BREGMAN MONOTONE OPERATOR SPLITTING
KENTA NIWA * AND W. BASTIAAN KLEIJN †

Abstract. Monotone operator splitting is a powerful paradigm that facilitates parallel processing for optimization problems where the cost function can be split into two convex functions. We propose a generalized form of monotone operator splitting based on Bregman divergence. We show that an appropriate design of the Bregman divergence leads to faster convergence than conventional splitting algorithms. The proposed Bregman monotone operator splitting (B-MOS) is applied to an application to illustrate its effectiveness. B-MOS was found to significantly improve the convergence rate.

Key words. Monotone operator splitting (MOS), Bregman divergence, Newton method, accelerated gradient descent (AGD)

AMS subject classifications. 90C25, 90C20, 68Q32

1. Introduction. Mathematical optimization is commonly used in a wide range of applications including image classification, speech recognition, and natural language processing. In recent years, the performance of optimization algorithms has improved drastically through the use of big data and large computing resources. Although we cannot provide a detailed overview of optimization theory because of the breadth of the field, we can provide a rough categorization on the basis of the three perspectives used in [1].

The first perspective is that of the problem formulation. Different formulations of the problem are often possible and lead to different solution methods. Optimization problems are commonly formulated as a cost minimization subject to a set of constraints. Linearly constrained minimization forms are particularly ubiquitous. Moreover, a dual formulation [2] forms an alternative that makes the optimization problem more tractable. A famous example dual formulation is the Lagrangian dual ascent problem [3], which is described in Sec. 2.1. For certain optimization problems, the cost function can be formulated as a summation of components. This is the problem formulation used in monotone operator splitting (MOS) e.g., [4, 5].

The second perspective is that of the solver method. In constructing a solver that applies to the defined problem, its convergence rate is an important factor. If we use a deterministic solution method, possible approaches are first order gradient descent (GD) [7], accelerated gradient descent (AGD) [8], and the Newton method and quasi-Newton method [9]. To handle large amounts of data, minibatch based stochastic optimization was introduced [10]. A method to estimate the convergence rate of stochastic optimization was provided in [11]. Recent algorithms, e.g., [12], [13], [14], [15], [16], are often combinations of a particular problem form and a particular solver. For example, stochastic dual coordinate ascent (SDCA) [17] applies first order stochastic gradient descent (SGD) to a risk minimization problem [18], whereas stochastic dual Newton ascent (SDNA) [19] applies second order SGD. If the cost function admits a formulation as a sum of two suitable convex functions, then this naturally leads to MOS approaches, e.g., [4, 5]. The alternating direction method of multipliers (ADMM) [20] is an example of MOS that applies Douglas-Rachford splitting [21] to the Lagrangian dual ascent problem.

The third perspective is how to run the solver effectively over multiple processors.
Recent progress on parallel computing architectures in the context of cloud computing and graphics processing unit (GPU) clusters has resulted in a large research effort towards running solvers in parallel on many processing nodes, usually for big data. Numerous parallel computing methods have been proposed. Pioneering methods include parallelized SGD [22], [23], the hogwild! algorithm [24], elastic averaging SGD [25], and communication-efficient coordinate ascent (COCOA) [26], which is a parallelization form of SDCA. The class of MOS based methods is particularly attractive for running over multiple processors as MOS naturally facilitates parallel computation. Many parallel algorithms based on MOS are variants of ADMM [27], [28]. Although ADMM is effective, its convergence rate is often relatively slow because it is based on Douglas-Rachford splitting. The primal-dual method of multipliers (PDMM) [29], [30] inherently converges faster as it is based on Peaceman-Rachford splitting [31].

In this paper, we focus on MOS methods. Several MOS solvers are well known, such as Peaceman-Rachford splitting [31], Douglas-Rachford splitting [21], forward-backward splitting [32] and Davis-Yin three-operator splitting [33]. Their variable update procedures are basically composed of operators such as the resolvent and Cayley operators as summarized in [4, 5]. The convergence rates of MOS solvers generally follow from the contractive property of the aforementioned operators. Penalty terms based on the $L_2$ norm are included in the variable update cost. This fact indicates that the variables are updated with a pre-determined step-size in a Euclidean metric and its convergence rate corresponds to that of first order GD. This suggests that we may obtain a faster convergence rate with MOS based solvers that correspond to, for example, Newton or accelerated gradient descent (AGD).

Our contribution is a generalization of MOS solvers to using Bregman divergence [34]. We aim at obtaining faster convergence rates in MOS solvers on the basis of a principle that similar to Newton or AGD methods in a conventional context. The Bregman divergence generalizes the Euclidean metric. Although the resolvent operator has been generalized using the Bregman divergence [35], the generalization of MOS solvers to include an appropriate Bregman divergence design has not been studied thus-far. We show that if the MOS solvers are generalized and the Bregman divergence is designed appropriately, then a faster convergence rate can be obtained. As discussed in the theory of Newton, AGD, and GD [36], their convergence rate differences are basically due to the eigenvalue range of a second order gradient (Hessian) of a convex cost. The eigenvalues should approach 1 for fast convergence. To manipulate the eigenvalue range, an approximation of the cost by a quadratic function is effective for Bregman divergence design. By means of a convergence rate analysis and numerical experiments, we show how the convergence rates are affected by the design of the Bregman divergence.

In Sec. 2, the theory of the new Bregman MOS (B-MOS) is described in the context of deterministic optimization. After Bregman MOS algorithms are constructed, their convergence rates are predicted. We explain how to design Bregman divergence to achieve fast convergence. In Sec. 3, the B-MOS solver is applied to a constrained minimization problem to illustrate the effectiveness of B-MOS as an example.

2. THEORY. In this section, we generalize the conventional monotone operator splitting algorithms to use Bregman divergence. The resulting formulation has additional degrees of freedom that can be optimized to minimize bounds on convergence rate. After defining the problem, we derive the basic algorithms in Sec. 2.2. The optimization of the Bregman divergence for fast convergence is discussed in Sec. 2.3.
We investigate the convergence rates for a set of algorithms in Sec. 2.4.

2.1. Problem Definition. We consider the problem of finding an infimum of a convex closed proper (CCP) function that can be split into two CCP functions as

\[ G(w) = G_1(w) + G_2(w), \]

where \( w \in \mathbb{R}^m \) is the latent variable to be optimized and \( G_i : \mathbb{R}^m \rightarrow \mathbb{R} \cup \{\infty\} \), \( i = 1, 2 \).

A fixed point \( w^* \) can be found by requiring that the subdifferential of (2.1) includes the zero vector,

\[ 0 \in \partial G_1(w) + \partial G_2(w), \]

where \( \partial \) is the subdifferential operator [37], and \( \in \) reflects that its output can be multi-valued. As each \( G_i(w) \) is a CCP function, \( \partial G_i(w) \) is maximally monotone [38].

To show that various relevant problems are of the form (2.1), we provide two examples.

Ex. 1: Constrained minimization problem

Let us suppose the constrained minimization problem is composed of a CCP function \( F_1 : \mathbb{R}^m \rightarrow \mathbb{R} \cup \{\infty\} \) and a CCP penalty term \( F_2 : \mathbb{R}^m \rightarrow \mathbb{R} \cup \{\infty\} \) as

\[ \inf_{p} F_1(p) \quad \text{s.t.} \quad F_2(p) \leq 0, \]

where \( p \in \mathbb{R}^m \) is the latent variable. It is well known that solving the problem (2.3) is equivalent to solving the following problem [39]:

\[ \inf_{p} F_1(p) + \mu F_2(p), \]

where \( \mu > 0 \). By replacing \( p \rightarrow w \), and setting \( G_1(w) = F_1(w), G_2(w) = \mu F_2(w) \), the constrained minimization problem (2.4) is of the form (2.1).

Ex. 2: Lagrangian dual ascent problem

For another constrained minimization problem, let us suppose the cost function to be minimized is composed of two CCP functions, \( F_1 : \mathbb{R}^p \rightarrow \mathbb{R} \cup \{\infty\} \) and \( F_2 : \mathbb{R}^q \rightarrow \mathbb{R} \cup \{\infty\} \), and let the variables be linearly constrained as

\[ \inf_{p,q} F_1(p) + F_2(q) \quad \text{s.t.} \quad Ap + Bq = c, \]

where the variables are \( p \in \mathbb{R}^p, q \in \mathbb{R}^q \) and where \( A \in \mathbb{R}^{m \times p}, B \in \mathbb{R}^{m \times q}, c \in \mathbb{R}^m \) specify the linear constraint parameters that relate the variables. Given the Lagrangian function \( \mathcal{L} \)

\[ \mathcal{L} = F_1(p) + F_2(q) + \langle \lambda, c - Ap - Bq \rangle. \]

When the dual problem exists [2], solving it instead of the primal problem is a natural strategy. The dual problem takes the form:

\[ \sup_{\lambda} \inf_{p,q} \mathcal{L} = \sup_{\lambda} \left( -F_1^*(A^T\lambda) - F_2^*(B^T\lambda) + \langle \lambda, c \rangle \right) \]

\[ = -\inf_{\lambda} \left( F_1^*(A^T\lambda) + F_2^*(B^T\lambda) - \langle \lambda, c \rangle \right), \]
where \( \lambda \in \mathbb{R}^m \) is a dual variable, \( ^T \) denotes the transposition, and \( F_i^* : \mathbb{R}^m \to \mathbb{R} \cup \{\infty\} \) is the convex conjugate (the Legendre transformation for the scalar case) of \( F_i \) [2] as

\[
F_i^*(A^T\lambda) = \sup_p (\langle \lambda, Ap \rangle - F_i(p)),
\]

(2.8)

\[
F_i^*(B^T\lambda) = \sup_q (\langle \lambda, Bq \rangle - F_i(q)).
\]

(2.9)

By replacing \( \lambda \to w \), and setting \( G_1(w) = F_i^*(A^Tw), G_2(w) = F_i^*(B^Tw) - \langle \lambda, c \rangle \), the Lagrangian dual ascent problem, which reformulates (2.7a) into the minimization (2.7b), is of the form (2.1).

### 2.2. Bregman Monotone Operator Splitting (B-MOS)

In this section, we generalize MOS solvers with the aim to obtain faster convergence. MOS methods have been studied as solvers for the problem (2.2) and are summarized well in [4, 5]. In the present paper, we focus on three well-known monotone operator splitting methods: namely Peaceman-Rachford (P-R) splitting [31], Douglas-Rachford (D-R) splitting [21], and forward-backward (F-B) splitting [32].

We generalize the conventional Euclidean distance metric used in MOS methods to the Bregman divergence (B-MOS). The motivation for the generalization is that our Bregman divergence based approach can be used to obtain significantly faster convergence rates than conventional Euclidean distance based solvers. (We will explain how to obtain fast convergence rates on the basis of Bregman divergence in Sec. 2.4.)

The Bregman divergence of a first point \( w \in \mathbb{R}^m \) and a second point \( z \in \mathbb{R}^m \) [34] is defined as

\[
B_D(w \parallel z) = D(w) - D(z) - \langle \nabla D(z), w - z \rangle,
\]

(2.10)

where \( D \) is any continuously differentiable strictly convex function, e.g., [40], and \( \nabla D \) denotes the gradient of \( D \). Note that for \( D(w) = \frac{1}{2\kappa} \langle w, w \rangle \) (\( \kappa > 0 \)) the Bregman divergence reduces to the Euclidean distance.

Associated with the Bregman divergence, we introduce the \( D \)-resolvent operator \( R_i \) (i = 1, 2) and the \( D \)-Cayley operator \( C_i \) (i = 1, 2) as

\[
R_i = (\nabla D + \partial G_i)^{-1}\nabla D = (I + \nabla D^{-1}\partial G_i)^{-1},
\]

(2.11)

\[
C_i = 2R_i - I
= 2(I + \nabla D^{-1}\partial G_i)^{-1} - I
= 2(I + \nabla D^{-1}\partial G_i)^{-1} - (I + \nabla D^{-1}\partial G_i)(I + \nabla D^{-1}\partial G_i)^{-1}
= (I - \nabla D^{-1}\partial G_i)(I + \nabla D^{-1}\partial G_i)^{-1}
\]

(2.12a)

\[
= (I - \nabla D^{-1}\partial G_i)R_i.
\]

(2.12b)

When the Bregman divergence is based on the Euclidean distance, i.e., \( D(w) = \frac{1}{2\kappa} \langle w, w \rangle \) and then \( \nabla D = \frac{1}{\kappa} I \), \( \nabla D^{-1} = \kappa I \), then the well-known (Euclidean) resolvent operator is obtained because \( R_i = (I + \kappa\partial G_i)^{-1} \), and the metric of \( C_i \) is then also Euclidean (e.g., [4]). The properties of the \( D \)-resolvent and \( D \)-Cayley operator are investigated in more detail in Appendix A.

We first derive Peaceman-Rachford splitting generalized using Bregman divergence (Bregman Peaceman-Rachford splitting) by reformulating the fixed-point condition (2.2):
\[ 0 \in \nabla D^{-1} \partial G_2(w) + \nabla D^{-1} \partial G_1(w), \]

\[ 0 \in (I + \nabla D^{-1} \partial G_2(w) - (I - \nabla D^{-1} \partial G_1)(w), \]

where \( I \) and \(-1\) are the identity operator and the inverse operator, respectively. Since \( D \) is a strictly convex function, \( \nabla D \) and its inverse \( \nabla D^{-1} \) are monotone operators that have a unique relation between input and output vectors. By setting \( w \in R_1(z) \), the fixed point condition (2.13) can be written as

\[ 0 \in (I + \nabla D^{-1} \partial G_2)R_1(z) - (I - \nabla D^{-1} \partial G_1)R_1(z), \]

\[ 0 \in R_1(z) - R_2C_1(z), \]

\[ 0 \in \frac{1}{2}(C_1 + I)(z) - \frac{1}{2}(C_2 + I)C_1(z). \]

Hence, we obtain the condition for a fixed point

\[ z \in C_2C_1(z). \]

**Appendix A** shows that the \( D \)-Cayley operator \( C_i \) is nonexpansive under relevant conditions. Hence, the iterative application of (2.14) generates a Cauchy sequence, and the iterations follow Banach-Picard fixed-point iterations, e.g., [41]. The iteration specified by (2.14) can be decomposed into simpler steps by introducing additional auxiliary variables \( x \in \mathbb{R}^m \) and \( y \in \mathbb{R}^m \):

\[ w^{t+1} = R_1(z') = (I + \nabla D^{-1} \partial G_1)^{-1}(z'), \]

\[ x^{t+1} = C_1(z') = (2R_1 - I)(z') = 2w^{t+1} - z', \]

\[ y^{t+1} = R_2(x^{t+1}) = (I + \nabla D^{-1} \partial G_2)^{-1}(x^{t+1}), \]

\[ z^{t+1} = C_2(x^{t+1}) = (2R_2 - I)(x^{t+1}) = 2y^{t+1} - x^{t+1}, \]

where (2.15) corresponds to the Bregman proximal point algorithm, e.g., [42]. This can be seen by first writing

\[ w \in R_1(z), \]

\[ w \in (I + \nabla D^{-1} \partial G_i)^{-1}(z), \]

\[ (I + \nabla D^{-1} \partial G_1)(w) \in z, \]

\[ 0 \in \nabla D^{-1} \partial G_1(w) + w - z, \]

\[ 0 \in \partial G_1(w) + \nabla D(w) - \nabla D(z). \]

Assuming the minimum exists, then the integral of (2.19) gives

\[ w^{t+1} = \arg\min_w \left( \| G_1(w) + B_D(w \mid z') \right). \]

From (2.20), we see that the metric of the cost function is generalized by using the Bregman divergence. By using (2.12a), the variable update using the \( D \)-Cayley operator can be obtained with (2.16). However, to show that the update cost is based on the Bregman divergence, we rewrite it with another formulation. By using (2.12b), the update procedure from \( w \in R_1(z) \) to \( x \in C_1(z) \) can be reformulated as

\[ x \in (I - \nabla D^{-1} \partial G_1)(w), \]

\[ 0 \in x - w + \nabla D^{-1} \partial G_1(w), \]

\[ 0 \in \nabla D(x) - \nabla D(w) + \partial G_1(w). \]
Algorithm 2.1 Bregman Peaceman-Rachford Splitting

Initialization of $z^0$

for $t = 0, \ldots, T - 1$
do
  $w^{t+1} = \arg \min_w (G_1(w) + B_D(w \| z^t))$
  $x^{t+1} = 2w^{t+1} - z^t$
  $y^{t+1} = \arg \min_y (G_2(y) + B_D(y \| x^t))$
  $z^{t+1} = 2y^{t+1} - x^{t+1}$
end for

Algorithm 2.2 Bregman Douglas-Rachford Splitting

Initialization of $z^0$

for $t = 0, \ldots, T - 1$
do
  $w^{t+1} = \arg \min_w (G_1(w) + B_D(w \| z^t))$
  $x^{t+1} = 2w^{t+1} - z^t$
  $y^{t+1} = \arg \min_y (G_2(y) + B_D(y \| x^t))$
  $z^{t+1} = \alpha(2y^{t+1} - x^{t+1}) + (1 - \alpha)z^t$
end for

Assuming that the minimum exists, then the integral of (2.21) gives

\[(2.22) \quad x^{t+1} = \arg \min_x (G_1(w^t) + B_D(x \| w^t)) \]

(2.22) also shows that the cost metric is generalized to a Bregman divergence. However, since the vector update with this procedure gives the same result as (2.16), we use the simple form (2.16) for the implementation of the $D$-Cayley operator hereafter. The resulting Bregman Peaceman-Rachford splitting algorithm is summarized in Algorithm 2.1.

Bregman Douglas-Rachford splitting, a generalization of Douglas-Rachford splitting, is obtained by introducing the averaging operator (e.g. [4]) into (2.14):

\[(2.23) \quad z \in \alpha C_2 C_1(z) + (1 - \alpha)z, \]

where $\alpha \in (0, 1)$. (2.23) can be decomposed into (2.15)-(2.17), augmented by

\[(2.24) \quad z^{t+1} = \alpha(2y^{t+1} - x^{t+1}) + (1 - \alpha)z^t. \]

The Bregman Douglas-Rachford splitting algorithm is summarized in Algorithm 2.2.

For another reformulation of (2.2), we derive forward-backward splitting generalized using the Bregman divergence (Bregman forward-backward splitting):

\[(2.25) \quad 0 \in \nabla D^{-1} \partial G_2(w) + \nabla D^{-1} \partial G_1(w), \quad 0 \in (I + \nabla D^{-1} \partial G_2)(w) - (I - \nabla D^{-1} \partial G_1)(w), \quad (I + \nabla D^{-1} \partial G_2)(w) \in (I - \nabla D^{-1} \partial G_1)(w), \quad w \in (I + \nabla D^{-1} \partial G_2)^{-1}(I - \nabla D^{-1} \partial G_1)(w), \quad w \in R_2(I - \nabla D^{-1} \partial G_1)(w). \]

Therefore, the update procedure is given by

\[(2.26) \quad x^{t+1} = (I - \nabla D^{-1} \partial G_1)(w^t), \quad (2.27) \quad w^{t+1} = R_2(x^{t+1}). \]
Algorithm 2.3 Bregman Forward-Backward Splitting

Algorithm 2.3 Bregman Forward-Backward Splitting

The Bregman forward-backward splitting algorithm is summarized in Algorithm 2.3.

In this subsection, several monotone operator splitting algorithms were generalized using the Bregman divergence. To exploit this generalization and make the algorithms converge faster, an appropriate $B_D(w \| z')$ must be selected. Preliminary steps for selecting an appropriate metric $B_D(w \| z')$ are provided in the next subsection.

2.3. Bregman Divergence Design Using a Quadratic Function. We now introduce the main idea of how to design $B_D(w \| z')$ for fast convergence. As explained in Sec. 2.2, the metric of variable space was generalized by using Bregman divergence instead of the Euclidean distance used in the traditional MOS solvers. As is shown in Appendix A and similar to the theory of conventional Newton, AGD, and GD [36], the convergence rate basically depends on the eigenvalue range of the second order gradient (Hessian) of a convex cost. This property is also applicable to overall MOS solvers because the contractive ratios of $D$-resolvent and $D$-Cayley operators vary with the eigenvalue range. Thus, we aim to find methods to control the eigenvalue range through Bregman divergence design.

Although there may be various methods to design $B_D(w \| z')$, we will use a general quadratic function and write

$$B_D(w \| z') = \frac{1}{2} \langle M_D(z'), w - z' \rangle.$$  \hspace{1cm} (2.28)

where $M_D(z')$ is any positive definite matrix, i.e., $O \prec M_D(z') \prec +\infty$. Then $D$ is

$$D(w) = \frac{1}{2} \langle M_D(z')w, w \rangle.$$  \hspace{1cm} (2.29)

Note that $M_D(z')$ can vary with the input vector $z'$, allowing us to select the best quadratic metric around $z'$ for each update of the optimization algorithm.

To model the eigenvalue range of a convex cost Hessian even if $G_i$ is not strictly convex, we characterize it with an upper/lower bound pair that is represented by locally quadratic functions around the present variable $z'$ that satisfies

$$\| \partial G_i(w) \| - \| \partial G_i(z') \|_2 \leq \| M_{G_i}(z')(w - z') \|_2,$$  \hspace{1cm} (2.30)

$$\| \partial G_i(w) \| - \| \partial G_i(z') \|_2 \geq \| K_{G_i}(z')(w - z') \|_2.$$  \hspace{1cm} (2.31)

where $M_{G_i}(z')$ and $K_{G_i}(z')$ are, respectively, a positive definite matrix and a semi-positive definite matrix, i.e., $O \preceq K_{G_i}(z') \prec M_{G_i}(z') \prec +\infty$. Since $K_{G_i}(z')$ is allowed to include zero, $G_i(z')$ does not need to be strongly convex (e.g. [37]) but is assumed to be Lipschitz continuous (e.g., [37]) to have a majorization function of the original cost as in (2.30). If $G_i$ is continuously differentiable, we can make the replacement $\partial G_i(z') = \nabla G_i(z')$ and $M_{G_i}(z') = \nabla^2 G_i(z') + \varepsilon I$, where $\varepsilon > 0$ is used to ensure the Lipschitz smoothness condition.

Now, we can model the eigenvalue range of the Hessian when we generalize the cost metric to Bregman divergence. Since convexity of the original cost was modeled

| Algorithm 2.3 Bregman Forward-Backward Splitting |
|--------------------------------------------------|
| Initialization of $w^0$                         |
| for $t = 0, \ldots, T - 1$ do                   |
| $x^{t+1} = w^t - \nabla D \partial G_i(w^t)$,   |
| $w^{t+1} = \arg\min_w \left( G_2(w) + B_D(w \| x^{t+1}) \right)$ |
| end for                                         |

The Bregman forward-backward splitting algorithm is summarized in Algorithm 2.3.
An advantage of this method is that, at least in principle, a learning rate does not whose diagonal elements are the same in it.

\[ \beta \]

where \( \Lambda \) whose elements vary adaptively. For the case of RMSProp, with

\[ M \]

where \( \Lambda \) and \( \Lambda \) are the maximum and minimum eigenvalues, respectively. The result of Appendix A shows that the optimal choice is that both \( \sigma_{\text{max}, i} \) and \( \sigma_{\text{min}, i} \) are 1.

In the following, we provide three Bregman divergence design methods. To obtain a scenario where both \( \sigma_{\text{max}, i} \) and \( \sigma_{\text{min}, i} \) approximate 1, we attempt to match \( M_D(z') \) with \( M_G(z') \). In addition, since our cost function is composed of two CCP functions as in (2.1), we distinguish two scenarios: i) designing \( M_D(z') \) such that it matches only \( M_G(z') \) (case 1) and ii) designing \( M_D(z') \) so as to match both \( M_G(z') \) and \( M_G(z') \) (case 2).

2.3.1. Newton. If quadratic assumption as seen in (2.30) and (2.31) is true or a good approximation in practice, the Newton method is an optimal approach for designing a full matrix \( M_D(z') \) so as to match \( M_G(z') \):

\[ M_{D}^{(\text{Newton})}(z') = \begin{cases} M_G(z') & \text{(case 1)} \\ (M_G(z')M_G(z'))^{1/2} & \text{(case 2)} \end{cases} \]

An advantage of this method is that, at least in principle, a learning rate does not need to be set. Although not discussed here, the usage of quasi-Newton methods such as BFGS [9] to approximate \( M_G(z') \) with a gradient of \( G_i \) falls in the same category.

2.3.2. Accelerated Gradient Descent (AGD). Although the fastest convergence is expected for the Newton method, the computational cost for calculating \( M_D(z') \) and its inverse may be high. To overcome this problem, accelerated gradient descent (AGD) methods can be used such as AdaGrad [43], Adam [44], and RMSProp [45]. In many of these methods, \( M_G(z') \) is approximated with a diagonal matrix whose elements vary adaptively. For the case of RMSProp, \( M_D(z') \) is represented by

\[ M_{D}^{(\text{AGD})}(z') = \begin{cases} L_G(z') & \text{(case 1)} \\ (L_G(z')L_G(z'))^{1/2} & \text{(case 2)} \end{cases} \]

where \( L_G(z') \) is assumed to be a diagonal matrix

\[ L_G(z') \leftarrow \beta L_G(z') + (1 - \beta) \text{Diag}(M_G(z')) \]

\[ L_G(z') \leftarrow \beta L_G(z') + (1 - \beta) \text{Diag}(M_G(z')) \]

where \( \beta \in (0, 1) \) is a moving average coefficient and \( \text{Diag}(\cdot) \) generates a diagonal matrix whose diagonal elements are the same in it.

2.3.3. Gradient Descent (GD). When \( M_G(z') \) is approximated as a scaled identical matrix, the Bregman divergence is identical to the Euclidean distance. For this case our generalized approach corresponds to the gradient descent (GD) method [7] with a specific learning step size,

\[ M_{D}^{(\text{GD})}(z') = \frac{1}{\kappa} I. \]
Although its computational cost in each update iteration is very low, this approach may not be included in the strategy to reduce the dynamic range of eigenvalues in \( M_G(z^t) \). This case corresponds to the usage of the traditional (Euclidean) resolvent and (Euclidean) Cayley operators (e.g. [4]) and hence conventional MOS solvers.

It is difficult to provide a model of the eigenvalue dynamic range differences for the three methods. It is reasonable to assume that, in general,

\[
1 \leq \frac{\sigma_{UB,i}^{(Newton)\ast}}{\sigma_{LB,i}^{(Newton)\ast}} \leq \frac{\sigma_{UB,i}^{(AGD)\ast}}{\sigma_{LB,i}^{(AGD)\ast}} \leq \frac{\sigma_{UB,i}^{(GD)\ast}}{\sigma_{LB,i}^{(GD)\ast}}.
\]

Moreover, it is reasonable to assume that \( \sigma_{UB,i}^{\ast} \) and \( \sigma_{LB,i}^{\ast} \) will be closest to 1 with the Newton method.

### 2.4. Convergence Rates on B-MOS Algorithms.

In this subsection, the convergence rates of B-MOS algorithms in Sec. 2.2 are derived. Since these algorithms are based on the \( D \)-resolvent and \( D \)-Cayley operators, B-MOS convergence rates depend strongly on their contractive ratio investigated in Appendix A. As discussed in Appendix A, the CCP function \( G_i(z) \) is assumed to be Lipschitz continuous as in (A.4) but need not be strongly convex as in (A.5) and the monotone operator \( \nabla D \) is restricted by the form (A.2) through this subsection.

We first discuss the convergence rate of Bregman Peaceman-Rachford splitting. From Theorem A.2 in Appendix A, we obtain that the contractive ratio of the Bregman Cayley operator \( C_i \) is given by

\[
\eta_t^i = \sqrt{1 - \frac{4\sigma_{LB,i}^t}{(1 + \sigma_{UB,i}^t)^2}},
\]

where \( \eta_t^i \) will approach 0 when the eigenvalues approach 1. For the input/output pairs of Bregman Peaceman-Rachford splitting, \( z^{t+1} = C_2C_1(z^t) \), \( z^t = C_2C_1(z^{t-1}) \), and the contractive ratio is represented by

\[
\|z^{t+1} - z^t\|_2 \leq \eta_t^1 \eta_t^2 \|z^t - z^{t-1}\|_2.
\]

By using \( \eta_{\max,i} = \max\{\eta_1^i, \ldots, \eta_n^i\} \), the contractive ratio for each iteration can be represented as a constant:

\[
\|z^{t+1} - z^t\|_2 \leq \eta_{\max,1} \eta_{\max,2} \|z^t - z^{t-1}\|_2.
\]

The difference between variable \( z^t \) and its fixed point \( z^* \) is represented by

\[
\|z^t - z^*\|_2 = \|z^t - z^{t+1} + z^{t+1} - z^{t+2} + \cdots - z^*\|_2 \\
\leq \sum_{l=t}^{\infty} \|z^l - z^{l+1}\|_2 \\
\leq \left( \sum_{j=1}^{\infty} (\eta_{\max,1} \eta_{\max,2})^j \right) \|z^{t+2} - z^{t+1}\|_2 \\
= \frac{\eta_{\max,1} \eta_{\max,2}}{1 - \eta_{\max,1} \eta_{\max,2}} \|z^{t+2} - z^{t+1}\|_2. \tag{2.41}
\]
Similarly, we obtain
\begin{equation}
\|z^{t+1} - z^*\|_2 \leq \frac{1}{1 - \eta_{\text{max},1} \eta_{\text{max},2}} \|z^t - z^{t+1}\|_2. \tag{2.42}
\end{equation}

From (2.41) and (2.42), the following inequality is satisfied as
\begin{equation}
\|z^{t+1} - z^*\|_2 \leq \eta_{\text{max},1} \eta_{\text{max},2} \|z^t - z^*\|_2. \tag{2.43}
\end{equation}

Thus, the convergence rate on Bregman Peaceman-Rachford splitting satisfies
\begin{equation}
\|z^t - z^*\|_2 \leq (\eta_{\text{max},1} \eta_{\text{max},2})^t \|z^0 - z^*\|_2. \tag{2.44}
\end{equation}

Next, we discuss the convergence rate of Bregman Douglas-Rachford splitting. From Theorem A.2 in Appendix A, we obtain that the contractive ratio of the D-Cayley operator given by
\begin{equation}
\|z^{t+1} - z^*\|_2 = \|\alpha C_2 C_1(z^t) + (1 - \alpha) z^t - z^*\|_2
\end{equation}
\begin{align*}
&\leq \alpha \|C_2 C_1(z^t) - z^*\|_2 + (1 - \alpha) \|z^t - z^*\|_2 \\
&\leq \alpha \eta_{\text{max},1} \eta_{\text{max},2} \|z^t - z^*\|_2 + (1 - \alpha) \|z^t - z^*\|_2 \\
&= (1 - \alpha + \alpha \eta_{\text{max},1} \eta_{\text{max},2}) \|z^t - z^*\|_2. \tag{2.45}
\end{align*}

Thus, the convergence rate on Bregman Douglas-Rachford splitting is bounded by
\begin{equation}
\|z^t - z^*\|_2 \leq (1 - \alpha + \alpha \eta_{\text{max},1} \eta_{\text{max},2})^t \|z^0 - z^*\|_2. \tag{2.46}
\end{equation}

Finally, we bound the convergence rate of Bregman forward-backward splitting. We have input/output pairs given by
\begin{equation}
w^t = R_2(I - \nabla D^{-1} \partial G_1)(w^{t-1}), \quad w^{t+1} = R_2(I - \nabla D^{-1} \partial G_1)(w^t).
\end{equation}

By subtracting these, we obtain
\begin{equation}
w^{t+1} - w^t = R_2(I - \nabla D^{-1} \partial G_1)(w^t) - R_2(I - \nabla D^{-1} \partial G_1)(w^{t-1}).
\end{equation}

Lipschitz continuous of \((I - \nabla D^{-1} \partial G_1)\) is represented by
\begin{align*}
&\| (I - \nabla D^{-1} \partial G_1)(w^t) - (I - \nabla D^{-1} \partial G_1)(w^{t-1}) \|_2^2 \\
= &\| w^t - w^{t-1} - (\nabla D^{-1} \partial G_1(w^t) - \nabla D^{-1} \partial G_1(w^{t-1})) \|_2^2 \\
= &\| w^t - w^{t-1} \|_2^2 - 2 \langle \nabla D^{-1} \partial G_1(w^t) - \nabla D^{-1} \partial G_1(w^{t-1}), w^t - w^{t-1} \rangle \\
&+ \| \nabla D^{-1} \partial G_1(w^t) - \nabla D^{-1} \partial G_1(w^{t-1}) \|_2^2 \\
\leq &\left(1 - 2 \sigma^t_{LB,1} + \sigma^t_{UB,1} \right) \| w^t - w^{t-1} \|_2^2. \tag{2.47}
\end{align*}

Since \(R_2\) is Lipschitz continuous with \(\frac{1}{1 + \sigma^t_{LB,2}}\) from Theorem A.1, the contractive ratio through the Bregman forward-backward splitting is
\begin{equation}
\|w^{t+1} - w^t\|_2 \leq \sqrt{1 - 2 \sigma^t_{LB,1} + \sigma^t_{UB,1}} \| w^t - w^{t-1} \|_2. \tag{2.48}
\end{equation}
By using \( \sigma_{\text{max},FB} = \max \left( \sqrt{1-2\sigma_{L1,1}^T+\sigma_{L1,2}^T}, 1+\sigma_{L1,2}^T \right) \), the contractive ratio can be represented as a constant:
\[
\|w^{t+1}-w^t\|_2 \leq \sigma_{\text{max},FB} \|w^t-w^{t-1}\|_2.
\]
Thus, the convergence rate on Bregman forward-backward splitting satisfies
\[
\|w^t-w^*\|_2 \leq (\sigma_{\text{max},FB})^t \|w^0-w^*\|_2.
\]

The theory of this section can be used to predict the convergence rate of the three B-MOS solvers discussed in Sec. 2.3. The convergence rate will be fast for the Newton and AGD methods because the eigenvalue range is concentrated near 1.

3. APPLICATION EXAMPLE. In this section, B-MOS solver is applied to a cross-entropy minimization problem with \( L_1 \) norm regularization as in [46]. We first briefly formulate the problem in Sec. 3.1 and provide its B-MOS solver implementation in Sec. 3.2. Through numerical experiments in Sec. 3.3, we will illustrate the effectiveness of B-MOS.

3.1. Problem Definition. The problem form is of the constrained minimization as explained in Sec. 2.1. We aim to solve the multi-class classification problem, using the cross-entropy as an error term:
\[
F_1(w) = \sum_{k=1}^{K} F_{1,k}(w_k),
\]
\[
F_{1,k}(w_k) = -\frac{1}{U} \sum_{u=1}^{U} s_{k,u} \log d_{k,u}(w_k),
\]
where \( w = [w_1^T, \ldots, w_K^T]^T \), \( U \) is the number of data samples, \( K \geq 2 \) is the number of classes to be discriminated, and \( s_{k,u} \in \{0, 1\} \) is a one-hot-function that indicates whether a data sample belongs to class \( k \). \( d_{k,u}(w_k) \) is obtained through nonlinear transformation using observed data \( \phi_u \in \mathbb{R}^m \) and variable to be optimized \( w_k \in \mathbb{R}^m \) as:
\[
a_{k,u}(w_k) = \langle w_k, \phi_u \rangle,
\]
\[
d_{k,u}(w_k) = \frac{\exp(a_{k,u}(w_k))}{\sum_{j=1}^{K} \exp(a_{j,u}(w_j))}.
\]

\( F_2 \) is a regularization term in the form on an \( L_1 \) norm:
\[
F_2(w) = \sum_{k=1}^{K} F_{2,k}(w_k),
\]
\[
F_{2,k}(w_k) = \mu \|w_k\|_1,
\]
where \( \mu > 0 \). The problem to be solved in this section is \( \inf_w (F_1(w) + F_2(w)) \).

As a preliminary step towards constructing a B-MOS solver, we provide the first and second order gradients of \( F_{1,k} \):
\[
\nabla F_{1,k}(w_k) = \sum_{u=1}^{U} (s_{k,u} - d_{k,u}(w_k)) \phi_u,
\]
\[
\nabla^2 F_{1,k}(w_k) = \sum_{u=1}^{U} d_{k,u}(w_k) (s_{k,u} - d_{k,u}(w_k)) \phi_u \phi_u^T.
\]
3.2. Algorithm Implementation. As a B-MOS solver, Bregman Peaceman-Rachford splitting is applied to the problem defined in Sec. 2.2. Following the description in Sec. 3.1, the update procedure is

\begin{align}
\mathbf{w}^{t+1}_k &= R_1(\mathbf{z}^t_k) = \arg \min_{\mathbf{w}_k} (F_{1,k}(\mathbf{w}_k) + B_{D_k}(\mathbf{w}_k \parallel \mathbf{z}^t_k)), \\
\mathbf{x}^{t+1}_k &= C_1(\mathbf{z}^t_k) = 2\mathbf{w}^{t+1}_k - \mathbf{z}^t_k, \\
\mathbf{y}^{t+1}_k &= R_2(\mathbf{x}^{t+1}_k) = \arg \min_{\mathbf{y}_k} (F_{2,k}(\mathbf{y}_k) + B_{D_k}(\mathbf{y}_k \parallel \mathbf{x}^{t+1}_k)) \\
\mathbf{z}^{t+1}_k &= C_2(\mathbf{x}^{t+1}_k) = 2\mathbf{y}^{t+1}_k - \mathbf{x}^{t+1}_k,
\end{align}

where \( B_{D_k}(\mathbf{w}_k \parallel \mathbf{z}^t_k) \) is given from (2.28) as

\begin{equation}
B_{D_k}(\mathbf{w}_k \parallel \mathbf{z}^t_k) = \frac{1}{2} \langle \mathbf{M}_{D_k}(\mathbf{z}^t_k)(\mathbf{w}_k - \mathbf{z}^t_k), \mathbf{w}_k - \mathbf{z}^t_k \rangle.
\end{equation}

As noted in Sec. 2.3, \( \mathbf{M}_{D_k}(\mathbf{z}^t_k) \) must be designed appropriately to obtain a fast convergence rate (e.g., Newton in (2.34) or AGD in (2.35)). Since the Hessian of the regularization term (\( L_1 \) norm) is a zero matrix except at discontinuity points, it is natural to design \( \mathbf{M}_{D_k}(\mathbf{z}^t_k) \) such that it matches only with the error term (the cross-entropy), case 1 in (2.34) and (2.35). We now summarize the implementation of \( \mathbf{M}_{D_k}(\mathbf{z}^t_k) \) for the three methods illustrated in Sec. 2.3:

\begin{equation}
\mathbf{M}_{D_k}(\mathbf{z}^t_k) = \begin{cases}
\mathbf{M}_{F_{1,k}}(\mathbf{z}^t_k) & \text{(Newton)} \\
L_{F_{1,k}}(\mathbf{z}^t_k) & \text{(AGD)} \\
\frac{1}{\varepsilon} \mathbf{I} & \text{(GD)}
\end{cases}
\end{equation}

where \( \mathbf{M}_{F_{1,k}}(\mathbf{z}^t_k) = \nabla^2 F_{1,k}(\mathbf{z}^t_k) + \varepsilon \mathbf{I} \) is used as discussed in Sec. 2.3.

Since the \( L_1 \)-norm includes discontinuous points, we will write down the \( \mathbf{y}_k \)-update procedure via (3.11) as

\begin{equation}
\mathbf{y}^{t+1}_k = \arg \min_{\mathbf{y}_k} \left( \mu \| \mathbf{y}_k \|_1 + B_{D_k}(\mathbf{y}_k \parallel \mathbf{x}^{t+1}_k) \right) = \mathbf{x}^{t+1}_k - \mu \mathbf{M}^{-1}_{D_k}(\mathbf{z}^t_k) \partial \| \mathbf{y}_k \|_1,
\end{equation}

where \( \partial \| \mathbf{y}_k \|_1 \) corresponds to the nonlinearly modified soft-threshold, i.e., its \( i \)-th component is calculated as

\begin{equation}
\partial \| y_{k,i} \|_1 = \begin{cases}
1 & (\chi_{k,i}^{t+1} > \mu) \\
\frac{1}{\mu} \chi_{k,i}^{t+1} & (-\mu \leq \chi_{k,i}^{t+1} \leq \mu) \\
-1 & (\chi_{k,i}^{t+1} < -\mu)
\end{cases}
\end{equation}

where \( \chi_{k,i} \) is the \( i \)-th component of \( \chi^{t+1} = \mathbf{M}_{D_k}(\mathbf{z}^t_k)\mathbf{x}^{t+1}_k \). The update procedures are summarized in Algorithm 3.1. Through the numerical experiments in the next subsection, we will show how the convergence rates change with the design of Bregman divergence and other conventional algorithms.

3.3. Numerical Experiments. We compared convergence rates for B-MOS with that of conventional ADMM [20] for the MNIST data set [47]. This data set is composed of images of \( K = 10 \) handwritten digits (\( U = 60,000 \) training and 10,000 evaluation data). Each input image \( \phi_u \) is composed of 784 (\( = 28 \times 28 \)) pixels and a bias for a total dimensionality of \( m = 785 \).

Four solvers were compared: (i) ADMM [20] in Algorithm 3.2, (ii) B-MOS with GD, (iii) B-MOS with AGD, and (iv) B-MOS with Newton. The B-MOS based
Algorithm 3.1 Bregman Peaceman-Rachford splitting based implementation

Initialization of $z^0_k$ for $t = 0, \ldots, T-1$

▷ Bregman divergence design

for all $k = 1, \ldots, K$ do

$M_{D_k}(z^0_k) = \begin{cases} M_{F_1,k}(z^0_k) \text{ (Newton)} \\ L_{F_1,k}(z^0_k) \text{ (AGD)} \\ \frac{1}{n} I \text{ (GD)} \end{cases}$

▷ Variable update

for all $k = 1, \ldots, K$ do

$w^{t+1}_k = \arg \min_{w_k} \left( F_{1,k}(w_k) + B_{D_k}(w_k \parallel z^t_k) \right)$,

$x^{t+1}_k = 2w^{t+1}_k - z^t_k$

▷ Calculating $\partial \|y_k\|_1$

for all $k = 1, \ldots, K$ do

$\chi^{t+1}_k = M_{D_k}(z^t_k) x^{t+1}_k$,

$\partial \|y_k\|_1 = \begin{cases} 1 & (\chi^{t+1}_{k,i} > \mu) \\ \frac{1}{\mu} \chi^{t+1}_{k,i} & (-\mu \leq \chi^{t+1}_{k,i} \leq \mu) \\ -1 & (\chi^{t+1}_{k,i} < -\mu) \end{cases}$

▷ Variable update

for all $k = 1, \ldots, K$ do

$y^{t+1}_k = x^{t+1}_k - \mu M^{-1}_{D_k}(z^t_k) \partial \|y_k\|_1$,

$z^{t+1}_k = 2y^{t+1}_k - x^{t+1}_k$

end for

Algorithm 3.2 Conventional ADMM

1: Initialization of $z^0_k, \varphi^0_k$
2: for $t \in 0, \ldots, T-1$ do
3: ▷ Variable update

for all $k = 1, \ldots, K$ do

$x^{t+1}_k = \arg \min_{x_k} \left( F_{1,k}(x_k) + \frac{1}{2\rho} \| x_k - z^t_k + \rho \varphi^t_k \|_2^2 \right)$,

$z^{t+1}_k = \arg \min_{z_k} \left( F_{2,k}(z_k) + \frac{1}{2\rho} \| x^{t+1}_k - z_k + \rho \varphi^t_k \|_2^2 \right)$

4: ▷ Dual variable update

for all $k = 1, \ldots, K$ do

$\varphi^{t+1}_k = \varphi^t_k + \frac{1}{\rho} (x^{t+1}_k - z^{t+1}_k)$

5: end for

methods (ii)-(iv) follow the Algorithm 3.1, design methods of $M_{D_k}(z^t_k)$. However, since our aim is to achieve fast convergence rates, only B-MOS with AGD and Newton are included in our experiments. All variables, such as $z^0_k$ were initialized by drawing from a normal distribution $\text{Norm}(0, 0.1)$. The algorithms have adjustable parameters and their values were selected to make the algorithms perform well. The selected values are shown in Table 1.
We evaluated B-MOS with three evaluation measures. To evaluate the convergence rates, we measured the variable error $E_1^t$ defined by the mean squared error (MSE) between the estimated variable $z_{tk}^t$ and its fixed point $z_k^*$:

\[ E_1^t = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{2Km} \| z_k^* - z_{tk}^t \|_2^2, \]

where $z_k^*$ is obtained by averaging the optimized results obtained by 10 trials of B-MOS with Newton. In total $T=1,000$ iterative updates were computed on a GPU (NVIDIA Tesla K40) and both the variable error and processing time were recorded. In addition the overall cost for the training data set to be minimized was measured as

\[ E_2^t = \sum_{k=1}^{K} F_{1,k}(z_{tk}^t) + \sum_{k=1}^{K} F_{2,k}(z_k^*). \]

As a third measure, the classification accuracy for evaluation data sets was also used.

Figure 1 shows the relationships between four kinds of methods and three kinds of evaluation measures. These results were drawn separately by the iteration number and the processing time. The experimental results show that B-MOS with Newton had the fastest convergence rates followed by B-MOS with AGD. Since B-MOS with GD is conventional Peaceman-Rachford splitting in this experiment and ADMM essentially follows the performance of Douglas-Rachford splitting [20], ADMM was the slowest of the discussed algorithms for this task. Although the computational cost per iteration was higher when using B-MOS with Newton or AGD than when using conventional methods, the Newton B-MOS method is also significantly faster according to the

---

### Table 1

Parameter settings.

| Parameter                          | Algorithm(s)          | Value   |
|------------------------------------|------------------------|---------|
| $L_1$ penalty coefficient, $\mu$  | all                    | 10      |
| Step-size, $\kappa$               | B-MOS GD               | 1E-4    |
| Moving averaging coefficient, $\beta$ | B-MOS AGD             | 0.8     |
| Constant to multiply a unit matrix, $\epsilon$ | B-MOS Newton/AGD | 1E4     |
| Step-size, $\rho$                 | ADMM                   | 1.0     |
processing time results. A major advantage of the new method is that we do not have to set a learning rate.

4. CONCLUSION. We considered the use of operator splitting to find the infimum of \( G(w) = G_1(w) + G_2(w) \), where \( G_1 \) and \( G_2 \) are convex, closed proper functions. We proposed a generalization of monotone operator splitting (MOS) based on Bregman divergence (B-MOS). The convergence rates of the generalized approach depend on the choice for the Bregman divergence. We found that a fast convergence rate can be achieved by designing the function \( D \) that characterizes the Bregman divergence \( B_D(w\|z) \) such that the eigenvalues of \( \nabla D^{-1}G_i \) are near 1. The fastest method introduces principles similar to those used in the Newton method in MOS. A major advantage of the new method is it eliminates the need to carefully set learning rates. The outcomes of our numerical experiments, in which the B-MOS solvers were applied to a constrained optimization problem, revealed that B-MOS solvers can significantly improve the convergence rate in practical optimization problems.

Appendix A. Attributes of \( D \)-Resolvent and \( D \)-Cayley Operators.

In this Appendix, we investigate the contractive properties of the \( D \)-resolvent and \( D \)-Cayley operators. To this purpose, we first study the operator \( \nabla D^{-1}\partial G_i \).

We restrict the operator \( D \) at iteration \( t + 1 \) to be of the form

\[
D(w) = \frac{1}{2}\langle M_D(z^t)w, w \rangle,
\]

where \( M_D(z^t) \) is positive semidefinite. We can also write

\[
\nabla D(w) = M_D(z^t)w,
\]

and hence

\[
\nabla D^{-1}(w) = M_D^{-1}(z^t)w.
\]

We also assume that a positive definite matrix \( M_{G_i}(z^t) \) and a semi-positive semidefinite matrix \( K_{G_i}(z^t) \) exist such that

\[
\|\partial G_i(w) - \partial G_i(z^t)\|_2 \leq \|M_{G_i}(z^t)(w - z^t)\|_2,
\]

\[
\|\partial G_i(w) - \partial G_i(z^t)\|_2 \geq \|K_{G_i}(z^t)(w - z^t)\|_2,
\]

where \( O \preceq K_{G_i}(z^t) \prec M_{G_i}(z^t) \prec +\infty \). It follows from (A.3), (A.4), and (A.5) and the properties of the matrix norm that

\[
\|\nabla D^{-1}(\partial G_i(w) - \partial G_i(z^t))\|_2 \leq \|M_D^{-1}(z^t)M_{G_i}(z^t)(w - z^t)\|_2,
\]

\[
\|\nabla D^{-1}(\partial G_i(w) - \partial G_i(z^t))\|_2 \geq \|M_D^{-1}(z^t)K_{G_i}(z^t)(w - z^t)\|_2.
\]

Let \( \sigma_{LB,i}^t \) be the minimum eigenvalue of \( M_D^{-1}(z^t)K_{G_i}(z^t) \) and let \( \sigma_{UB,i}^t \) be the maximum eigenvalue of \( M_D^{-1}(z^t)M_{G_i}(z^t) \). Then, it follows from (A.6) and (A.7) that

\[
\|\nabla D^{-1}\partial G_i(w) - \nabla D^{-1}\partial G_i(z^t)\|_2 \leq \sigma_{UB,i}^t\|w - z^t\|_2,
\]

\[
\|\nabla D^{-1}\partial G_i(w) - \nabla D^{-1}\partial G_i(z^t)\|_2 \geq \sigma_{LB,i}^t\|w - z^t\|_2,
\]

where \( \sigma_{UB,i}^t \geq \sigma_{LB,i}^t \geq 0 \).

We can now investigate the contractive properties of the \( D \)-resolvent and \( D \)-Cayley operators. They will be represented by using \( \sigma_{UB,i}^t \) and \( \sigma_{LB,i}^t \).
Theorem A.1. Contractive ratio of D-resolvent operator

Assume that the CCP function $G_i$ is Lipschitz continuous as in (A.4) and satisfies (A.5), where the monotone operator $\nabla D$ is restricted by the form (A.2). Then, the contractive ratio for the input/output pairs on the D-resolvent operator $R_i$ satisfies

\begin{equation}
\frac{1}{1+\sigma_{UB,i}^t} \|z^t - z^{t-1}\|_2 \leq \|R_i(z^t) - R_i(z^{t-1})\|_2 \leq \frac{1}{1+\sigma_{LB,i}^t} \|z^t - z^{t-1}\|_2.
\end{equation}

Proof. The input/output pairs for the D-resolvent operator $R_i = (1+\nabla D^{-1}\partial G_i)^{-1}$ are $w^t = R_i(z^{t-1})$, $w^{t+1} = R_i(z^t)$. They are reformulated as

$$(I + \nabla D^{-1}\partial G_i)(w^t) = z^{t-1}, \quad (I + \nabla D^{-1}\partial G_i)(w^{t+1}) = z^t.$$ 

By subtracting these, we obtain

\begin{equation}
(I + \nabla D^{-1}\partial G_i)(w^{t+1}) - (I + \nabla D^{-1}\partial G_i)(w^t) = z^t - z^{t-1}.
\end{equation}

Since $(I + \nabla D^{-1}\partial G_i)$ is strongly monotone with $(1+\sigma_{LB,i}^t)^{-1}$, its inverse operator $(I + \nabla D^{-1}\partial G_i)^{-1} = R_i$ is Lipschitz smooth with $(1+\sigma_{LB,i}^t)^{-1}$, e.g., [4]. Hence, the upper bound in (A.10) is proven. Since $\sigma_{LB,i}^t \geq 0$, this shows the nonexpansive property of D-resolvent operator and this fact was first proven in [35]. By taking the norm of (A.11), we obtain

\begin{equation}
\|w^{t+1} - w^t\|_2 + \|\nabla D^{-1}\partial G_i(w^{t+1}) - \nabla D^{-1}\partial G_i(w^t)\| \geq \|z^t - z^{t-1}\|_2.
\end{equation}

By taking the Lipschitz inequality into account, the lower bound in (A.10) is obtained. 

Theorem A.2. Contractive ratio of D-Cayley operator

Assume that the CCP function $G_i$ is Lipschitz continuous as in (A.4) and satisfies (A.5), where the monotone operator $\nabla D$ is restricted by the form (A.2). Then, the contractive ratio for the input/output pairs of the D-Cayley operator $C_i$ satisfies

\begin{equation}
\|C_i(z^t) - C_i(z^{t-1})\|_2 \leq \eta_i^t \|z^t - z^{t-1}\|_2,
\end{equation}

where $\eta_i^t (0 \leq \eta_i^t \leq 1)$ is defined by

\begin{equation}
\eta_i^t = \sqrt{1 - \frac{4\sigma_{LB,i}^t}{(1+\sigma_{UB,i}^t)^2}}.
\end{equation}

Proof. When we have $w^t = R_i(z^{t-1})$ and $w^{t+1} = R_i(z^t)$ of Theorem A.1 holds, we obtain the following relationship by multiplying $(w^{t+1} - w^t)^T$ with (A.11) as

\begin{equation}
\|w^{t+1} - w^t\|^2_2 + \langle w^{t+1} - w^t, \nabla D^{-1}\partial G_i(w^{t+1}) - \nabla D^{-1}\partial G_i(w^t) \rangle = \langle w^{t+1} - w^t, z^t - z^{t-1} \rangle.
\end{equation}

By taking strong monotonicity into account, we obtain

\begin{equation}
(1+\sigma_{LB,i}^t) \|w^{t+1} - w^t\|^2_2 \leq \langle w^{t+1} - w^t, z^t - z^{t-1} \rangle.
\end{equation}

By taking the squared $L_2$ norm for the D-Cayley input/output pairs $x^t = C_i(z^{t-1})$, $x^{t+1} = C_i(z^t)$, we obtain

\begin{equation}
\|x^{t+1} - x^t\|_2^2 = 2\|w^{t+1} - w^t\|_2^2 - \|z^t - z^{t-1}\|_2^2 \geq 4\|w^{t+1} - w^t\|_2^2 - 4\langle w^{t+1} - w^t, z^t - z^{t-1} \rangle + \|z^t - z^{t-1}\|_2^2.
\end{equation}

\begin{equation}
\langle z^t - z^{t-1} \rangle \leq \frac{1}{2}\|z^t - z^{t-1}\|_2^2.
\end{equation}
where (A.15) is used for reforming (A.16a) into (A.16b), and this proves the nonexpansive property of $C_i$. Combining (A.15) and (A.16a) results in

$$\|x^{t+1} - x^t\|^2 \leq \|z^t - z^{t-1}\|^2 - 4\sigma_{LB,i}^t \|w^{t+1} - w^t\|^2.$$ 

With the lower bound of (A.10), we obtain

$$\|x^{t+1} - x^t\|^2 \leq \left( 1 - \frac{4\sigma_{LB,i}^t}{(1 + \sigma_{UB,i}^t)^2} \right) \|z^t - z^{t-1}\|^2.$$

Therefore, we obtain (A.13).

We can find the optimal values for $\sigma_{UB,i}^t$ and $\sigma_{LB,i}^t$ as follows. First, we note that $\sigma_{min}^t \geq 0$ because of the semi-positive definiteness of $K_{G_i}(z^t)$ and positive definiteness of $M_D(z^t)$. Let us optimize $\sigma_{LB,i}^t$ given $\sigma_{UB,i}^t$. It is clear that this is the case for

$$\sigma_{LB,i}^t = \min(\sigma_{UB,i}^t, \frac{1}{4} (1 + \sigma_{UB,i}^t)^2).$$

This means that $\sigma_{LB,i}^t = \sigma_{UB,i}^t = \frac{1}{4} (1 + \sigma_{UB,i}^t)^2$ only if $\sigma_{UB,i}^t = 1$ and the contraction factor $\eta_i^t$ is then equal to 0. For $0 \leq \sigma_{UB,i}^t < 1$ or $\sigma_{LB,i}^t > 1$, we have that the optimal contraction factor when $\sigma_{LB,i}^t = \sigma_{UB,i}^t$. Thus, the contraction factor $\eta_i^t$ satisfies

$$0 \leq \left( 1 - \frac{4\sigma_{UB,i}^t}{(1 + \sigma_{UB,i}^t)^2} \right) \leq \eta_i^t \leq 1.$$

We conclude that for optimal contraction $\sigma_{UB,i}^t = \sigma_{UB,i}^t = 1$ and that, moreover, for a given $\sigma_{UB,i}^t$, it is optimal to minimize the dynamic range of the eigenvalues $\sigma_{UB,i}^t/\sigma_{LB,i}^t \to 1$.

REFERENCES

[1] T. Y. Liu, W. Chen and T. Wang, Recent advances in distributed machine learning, http://www.aaai.org/Conferences/AAAI/2017/aaai17tutorials.php, 2017.
[2] W. Fenchel, On conjugate convex functions, Canad. J. Math, 1, 73–77, 1949.
[3] H. Uzawa, Gradient method for concave programming, ii: global stability in the strictly concave case, Rendus de l’Academie des Sciences, 25, 536–538, 1847.
[4] D. F. Shanno, Conditioning of quasi-Newton methods for function minimization, Mathematics of Computation, 24(114), 647–656, 1970.
[5] H. Robbins and S. Monro, A stochastic approximation method, The Annals of Mathematical Statistics, 400–407, 1951.
[6] V. N. Vapnik, Statistical learning theory, 1, Wiley New York, 1998.
[7] A. Cauchy, Méthode générale pour la résolution des systemes d’équations simultanées, Comptes Rendus de l’Academie des Sciences, 25, 536–538, 1847.
[8] Y. Nesterov, A method of solving a convex programming problem with convergence rate o (1/k2), Soviet Mathematics Doklady, 27, 372–376, 1983.
[9] D. F. Shanno, Conditioning of quasi-Newton methods for function minimization, Mathematics of Computation, 24(111), 647–656, 1970.
[10] H. Robbins and S. Monro, A stochastic approximation method, The Annals of Mathematical Statistics, 400–407, 1951.
[11] O. Bousquet, and L. Bottou, The tradeoffs of large scale learning, Advances in Neural Information Processing Systems, 161–168, 2008.
[12] R. Johnson and T. Zhang, Accelerating stochastic gradient descent using predictive variance reduction, Advances in Neural Information Processing Systems, 315–323, 2013.
[13] L. Xiao and T. Zhang, A proximal stochastic gradient method with progressive variance reduction, SIAM Journal on Optimization, 24(4), 2057–2075, 2014.
[14] N. L. Roux, M. Schmidt and F. R. Bach, A stochastic gradient method with an exponential convergence rate for finite training sets, Advances in Neural Information Processing Systems, 2663–2671, 2012.
[15] A. Defazio, F. Bach and S. Lacoste-Julien, Saga: a fast incremental gradient method with support for non-strongly convex composite objectives, *Advances in Neural Information Processing Systems*, 1646–1654, 2014.

[16] T. Suzuki, Stochastic dual coordinate ascent with alternating direction method of multipliers, *International Conference on Machine Learning (ICML 2014)*, 736–744, 2014.

[17] S. Shalev-Shwartz and T. Zhang, Stochastic dual coordinate ascent methods for regularized loss minimization, *Journal of Machine Learning Research*, 14, 567–599, 2013.

[18] V. N. Vapnik, *Statistical learning theory*, 1, Wiley New York, 1998.

[19] Z. Qu, P. Richtárik, M. Takáč, and O. Fercoq, SDNA: stochastic dual Newton ascent for empirical risk minimization, *International Conference on Machine Learning (ICML 2016)*, 1823–1832, 2016.

[20] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, *Computers & Mathematics with Applications*, 2(1), 17–40, 1976.

[21] J. Douglas and H. H. Rachford, On the numerical solution of heat conduction problems in two and three space variables, *Transactions of the American Mathematical Society*, 82(2), 421–439, 1956.

[22] M. Zinkevich, M. Weimer, L. Li and A. L. Smola, Parallelized stochastic gradient descent, *Advances in Neural Information Processing Systems*, 2595–2603, 2010.

[23] Y. Zhang, M. J. Wainwright and J. C. Duchi, Communication-efficient algorithms for statistical optimization, *Advances in Neural Information Processing Systems*, 1502–1510, 2012.

[24] B. Recht, C. Re, S. Wright and F. Niu, Hogwild: a lock-free approach to parallelizing stochastic gradient descent, * Advances in Neural Information Processing Systems*, 693–701, 2011.

[25] S. Zhang, A. E. Choromanska and Y. LeCun, Deep learning with elastic averaging SGD, *Advances in Neural Information Processing Systems*, 685–693, 2015.

[26] M. Jaggi, V. Smith, M. Takáč, J. Terhorst, S. Krishnan, T. Hofmann and M. I. Jordan, Communication-efficient distributed dual coordinate ascent, *Advances in Neural Information Processing Systems*, 3068–3076, 2014.

[27] E. Wei and A. Ozdaglar, Distributed alternating direction method of multipliers, In *IEEE 51st Annual Conference on Decision and Control (CDC 2012)*, 5445–5450, 2012.

[28] R. Zhang and J. Kwok, Asynchronous distributed admm for consensus optimization, *International Conference on Machine Learning (ICML 2014)*, 1701–1709, 2014.

[29] G. Zhang and R. Heusdens, Distributed optimization using the primal-dual method of multipliers, *IEEE Transactions on Signal and Information Processing over Networks*, 4(1), 173–187, 2017.

[30] T. Sherson, R. Heusdens and W. B. Kleijn, Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory, *arXiv preprint arXiv:1706.08654*, 2017.

[31] D. W. Peaceman and H. H. Rachford, The numerical solution of parabolic and elliptic differential equations, *Journal of the SIAM*, 3(1), 28–41, 1955.

[32] G. B. Passty, Ergodic convergence to a zero of the sum of monotone operators in Hilbert space, *Journal of Mathematical Analysis and Applications*, 72(2), 383–390, 1979.

[33] D. Davis and W. Yin, A three-operator splitting scheme and its optimization applications, *Set-valued and variational analysis*, 25(4), 829–858, 2017.

[34] L. M. Bregman, The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming, *USSR Computational Mathematics and Mathematical Physics*, 7(3), 200–217, 1967.

[35] H. H. Bauschke, M. Jonathan and P. L. Combettes, Bregman monotone optimization algorithms, *SIAM Journal on control and optimization*, 42(2), 596–636, 2003.

[36] R. Battiti, First-and second-order methods for learning: between steepest descent and Newton’s method, *Neural computation, MIT Press*, 4(2), 596–636, 2003.

[37] R. T. Rockafellar, *Convex analysis*, Princeton university press, 1970.

[38] G. J. Minty, Monotone networks, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 257, 194–212, 1960.

[39] R. Tibshirani, Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288, 1996.

[40] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[41] V. Berinde, *Iterative approximation of fixed points*, Springer, 2007.

[42] J. Eckstein, Nonlinear proximal point algorithms using Bregman functions, with applications to convex programming, *Mathematics of Operations Research*, 18(1), 202–226, 1993.

[43] J. Duchi, E. Hazan and Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, *Journal of Machine Learning Research*, 12, 2121–2159, 2011.

[44] D. Kingma and J. Ba, Adam: A method for stochastic optimization, *The International
[45] T. Tieleman and G. Hinton, Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, *COURSERA: Neural Networks for Machine Learning*, 4(2), 26–31, 2012.

[46] G. C. Cawley, N. L. Talbot and M. Girolami, Sparse multinomial logistic regression via Bayesian L1 regularisation, *Advances in neural information processing systems*, 209–216, 2007.

[47] Y. LeCun, C. Cortes and C. J. Burges, MNIST handwritten digit database, *AT&T Labs. Available: http://yann.lecun.com/exdb/mnist*, 2010.