k+1}, \quad \beta_k = \frac{1}{2L_\psi}, \quad \rho_k = \frac{k\beta_k}{2}.$$

Step 2. Compute $W_{md}^k = (1-\alpha_k)W_{ag}^{k-1} + \alpha_k W^{k-1}$ and $Z_{md}^k = (1-\alpha_k)Z_{ag}^{k-1} + \alpha_k Z^{k-1}$.

Step 3. Compute W^k and Z^k according to

- Compute $N_{2k} = W^{k-1} - \rho_k \nabla\psi_W(W_{md}^k)$ and $M_{2k} = Z^{k-1} - \rho_k \nabla\psi_Z(Z_{md}^k)$;
- Set $\tilde{N}_{2k} = (N_{2k} + N_{2k}^\top)/2$ and $\hat{M}_{2k} = M_{2k} - \mathrm{Diag}(\mathrm{diag}(M_{2k}))$;
- Compute $\tilde{N}_{2k} = V^k \Sigma_k (V^k)^\top$, determine $V_{\mathcal{L}}^k$, and then compute

$$W^k = V_{\mathcal{L}}^k {V_{\mathcal{L}}^k}^\top;$$

- Compute

$$Z^k = \max\Big\{\frac{\hat{M}_{2k} + \hat{M}_{2k}^\top}{2}, 0\Big\}.$$

Step 4. Compute W_{ag}^k and Z_{ag}^k according to

- Compute $N_{2k}^{ag} = W_{md}^k - \beta_k \nabla\psi_W(W_{md}^k)$ and $M_{2k}^{ag} = Z_{md}^k - \beta_k \nabla\psi_Z(Z_{md}^k)$;
- Set $\tilde{N}_{2k}^{ag} = (N_{2k}^{ag} + (N_{2k}^{ag})^\top)/2$ and $\hat{M}_{2k}^{ag} = M_{2k}^{ag} - \mathrm{Diag}(\mathrm{diag}(M_{2k}^{ag}))$;
- Compute $\tilde{N}_{2k}^{ag} = U^k \Sigma_k (U^k)^\top$, determine $U_{\mathcal{L}}^k$, and then compute

$$W_{ag}^k = U_{\mathcal{L}}^k {U_{\mathcal{L}}^k}^\top;$$

- Compute

$$Z_{ag}^k = \max\Big\{\frac{\hat{M}_{2k}^{ag} + (\hat{M}_{2k}^{ag})^\top}{2}, 0\Big\}.$$

Step 5. Let $k-1 := k$, go to Step 1.

From [5, Lemma 1, Corollary 1], we can get the following theorem which further indicates that AGM converges globally.

Theorem 3.1. *Let $\{Z^k, W^k\}$ be the sequence generated by the AGM with properly choosing α_k, β_k and ρ_k. Then for any $N \geq 1$, we have*

$$\min_{k=1,\cdots,N} \left\| \mathcal{G}(Y^k_{md}, \nabla\psi(Y^k_{md}), \beta_k) \right\|^2 \leq 24L_\psi \Big[\frac{4L_\psi \|Y^0 - Y^*\|^2}{N(N+1)(N+2)} + \frac{L_f}{N}(\|Y^*\|^2 + M^2) \Big],$$

where $Y^k = [Z^k; W^k]$ and Y^* is a critical point for problem (3.4), which means that $\mathcal{G}(Y^k_{md}, \nabla\psi(Y^k_{md}), \beta_k)$ vanishes if N is large enough and $\{Z^k, W^k\}$ converges globally from [5, Lemma 3].

The Theorem indicates that, one can find an approximated solution $\tilde{Y}$ with a tolerance $\epsilon > 0$ such that $\|\mathcal{G}(\tilde{Y}, \nabla\psi(\tilde{Y}), \frac{1}{2L_\psi})\|^2 \leq \epsilon$ in at most $\mathcal{O}\big(L_\psi^{2/3}/\epsilon^{1/3} + L_\psi L_f/\epsilon\big)$ iterations.

4 Numerical Experiments

This section is devoted to evaluating the feasibility and efficiency of AGM on the standard motion segmentation dataset – Hopkins 155 [13] and the Extended Yale Database B [14]. All experiments are performed under MAC OS and Matlab R2018a running on a MacBook Air with an Intel Core i5 CPU at 1.60 GHz and 8 GB of memory. In all the tests given below, we use zero matrices as starting points, and simply terminate the iterative process when the relative changes of two consecutive iterations are sufficiently small, i.e.,

$$\text{RelErr} := \frac{\|Z^{k+1} - Z^k\|}{\|Z^k\|} \leq \epsilon,$$

where "$\epsilon \geq 0$" is a given margin of error. Specifically, if we can't achieve convergence within the maximum the number of iterations 1000, the iterative process is forcefully terminate. The quality of an optimal solution is measured by using the usual clustering error defined as:

$$\text{CluErr} := 1 - \frac{1}{n}\sum_{i=1}^{n} \delta\big(a_i, \text{map}(b_i)\big),$$

where a_i and b_i represent the output label and the true one of the i-th point respectively, $\delta(x,y) = 1$ if $x = y$ and 0 otherwise, and $\text{map}(b_i)$ is the best mapping function that permutes clustering labels to match the true labels.

To be more clearly evaluate the efficiencies and stabilities of AGM for solving (1.7), we also do performance comparisons with the state-of-the-art algorithms BDR and ABCGD. For both algorithms, we use the Matlab packages provided by the authors and set all the parameters as default. Before we begin our test, we briefly review the iterative frame of ABCGD to make it is easier to follow. The algorithm ABCGD is proposed by Kong at her thesis [6], which employs the famous Nesterov's accelerated gradient method to solve (1.7) with equivalent form (3.1). Starting from $\tilde{Z}^k$, it is from [6] we know that the iterative scheme of ABCGD is the following:

$$\begin{cases} W^{k+1} = \arg\min_W \left\{ \delta_{\mathcal{C}_1}(W) + \gamma \langle \mathrm{Diag}(\tilde{Z}^k 1) - \tilde{Z}^k, W \rangle + \frac{1}{2}\|W - W^k\|^2_{\mathcal{W}_1} \right\}, \\ Z^{k+1} = \arg\min_Z \left\{ \delta_{\mathcal{C}_2}(Z) + \gamma \langle \mathrm{Diag}(Z1) - Z, W^{k+1} \rangle + \frac{1}{2}\|X - XZ\|^2 + \frac{1}{2}\|Z - Z^k\|^2_{\mathcal{W}_2} \right\}, \\ t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}, \\ \tilde{Z}^{k+1} = Z^{k+1} + \frac{t_k - 1}{t_{k+1}}\big(Z^{k+1} - Z^k\big). \end{cases}$$

where $t_0 > 0$ is a constant, and $\mathcal{W}_1$ and $\mathcal{W}_2$ are self-adjoint semi-positive definite linear operators to make each subproblem easier to compute. For more details, one may refer to [6].

$\boxed{4.1}$ Motion Segmentation

The Hopkins 155 data set provides fact-based movement labels and outlier-free feature trajectories (x-, y-coordinates) by the pictures with moderate noises. The number of feature trajectories with different colors in every sequence ranges from 39 to 556, and the frame from 15 to 100. In the affine camera model, the movement track is in an affine subspace which is three dimensional at best, as a result of which the subspace clustering methods can be applied to motion segmentation, and every sequence actually is a separate clustering task. In this test, the primitive 2F-dimensional feature trajectories are used, where F stands for the number of frames in the video sequences. Exactly, if there is a set of feature points $x_{f_i} \in \mathbb{R}^2$ with $i = 1, \ldots, N$ and every frame is expressed as $f = 1, \ldots, F$ in the video. Then, under the affine projection model, the feature trajectory is formed as $y_i = [x_{1i}^\top, x_{2i}^\top, \ldots, x_{Fi}^\top]^\top \in \mathbb{R}^{2F}$ by superimposing the feature point x_{f_i}. Since the trajectories are relevant to the single rigid movement in the affine subspace of $\mathbb{R}^{2F}$ which is at most four dimensions, it is composed by l rigid movements in the union of l low-dimensional subspaces of $\mathbb{R}^{2F}$. Therefore, the problem of affine multi-view motion segmentation can be simplified as a subspace clustering problem.

In this test, we choose the model parameter as $\gamma = 0.04$ which is same as the one in AGM, BDR and ABCGD, and set the error tolerance as $\epsilon = 1e-3$. In each test, we report the results obtained by all the methods with respect to the sequence name in Hopkins 155 (Name), the number of motions in this sequence (Motions), the number of iterations (Iter), the computing time (Time), the relative changes of final two consecutive iterations (RelErr), and the clustering errors of the final solution (CluErr). The detailed computational results of each algorithm for these problems are reported in Table 3- 5.

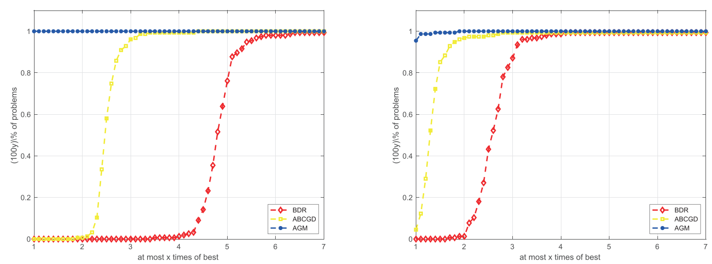
It can be seen from Table 3 -5 that, the quality of CluErr and RelErr of the solutions produced by BDR, ABCGD, and AGM are almost the same, but the computing times and the number of iterations are significantly different. For reporting the performance of algorithms preferably, we compute the average values of Time, Iter, and CluErr produced by BDR, ABCGD and AGM, and then display them in Table 1. As can be seen from this table that, AGM performs much better than BDR and ABCGD, and particularly, AGM is faster than the state-of-the-art algorithm ABCGD and at least two times faster than BDR for the vast majority of the tested problems.

To more clearly show the performance of each algorithm, we draw the profiles of Dolan and Moré [2] regarding to computing time and iterations. We recall that a point (x, y) is in the performance profile curve of a method if and only if it can solve exactly $(100y)\%$ of all

Table 1: Average result of AGM, BDR and ABCGD

Motions	AGM			BDR			ABCGD		
	Iter	Time	CluErr	Iter	Time	CluErr	Iter	Time	CluErr
2	57.4	2.840	0.0692	277.1	7.521	0.0900	144.3	3.741	0.0977
3	59.5	6.484	0.1414	284.4	16.870	0.1956	147.0	8.331	0.1689
all	57.90	3.663	0.0855	278.75	9.632	0.1136	144.94	4.7775	0.1138

the tested problems at most x times worse than any other methods. In short, the top curved shape at the figure means that the corresponding algorithm is a winner. The performance profiles of all the algorithms are plotted in Figure 1. Observing Figure 1, it is clear that, in each plot, the blue broken line is always at the top and the yellow dashed line at the second, which indicates that AGM performs better than ABCGD, and they both perform better than BDR.



(a) (b)

Figure 1: Performance profiles of BDR, ABCGD and AGM based on (a) iterations and (b) computing time.

4.2 Face Clustering

In this part, we further evaluate the practical abilities of AGM on the Extended Yale B database which is available at `http://vision.ucsd.edu/leekc/ExtYaleDatabase/ExtYaleB.html`. The Extended Yale B database consists the frontal face images of 28 human subjects under 9 poses and 64 illumination conditions. The data set partitions these images into 38 classes and each one contains 64 face images with 192×168 pixels. To reduce the computation and memory cost, we downsample each image to 32×32 pixels and then vectorize it as a vector with length 1024. Besides, to avoid overflows, we normalize each data into an unit length. We construct the data matrix X from subsets which consist of different numbers of subjects $\kappa \in \{2, 3, 5, 8, 10\}$ from the Extended Yale B database. For each κ, we randomly sample κ number subjects face images from this data set to construct the data matrix $X \in \mathbb{R}^{m \times n}$, where $m = 1024$ and $n = 64\kappa$. Then the subspace clustering methods can be performed on X and the segmentation accuracies are recorded. We run 20 times of each algorithm and list the results of the mean of segmentation accuracy, running time (Time), and number of iterations (Iter) for each algorithm in Table 2.

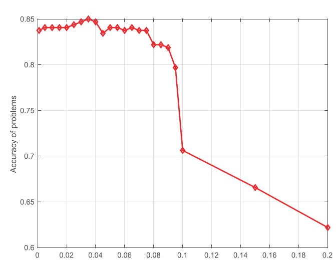
It can be seen from the first column in Table 2 that AGM produce higher quality segmentation accuracies than BDR and ABCGD, and as the number of cluster increases, the accuracy decreases monotonously. While we turn our attention to the other columns regarding to iterations, we can find that AGM only needs two to three hundreds iterations, but

ABCGD needs at least one thousand and BDR requires at least two thousand iterations. The phenomenon is not surprising, because the AGM not only converges globally but also has the ability to accelerate its iterative points, so that the number of iterations should be reduced greatly. Moreover, the third column shows that AGM is at leat two times faster than ABCGD and about four times than BDR. At last, we also see that as the number of cluster increases, the computing time and the iterations both increase correspondingly. Taking everything together, this experiment once again demonstrates the effectiveness of our AGM for the challenging face clustering task on the Extended Yale B database.

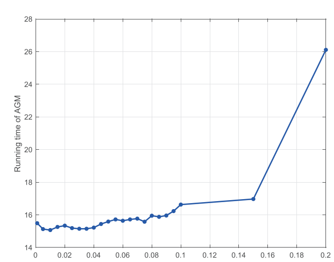
Table 2: The mean of segmentation accuracy(%), running time(s), and iterations of each algorithm.

NCluster	AGM			BDR			ABCGD		
	Accuracy	Time	Iter	Accuracy	Time	Iter	Accuracy	Time	Iter
2	100.000	1.120	293	100.000	3.020	1965	100.000	1.480	1012
3	95.833	1.790	326	94.792	7.150	2039	94.792	4.300	1058
5	96.875	4.650	315	96.875	18.540	2081	96.875	8.200	1070
8	84.375	11.780	315	73.047	41.320	2106	73.047	26.000	1080
10	84.688	18.950	313	69.062	67.570	2145	69.062	37.150	1098

To end this part, we test the influence of the regular parameter γ on segmentation accuracy and algorithm's performance. In this test, we set the error tolerance as $\epsilon = 1e-4$ and choose the parameter values γ from 0.001 to 0.01 with apart 0.001, and then from 0.01 to 0.1 with apart 0.01, and then from 0.1 to 0.2 with apart 0.05. In this test, we use 10 subjects from the Extended Yale B database to observe the segmentation accuracy and computing time when the regular parameter γ increases. The behavior is drawn in Figure 2. It can be seen from the figure that with the increase of γ, the segmentation accuracy and computing time almost remain unchanged, and then change rapidly from the point $\gamma = 0.09$, that is, it needs to spend more computing time but getting lower segmentation accuracy. From this simple test, we can conclude that the parameter regular value $\gamma = 0.02$ to 0.08 are all suitable choices.



(a)



(b)

Figure 2: Changes of the segmentation accuracy (a) and Running time (b) of AGM as the regular parameter γ increases.

5 Concluding Remarks

The sparse subspace clustering problem was recently characterized as a block diagonal matrix regularized nonconvex minimization problem. The earliest algorithm BDR targeted to a penalty model but not the original model (1.7) itself. The recent algorithm ABCGD has the ability to solve the original model (1.7), but its convergence is still not known. To remedy these deficiencies, this paper proposed an efficient algorithm with convergence guaranteed to solve the original model (1.7). The algorithm is an implementation of AGM [5] in which we showed that each subproblem is easily implementable by taking full use of the favourable structure of the constraints. We showed that the generated sequence converges globally to a critical point of the original model (1.7) if the stepsize is chosen properly. We have tested the proposed algorithm on the Hopkins 155 and Extended Yale B real datasets and did performance comparisons with BDR, BCD, and ABCGD. The results demonstrated that the proposed AGM is faster than the ABCGD, and highly faster than BDR. At last but not at least, we must emphasize that the stepsize is heavily depending on the Lipschitz constant of the differentiable term, which is not an easy task to evaluate. Therefore, it is an interesting topic for further research.

Acknowledgements

We would like to thank the anonymous referees for their useful comments and suggestions which improved this paper greatly.

References

[1] A. Beck, and M. Teboulle, A fast iterative shrinkage-thresholding algorithm for linear inverse problems, *SIAM J. Imaging Sci.* 2 (2009) 183–202.

[2] E.D. Dolan and J.J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.* 91 (2002) 201–213.

[3] E. Elhamifar and R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, *IEEE T. Pattern. Anal.* 35 (2013) 2765–2781.

[4] J. Feng, Z. Lin, H. Xu, and S. Yan, Robust subspace segmentation with block-diagonal prior, in: *Computer Vision and Pattern Recognition*, 2014, pp. 3818–3825.

[5] S. Ghadimi, and G. H. Lan, Accelerated gradient methods for nonconvex nonlinear and stochastic programming, *Math. Program.* 156 (2016) 59–99.

[6] Y. Kong, Research on the block coordinate descent methods for sparse subspace clustering problems, Henan University, M.Sc. degree thesis, 2020.

[7] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu and Y. Ma, Robust recovery of subspace structures by low-rank representation, *IEEE T. Pattern. Anal.* 35 (2013) 171–184.

[8] G. Liu, Z. Lin and Y. Yu, Robust subspace segmentation by low-rank representation, *ICML* 3 (2010) 663–670.

[9] C. Lu, J. Feng, Z. Lin, T. Mei and S. Yan, Subspace clustering by block diagonal representation, *IEEE T. Pattern. Anal.* 41(2019) 487–501.

[10] D.J. Luo, F.P. Nie, C.Ding and H. Huang, Multi-subspace representation and discovery, in: *Machine Learning and Knowledge Discovery in Databases*, Springer, 2011, pp. 405–420.

[11] Y. Nesterov, A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$, *Soviet Mathematics Doklady* 27(1983) 372–376.

[12] R.T. Rockafellar, *Convex Analysis*, Princeton University Press, 1970.

[13] R. Tron and R. Vidal, A benchmark for the comparison of 3-D motion segmentation algorithms, in: *Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[14] Y. Xiao, S.-Y. Wu and D.H. Li, Splitting and linearizing augmented Lagrangian algorithm for subspace recovery from corrupted observations, *Adv. Comput. Math.* 38 (2013) 837–858.

[15] S.J. Wright, Coordinate descent algorithms, *Math. Program.* 151(2015) 3–34.

[16] L. Zhuang, H. Gao, Z. Lin, Y. Ma, X. Zhang and N. Yu, Non-negative low rank and sparse graph for semi-supervised learning, in: *Computer Vision and Pattern Recognition*, 2012, pp. 2328–2335.

Manuscript received 15 April 2021
revised 29 June 2021
accepted for publication 12 October 2021

Hongwu Li
Schoole of Science, Beijing University of Technology
Beijing 100124, China
School of Mathematics and Statistics
Nanyang Normal University, Nanyang 473061, China
E-mail address: lihongwu2018@163.com

Haibin Zhang
School of Science, Beijing University of Technology
Beijing 100124, P.R. China
E-mail address: zhanghaibin@bjut.edu.cn

Yunhai Xiao
Center for Applied Mathematics of Henan Province
Henan University, Zhengzhou 450046, P.R. China
E-mail address: yhxiao@henu.edu.cn

Table 3: Numerical results of BDR, ABCGD and AGM (I)

Name	Mt	BDR				ABCGD				AGM			
		Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr
1R2RC	3	283	25.19	9.97e-04	0.205	148	10.98	9.97e-04	0.191	61	11.37	9.74e-04	0.189
1R2RCR	2	303	2.05	1.00e-03	0.000	148	0.87	9.96e-04	0.000	54	0.70	9.97e-04	0.000
1R2RCR_g12	2	272	14.03	9.97e-04	0.240	144	6.22	9.97e-04	0.243	59	5.21	9.90e-04	0.238
1R2RCR_g13	2	268	15.36	9.98e-04	0.188	137	9.06	1.00e-03	0.178	57	6.04	9.96e-04	0.072
1R2RCR_g23	3	288	18.64	9.98e-04	0.114	141	8.05	1.00e-03	0.105	60	5.84	9.80e-04	0.069
1R2RCT_A	2	313	3.65	9.97e-04	0.000	167	1.82	9.96e-04	0.000	60	1.08	9.79e-04	0.000
1R2RCT_A_g12	2	299	9.31	9.98e-04	0.141	150	3.57	9.99e-04	0.081	59	2.60	9.93e-04	0.074
1R2RCT_A_g13	2	265	8.70	9.98e-04	0.051	126	4.21	9.96e-04	0.060	56	3.76	9.79e-04	0.051
1R2RCT_A_g23	3	298	23.21	9.98e-04	0.131	147	11.45	9.98e-04	0.246	59	7.52	9.97e-04	0.115
1R2RCT_B	2	296	2.49	9.99e-04	0.000	158	1.79	9.97e-04	0.000	60	1.05	9.74e-04	0.000
1R2RCT_B_g12	2	270	7.45	9.97e-04	0.138	134	4.47	9.93e-04	0.129	56	3.10	9.79e-04	0.107
1R2RCT_B_g13	2	285	13.21	9.99e-04	0.123	147	8.16	9.97e-04	0.113	60	5.52	9.79e-04	0.102
1R2RCT_B_g23	2	290	3.11	9.96e-04	0.000	140	1.65	9.95e-04	0.000	54	1.18	9.84e-04	0.000
1R2RC_g12	2	274	8.33	9.98e-04	0.254	136	3.96	9.99e-04	0.260	57	3.47	9.98e-04	0.254
1R2RC_g13	2	268	11.01	9.99e-04	0.192	133	5.01	9.99e-04	0.197	57	4.52	9.81e-04	0.197
1R2RC_g23	3	279	29.19	9.97e-04	0.131	141	15.89	9.97e-04	0.131	61	12.88	9.79e-04	0.131
1R2TCR	3	287	25.48	9.97e-04	0.155	140	12.27	9.98e-04	0.167	62	12.67	9.91e-04	0.151
1R2TCRT	2	257	2.51	9.97e-04	0.000	125	1.20	9.91e-04	0.000	53	1.14	9.88e-04	0.000
1R2TCRT_g12	2	279	12.30	9.97e-04	0.013	146	7.28	9.97e-04	0.024	62	6.68	9.91e-04	0.000
1R2TCRT_g13	2	271	16.14	1.00e-03	0.021	125	7.07	1.00e-03	0.000	59	6.91	9.83e-04	0.174
1R2TCRT_g23	2	295	3.55	9.98e-04	0.000	147	1.77	9.97e-04	0.000	55	1.87	9.82e-04	0.000
1R2TCR_g12	2	284	14.67	9.98e-04	0.027	140	6.52	9.96e-04	0.024	60	6.50	9.84e-04	0.027
1R2TCR_g13	2	267	19.55	9.97e-04	0.129	138	9.79	9.93e-04	0.131	60	9.05	9.88e-04	0.129
1R2TCR_g23	3	305	24.63	9.98e-04	0.116	151	11.94	9.99e-04	0.116	60	8.98	9.86e-04	0.122
1RT2RCR	3	284	17.51	9.97e-04	0.088	149	10.62	9.99e-04	0.095	61	8.50	9.80e-04	0.079
1RT2RCRT	2	286	2.53	1.00e-03	0.000	157	1.69	9.95e-04	0.000	57	1.22	9.82e-04	0.000
1RT2RCRT_g12	2	263	8.98	9.97e-04	0.070	142	5.75	9.96e-04	0.079	60	4.00	9.89e-04	0.000
1RT2RCRT_g13	2	266	8.18	9.98e-04	0.103	138	4.09	1.00e-03	0.103	57	3.30	9.95e-04	0.103
1RT2RCRT_g23	2	293	4.82	9.96e-04	0.000	154	2.87	9.98e-04	0.000	58	1.89	9.97e-04	0.000
1RT2RCR_g12	2	277	10.81	9.98e-04	0.114	143	5.60	9.97e-04	0.143	59	4.44	9.86e-04	0.140
1RT2RCR_g13	2	257	8.35	9.99e-04	0.171	126	4.42	9.95e-04	0.163	54	3.57	9.73e-04	0.163
1RT2RCR_g23	3	302	8.49	9.98e-04	0.119	150	4.12	9.99e-04	0.137	59	3.12	9.97e-04	0.125
1RT2RTCRT_A	2	298	2.29	9.97e-04	0.000	159	1.23	9.96e-04	0.000	54	0.82	9.82e-04	0.000
1RT2RTCRT_A_g12	2	272	4.46	9.96e-04	0.101	136	2.13	9.93e-04	0.125	56	1.83	9.83e-04	0.101
1RT2RTCRT_A_g13	2	251	3.09	9.99e-04	0.167	125	1.44	9.95e-04	0.163	54	1.27	9.72e-04	0.145
1RT2RTCRT_A_g23	3	299	14.00	1.00e-03	0.180	154	7.20	9.94e-04	0.177	61	5.43	9.83e-04	0.157
1RT2RTCRT_B	2	318	2.21	1.00e-03	0.000	173	1.12	9.95e-04	0.000	64	0.93	9.84e-04	0.000
1RT2RTCRT_B_g12	2	284	7.66	1.00e-03	0.153	146	3.78	9.96e-04	0.037	60	3.18	9.81e-04	0.016
1RT2RTCRT_B_g13	2	270	6.74	9.97e-04	0.212	139	3.34	9.95e-04	0.199	56	2.50	9.83e-04	0.199
1RT2RTCRT_B_g23	3	293	7.35	9.99e-04	0.110	149	3.71	9.98e-04	0.110	58	2.67	9.84e-04	0.094
1RT2TC	3	289	6.55	9.96e-04	0.145	159	3.68	9.93e-04	0.125	61	2.71	9.91e-04	0.132
1RT2TCRT_A	2	290	1.99	9.99e-04	0.000	154	1.19	9.96e-04	0.000	57	0.73	9.85e-04	0.000
1RT2TCRT_A_g12	2	262	3.15	9.99e-04	0.005	142	1.71	9.95e-04	0.434	58	1.32	9.84e-04	0.434
1RT2TCRT_A_g13	2	244	2.90	9.85e-04	0.192	127	1.41	9.94e-04	0.174	55	1.26	9.85e-04	0.179
1RT2TCRT_A_g23	3	262	8.61	9.99e-04	0.122	143	4.06	9.95e-04	0.104	59	3.43	9.82e-04	0.104
1RT2TCRT_B	2	299	1.60	9.96e-04	0.000	163	0.84	9.98e-04	0.000	57	0.59	9.79e-04	0.000
1RT2TCRT_B_g12	2	263	4.19	9.97e-04	0.020	145	2.28	9.93e-04	0.024	60	1.88	9.95e-04	0.000
1RT2TCRT_B_g13	2	242	4.47	9.98e-04	0.130	128	2.24	9.98e-04	0.120	55	1.97	9.95e-04	0.123
1RT2TCRT_B_g23	2	300	3.49	1.00e-03	0.000	155	1.46	9.93e-04	0.000	59	1.14	9.95e-04	0.000
1RT2TC_g12	2	286	3.02	9.96e-04	0.000	144	1.41	9.94e-04	0.000	53	1.06	9.97e-04	0.000
1RT2TC_g13	2	279	3.13	9.99e-04	0.092	140	1.40	9.99e-04	0.135	52	1.13	9.84e-04	0.135
1RT2TC_g23	3	303	25.27	9.99e-04	0.273	148	14.63	9.95e-04	0.271	62	9.85	9.86e-04	0.259
2R3RTC	3	305	16.27	9.99e-04	0.146	149	9.13	9.94e-04	0.156	62	5.51	9.85e-04	0.144
2R3RTCRT	2	256	2.96	9.98e-04	0.000	135	2.01	9.97e-04	0.000	53	1.23	9.79e-04	0.000
2R3RTCRT_g12	2	300	6.26	9.99e-04	0.231	154	3.30	9.94e-04	0.265	61	2.23	9.87e-04	0.213
2R3RTCRT_g13	2	292	8.84	9.98e-04	0.069	137	4.21	9.98e-04	0.176	58	3.18	9.96e-04	0.170
2R3RTCRT_g23	2	263	4.35	9.98e-04	0.000	136	2.54	9.93e-04	0.000	52	1.63	9.98e-04	0.000
2R3RTC_g12	2	260	7.54	9.99e-04	0.390	124	3.64	9.97e-04	0.384	56	3.10	9.82e-04	0.369
2R3RTC_g13	2	296	14.88	9.99e-04	0.039	139	8.03	9.99e-04	0.051	60	5.46	9.98e-04	0.022
2R3RTC_g23	3	291	31.15	1.00e-03	0.445	137	15.13	9.99e-04	0.345	59	11.29	9.96e-04	0.102
2RT3RC	3	292	26.18	1.00e-03	0.247	134	12.80	1.00e-03	0.215	59	9.89	9.74e-04	0.280
2RT3RCR	2	267	3.62	9.98e-04	0.000	139	2.02	1.00e-03	0.000	54	1.42	9.74e-04	0.000

Table 4: Numerical results of BDR, ABCGD and AGM (II)

Name	Mt	BDR				ABCGD				AGM			
		Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr
2RT3RCR_g12	2	291	14.26	9.99e-04	0.093	138	6.03	9.98e-04	0.083	58	4.63	9.89e-04	0.013
2RT3RCR_g13	2	272	15.31	1.00e-03	0.428	130	7.18	9.99e-04	0.234	57	5.74	9.79e-04	0.416
2RT3RCR_g23	3	301	12.50	1.00e-03	0.125	159	5.71	9.95e-04	0.143	63	4.12	9.80e-04	0.140
2RT3RCT_A	2	261	2.67	9.97e-04	0.000	138	1.26	9.95e-04	0.000	57	0.92	9.94e-04	0.000
2RT3RCT_A_g12	2	277	4.44	9.97e-04	0.031	148	1.84	9.93e-04	0.004	61	1.51	9.65e-04	0.004
2RT3RCT_A_g13	2	287	6.83	9.98e-04	0.047	145	2.97	9.96e-04	0.054	59	2.44	9.80e-04	0.047
2RT3RCT_A_g23	3	299	25.27	9.97e-04	0.308	147	10.44	9.96e-04	0.308	61	7.97	9.99e-04	0.302
2RT3RCT_B	2	261	2.10	1.00e-03	0.000	154	1.05	9.99e-04	0.000	50	0.72	9.88e-04	0.000
2RT3RCT_B_g12	2	263	8.92	9.97e-04	0.401	134	3.97	1.00e-03	0.390	59	3.54	9.85e-04	0.398
2RT3RCT_B_g13	2	285	16.39	9.98e-04	0.059	138	7.75	9.99e-04	0.374	59	6.22	9.92e-04	0.031
2RT3RCT_B_g23	2	260	3.36	9.98e-04	0.000	145	2.30	9.98e-04	0.000	56	1.37	9.87e-04	0.000
2RT3RC_g12	2	278	16.13	9.98e-04	0.411	132	6.90	9.99e-04	0.381	58	5.69	9.81e-04	0.000
2RT3RC_g13	2	278	20.61	9.99e-04	0.271	129	8.75	9.95e-04	0.163	57	7.42	9.97e-04	0.119
2RT3RC_g23	3	296	11.38	9.98e-04	0.179	157	6.53	9.98e-04	0.140	61	4.02	9.94e-04	0.160
2RT3RTCRT	2	279	2.27	1.00e-03	0.000	154	1.22	9.98e-04	0.000	60	0.87	9.74e-04	0.000
2RT3RTCRT_g12	2	287	5.39	9.99e-04	0.278	159	2.98	9.94e-04	0.275	62	2.00	9.97e-04	0.275
2RT3RTCRT_g13	2	276	6.29	9.98e-04	0.221	147	3.33	9.93e-04	0.218	59	2.41	9.87e-04	0.183
2RT3RTCRT_g23	3	284	29.67	1.00e-03	0.424	144	13.30	9.96e-04	0.342	60	12.11	9.82e-04	0.379
2T3RCR	3	292	30.84	9.98e-04	0.409	143	14.57	9.98e-04	0.438	61	11.63	9.92e-04	0.409
2T3RCRT	3	294	24.76	9.98e-04	0.118	144	11.50	9.95e-04	0.122	61	9.13	9.87e-04	0.049
2T3RCRTP	2	283	3.50	1.00e-03	0.000	147	2.28	9.99e-04	0.000	59	1.29	9.85e-04	0.000
2T3RCRTP_g12	2	290	10.34	9.99e-04	0.331	148	6.24	9.95e-04	0.381	61	3.89	9.87e-04	0.000
2T3RCRTP_g13	2	286	15.48	9.98e-04	0.141	138	9.28	9.98e-04	0.078	59	6.10	9.96e-04	0.078
2T3RCRTP_g23	2	293	2.22	9.99e-04	0.000	162	1.20	9.99e-04	0.000	58	1.09	9.66e-04	0.000
2T3RCRT_g12	2	282	20.47	9.98e-04	0.178	139	10.09	9.99e-04	0.022	59	7.02	9.98e-04	0.000
2T3RCRT_g13	2	284	21.62	9.99e-04	0.457	135	8.69	9.99e-04	0.472	59	7.74	9.99e-04	0.478
2T3RCRT_g23	2	257	2.95	9.99e-04	0.000	147	1.19	9.99e-04	0.000	58	0.98	9.80e-04	0.000
2T3RCR_g12	2	286	15.01	1.00e-03	0.385	147	7.73	1.00e-03	0.385	61	6.05	9.84e-04	0.381
2T3RCR_g13	2	273	19.83	9.99e-04	0.399	139	11.51	9.96e-04	0.386	58	7.54	9.93e-04	0.326
2T3RCR_g23	3	294	22.81	9.98e-04	0.192	148	10.31	1.00e-03	0.094	60	7.70	9.78e-04	0.066
2T3RCTP	2	244	4.54	9.99e-04	0.000	133	2.21	9.97e-04	0.000	55	1.82	9.83e-04	0.000
2T3RCTP_g12	2	286	8.60	9.98e-04	0.204	145	3.70	9.98e-04	0.163	58	3.14	9.81e-04	0.000
2T3RCTP_g13	2	283	11.26	9.99e-04	0.036	139	4.44	9.98e-04	0.006	56	3.74	9.94e-04	0.078
2T3RCTP_g23	3	311	30.49	9.99e-04	0.327	150	13.91	9.99e-04	0.308	62	10.56	9.90e-04	0.351
2T3RTCR	2	272	7.22	9.97e-04	0.000	136	3.34	9.97e-04	0.000	54	2.93	9.88e-04	0.000
2T3RTCR_g12	2	283	10.40	1.00e-03	0.009	140	4.88	9.96e-04	0.003	59	4.77	9.90e-04	0.003
2T3RTCR_g13	2	293	13.81	9.97e-04	0.360	144	9.67	9.95e-04	0.305	59	5.43	9.89e-04	0.292
2T3RTCR_g23	2	280	0.44	9.99e-04	0.026	150	0.25	9.99e-04	0.000	64	0.21	9.78e-04	0.000
arm	3	276	1.58	9.98e-04	0.000	147	1.33	9.97e-04	0.000	44	0.50	9.90e-04	0.000
articulated	2	267	0.47	9.97e-04	0.000	145	0.46	9.98e-04	0.000	49	0.17	9.95e-04	0.000
articulated_g12	2	320	0.97	9.96e-04	0.000	174	0.67	9.99e-04	0.000	65	0.36	9.86e-04	0.000
articulated_g13	2	265	0.90	9.98e-04	0.000	148	0.71	9.93e-04	0.000	56	0.35	9.97e-04	0.000
articulated_g23	2	290	7.23	9.98e-04	0.003	158	4.23	9.96e-04	0.003	63	3.01	9.85e-04	0.003
cars1	3	297	6.71	9.98e-04	0.054	161	3.90	9.97e-04	0.054	64	2.81	9.93e-04	0.051
cars10	2	311	4.24	9.99e-04	0.000	168	2.15	9.99e-04	0.000	64	1.74	1.00e-03	0.000
cars10_g12	2	305	3.83	1.00e-03	0.000	160	3.26	9.99e-04	0.000	61	1.63	9.80e-04	0.036
cars10_g13	2	310	2.05	9.99e-04	0.101	166	1.19	9.99e-04	0.094	62	0.87	9.94e-04	0.082
cars10_g23	2	298	25.03	9.97e-04	0.333	145	11.43	9.96e-04	0.451	61	9.00	9.94e-04	0.004
cars2	3	290	30.84	9.97e-04	0.389	153	15.89	9.98e-04	0.437	64	11.86	9.85e-04	0.000
cars2B	2	247	0.60	9.98e-04	0.000	162	0.37	9.93e-04	0.000	54	0.25	9.76e-04	0.000
cars2B_g12	2	294	27.78	9.99e-04	0.492	145	13.75	9.94e-04	0.472	62	9.90	9.73e-04	0.000
cars2B_g13	2	282	24.83	9.97e-04	0.000	151	10.35	9.99e-04	0.000	62	8.90	9.90e-04	0.000
cars2B_g23	3	241	1.30	9.98e-04	0.065	131	0.47	9.97e-04	0.033	55	0.48	9.87e-04	0.000
cars2_06	2	284	0.30	9.99e-04	0.000	174	0.15	9.97e-04	0.000	56	0.08	9.79e-04	0.000
cars2_06_g12	2	215	0.63	1.00e-03	0.000	126	0.36	9.97e-04	0.000	52	0.28	9.74e-04	0.000
cars2_06_g13	2	231	0.81	9.98e-04	0.000	125	0.38	1.00e-03	0.000	51	0.33	9.85e-04	0.010
cars2_06_g23	3	262	4.03	1.00e-03	0.429	147	1.67	9.97e-04	0.000	57	1.59	9.94e-04	0.000
cars2_07	2	317	0.22	9.98e-04	0.000	159	0.09	9.99e-04	0.000	50	0.07	9.19e-04	0.000
cars2_07_g12	2	253	2.45	9.97e-04	0.479	137	1.17	9.97e-04	0.000	53	1.02	9.91e-04	0.000
cars2_07_g13	2	258	3.04	1.00e-03	0.000	144	1.25	9.94e-04	0.000	50	0.99	9.89e-04	0.000
cars2_07_g23	3	250	30.74	9.97e-04	0.341	144	15.86	9.94e-04	0.356	58	12.48	9.75e-04	0.402
cars3	2	289	1.02	9.99e-04	0.000	160	0.59	9.91e-04	0.000	57	0.41	9.97e-04	0.000

Table 5: Numerical results of BDR, ABCGD and AGM (III)

Name	Mt	BDR				ABCGD				AGM			
		Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr	Iter	Time	RelErr	CluErr
cars3_g12	2	233	17.47	1.00e-03	0.000	137	11.62	9.95e-04	0.386	57	7.78	9.74e-04	0.000
cars3_g13	2	271	28.57	1.00e-03	0.000	129	12.10	9.94e-04	0.239	57	10.00	9.85e-04	0.000
cars3_g23	2	268	1.62	9.99e-04	0.000	150	0.79	9.95e-04	0.000	61	0.78	9.96e-04	0.000
cars4	3	229	11.43	9.99e-04	0.000	141	6.37	9.98e-04	0.056	58	4.76	9.79e-04	0.000
cars5	2	294	0.57	9.99e-04	0.000	157	0.34	9.93e-04	0.000	36	0.14	1.00e-03	0.000
cars5_g12	2	237	9.70	9.99e-04	0.000	112	4.53	9.95e-04	0.073	55	3.60	9.85e-04	0.000
cars5_g13	2	235	10.26	9.98e-04	0.000	120	4.67	1.00e-03	0.000	55	3.78	9.84e-04	0.000
cars5_g23	2	265	21.25	9.99e-04	0.000	115	7.82	9.98e-04	0.000	54	6.96	9.97e-04	0.000
cars6	2	277	25.30	9.98e-04	0.004	145	12.64	9.96e-04	0.267	61	9.79	9.75e-04	0.014
cars7	2	303	2.68	9.99e-04	0.000	162	1.31	9.95e-04	0.000	63	1.37	9.79e-04	0.000
cars8	3	275	3.69	9.99e-04	0.305	157	1.75	9.98e-04	0.309	60	1.33	9.94e-04	0.309
cars9	2	325	0.49	9.99e-04	0.000	169	0.28	9.99e-04	0.000	65	0.21	9.75e-04	0.000
cars9_g12	2	256	2.24	9.97e-04	0.005	152	1.27	9.94e-04	0.011	59	1.01	9.77e-04	0.000
cars9_g13	2	273	2.01	9.96e-04	0.011	142	1.04	9.95e-04	0.011	58	0.82	9.87e-04	0.011
cars9_g23	5	273	2.01	9.96e-04	0.011	142	1.04	9.95e-04	0.011	58	0.82	9.87e-04	0.011
dancing	2	291	0.82	9.92e-04	0.455	151	0.38	9.98e-04	0.424	51	0.27	9.83e-04	0.444
head	2	250	1.30	9.99e-04	0.000	137	0.60	9.97e-04	0.000	54	0.50	9.86e-04	0.000
kanatani1	2	304	0.41	9.98e-04	0.000	167	0.21	9.97e-04	0.000	62	0.18	9.93e-04	0.000
kanatani2	2	251	0.39	9.94e-04	0.178	118	0.19	9.96e-04	0.192	50	0.14	9.81e-04	0.192
kanatani3	2	278	25.97	9.99e-04	0.002	158	12.94	9.98e-04	0.024	50	0.21	9.81e-04	0.192
people1	2	278	21.25	9.98e-04	0.002	145	10.02	9.99e-04	0.275	62	9.24	9.80e-04	0.000
people2	3	268	2.02	1.00e-03	0.029	149	1.07	9.98e-04	0.029	58	0.99	9.89e-04	0.029
three-cars	2	291	0.69	9.95e-04	0.000	167	0.36	9.96e-04	0.000	63	0.27	9.95e-04	0.000
three-cars_g12	2	265	1.27	9.99e-04	0.000	142	0.54	9.97e-04	0.000	54	0.41	9.87e-04	0.016
three-cars_g13	2	311	1.79	1.00e-03	0.039	170	0.69	9.96e-04	0.031	58	0.49	9.86e-04	0.039
three-cars_g23	2	269	2.76	1.00e-03	0.000	139	1.13	1.00e-03	0.000	59	1.02	9.67e-04	0.000
truck1	2	291	9.78	9.97e-04	0.027	158	4.53	9.96e-04	0.027	62	4.12	9.95e-04	0.027
truck2	3	262	0.67	9.97e-04	0.415	142	0.32	9.93e-04	0.043	54	0.42	9.97e-04	0.043
two_cranes	2	250	0.35	9.95e-04	0.000	131	0.20	9.95e-04	0.000	55	0.16	9.71e-04	0.000
two_cranes_g12	2	275	0.44	9.96e-04	0.000	146	0.27	9.95e-04	0.000	53	0.20	9.72e-04	0.000
two_cranes_g13	2	255	0.13	1.00e-03	0.128	132	0.08	9.98e-04	0.154	73	0.15	9.73e-04	0.180