Communication efficient quasi-Newton distributed optimization based on the Douglas-Rachford envelope

Dingran Yi and Nikolaos M. Freris

***Abstract*—We consider distributed optimization in the client-server setting. By use of Douglas-Rachford splitting to the dual of the sum problem, we design a BFGS method that requires minimal communication (sending/receiving one vector per round for each client). Our method is line search free and achieves superlinear convergence. Experiments are also used to demonstrate the merits in decreasing communication and computation costs.**

***Index Terms*—distributed optimization, quasi-Newton, Douglas-Rachford splitting, superlinear convergence**

I. Introduction

Distributed optimization has received a lot of attention in signal processing, machine learning, and control. A common setting is a set of users with local data and computational capabilities coordinated by a server to

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \sum_{i=1}^{m} f_i(x) + \frac{\lambda}{2}\|x\|^2 \tag{1}$$

where f_i pertains to client i and $\frac{\lambda}{2}\|\cdot\|$ is a regularizer to reduce model complexity.

There is a wealth of first order methods (e.g., gradient descent-based) [1] developed for (1), due to their efficiency and simplicity. Nonetheless, they suffer from slow convergence. A natural choice to remedy this (that we also adopt in this paper) is to resort to second order, i.e., Newton's method, which can achieve superlinear rate. Nonetheless, this poses serious challenges in terms of computation and communication costs. For second order methods, the server needs to compute the Newton direction. The issue is that this requires to communicate Hessian matrices in addition to gradients [2] ($\mathcal{O}(d^2)$ cost), which is unsuitable for high-dimensional problems. Several methods have been devised to alleviate this. In [3], the server updates the aggregate Hessian over d rounds during which clients communicate a single column of their local Hessian. However, the incurred delay is undesirable when $d >> 1$. There are also methods that apply compression [4], [5] and SVD decomposition [6], [7] on local Hessians before transmitting to the server but they cannot guarantee that the cost can be reduced to $\mathcal{O}(d)$. Also SVD decomposition imposes additional computational overhead on the user devices. Another approach is where clients compute a direction locally and the server aggregates, with some criterion to ensure convergence: [8] considers the angle between local direction and aggregate gradient, while [9] requires a series of additional communication exchanges per round (i.e., an inner loop). Nonetheless, only linear convergence can be established. In conclusion, the aforementioned methods require computing Hessian matrices which imposes substantial computational burden on user devices. Quasi-Newton (that approximates the curvature from gradients) can ease this problem to some extent [10], [11]. However, these methods all consider local problems as individuals and then try to estimate the sum. It reflects on the slow convergence rate or heavy communication cost. In this paper, we adopt a different approach. We consider the Douglas-Rachford (D-R) splitting [12] on the dual problem of (1) and use the corresponding envelope function [13] as the target of quasi-Newton optimization (based on the BFGS algorithm [14, Chapter 6]). This promotes a decoupling that is key to a communication-efficient distributed implementation. In specific, clients locally compute and communicate gradients pertaining to the envelope and the server performs the update. Under standard assumptions (strongly convex and Lipschitz continuous gradient), we establish global convergence with superlinear rate while maintaining the communication cost at minimum (single upload/download of a vector of size d per client for each round). Another novel contribution is that this is achieved without line search, the significance of which is elaborated next.

Backtracking line search is indispensable to ensure global convergence (i.e., from an arbitrary starting point) with (asymptotic) quadratic/superlinear rate in Newton/quasi-Newton methods [14]. In the distributed setting, this results in an 'inner-loop' of additional communication exchanges per round [8]: this not only incurs extra communication and computation burden, but also slows down the algorithm in terms of actual time. In this regard, [3], [11] establish convergence when initiating near optimality (so that line search is not needed and a unit step size can be used); this is quite a restrictive assumption in practical scenarios. [15] proposes a line search free BFGS method based on Hessian-vector products and greedy selection from base vectors. Nonetheless, in the distributed setting this would require to communicate approximated Hessian matrices. Besides, [2], [16] adopt an adaptive stepsize selection (in the place of line search) based on gradient norm. However, this is only applicable for (exact) Newton's method (thus requires matrix exchanges; see also Sec. V). Our approach is more similar to the MBFGS in [17], [18] which checks

School of Computer Science, University of Science and Technology of China, Hefei, Anhui, 230027, China. Emails: ydr0826@mail.ustc.edu.cn, nfr@ustc.edu.cn.

the function value only once to determine the stepsize (thus the communication overhead is just one scalar). We go a step beyond this, by designing a new criterion that decreases the computation cost (see Sec. II.B and Fig. 2).

Contributions:

1) We design a second order method for distributed optimization via applying BFGS on the Douglas-Rachford envelope of the dual problem. We demonstrate that it attains an efficient implementation with minimal communication costs per round.
2) We establish global convergence with superlinear rate *without line search*. This is key to communication efficiency and is attained by a new adaptive stepsize selection mechanism featuring low computational effort.
3) Experiments demonstrate noticeable advantages in terms of communication and computation savings over leading baseline methods.

Our novelty has two aspects: 1) we demonstrate that quasi-Newton acceleration on distributed optimization is possible in an efficient manner in the sense of minimal communication; 2) we derive a novel adaptive stepsize selection scheme which needs at most one more scalar communication per round.

II. Algorithm Establishment

Problem (1) can be reformulated as

$$\begin{aligned} \underset{x\in\mathbb{R}^{md},\theta\in\mathbb{R}^d}{\text{minimize}} \quad & F(x)+\frac{\lambda}{2}\|\theta\|^2 \\ \text{s.t.} \quad & x_i-\theta=0,\ \ i=1,\dots,m \end{aligned} \tag{2}$$

where $F(x)=\sum_{i=1}^m f_i(x_i)$ and $x=[x_1^\top,\dots,x_m^\top]^\top$, We also define $\hat{x}:=\frac{1}{m}\sum_{j=1}^m x_j\in\mathbb{R}^d$ and $\bar{x}:=(\hat{x},\dots,\hat{x})\in\mathbb{R}^{md}$, with the same notation applying to averaging other user variables (at the server). In the following, we first show that solving the dual problem by means of quasi-Newton minimization of the Douglas-Rachford envelope admits an efficient distributed implementation, and proceed to discuss the rule for determining the stepsize so as to avoid additional communication/computation costs (pertaining to line search).

A. Distributed method based via BFGS on the dual problem

The dual of (2) is equivalent to the following problem:

$$\underset{y\in\mathbb{R}^{md}}{\text{minimize}} \quad h_1(y)+h_2(y), \tag{3}$$

where $h_1(y):=F^\star(-y)$ is the conjugate function [19, Chapter 3], and $h_2(y):=\frac{1}{2\lambda}\left\|\sum_{i=1}^m y_i\right\|^2$ comes from the dual of quadratic (all norms are Euclidean in this paper). It follows from [13] that this is equivalent to finding a stationary point of the *Douglas-Rachford envelope*, solving (3) is equivalent to finding the stationary point of the following Douglas-Rachford envelope function

$$H_\gamma^{\mathrm{DR}}(y):=h_2^\gamma(y)-\gamma\left\|\nabla h_2^\gamma(y)\right\|^2+h_1^\gamma(y-2\gamma\nabla h_2^\gamma(y)), \tag{4}$$

where $h^\gamma(y):=\inf_z\left\{h(z)+\frac{1}{2\gamma}\|y-z\|^2\right\}$ is the Moreau envelope [20]. Note that using gradient descent to minimize (4) gives rise to the celebrated ADMM method [13]. Thus, ADMM is a first order method that can only attain linear convergence (this can also be understood by the fact that the dual update is carried via gradient ascent). In this paper, we consider second order acceleration by means of applying a quasi-Newton method for minimizing the envelope function based on gradient evaluation. The latter can be calculated as:

$$\nabla H_\gamma^{\mathrm{DR}}(y)=\frac{1}{\gamma}\left(I-2\tau Q\right)\left(y-\tau\bar{y}-\mathrm{prox}_{\gamma h_1}(y-2\tau\bar{y})\right),$$

where $Q:=\frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top\otimes I_d\in\mathbb{R}^{md\times md}$ is the averaging matrix and $\tau:=\frac{m\gamma}{m\gamma+\lambda}$. Distributed implementation is possible in view of the fact that $\mathrm{prox}_{\gamma h_1}(y-2\tau\bar{y})$ can be computed at the server based on local computations carried (in parallel) at the users, For

$$x_i=\underset{\tilde{x}_i}{\mathrm{argmin}}\left\{f_i(\tilde{x}_i)+(y_i-2\tau\hat{y})^\top\tilde{x}_i+\frac{\gamma}{2}\|\tilde{x}_i\|^2\right\}, \tag{5}$$

which are intrepreted as *primal variables* (i.e., the local models in (2)), $\mathrm{prox}_{\gamma h_1}(y-2\tau\bar{y})=y-2\tau\bar{y}+\gamma x$. To conclude:

$$\nabla H_\gamma^{\mathrm{DR}}(y)=\left(\frac{\tau}{\gamma}-\frac{2\tau^2}{\gamma}\right)\bar{y}-x+2\tau\bar{x}.$$

Moreover, we change the order of the three steps of BFGS update for a consistent description of our method:

$$\begin{aligned} B_k^{-1}=&B_{k-1}^{-1}+\frac{(s_{k-1}^\top z_{k-1}+z_{k-1}^\top B_{k-1}^{-1}z_{k-1})(s_{k-1}s_{k-1}^\top)}{(s_{k-1}^\top z_{k-1})^2} \\ &-\frac{B_{k-1}^{-1}z_{k-1}s_{k-1}^\top+s_{k-1}z_{k-1}^\top B_{k-1}^{-1}}{s_{k-1}^\top z_{k-1}}, \end{aligned} \tag{6a}$$

$$p^k=B_k^{-1}\nabla H_\gamma^{\mathrm{DR}}(y^k), \tag{6b}$$

$$y^{k+1}=y^k-\eta^k p^k, \tag{6c}$$

with $s_k=y^{k+1}-y^k, z_k=\nabla H_\gamma^{\mathrm{DR}}(y^{k+1})-\nabla H_\gamma^{\mathrm{DR}}(y^k)$. Here B_k^{-1} is to approximate the inverse of Hessian and p^k represents the update direction. Note that to initialize the iteration, we need two different variables y^0, y^1 (y^1 doesn't need to be determined from y^0) and clients run (5) so that server can get $\nabla H_\gamma^{\mathrm{DR}}(y^0)$ and $\nabla H_\gamma^{\mathrm{DR}}(y^1)$; also client i needs to store $y_i^1-2\tau\hat{y}_i^1$ for later update.

B. Stepsize selection

In BFGS, superlinear convergence is asymptotic (for some neighborhood of the optimal solution, where a unit stepsize can be used). For global convergence, backtracking line search based on evaluating the function value is rudimentary. In our setting, this is highly unattractive as it would require to run (5) multiple times, thus inducing delay and additional communication and computation costs. To remedy this, we devise a new adaptive stepsize selection mechanism with minimal costs and represent it in Alg. 1. Given a constant $\sigma\in\left(0,\frac{1}{2}\right)$, at the beginning of round k, the server calculates (line 2):

$$\begin{aligned} q_{k-1}=&\frac{\left\|s_{k-1}-B_{k-1}^{-1}z_{k-1}\right\|}{\left\|B_{k-1}^{-1}s_{k-1}\right\|}+\frac{1}{\gamma}\left\|\eta^{k-1}p^{k-1}\right\| \\ &+\left\|\nabla H_\gamma^{\mathrm{DR}}(y^{k-1})\right\|, \end{aligned} \tag{7}$$

In round k after step (6b), server decides stepsize with the two conditions:

$$q_{k-1} \geq \frac{(1-2\sigma)(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{4\left\|p^k\right\|^2}, \tag{8a}$$

$$H_\gamma^{\mathrm{DR}}(y^k - p^k) \leq H_\gamma^{\mathrm{DR}}(y^k) - \sigma(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k). \tag{8b}$$

Based on this rule we can establish that

$$\begin{aligned} H_\gamma^{\mathrm{DR}}(y^k) =& \frac{m\lambda^2 - m^2\gamma^2}{2\gamma(m\gamma+\lambda)^2}\left\|\bar{y}^k\right\|^2 + \frac{\gamma}{2}\left\|x^k\right\|^2 - F(x^k) \\ & - (x^k)^\top(y^k - \frac{2m\gamma}{m\gamma+\lambda}\bar{y}^k + \gamma x^k). \end{aligned}$$

And we define $v_i^k = -\frac{\gamma}{2}\left\|x_i^k\right\|^2 - f_i(x_i^k) - (x_i^k)^\top(y_i^k - 2\tau\hat{y}^k)$ to represent client i's part of the formula above. We further define two boolean variables $\mathcal{A}$ and $\mathcal{B}$. $\mathcal{A}$ is 1 if (8a) holds and 0 otherwise; $\mathcal{B}$ is 1 if (8b) holds and 0 otherwise. Our mechanism can be summarized to be: if $\mathcal{A} \vee^\neg \mathcal{B}$, let the stepsize η^k be $\frac{\delta(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{\|p^k\|^2}$. Otherwise, the server takes unit stepsize. To be more specific, the server first check whether (8a) holds, if it does, we have determined that $y^{k+1} = y^k - \frac{\delta(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{\|p^k\|^2}p^k$. To obtain $\nabla H_\gamma^{\mathrm{DR}}(y^{k+1})$, instead of sending $y_i^{k+1} - 2\tau\hat{y}^{k+1}$ to client to implement (5), we only use the difference, i.e., $\Delta_i^k := \eta^k(p_i^k - 2\tau\hat{p}^k)$ (line 6). If (8a) doesn't hold, server further checks whether (8b) holds, for which we first need try whether unit stepsize can be taken (line 9). In this case, client only sends back v_i^{k+1} first (line 10) to save communication cost. If (8b) holds, client doesn't need extra computation. If not, client implements (5) one more time to obtain x_i^{k+1} (line 16). From this we can also see sending Δ_i^k (line 9) is communication saving, because if (8b) doesn't hold, server just needs to send one more scalar (line 15).

We design in this way because if we only have condition (8b) and find it doesn't hold, (5) needs to be implemented one more time and thus increase computation cost. We tend to establish another condition, which should be easy to check and can indicate whether (8b) holds to some extend. It can be understood that BFGS enjoys superliear since as the iteration goes, the variable is close to minima and the approximated matrix is close to the Hessian. Recall the expression of q in (7), the first part measures how close the approximated matrix is to Hessian and the second part represent the distance between variable and minima. However, when we are determining η^k, q_k is not known yet. We use q_{k-1} to estimate and when it's small enough (i.e., $^\neg\mathcal{A}$), it is very likely that (8b holds. Therefore, only $^\neg\mathcal{A}$, we check (8b) and thus save the computation cost. We summarize the steps in Algorithm QND2R (**Q**uasi **N**ewton **D**istributed **D**ouglas **R**achford).

Algorithm 1 QND2R (server view)

initialization: $y^0, y^1, \nabla H_\gamma^{\mathrm{DR}}(y^0), \nabla H_\gamma^{\mathrm{DR}}(y^0), B_0$

1: **for** $k = 1, 2, 3, \ldots$ **do**
2: calculate q_{k-1} based on (7)
3: update B_k^{-1} and p^k based on (6a) and (6b)
4: set $\eta^k = \frac{\delta(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{\|p^k\|^2}$
5: **if** $\mathcal{A}$ **then**
6: send $\mathcal{A}$ and $\Delta_i^k = \eta^k\left(p_i^k - 2\tau\hat{p}_i^k\right)$ to run Alg. 2
7: receive $\left\{x_i^{k+1}, v_i^{k+1}\right\}$, go to line 18
8: **else**
9: send $\mathcal{A}$ and $\Delta_i^k = p_i^k - 2\tau\hat{p}_i^k$ to run Alg. 2
10: receive v_i^{k+1}
11: **end if**
12: **if** $\mathcal{B}$ **then**
13: send $\mathcal{B}$ and receive x_i, $\eta^k = 1$
14: **else**
15: send $\mathcal{B}$ and η^k
16: discard v_i^{k+1} in line 10 and receive $\left\{x_i^{k+1}, v_i^{k+1}\right\}$
17: **end if**
18: $y^{k+1} = y^k - \eta^k p^k$
19: $H_\gamma^{\mathrm{DR}}(y^{k+1}) = \frac{m\lambda^2 - m^2\gamma^2}{2\gamma(m\gamma+\lambda)^2}\left\|\bar{y}^{k+1}\right\|^2 + \sum_{i=1}^m v_i^{k+1}$
20: $\nabla H_\gamma^{\mathrm{DR}}(y^{k+1}) = \frac{m\lambda - m^2\gamma}{(m\gamma+\lambda)^2}\bar{y}^{k+1} - x^{k+1} + \frac{2m\gamma}{m\gamma+\lambda}\bar{x}^{k+1}$
21: **end for**

Algorithm 2 (client i)

initialization: $u_i := y_i^1 - \frac{2m\gamma}{m\gamma+\lambda}\bar{y}_i^1$

1: receive input $\mathcal{A}$ and Δ_i^k from server
2: compute $x_i^{k+1} = \operatorname{argmin}_{x_i}\left\{f_i(x_i) + (u_i - \Delta_i^k)^\top x_i + \frac{\gamma}{2}\|x_i\|^2\right\}$
3: $v_i^{k+1} = -\frac{\gamma}{2}\left\|x_i^k\right\|^2 - f_i(x_i^k) - (x_i^k)^\top(u_i - \Delta_i^k)$
4: **if** $\mathcal{A}$ **then**
5: send $\left\{x_i^{k+1}, v_i^{k+1}\right\}$, let $u_i = u_i - \Delta_i^k$, go to line 16
6: **else**
7: send v_i^{k+1} and receive $\mathcal{B}$
8: **if** $\mathcal{A}$ **then**
9: send x_i, let $u_i = u_i - \Delta_i^k$
10: **else**
11: receive η^k and compute
$x_i^{k+1} = \operatorname{argmin}_{x_i}\left\{f_i(x_i) + (u_i - \eta^k\Delta_i^k)^\top x_i + \frac{\gamma}{2}\|x_i\|^2\right\}$
12: $v_i^{k+1} = -\frac{\gamma}{2}\left\|x_i^{k+1}\right\|^2 - f_i(x_i^{k+1}) - x_i^\top(u_i - \eta^k\Delta_i^k)$
13: send $\left\{x_i^{k+1}, v_i^{k+1}\right\}$, let $u_i = u_i - \eta^k\Delta_i^k$
14: **end if**
15: **end if**
16: exit

III. CONVERGENCE ANALYSIS

Our analysis is based on the following.

Assumption 1. *Each f_i is twice differentiable, strongly convex, with Lipschitz continuous gradient, i.e., there exist $0 \leq L_1 \leq L_2$ s.t. for $i = 1, \ldots, m$,*

$$L_1 I_d \preceq \nabla^2 f_i(x) \preceq L_2 I_d, \quad x \in \mathbb{R}^d.$$

Assumption 2. *The Hessian of f_i is Lipschitz continuous, i.e., there exists constant L_3, s.t. for $i = 1, \ldots, m$,*

$$\left\|\nabla^2 f_i(x) - \nabla^2 f_i(y)\right\| \leq L_3\left\|x - y\right\|, \quad x, y \in \mathbb{R}^d.$$

Property 1. *Under Assumption 1, for given y and x obtained from (5), we have $\|x - \bar{x}\|^2 + \left\|\sum_{i=1}^m\left(\nabla f_i(x_i) + \frac{\lambda}{m}x_i\right)\right\|^2 \leq \left(\frac{\gamma^2}{\tau^2} + \gamma^2 + 1\right)\left\|\nabla H_\gamma^{DR}(y)\right\|^2$. Further, suppose that $y^\star$ is the*

stationary point of (4), the corresponding $x^\star$ from satisfies $x_1^\star = \cdots = x_m^\star$ and each $x_i^\star$ solves (1).

Property 2. *Under Assumptions 1 and 2, $f_i^\star$ is strongly convex with Lipschitz continuous gradient with parameters $\frac{1}{L_2}$ and $\frac{1}{L_1}$. Meanwhile, $\nabla^2 f_i^\star$ exists and is continuous with parameter $\frac{L_3}{L_1^3}$*

Then we show the properties of the envelope function H_γ^{DR} defined in (4).

Lemma 1. *Under Assumptions 1 and 2, by choosing $\gamma = \frac{\lambda}{3m}$, H_γ^{DR} is strongly convex with Lipschitz continuous gradient with parameters $\min\left\{\frac{1}{8\gamma}, \frac{1}{L_2+\gamma}\right\}$ and $\frac{1}{\gamma}$ respectively. Meanwhile the Hessian is continuous with parameter $\frac{L_3 L_2^3}{L_1^3}$.*

A positive attribute of this analysis is that the hyperparameter can be easily selected without any loss on properties of the individual loss functions (i.e., using only λ, m that are directly accessible by the server).

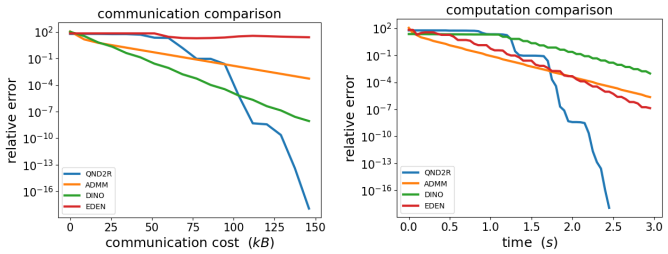
Theorem 1. *Under Assumptions 1 and 2, by choosing $\gamma = \frac{\lambda}{3m}$,$\sigma \in \left(0, \frac{1}{2}\right), \delta \in (0, \frac{\lambda}{3m})$, and B_0 to be some positive definite matrix, the sequence $\{y^k\}$ generated by Alg. 1 converges superlinearly to the unique minima $y^\star$ of (4). And for large enough k, unit stepsize is always taken.*

IV. Experiments

We evaluate our algorithm on a distributed logistic regression, which is a classical convex optimization problem:

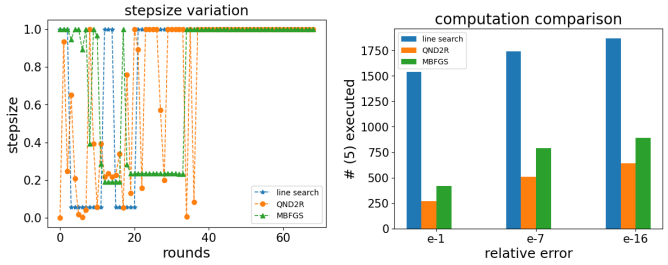
$$f_i(x) := \frac{1}{m_i}\sum_{j=1}^{m_i}\left[\ln\left(1+e^{w_j^T x}\right) + (1-y_j)w_j^T x\right],$$

where m_i is the number of data points held by each agent and $\{w_j, y_j\}_{j=1}^{m_i} \subset \mathbb{R}^d \times \{0,1\}$ are labeled samples. We used data from LIBSVM. We take 5,000 data points with dimension $d = 22$, and distribute them across $m = 10$ agents before ordering by label. We assume that communication is accurate and timely. We compare with two other papers DINO [8], EDEN [3] as well as standard ADMM since they avoid Hessian communication. We don't compare with papers using compression techniques like [4]–[7] because they need compression ratio tuning and matrix decomposition computation. For our method, since the server doesn't contain a model, the relative error is set to be $\left\|\sum_{i=1}^m \nabla f_i(x_i^k) + \frac{\lambda}{m}x_i\right\|^2 + \|x - \bar{x}\|^2$, where the first part represents the gradient and the second part represents the consensus error. For others, the relative error is set to be $\left\|\lambda x^k + \sum_{i=1}^m \nabla f_i(x^k)\right\|^2$. Fig. 1 show the comparison with algorithms above in terms of communication (a) and computation (b) cost. Our method QND2R gives a clear superlinear convergence and outperforms baselines. Fig. 2 illustrate the effect of our stepsize selection mechanism. We compare with inexact line search and MBFGS used in [17], [18]. Fig. 2.a shows the stepsize variation, in which our method finally take a unit stepsize. As expected, QND2R keeps the unit stepsize at the latest stage, however, Fig. 2.b shows our method requires least computation for target accuracy since the computation in our setting can be measured by times (5) executed. To be specific, our method gives $35.7\%, 35.4\%, 28.1\%$ computation savings for the three target accuracy respectively compared with MBFGS.



(a) (b)

Fig. 1: Relative error reduction vs. (a) total communication cost and (b) total computation cost (run-time).



(a) (b)

Fig. 2: Performance of different stepsize mechanism vs. stepsize variation (a) and computation cost (b).

V. Discussions

Central model:
The server does not contain a model in our implementation. If one is desirable, any single model or the averaged model, i.e., $\frac{1}{m}\sum_{j=1}^m x_j$ can be considered. The guarantee is because of Property 1: the consensus error and distance to optimality are upper bounded by the gradient norm of the envelope.

Regularizer choice:
In (1), we consider quadratic instead of general regularizers. This is because when establishing envelope in (4), we directly use formula of conjugate function to give a concise algorithm as well as clear values for hyperparameters. For another regularizer, one can use the same analysis to obtain the coefficients and hyperparameters for it.

More on stepsize selection:
In our setting, checking the decrease on the function value needs to solve (5), which means if unit stepsize does not work in one round, methods that only consider 8b will waste the computation. Thus, we establish 8a so that we can avoid such waste as much as possible. The effect of this choice is observed in Fig. 2. Although our method takes more conservative stepsize (Fig. 2.a), this is carried with much less computation effort, thus leading to substantial overall savings (Fig. 2.b).

REFERENCES

[1] Dušan Jakovetić, Joao Xavier, and José MF Moura. Fast distributed gradient methods. *IEEE Transactions on Automatic Control*, 59(5):1131–1146, 2014.

[2] J. Zhang, K. You, and T. Başar. Distributed adaptive Newton methods with global superlinear convergence. *Automatica*, 138:110–156, 2022.

[3] C. Liu, L. Chen, L. Luo, and J. C. Lui. Communication efficient distributed Newton method with fast convergence rates. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1406–1416, 2023.

[4] M. Safaryan, R. Islamov, X. Qian, and P. Richtarik. FedNL: Making Newton-type methods applicable to federated learning. In *International Conference on Machine Learning*, pages 18959–19010, 2022.

[5] R. Islamov, X. Qian, S. Hanzely, M. Safaryan, and P. Richtárik. Distributed Newton-type methods with communication compression and Bernoulli aggregation. *Transactions on Machine Learning Research*, 2023.

[6] A. Agafonov, D. Kamzolov, R. Tappenden, A. Gasnikov, and M. Takáč. FLECS: A federated learning second-order framework via compression and sketching. *arXiv:2206.02009*, 2022.

[7] N. D. Fabbro, S. Dey, M. Rossi, and L. Schenato. SHED: A Newton-type algorithm for federated learning based on incremental Hessian eigenvector sharing. *Automatica*, 160:111460, 2024.

[8] R. Crane and F. Roosta. DINO: Distributed Newton-type optimization method. In *International Conference on Machine Learning*, pages 2174–2184, 2020.

[9] M. Eisen, A. Mokhtari, and A. Ribeiro. A primal-dual quasi-Newton method for exact consensus optimization. *IEEE Transactions on Signal Processing*, 67(23):5983–5997, 2019.

[10] M. Eisen, A. Mokhtari, and A. Ribeiro. Decentralized quasi-Newton methods. *IEEE Transactions on Signal Processing*, 65(10):2613–2628, 2017.

[11] S. Soori, K. Mishchenko, A. Mokhtari, M. M. Dehnavi, and M. Gurbuzbalaban. DAve-QN: A distributed averaged quasi-Newton method with local superlinear convergence rate. In *International Conference on Artificial Intelligence and Statistics*, pages 1965–1976, 2020.

[12] J. Eckstein and D. P. Bertsekas. On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Mathematical programming*, 55:293–318, 1992.

[13] P. Patrinos, L. Stella, and A. Bemporad. Douglas-Rachford splitting: Complexity estimates and accelerated variants. In *IEEE Conference on Decision and Control*, pages 4234–4239, 2014.

[14] S. J. Wright. Numerical optimization, 2006.

[15] Y. Du and K. You. Distributed adaptive greedy quasi-Newton methods with explicit non-asymptotic convergence bounds. *Automatica*, 165:111629, 2024.

[16] B. Polyak and A. Tremba. New versions of Newton method: step-size choice, convergence domain and under-determined equations. *Optimization Methods and Software*, 35(6):1272–1303, 2020.

[17] L. Zhang. A globally convergent BFGS method for nonconvex minimization without line searches. *Optimization Methods and Software*, 20(6):737–747, 2005.

[18] L. Liu, Z. Wei, and X. Wu. The convergence of a new modified BFGS method without line searches for unconstrained optimization or complexity systems. *Journal of Systems Science and Complexity*, 23(4):861–872, 2010.

[19] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[20] J. J. Moreau. Fonctions convexes duales et points proximaux dans un espace hilbertien. *Comptes rendus hebdomadaires des séances de l'Académie des sciences*, 255:2897–2899, 1962.

[21] F. Abboud, M. Stamm, E. Chouzenoux, J.C. Pesquet, and H. Talbot. Distributed algorithms for scalable proximity operator computation and application to video denoising. *Digital Signal Processing*, 128:103610, 2022.

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

[23] X. Zhou. On the Fenchel duality between strong convexity and Lipschitz continuous gradient. *arXiv:1803.06573*, 2018.

[24] R. H. Byrd and J. Nocedal. A tool for the analysis of quasi-Newton methods with application to unconstrained minimization. *SIAM Journal on Numerical Analysis*, 26(3):727–739, 1989.

APPENDIX

In this section, we first give discussion on other proximal methods for distributed optimization and then full proofs for analysis.

Distributed proximal method: Splitting is a classical proximal method for optimization. As stated in [21], primal-dual methods are prominently used with the advantage that they usually do not need to compute the inverse of these linear operators. Algorithm for distributed setting is studied. [21] establishes the algorithm based on dual forward-backward splitting while the well-known ADMM [22] is actually dual Douglas-Rachford splitting. However, these splitting methods are first order method and thus suffer from slow convergence rate. Since the Douglas-Rachford envelope proposed in [13] offers a new tool to analyze the convergence, we look for the feasibility to establish a second order method and propose QND2R.

proof for Property. 1. Since $\nabla H_\gamma^{\mathrm{DR}}(y) = (\tau/\gamma - 2\tau^2/\gamma)\bar{y} - x + 2\tau\bar{x}$ and $\bar{y} = \mathbf{1}_m \otimes \hat{y}$, $\bar{x} = \mathbf{1}_m \otimes \hat{x}$, we have $\|x - \bar{x}\|^2 \leq \left\|\nabla H_\gamma^{\mathrm{DR}}(y)\right\|$. From (5) we obtain $\nabla f_i(x_i) + y_i - 2\tau\hat{y} + \gamma x_i = 0$, therefore, adding from 1 to m over i and substitute τ with $\frac{m\gamma}{m\gamma+\lambda}$ we have

$$
\begin{aligned}
0 =& \sum_{i=1}^m \left(\nabla f_i(x_i) + y_i - 2\tau\hat{y} + \gamma x_i\right) \\
=& m\hat{y} - 2m\tau\hat{y} + m\gamma\hat{x} + \sum_{i=1}^m \nabla f_i(x_i) \\
=& \frac{m\gamma+\lambda}{m} \sum_{i=1}^m \left(\left[\nabla H_\gamma^{\mathrm{DR}}(y)\right]_i + x_i\right) - m\gamma\hat{x} + \sum_{i=1}^m \nabla f_i(x_i),
\end{aligned}
$$

where $\left[\nabla H_\gamma^{\mathrm{DR}}(y)\right]_i$ means a sub-vector from entry $id - d + 1$ to id. Therefore,

$$
\begin{aligned}
&\left\|\sum_{i=1}^m \left(\nabla f_i(x_i) + \frac{\lambda}{m} x_i\right)\right\|^2 \\
\leq& \frac{(m\gamma+\lambda)^2}{m^2} \left\|\nabla H_\gamma^{\mathrm{DR}}(y)\right\|^2 + \gamma^2 \|x - \bar{x}\|^2 \\
\leq& \left(\frac{\gamma^2}{\tau^2} + \gamma^2\right) \left\|\nabla H_\gamma^{\mathrm{DR}}(y)\right\|^2.
\end{aligned}
$$

If $\nabla H_\gamma^{\mathrm{DR}}(y) = 0$, we have $x = \bar{x}$, i.e., $x_1 = \cdots = x_m$. Meanwhile we have $\sum_{i=1}^m \left(\nabla f_i(x_i) + \frac{\lambda}{m} x_i\right) = 0$ so each x_i solves (1). □

proof for Property. 2. The first part can be directly obtained from [23]. f_i is closed and strongly convex, for any y, there exists a unique x such that $y = f_i(x)$, therefore, $\nabla f_i^\star(y) = x$, which means the gradient of $f_i^\star$ exists. For x_0 and any sequence $\{x_j\}$ that converge to x_0, we have (assume that $x_j \neq x_0$ for any j)

$$
\frac{\left\|\nabla^2 f_i(x_0)(x_j - x_0) - (\nabla f_i(x) - \nabla f_j(x_0))\right\|}{\|x_j - x_0\|} \to 0.
$$

For $y_0 = \nabla f_i(x_0)$, and the sequence $\{y_j\}$ such that $y_j = \nabla f_i(x_j)$, we have

$$\frac{\left\|\nabla^2 f_i(x_0)\left(\nabla f_i^\star(y_j) - \nabla f_i^\star(y_0)\right) - (y_j - y_0)\right\|}{\|x_j - x_0\|} \to 0.$$

$\nabla^2 f_i(\cdot)$ is upper and lower bounded, so

$$\frac{\left\|\left(\nabla f_i^\star(y_j) - \nabla f_i^\star(y_0)\right) - \left(\nabla^2 f_i(x_0)\right)^{-1}(y_j - y_0)\right\|}{\|y_j - y_0\|} \to 0,$$

which means $\nabla^2 f_i^\star(y_0) = \left(\nabla^2 f_i(x_0)\right)^{-1}$. For any x_0 and x_1, since $\left\|I - (\nabla^2 f_i(x_1))^{-1}\nabla^2 f_i(x_0)\right\| \le \frac{L_3}{L_1}\|x_1 - x_0\|$, we have $\left\|(\nabla^2 f_i(x_0))^{-1} - (\nabla^2 f_i(x_1))^{-1}\right\| \le \frac{L_3}{L_1^2}\|x_1 - x_0\|$. Substituting with conjugate function and y variables, we obtain $\left\|\nabla^2 f_i^\star(y_0) - \nabla^2 f_i^\star(y_1)\right\| \le \frac{L_3}{L_1^3}\|y_1 - y_0\|$. □

proof for Lemma. 1. Based on Property 2, we can further calculate that

$$\nabla^2 H_\gamma^{\mathrm{DR}}(y) = \gamma^{-1}(I - 2\tau Q)\left((I - \tau Q) - \left(I + \gamma\nabla^2 h_1(y)\right)^{-1}(I - 2\tau Q)\right).$$

Since $h_1(\cdot)$ is continuous with parameter $\frac{L_3}{L_1^3}$, with the same trick used in Property 2, we obtain $\nabla^2 H_\gamma^{\mathrm{DR}}(y)$ is continuous with parameter $\frac{L_3 L_2^3}{L_1^3}$. Next, we calculate $z^\top \nabla^2 H_\gamma^{\mathrm{DR}}(y) z$ to show that $\nabla^2 H_\gamma^{\mathrm{DR}}(y)$ is uniformly lower and upper bounded.

$$\begin{aligned}&\gamma z^\top \nabla^2 H_\gamma^{\mathrm{DR}}(y) z\\ =&z^\top z - \left(3\tau - 2\tau^2\right)\bar{z}^\top \bar{z}\\ &- (z - 2\tau\bar{x})^\top\left(I + \gamma\nabla^2 h_1(y)\right)^{-1}(z - 2\tau\bar{x}).\end{aligned}$$

Since $\frac{1}{1+\gamma/L_1} I \preceq \left(I + \gamma\nabla^2 h_1(y)\right)^{-1} \preceq \frac{1}{1+\gamma/L_2} I$ and by choosing $\gamma = \frac{\lambda}{3m}$, $\tau = \frac{m\gamma}{m\gamma+\lambda} = \frac{1}{4}$, we have

$$\gamma z^\top \nabla^2 H_\gamma^{\mathrm{DR}}(y) z \le z^\top z - \frac{5}{8}\bar{z}^\top \bar{z} - \frac{z^\top z - \frac{3}{4}\bar{z}^\top \bar{z}}{1 + \gamma/L_1} \le z^\top z,$$

which means $\nabla^2 H_\gamma^{\mathrm{DR}}(y)$ is upper bounded by $\frac{1}{\gamma} I$. Similarly, we obtain

$$\begin{aligned}\gamma z^\top \nabla^2 H_\gamma^{\mathrm{DR}}(y) z \ge& z^\top z - \frac{5}{8}\bar{z}^\top \bar{z} - \frac{z^\top z - \frac{3}{4}\bar{z}^\top \bar{z}}{1 + \gamma/L_2}\\ \ge& \min\left\{\frac{1}{8}, \frac{\gamma}{L_2 + \gamma}\right\}\bar{z}^\top \bar{z},\end{aligned}$$

so $\nabla^2 H_\gamma^{\mathrm{DR}}(y)$ is lower bounded by $\min\left\{\frac{1}{8\gamma}, \frac{1}{L_2+\gamma}\right\} I$. □

proof for Theorem. 1. It follows from (6a) that B_k is positive definite for all k. We proceed to establish that $H_\gamma^{\mathrm{DR}}(y^k)$ is a decreasing sequence. If $\eta^k = \frac{\delta (p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{\|p^k\|^2}$, we have

$$\begin{aligned}&H_\gamma^{\mathrm{DR}}(y^{k+1}) - H_\gamma^{\mathrm{DR}}(y^k) = \nabla H_\gamma^{\mathrm{DR}}(\xi^k)^\top s_k\\ \le&\nabla H_\gamma^{\mathrm{DR}}(y^k)^\top s_k + \left\|\nabla H_\gamma^{\mathrm{DR}}(\xi^k) - \nabla H_\gamma^{\mathrm{DR}}(y^k)\right\|\|s_k\|\\ \le&(\eta^k)^2\left\|p^k\right\|^2 - \eta^k \nabla H_\gamma^{\mathrm{DR}}(y^k)^\top p^k \le (\delta - 1)\eta^k \nabla H_\gamma^{\mathrm{DR}}(y^k)^\top p^k,\end{aligned}$$

where ξ^k is on the segment between y^k and y^{k+1}. If $\eta^k = 1$, of course the sequence is decreasing. We then show the norm of gradient will converge to zero. According to [24, Thm 2.1], there exist $\beta_1, \beta_2, \beta_3$ such that

$$\|B_k s_k\| \le \beta_1\|s_k\|, \ \ \beta_2\|s_k\|^2 \le s_k^\top B_k s_k \le \beta_3\|s_k\|^2$$

hold for infinitely many k. Denoting the subsequence $\{k'\}$, we have

$$\eta^{k'} \ge \min\left\{1, \frac{\delta (p^{k'})^\top B_{k'} p^{k'}}{\|p^{k'}\|}\right\} \ge \min\{1, \delta\beta_2\}.$$

$-B_{k'} s_{k'} = \eta^{k'} B_{k'} p^{k'} = \eta^{k'} \nabla H_\gamma^{\mathrm{DR}}(y^{k'})$, so $\left\|\nabla H_\gamma^{\mathrm{DR}}(y^{k'})\right\| \le \left\|p^{k'}\right\|$, which means

$$\begin{aligned}&\nabla H_\gamma^{\mathrm{DR}}(y^{k'})^\top p^{k'} = (p^{k'})^\top B_{k'} p^{k'} = \frac{1}{(\eta^{k'})^2}(s_{k'})^\top B_{k'} s_{k'}\\ \ge&\frac{\beta_2}{(\eta^{k'})^2}\|s_{k'}\|^2 = \beta_2\left\|p^{k'}\right\|^2 \ge \beta_2\left\|\nabla H_\gamma^{\mathrm{DR}}(y^{k'})\right\|^2.\end{aligned}$$

In the following, we use the trick of contradiction: if $\left\|\nabla H_\gamma^{\mathrm{DR}}(y^k)\right\| \nrightarrow 0$, since $H_\gamma^{\mathrm{DR}}(y^k)$ is a decreasing sequence, we have $\left\|\nabla H_\gamma^{\mathrm{DR}}(y^{k'})\right\| \nrightarrow 0$. Because

$$H_\gamma^{\mathrm{DR}}(y^{k'+1}) - H_\gamma^{\mathrm{DR}}(y^{k'}) \le (\delta - 1)\eta^{k'}\beta_2\left\|\nabla H_\gamma^{\mathrm{DR}}(y^{k'})\right\|^2,$$

we have $H_\gamma^{\mathrm{DR}}(y^{k'}) \to -\infty$, which conflicts with the strong convexity of H_γ^{DR}. Therefore $\left\|\nabla H_\gamma^{\mathrm{DR}}(y^k)\right\| \to 0$.

We now establish the superlinear convergence, which mainly comes from the fact that a unit stepsize can be always chosen for large enough k. Following our algorithm, this is equivalent to showing that for large enough k, condition in line 12 holds while that in line 5 does not. From [24, Thm 3.2], denoting by $\nabla_\star^2$ the Hessian at $y^\star$, we have $\lim_{k\to\infty}\frac{\left\|(B_k - \nabla_\star^2)s_k\right\|}{\|s_k\|} = 0$ and $\|B_k\|, \left\|B_k^{-1}\right\|$ are uniformly bounded. There exists some ξ^k between y^k and y^{k+1}, such that $z_k = \nabla^2 H_\gamma^{\mathrm{DR}}(\xi^k)s_k$, then

$$\|B_k s_k - z_k\| \le \left\|(B_k - \nabla_\star^2)s_k\right\| + \left\|(\nabla^2 H_\gamma^{\mathrm{DR}}(\xi^k) - \nabla_\star^2)s_k\right\|,$$

and

$$\left\|(B_k - \nabla_\star^2)s_k\right\| \le \|B_k s_k - z_k\| + \left\|(\nabla^2 H_\gamma^{\mathrm{DR}}(\xi^k) - \nabla_\star^2)s_k\right\|.$$

By letting $\tilde{q}_k = \frac{\|B_k s_k - z_k\|}{\|s_k\|} + M(\left\|\nabla H_\gamma^{\mathrm{DR}}(y^k)\right\| + \frac{1}{\gamma}\left\|\eta^k p^k\right\|)$, where $M := \frac{L_3 L_2^3}{L_1^3}(L_2 + 8\gamma)$ is related to the Hessian continuity parameter $\frac{L_3 L_2^3}{L_1^3}$ and strong convexity parameter $\min\left\{\frac{1}{8\gamma}, \frac{1}{L_2+\gamma}\right\}$ obtained in Lemma 1, we have $\frac{\left\|(B_k - \nabla_\star^2)s_k\right\|}{\|s_k\|} \le \tilde{q}_k \to 0$. Therefore, $q_k := \frac{\left\|s_k - B_k^{-1} z_k\right\|}{\left\|B_k^{-1} s_k\right\|} + \left\|\nabla H_\gamma^{\mathrm{DR}}(y^k)\right\| + \frac{1}{\gamma}\left\|\eta^k p^k\right\| \to 0$. From the boundedness of B_k and B_k^{-1}, $\frac{(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k)}{\|p^k\|^2}$ has a uniform lower bound, which means that line 8 in Alg. 1 is eventually executed. Moreover, we have

$$|(p^k)^\top \nabla H_\gamma^{\mathrm{DR}}(y^k) - (p^k)^\top \nabla_\star^2 p^k| \le \tilde{q}_k\left\|p^k\right\|^2.$$

Since

$$
\begin{aligned}
&H_\gamma^{\mathrm{DR}}(y^k+p^k)-H_\gamma^{\mathrm{DR}}(y^k)\\
=&(p^k)^\top\nabla H_\gamma^{\mathrm{DR}}(y^k)+\frac{1}{2}(p^k)^\top\nabla^2 H_\gamma^{\mathrm{DR}}(\xi_k)p^k\\
\le&\frac{1}{2}(p^k)^\top\nabla H_\gamma^{\mathrm{DR}}(y^k)+2\tilde{q}_k\left\|p^k\right\|^2\\
=&\sigma(p^k)^\top\nabla H_\gamma^{\mathrm{DR}}(y^k)+2\tilde{q}_k\left\|p^k\right\|^2+\frac{1-2\sigma}{2}(p^k)^\top\nabla H_\gamma^{\mathrm{DR}}(y^k),
\end{aligned}
$$

if $\tilde{q}_k \le -\frac{(2\sigma-1)(p^k)^\top\nabla H_\gamma^{\mathrm{DR}}(y^k)}{4\|p^k\|^2}$, a unit stepsize is accepted. $\square$