Residual Expansion Algorithm: Fast and Effective Optimization for Nonconvex Least Squares Problems

Daiki Ikami Toshihiko Yamasaki Kiyoharu Aizawa
The University of Tokyo, Japan
{ikami, yamasaki, aizawa}@hal.t.u-tokyo.ac.jp

Abstract

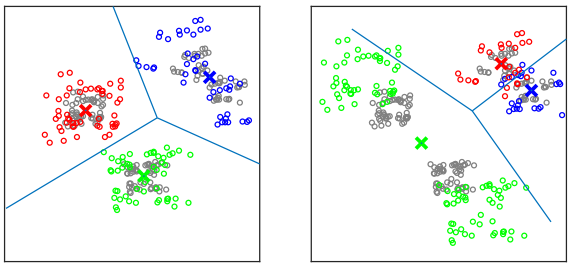
We propose the residual expansion (RE) algorithm: a global (or near-global) optimization method for nonconvex least squares problems. Unlike most existing nonconvex optimization techniques, the RE algorithm is not based on either stochastic or multi-point searches; therefore, it can achieve fast global optimization. Moreover, the RE algorithm is easy to implement and successful in high-dimensional optimization. The RE algorithm exhibits excellent empirical performance in terms of k-means clustering, point-set registration, optimized product quantization, and blind image deblurring.

1. Introduction

Many problems in computer vision and machine learning can be formulated as optimization problems. If we can formulate a problem as a convex optimization, we can solve it by convex optimization techniques such as gradient-based methods. However, most optimization problems are nonconvex and often have many local minima. In these cases, convex optimization techniques can find only local minima.

Global optimization of nonconvex problems is an NP-hard problem in most cases. Therefore, heuristic methods are often used to find a global (or near-global) optimum. There are two major approaches: good initialization and stochastic optimization. The former is fast and effective if we can obtain a good initial guess [2]; however, many optimization problems do not provide this. The latter is random search or multiple-point search, which is represented by simulated annealing (SA) [13], particle swarm optimization (PSO) [12], and genetic algorithms (GA) [19]. Although these methods are effective with low-dimensional optimization problems, it is difficult to obtain good solutions with high-dimensional ones. Moreover, these approaches often require excessive computation time to obtain a good solution.

In this paper, we propose a fast and effective optimization method for nonconvex least squares (LS) problems such as k-means clustering and point-set registration. First, we propose a novel measure of convergence called RE convergence: this represents how far we can expand data points along their residual directions under convergence. Fig. 1 shows k-means results and expanded data points. Fig. 1(a) depicts convergence on expanded data while Fig. 1(b) shows a case that is not converged. We presume that RE convergence is associated with global convergence. In fact, we can prove that the solution that is stable on a large expansion is the global optimum in the case of a one-dimensional quartic minimization problem.



(a) The global optimum result. (b) The local minimum result.

Fig 1: K-means clustering results with different RE convergence. Gray circles denote original data points and red, blue, and green circles denote α-expanded data points from each cluster center. Fig. 1(a) shows α RE convergence with $\alpha = 1$; however, Fig. 1(b) does not show this. In this case, the solution with a larger RE constant achieves the global optimum. The details of RE convergence and the RE constant are described in Section 3.

Additionally, we propose a heuristic algorithm to find a solution that is stable on the large expansion, which we term the residual expansion (RE) algorithm. This algorithm is based on neither multiple-point search nor random search, and thus fast computation can be achieved.

Our contribution is as follows:

1. We propose a novel concept of convergence, RE con-

vergence. We show the relationship between RE convergence and the global optimum.
2. We propose the RE algorithm, which can be applied for any nonconvex LS problem. We show that the RE algorithm is fast, effective, and easy to implement.
3. We show the RE algorithm's excellent performance for various nonconvex LS problems such as k-means clustering, point-set registration, optimized product quantization, and blind image deblurring.

2. Related works

2.1. Nonconvex least squares problems

We focus on nonconvex LS problems, of which many exist. In this paper, we study the following four important problems in computer vision and machine learning.

2.1.1 K-means clustering

K-means clustering is one of the most popular clustering methods with various applications such as quantization [11], feature learning [8], and segmentation [1]. K-means clustering assigns data vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ to the nearest representative clusters. The optimization problem can be formulated as

$$\min_{\mathbf{C},\mathbf{Z}} \frac{1}{2}\|\mathbf{X} - \mathbf{C}\mathbf{Z}\|_F^2 \tag{1}$$
$$\text{s.t. } z_{ij} = \{0,1\}, \|\mathbf{z}_i\|_1 = 1,$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ is a data matrix, $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_k] \in \mathbb{R}^{d\times k}$ is a matrix of cluster centroids, and $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n] \in \mathbb{R}^{k\times n}$ is an assignment matrix.

The most popular optimization method is Lloyd's algorithm [17], which has an update step (fix $\mathbf{Z}$ and update $\mathbf{C}$) and an assignment step (fix $\mathbf{C}$ and update $\mathbf{Z}$). Hartigan's algorithm [10] achieves better clustering than Lloyd's algorithm because the set of local minima of Hartigan's algorithm is a subset of those of Lloyd's algorithm [26, 25]. For good initialization, k-means++ [2] is often used because of its efficiency and effectiveness.

2.1.2 Point-set registration

Point-set registration is a fundamental problem in computer vision. Here we consider a rigid 3D-point-set registration problem: Given source point sets $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_n] \in \mathbb{R}^{3\times n}$ and target point sets $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_m] \in \mathbb{R}^{3\times m}$, we estimate the best rigid transformation parameters.

In this paper, we consider the following optimization problem with a point-to-point cost function:

$$\min_{\mathbf{R},\mathbf{t},\mathbf{Z}} \frac{1}{2}\|\mathbf{R}\mathbf{X} + \mathbf{t}\mathbf{1}^\top - \mathbf{Y}\mathbf{Z}\|_F^2 \tag{2}$$
$$\text{s.t. } z_{ij} = \{0,1\}, \|\mathbf{z}_i\|_1 = 1, \mathbf{R}^\top\mathbf{R} = \mathbf{I},$$

where $\mathbf{R} \in \mathrm{SO}(3)$ is a rotation matrix, $\mathbf{t} \in \mathbb{R}^3$ is a translation vector, and $\mathbf{Z} = [\mathbf{z}_1, \dots, \mathbf{z}_n] \in \mathbb{R}^{k\times n}$ is an assignment matrix. $\mathbf{I}$ is an identity matrix and $\mathbf{1}$ is a vector of all ones.

The iterative closest point (ICP) algorithm [3] is a well-known alternating optimization method: it fixes $\mathbf{Z}$ and updates $\mathbf{R}$, $\mathbf{t}$, and then fixes $\mathbf{R}, \mathbf{t}$ and updates $\mathbf{Z}$. To obtain a global minimum, some studies adopt stochastic optimization, such as GA [24], PSO [27], and SA [18]. Recently, Yang *et al*. proposed Go-ICP [29], which guarantees global optimality by using the branch-and-bound algorithm. However, it requires significant computation time.

2.1.3 Optimized product quantization

Optimized product quantization (OPQ) [9, 22], which is an extension of product quantization (PQ), is an efficient fast approximate nearest neighbor search method. The optimization problem in OPQ is described by

$$\min_{\mathbf{R},\mathbf{C},\mathbf{Z}} \frac{1}{2}\sum_{i=1}^{N} \left\| \mathbf{x}_i - \mathbf{R} \begin{bmatrix} \mathbf{C}^{(1)}\mathbf{z}_i^{(1)} \\ \vdots \\ \mathbf{C}^{(M)}\mathbf{z}_i^{(M)} \end{bmatrix} \right\|_2^2 \tag{3}$$
$$\text{s.t. } z_{ij}^{(m)} = \{0,1\}, \left\|\mathbf{z}_i^{(m)}\right\|_1 = 1, \mathbf{R}^\top\mathbf{R} = \mathbf{I},$$

where $\mathbf{X}, \mathbf{C}, \mathbf{Z}$ have the same meaning as in Section 2.1.1 and $\mathbf{R}$ is a rotation matrix.

The optimization problem of Eq. (3) can be solved by alternating optimization of $\mathbf{R}$, $\mathbf{C}$, and $\mathbf{Z}$ [9, 22]. Ge *et al*. also proposed a parametric optimization method that assumes the data follows a parametric Gaussian distribution [9].

2.1.4 Blind image deblurring

Blind image deblurring has long been a challenging problem in computer vision. From a blurred image $\mathbf{B} \in \mathbb{R}^{h\times w}$, we estimate an original image $\mathbf{I} \in \mathbb{R}^{h\times w}$ and blur kernel $\mathbf{k} \in \mathbb{R}^{k\times k}$ by minimizing the following cost function:

$$\min_{\mathbf{I},\mathbf{k}} \frac{1}{2}\|\mathbf{I}\otimes\mathbf{k} - \mathbf{B}\|_F^2 + \gamma_I R_I(\mathbf{I}) + \gamma_k R_k(\mathbf{k}), \tag{4}$$

where $R_I(\mathbf{I})$ and $R_k(\mathbf{k})$ are the regularization terms, and $\otimes$ denotes the convolution operator. For $R_I(\mathbf{I})$, L0-norm (or approximately L0-norm) [28, 23], or L1/L2 functions [15] are proposed to enforce the sharp edges of the original image. For $R_k(\mathbf{k})$, L2-norm [28, 23] or L1-norm [15] are often used. We refer to the paper [16] for a recent comparative study of blind image deblurring.

We can minimize Eq. (4) by alternating optimization of $\mathbf{I}$ and $\mathbf{k}$. For fast and effective optimization, a coarse-to-fine strategy [7, 15, 28, 23] is generally employed.

2.2. Nonconvex optimization techniques

Most nonconvex optimization techniques are based on stochastic optimization, including GA [19], PSO [12], and

SA [13]. These methods generally do not work well or require significant computation time for high-dimensional optimization problems. Several studies [4, 14, 18, 27, 24] have employed these nonconvex optimization techniques to our target problems described in Section 2.1; however, these methods are not often used in practice.

Our approach is related to graduated nonconvexity (GNC) [5], which first solves a simplified optimization problem and then gradually transforms the problem into the original nonconvex problem. The basic concept of graduated optimization methods is *extinguishing local minima* by using a convexified objective function, and then gradually changing the objective function to the original function. We refer readers to [20] for a recent survey of graduated optimization. In contrast to GNC, our approach is explicitly *to escape from poor local minima* by using a largely expanded objective function and then gradually transforming it into the original function, as described in Section 4.

3. Residual expansion convergence

First, we describe RE convergence, which indicates how we can expand data along their residual directions. RE convergence is a measure of the depth of convergence, and our proposed algorithm is based on this concept. We show a relationship between the global optimum and RE convergence.

We discuss a nonconvex least squares (LS) optimization problem whose objective function is given by

$$E(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{f}(\boldsymbol{\theta})\|_2^2. \tag{5}$$

Our definitions are as follows.

Definition 3.1 (Residual Expansion)**.** *Let $\boldsymbol{\theta}^*$ be a local minimum point of Eq. (5). We define the α-expanded objective function $E_\alpha(\boldsymbol{\theta})$:*

$$E_\alpha(\boldsymbol{\theta}) = \frac{1}{2}\|\hat{\mathbf{y}} - \mathbf{f}(\boldsymbol{\theta})\|_2^2. \tag{6}$$

where $\hat{\mathbf{y}}$ is constructed by expanding $\mathbf{y}$ in the residual direction with a magnitude of α as

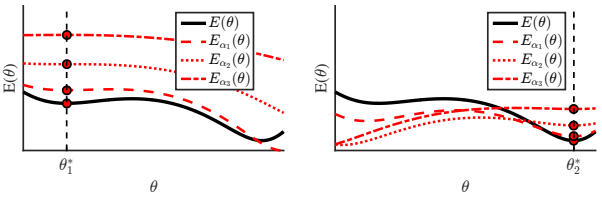
$$\hat{\mathbf{y}} = \mathbf{y} + \alpha(\mathbf{y} - \mathbf{f}(\boldsymbol{\theta}^*)), \tag{7}$$

We call the operation that constructs the α-expanded objective function residual expansion (with α).

Definition 3.2 (α RE convergence)**.** *$\boldsymbol{\theta}^*$ is called α RE convergence if there exists a constant $\alpha \geq 0$ such that $\boldsymbol{\theta}$ is still a local optimum of $E_\alpha(\boldsymbol{\theta})$. In particular, the maximum (or supremum) constant is called the RE constant*[1].

Our hypothesis is that *a solution with a larger α-RE constant is closer to the global optimum solution.* This hypothesis holds in the case of quartic minimization, as presented in section 3.1.1.

[1]If $\boldsymbol{\theta}^*$ is always a local minimum of $E_\alpha(\boldsymbol{\theta}^*)$ under all $\alpha \geq 0$, we define the RE constant as ∞.



(a) θ_1^* and its expanded objective function. (b) θ_2^* and its expanded objective function.

Fig 2: Expanded objective functions $E_\alpha(\theta)$ with different local minima, θ_1^* and θ_2^*. Red broken lines denote different α-expanded objective functions $E_\alpha(\theta)$ with $\alpha_1 < \alpha_2 < \alpha_3$. θ_2^* is still a local minimum of $E_{\alpha_3}(\theta)$, while θ_1^* is not.

3.1. Unconstrained and differentiable problems

We consider one of the simplest cases: unconstrained and differentiable LS problems. Given a local optimum $\boldsymbol{\theta}^*$, we can obtain first- and second-order derivatives of the α-expanded objective function $E_\alpha(\boldsymbol{\theta})$ at $\boldsymbol{\theta}^*$ as

$$\nabla E_\alpha(\boldsymbol{\theta}^*) = (1+\alpha)\mathbf{J}^\top(\boldsymbol{\theta}^*)(\mathbf{y} - \mathbf{f}(\boldsymbol{\theta}^*)) = \mathbf{0}, \tag{8}$$

$$\nabla^2 E_\alpha(\boldsymbol{\theta}^*) = \mathbf{J}^\top(\boldsymbol{\theta}^*)\mathbf{J}(\boldsymbol{\theta}^*) + (1+\alpha)\mathbf{S}(\boldsymbol{\theta}^*). \tag{9}$$

where J is a Jacobian matrix and $\mathbf{S}(\boldsymbol{\theta}^*)$ is

$$\mathbf{S}(\boldsymbol{\theta}^*) = \sum_i \nabla^2 f_i(\boldsymbol{\theta}^*)(y_i - f_i(\boldsymbol{\theta}^*)). \tag{10}$$

Eq. (8) means that $\boldsymbol{\theta}^*$ is always a stationary point of $E_\alpha(\boldsymbol{\theta})$. Therefore, $\boldsymbol{\theta}^*$ is a local minimum of $E_\alpha(\boldsymbol{\theta})$ if and only if $\nabla^2 E_\alpha(\boldsymbol{\theta})$ is a positive semi-definite (PSD) matrix. If $\mathbf{S}$ is not a PSD matrix, there is a $\alpha \geq 0$ which satisfies the fact that $\nabla^2 E_\alpha(\boldsymbol{\theta})$ is not a PSD matrix.

Fig. 2 shows examples of α-expanded objective functions. Residual expansion elevates the objective function around $\boldsymbol{\theta}^*$, and if α is sufficiently large then it ceases to be a local minimum.

One-dimensional quartic minimization: Here we consider a quartic minimization problem—in particular, one that can be formulated as an LS problem:

$$E(\theta) = \frac{1}{2}\left((y_1 - \theta^2)^2 + (y_2 - \theta)^2\right). \tag{11}$$

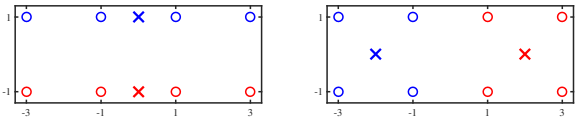
We consider the case where Eq. (11) has two local minima θ_1 and θ_2. The following theorem then holds:

Theorem 1. *Let θ_1 and θ_2 be local minima points of Eq. (11) with RE constants of α_1 and α_2, respectively. θ_1 is the global minimum point if $\alpha_1 > \alpha_2$ and θ_2 is the global minimum point otherwise.*

Proof. Please refer to the supplementary materials. □

3.2. General relationship between the α RE convergence and the global optimum

It is not obvious when our hypothesis, i.e., that a solution with a larger RE constant is closer to the global optimum, is valid. Unfortunately, we can easily find a counterexample



(a) K-means clustering result of a poor local minimum.

(b) K-means clustering result of a global minimum.

Fig 3: Different k-means clustering results with $k = 2$. The result (a) has an RE constant of $\alpha = \infty$; however, this is a poor local minimum. On the other hand, the result (b) has finite RE constant; however, this is a global minimum.

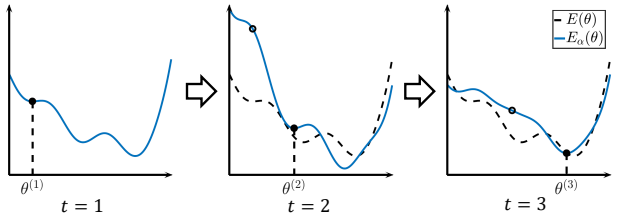


Fig 4: Conceptual view of the RE algorithm. The algorithm iterates parameter updating and residual expansion, which elevates the objective function for the current solution.

in k-means clustering, as shown in Fig. 3. However, our algorithm, which aims to find a solution with a large RE constant, works well from an empirical perspective in many nonconvex LS problems.

4. Residual expansion algorithm

In this section, we propose the RE algorithm, which aims to find a solution with a large RE constant. Since it is difficult to find the solution with the largest RE constant exactly, we employ a heuristic strategy.

The RE algorithm has two steps: parameter updating and residual expansion. We show an intuitive illustration of the algorithm in Fig. 4. For the residual expansion step, we expand data along their residual direction. This results in elevating the objective function around the current solution as in Fig. 2. For the parameter-updating step, instead of minimizing the original function Eq. (5), we minimize the following expanded objective function in each iteration:

$$E_t(\boldsymbol{\theta}) = \frac{1}{2}\|\hat{\mathbf{y}}^{(t)} - \mathbf{f}(\boldsymbol{\theta})\|_2^2, \tag{12}$$

where $\hat{\mathbf{y}}^{(t)}$ is an expanded data vector:

$$\hat{\mathbf{y}}^{(t)} = \mathbf{y} + \alpha^{(t)}\mathbf{r}^{(t)}, \tag{13}$$

$$\mathbf{r}^{(t)} = p^{(t)}(\mathbf{y} - \mathbf{f}(\boldsymbol{\theta}^{(t)})) + (1 - p^{(t)})\mathbf{r}^{(t-1)}. \tag{14}$$

where α and $0 < p \leq 1$ are expansion and momentum parameters, respectively. Note that, if $p = 1$, Eq. (12) is an exactly $\alpha^{(t)}$-expanded objective function on $\boldsymbol{\theta}^{(t)}$. The momentum parameter is important for achieving good performance and ensuring that the RE algorithm does not to diverge, as described later.

Algorithm 1 Residual expansion algorithm.

Input: Expansion parameter $\alpha^{(t)} \to 0$, momentum $p^{(t)}$.
Initialize: $t = 0, \hat{\mathbf{y}}^{(0)} = \mathbf{y}, \mathbf{r}^{(0)} = \mathbf{0}$
1: **while** not converged **do**
2: Update $\boldsymbol{\theta}$ by Eq. (12) (or Eq. (26))
3: $\mathbf{r}^{(t+1)} = p^{(t)}\left(\mathbf{y} - \mathbf{f}\left(\boldsymbol{\theta}^{(t+1)}\right)\right) + \left(1 - p^{(t)}\right)\mathbf{r}^{(t)}$
4: $\hat{\mathbf{y}}^{(t+1)} = \mathbf{y} + \alpha^{(t)}\mathbf{r}^{(t+1)}$
5: $t = t + 1$
6: **end while**
Output: $\boldsymbol{\theta}$

The RE algorithm iterates through parameter updating by minimizing Eq. (12) and residual expansions by Eq. (13) and Eq. (14). We use a large $\alpha^{(0)}$ initially to find a solution with a large RE constant. Then we decrease $\alpha^{(t)}$ gradually to achieve convergence, analogous to a temperature parameter in SA. We summarize the RE algorithm in Alg. 1.

4.1. Parameter setting for convergence

The RE algorithm has two parameters, α and p, for each iteration. We decrease $\alpha^{(t)}$ to 0 for convergence. when $\alpha = 0$, there is no residual expansion and RE algorithm is guaranteed to converge if the original LS problem has a convergence-guaranteed algorithm. However, inadequate parameters of α and p cause unstable optimization. We consider the norm of $\mathbf{r}^{(t+1)}$. We can obtain

$$\begin{aligned}\left\|\mathbf{r}^{(t+1)}\right\|_2^2 &= \left\|p\left(\mathbf{y} - \mathbf{f}\left(\boldsymbol{\theta}^{(t+1)}\right)\right) + (1-p)\,\mathbf{r}^{(t)}\right\|_2^2 \\ &\sim (1 - p - \alpha p)^2 \left\|\mathbf{r}^{(t)}\right\|_2^2. \end{aligned} \tag{15}$$

We use $\mathbf{f}\left(\boldsymbol{\theta}^{(t+1)}\right) \sim \hat{\mathbf{y}}^{(t)}$ for the last approximation of Eq. (15). Eq. (15) suggests $(1 - p - \alpha p)^2 \leq 1$ to make the RE algorithm stable. A good way to determine these values of α and p is described in Section 5.

4.2. Advantages of the RE algorithm

Our algorithm consists of two steps of parameter updating and residual expansion. Parameter updating is based simply on a typical LS problem approach. Therefore, if there is a source code which minimizes Eq. (5), for example, by alternative optimization or gradient methods, then we can implement our algorithm by adding a residual expansion step to the existing code, which can be done in a few lines of code.

Moreover, the computational complexity of residual expansion is generally less than that of parameter updating. Therefore, we can achieve faster optimization than most nonconvex optimization techniques based on multi-point search or random search, such as SA and GA.

As described in Section 2.2, GNC is a similar approach

to ours; however GNC often does not apply for LS problems. Our algorithm can be applied for any nonconvex LS problem provided that there is a method for finding a local optimum, such as Lloyd's algorithm for k-means clustering and ICP algorithms for point-set registration.

5. Alternate direction method of multipliers for least squares problems

In this section, we apply the alternate direction method of multipliers (ADMM) [6] to solve Eq. (5). We show that ADMM is a special case of the RE algorithm for LS problems. Moreover, ADMM suggests a modified RE algorithm for regularized LS problems.

We introduce an auxiliary variable $\mathbf{z} = \mathbf{y} - \mathbf{f}(\boldsymbol{\theta})$ and rewrite Eq. (5) as a constrained optimization problem:

$$\min_{\mathbf{z},\boldsymbol{\theta}} \frac{1}{2}\|\mathbf{z}\|_2^2 \quad \text{s.t. } \mathbf{z} = \mathbf{y} - \mathbf{f}(\boldsymbol{\theta}). \tag{16}$$

We can construct the augmented Lagrangian function of Eq. (16) as

$$L_t(\boldsymbol{\theta},\mathbf{z},\lambda) = \frac{1}{2}\|\mathbf{z}\|_2^2 + \lambda^\top(\mathbf{z}-\mathbf{y}+\mathbf{f}(\boldsymbol{\theta})) + \frac{\mu^{(t)}}{2}\|\mathbf{z}-\mathbf{y}+\mathbf{f}(\boldsymbol{\theta})\|_2^2. \tag{17}$$

We take the alternating direction approach for solving Eq. (17) and then obtain update rules as

$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\boldsymbol{\theta}} L_t(\boldsymbol{\theta}, \mathbf{z}^{(t)}, \lambda^{(t)}) \tag{18}$$

$$\mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z}} L_t(\boldsymbol{\theta}^{(t+1)}, \mathbf{z}, \boldsymbol{\lambda}^{(t)}) \tag{19}$$

$$\boldsymbol{\lambda}^{(t+1)} = \boldsymbol{\lambda}^{(t)} + \mu^{(t)}\left(\mathbf{z}^{(t+1)} - \mathbf{y} + \mathbf{f}\left(\boldsymbol{\theta}\right)^{(t+1)}\right) \tag{20}$$

5.1. Relation to the RE algorithm

We can simplify Eq. (18), Eq. (19) and Eq. (20) as

$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\boldsymbol{\theta}} \frac{\mu^{(t)}}{2}\left\|\mathbf{y} + \left(\frac{1-\mu^{(t)}}{\mu^{(t)}}\right)\mathbf{z}^{(t)} - \mathbf{f}(\boldsymbol{\theta})\right\|_2^2, \tag{21}$$

$$\mathbf{z}^{(t+1)} = \left(\frac{1}{1+\mu^{(t)}}\right)\mathbf{z}^{(t)} + \left(\frac{\mu^{(t)}}{1+\mu^{(t)}}\right)\left(\mathbf{y} - \mathbf{f}\left(\boldsymbol{\theta}^{(t+1)}\right)\right). \tag{22}$$

Details of the derivation are described in the supplementary material. This is a special case of the RE algorithm of Eq. (13) and Eq. (14) with

$$\alpha^{(t)} = (1-\mu^{(t)})/\mu^{(t)}, \tag{23}$$

$$p^{(t)} = \mu^{(t)}/(1+\mu^{(t)}). \tag{24}$$

There are two main advantages to using ADMM. First, we can choose only μ instead of parameters α and p in the general RE algorithm. Eq. (23) and Eq. (24) always satisfy $(1-p-\alpha p)^2 < 1$, which is a condition necessary for avoiding divergence to infinity, as described in Section 4.1, and this update achieves good performance in experiments. Second, ADMM can treat regularized LS optimization problems, such as blind image deblurring (Eq. (4)). We will describe this in the next section.

Algorithm 2 RE algorithm based on ADMM.

Input: Penalty parameter $\mu^{(t)} \to 1$.
Initialize: $t = 0, \hat{\mathbf{y}}^{(0)} = \mathbf{y}, \mathbf{r}^{(0)} = \mathbf{0}$.
1: **while** not converged **do**
2: Update $\boldsymbol{\theta}$ by Eq. (27).
3: $\mathbf{r}^{(t+1)} = \left(\frac{\mu^{(t)}}{1+\mu^{(t)}}\right)\left(\mathbf{y} - \mathbf{f}\left(\boldsymbol{\theta}^{(t+1)}\right)\right) + \left(\frac{1}{1+\mu^{(t)}}\right)\mathbf{r}^{(t)}$
4: $\hat{\mathbf{y}}^{(t+1)} = \mathbf{y} + \left(\frac{1-\mu^{(t)}}{\mu^{(t)}}\right)\mathbf{r}^{(t+1)}$
5: $t = t + 1$
6: **end while**
Output: $\boldsymbol{\theta}$

5.2. Regularized least squares problems

We consider a regularized LS problem as follows:

$$E(\boldsymbol{\theta}) = \frac{1}{2}\|\mathbf{y} - \mathbf{f}(\boldsymbol{\theta})\|_2^2 + \gamma R(\boldsymbol{\theta}). \tag{25}$$

When we apply the RE algorithm in a straightforward manner, we can obtain the following objective function in each iteration:

$$E_t(\boldsymbol{\theta}) = \frac{1}{2}\|\hat{\mathbf{y}}^{(t)} - \mathbf{f}(\boldsymbol{\theta})\|_2^2 + \gamma R(\boldsymbol{\theta}). \tag{26}$$

In the case of ADMM, from Eq. (21), the objective function is as follows:

$$E_t(\boldsymbol{\theta}) = \frac{\mu^{(t)}}{2}\|\hat{\mathbf{y}}^{(t)} - \mathbf{f}(\boldsymbol{\theta})\|_2^2 + \gamma R(\boldsymbol{\theta}). \tag{27}$$

We can find that the difference between Eq. (26) and Eq. (27) is simply the coefficient of the squared term. We find that minimizing Eq. (27) achieves better performance than minimizing Eq. (26). We summarize the RE algorithm based on ADMM in Alg. 2.

6. Implementation details

We used the RE algorithm based on ADMM (Alg. 2) unless otherwise stated. In the update of $\boldsymbol{\theta}$ (line 2 in Alg. 2), we perform alternating optimization with a single iteration; for example, with k-means clustering, the cluster centers and assignments are updated only once. The four problems we treat in this paper can be minimized by alternating optimization.

For the parameter μ, we adapt $\mu^{(t+1)} = \min(\rho\mu^{(t)}, 1)$, where $\rho = \exp(-\log(\mu^{(0)})/T)$ to satisfy $\mu^{(T)} = 1$. Therefore, we only need to determine the two parameters $\mu^{(0)}$ and T in our method.

7. Experimental results

We evaluate the performance of the RE algorithm on four nonconvex LS problems: k-means clustering, 3D point set registration, OPQ, and single blind image deblurring. All experiments were executed on an Intel Core i5-4200U CPU (1.60 GHz) with 8GB of RAM, and were implemented

Table 1: Clustering results on synthetic data. Mean relative errors of the RE algorithm with different $\mu^{(0)}$ and N are reported.

(a) Synthetic data A with $k = 100$.

	$\mu^{(0)} = 0.5$	$\mu^{(0)} = 0.2$	$\mu^{(0)} = 0.1$	$\mu^{(0)} = 0.01$
$T = 30$	0.905	0.894	0.902	0.921
$T = 100$	0.854	0.856	0.862	0.876
$T = 300$	0.843	0.844	0.846	0.854
$T = 1,000$	0.837	0.839	0.840	0.843

(b) Synthetic data B with $k = 10$.

	$\mu^{(0)} = 0.5$	$\mu^{(0)} = 0.2$	$\mu^{(0)} = 0.1$	$\mu^{(0)} = 0.01$
$T = 30$	2.699	1.758	1.493	0.998
$T = 100$	2.784	1.209	0.789	0.630
$T = 300$	2.722	1.036	0.708	0.552
$T = 1,000$	2.749	0.994	0.630	0.552

Table 2: Clustering results on synthetic data. The mean / min / max relative errors and the average elapsed time are reported.

(a) Synthetic data A with $k = 100$.

		Relative error			Elapsed time [sec]
		Mean	Min	Max	
Random seeding		1.246	1.102	1.473	0.058
k-means++ [2]		1.000	0.944	1.081	0.244
Hartigan's algorithm [10]		0.925	0.881	0.981	0.359
RE algorithm ($\mu^{(0)} = 0.1$)	$T = 30$	0.902	0.875	0.942	0.258
	$T = 100$	0.862	0.846	0.873	0.780
	$T = 300$	0.846	0.836	0.856	2.29
	$T = 1,000$	0.840	0.831	0.850	7.61

(b) Synthetic data B with $k = 10$.

		relative error			elapsed time [sec]
		mean	min	max	
Random seeding		4.277	0.552	22.680	0.0194
k-means++ [2]		1.000	0.552	8.743	0.0271
Hartigan's algorithm [10]		1.000	0.552	8.743	0.0429
RE algorithm ($\mu^{(0)} = 0.1$)	$T = 30$	1.493	0.552	6.777	0.0699
	$T = 100$	0.789	0.552	2.500	0.176
	$T = 300$	0.708	0.552	2.500	0.473
	$T = 1,000$	0.630	0.552	2.500	1.51

in MATLAB[2] except for Go-ICP [29]. For Go-ICP and its comparison experiment, we used the publicly available code[3] implemented in C++.

7.1. K-means clustering

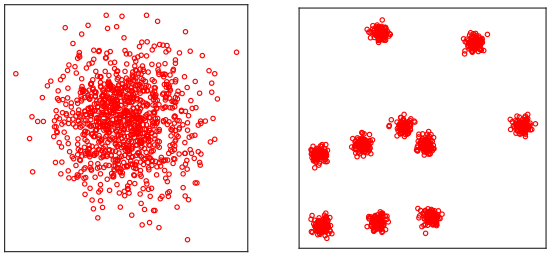
We compared our method with k-means++ [2], which is a good initialization method, and Hartigan's algorithm [10]. For Hartigan's algorithm, we first used Lloyd's algorithm [17] with k-means++ initialization for fast computation. We reported the total time of Lloyd's algorithm and Hartigan's algorithm. For the other method, we used Lloyd's algorithm for optimization. We used random initialization for the RE algorithm. For error measurement, we used the objective function value of Eq. (1) and reported relative error from the value of k-means++ (therefore, the relative error of k-means++ is always 1).

7.1.1 Synthetic data experiments

We start with two synthetic datasets as shown in Fig. 5. We repeated each method 50 times from different initializations and report the average relative errors. Table 1 shows the results of our method with different $\mu^{(0)}$ and T. We found that larger T achieved better performance. We also found that smaller $\mu^{(0)}$ achieved better performance in dataset B; however, larger $\mu^{(0)}$ achieved better performance in dataset A. This indicates that the best setting $\mu^{(0)}$ is different for different data distributions. Intuitively, dataset B requires a larger residual expansion (in other words, small $\mu^{(0)}$) to escape from a poor local minimum, while dataset A requires a smaller residual expansion.

[2]Our codes will be available if the paper is accepted.

[3]http://iitlab.bit.edu.cn/mcislab/~yangjiaolong/go-icp/



(a) Synthetic dataset A. (b) Synthetic dataset B.

Fig 5: Synthetic data (1,000 two-dimensional points).

We show comparison results in Table 2. We repeated each method 50 times from different initializations. K-means++ worked well with dataset B. On the other hand, Hartigan's algorithm can improve the results of k-means++ in dataset A; however, this does not work in dataset B. The RE algorithm worked best in both cases, even though it was initialized by random seeding. Moreover, the RE algorithm with $T = 30$ achieved comparable speed to k-means++ with better performance for dataset A.

7.1.2 Real-world data experiments

We used two real-world datasets for comparison: the cloud dataset[4] and the COIL20 dataset [21]. We performed experiments in the same manner as in Section 7.1.1.

Table 3 shows comparative results. In the cloud dataset, k-means++ achieves faster and better clustering than random seeding. The RE algorithm with $T = 30$ achieved better clustering than k-means++ with about 1.8 times the computational cost. The RE algorithm with $T = 1000$ per-

[4]https://archive.ics.uci.edu/ml/datasets/Cloud

Table 3: Clustering results on real data. The mean / min / max of the relative error and the average elapsed time are reported.

(a) Cloud dataset ($\mathbf{X} \in \mathbf{R}^{10\times 1024}$) with $k = 10$.

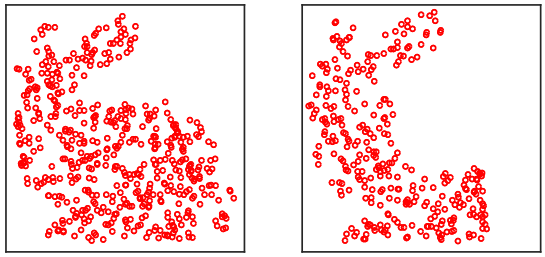
		Relative error			Elapsed time [sec]
		Mean	Min	Max	
Random seeding		1.255	1.003	1.438	0.0444
k-means++ [2]		1.000	0.920	1.097	0.0395
Hartigan's algorithm [10]		0.994	0.920	1.093	0.0593
RE algorithm ($\mu^{(0)} = 0.1$)	$T = 30$	0.980	0.920	1.031	0.0719
	$T = 100$	0.941	0.920	0.986	0.183
	$T = 300$	0.926	0.920	0.983	0.516
	$T = 1,000$	0.920	0.920	0.921	1.63

(b) COIL20 dataset ($\mathbf{X} \in \mathbf{R}^{1024\times 1440}$) with $k = 20$.

		Relative error			Elapsed time [sec]
		Mean	Min	Max	
Random seeding		0.999	0.953	1.076	2.22
k-means++ [2]		1.000	0.962	1.038	3.52
Hartigan's algorithm [10]		0.990	0.960	1.021	8.07
RE algorithm ($\mu^{(0)} = 0.1$)	$T = 30$	0.951	0.939	0.977	4.37
	$T = 100$	0.945	0.938	0.960	12.6
	$T = 300$	0.942	0.938	0.950	36.6
	$T = 1,000$	0.941	0.938	0.956	119

Table 4: Point set registration results. We reported the number of successes with each different rotation angle. We also reported the average elapsed time for all 150 point sets.

		Number of successes			Number of iterations
		$\phi = \pi/3$	$\phi = 5\pi/12$	$\phi = \pi/2$	
ICP algorithm		26	4	1	46.7
RE algorithm ($\mu^{(0)} = 0.1$)	$T = 30$	46	25	2	47.4
	$T = 100$	49	31	5	110.1
	$T = 300$	49	33	6	310.2
	$T = 1,000$	49	36	6	1008



(a) Source model (500 points). (b) Target model (313 points).

Fig 6: Point set models.

formed best, and found the near-global optimum in every case. For the COIL20 dataset, although k-means++ and Hartigan's algorithm did not work well, the RE algorithm significantly outperformed the other methods.

7.2. Point set registration

We compared the RE algorithm with the ICP algorithm and Go-ICP [29]. Go-ICP is known as a method that can achieve global optimization. We used the bunny model from the Stanford3D dataset[5], as in Fig. 6. For the target model, we used a partial point set as in Fig. 6(b). In the experiments, point sets were normalized within a cube of $[-1, 1]^3$.

We made a rotation matrix $\mathbf{R}_{gt}$ from a random rotation axis and the rotation angle ϕ. The target point set was constructed by this rotation matrix, and we added Gaussian noise with a standard deviation of $\sigma = 0.03$. We performed 50 tests with different random rotation axes at each rotation angle $\phi = \pi/3, 5\pi/12, \pi/2$. For measurement of the error, we used the objective value Eq. (2) and regarded the results as successful if the objective error was less than 1 (this value

[5]http://graphics.stanford.edu/data/3Dscanrep/

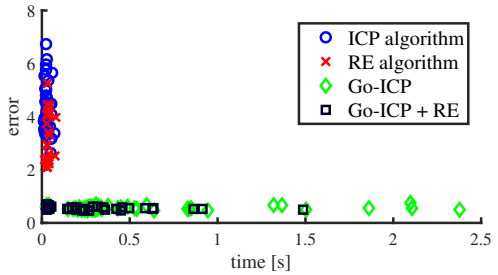


Fig 7: Point set registration results with $\phi = 5\pi/12$. We plotted the objective value and computation time over 50 trials. For the RE algorithm, we set $T = 30$. The average computational times for the ICP algorithm, RE algorithm, Go-ICP, and RE + Go-ICP were 0.0304, 0.0347, 0.551, and 0.231 seconds, respectively.

is approximately twice of the average objective value using Go-ICP, as in Fig. 7).

We first show the comparison results between the RE algorithm and the ICP algorithm as in Table 4. The RE algorithm with $T = 30$ achieved a better success rate with almost the same number of iterations as the ICP algorithm. Using a large T can improve the results to a small extent.

We also compare our method to Go-ICP [29]. Go-ICP has two steps: the ICP algorithm and the branch-and-bound algorithm. We compared the original Go-ICP and RE + Go-ICP, which has the two steps of ICP with the RE and branch-and-bound algorithms. Fig. 7 plots all 50 results in $\phi = 5\pi/12$. Note that this comparison was implemented entirely in C++. Go-ICP always found the global optimum solution; however, it required significant computation. RE + Go-ICP reduced computational cost while achieving global optimization.

7.3. Optimized product quantization

We show that the RE algorithm is successful in OPQ optimization problems. We compare our method with the alternating optimization method [9, 22]. For a dataset, we used SIFT 1M [11], which contains 100,000 128-dimensional SIFT descriptors for training. We set the subspace number $M = 8$ and cluster number $k = 256$, which are often used in the field of approximate nearest neighbor search. For error measurement, we used the objective func-

Table 5: Deblurring results. We reported PSNR values [dB] for each method. The best and second-best results are highlighted in bold and italics, respectively.

Image	#1				#2				#3				#4				#5			
Kernel	#1	#2	#3	#4	#1	#2	#3	#4	#1	#2	#3	#4	#1	#2	#3	#4	#1	#2	#3	#4
Pan *et al*. [23]	13.6	13.2	20.3	13.1	14.9	16.5	14.1	**11.2**	11.8	*20.6*	19.4	*10.0*	13.7	11.8	*19.1*	11.9	19.6	**19.2**	16.8	*14.1*
Alg. 1	**20.2**	**20.2**	**21.3**	**16.4**	*17.0*	*16.9*	**15.8**	7.4	**20.9**	18.8	*20.4*	6.5	**23.6**	*20.4*	**19.7**	**12.5**	*21.3*	*18.2*	*17.5*	9.4
Alg. 2	**20.2**	*19.6*	*20.5*	*15.7*	**17.2**	**17.1**	**15.8**	*8.4*	*19.5*	**21.5**	**20.7**	**14.2**	**23.6**	**20.5**	*19.1*	*11.9*	**21.9**	17.7	**18.6**	**15.2**

(a) Latent image. (b) Blurred image. (c) Result using [23]. (d) Result using Alg. 2.

Fig 9: An example of the results of deblurring (image #1, kernel #4).

tion value of Eq. (3). For our method, we set $\mu^{(0)} = 0.5$.

We plot the objective function value versus iteration number in Fig. 8. We performed five repetitions using different initializations and report the average objective values obtained. Our method improved the objective function value; moreover, it achieved rapid convergence in the cases of $T = 30$ and $T = 100$. The RE algorithm elevates the objective function around the current solution; in other words, it transforms the gradient for the current solution into a steeper gradient, potentially causing rapid convergence.

7.4. Blind image deblurring

We evaluated our method with single blind image deblurring. There are many formulations for blind image deblurring. In this paper, we followed Pan *et al*.'s formulation [23], which can be minimized by alternating optimization. We compared three methods as follows: a coarse-to-fine strategy [23] and RE algorithms based on Alg. 1 and Alg. 2. We used the uniform blurred text images from the dataset provided by Lai *et al*. [16], which contains five latent images and four blurring kernels for a total of 20 blurred images. For all methods, we used the same objective function parameters, such as the regularization coefficients. For our method, we set $\mu^{(0)} = 0.2$ and $T = 100$.

We show the results in Table 5. Our method significantly outperforms Pan's method [23] and is successful for a significantly blurred image, as in Fig. 9. We found that Alg. 2 is superior to Alg. 1 in the cases of {image #3, kernel #4} and {image #5, kernel #4}. Note that these results are obtained by minimizing the same objective function, however using different optimization methods. Therefore our method likely improves upon other methods which use different objective functions [15, 28].

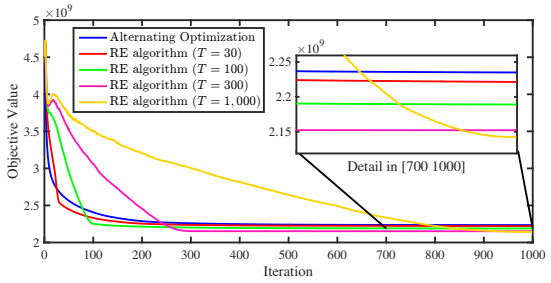


Fig 8: Objective value of Eq. (3) versus iteration number of OPQ optimization. We report average results over five different initializations.

8. Conclusion

We proposed the RE algorithm, which is a novel global optimization algorithm for nonconvex LS problems. This method is based on a novel measurement of global convergence called RE convergence. We presented theoretical analysis of RE convergence and empirical results showing excellent performance of the RE algorithm for various optimization problems.

There remain many open questions in both theoretical and empirical aspects. We can guarantee that the solution with the largest RE constant is the global optimum in limited cases. To which problems this applies remains unknown. We plan to investigate the applicability of the RE algorithm to other nonconvex optimization problems, including non-LS problems.

Acknowledgement

This research is partially supportted by CREST (JPMJCR1686) and KAKENHI15K12025

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 34(11):2274–2282, 2012.

[2] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *SODA*, pages 1027–1035, 2007.

[3] P. J. Besl and H. D. McKay. A method for registration of 3-D shapes. *TPAMI*, 14(2):239–256, 1992.

[4] G. Blais and M. D. Levine. Registering multiview range data to create 3D computer objects. *TPAMI*, 17(8):820–824, 1995.

[5] A. Blake and A. Zisserman. *Visual reconstruction*. MIT Press, 1987.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

[7] S. Cho and S. Lee. Fast motion deblurring. *ACM ToG*, 28(5):145:1–145:8, 2009.

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[9] T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization for approximate nearest neighbor search. In *CVPR*, pages 2946–2953, 2013.

[10] J. A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[11] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *TPAMI*, 33(1):117–128, 2011.

[12] J. Kennedy and R. Eberhart. Particle swarm optimization. In *ICNN*, volume 4, pages 1942–1948, 1995.

[13] S. Kirkpatrick, M. P. Vecchi, et al. Optimization by simmulated annealing. *science*, 220(4598):671–680, 1983.

[14] K. Krishna and M. N. Murty. Genetic K-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439, 1999.

[15] D. Krishnan, T. Tay, and R. Fergus. Blind deconvolution using a normalized sparsity measure. In *CVPR*, pages 233–240, 2011.

[16] W.-S. Lai, J.-B. Huang, Z. Hu, N. Ahuja, and M.-H. Yang. A comparative study for single image blind deblurring. In *CVPR*, June 2016.

[17] S. Lloyd. Least squares quantization in PCM. *TIT*, 28(2):129–137, 1982.

[18] J. Luck, C. Little, and W. Hoff. Registration of range data using a hybrid simulated annealing and iterative closest point algorithm. In *ICRA*, volume 4, pages 3739–3744, 2000.

[19] M. Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[20] H. Mobahi and J. W. Fisher III. On the link between gaussian homotopy continuation and convex envelopes. In *International Workshop on Energy Minimization Methods in CVPR*, pages 43–56, 2015.

[21] S. A. Nene, S. K. Nayar, H. Murase, et al. Columbia object image library (COIL-20). Technical report.

[22] M. Norouzi and D. J. Fleet. Cartesian k-means. In *CVPR*, pages 3017–3024, 2013.

[23] J. Pan, Z. Hu, Z. Su, and M.-H. Yang. Deblurring text images via l0-regularized intensity and gradient prior. In *CVPR*, pages 2901–2908, 2014.

[24] L. Silva, O. R. P. Bellon, and K. L. Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *TPAMI*, 27(5):762–776, 2005.

[25] N. Slonim, E. Aharoni, and K. Crammer. Hartigan's k-means versus Lloyd's k-means: Is it time for a change? In *IJCAI*, pages 1677–1684, 2013.

[26] M. Telgarsky and A. Vattani. Hartigan's method: k-means clustering without Voronoi. In *AISTATS*, pages 820–827, 2010.

[27] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby. An approach to multimodal biomedical image registration utilizing particle swarm optimization. *TEVC*, 8(3):289–301, 2004.

[28] L. Xu, S. Zheng, and J. Jia. Unnatural L0 sparse representation for natural image deblurring. In *CVPR*, pages 1107–1114, 2013.

[29] J. Yang, H. Li, and Y. Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In *ICCV*, pages 1457–1464, 2013.