# Incremental Cubic Regularized SR1 Quasi-Newton Method and the Applications in Large-Scale Problems

Jinmin Hou
*China Fire and Rescue Institute,*
Beijing, China

Heric Tsang
*Zhengzhou Institute, China University of Geosciences*
Zhengzhou, China

***Abstract*—This paper delves into an innovative incremental cubic regularized symmetric rank-1 (SR1) method (ICuREG-SR1). By incorporating the cubic regularization technique into SR1, we successfully address the issue of indefinite resulting matrix in SR1. Our core strategy is to adopt an incremental optimization scheme, gradually updating the information of the objective function, which typically involves a sum of multiple independent functions, and is very common in large-scale machine learning tasks. Through numerical experiments on multiple machine learning problems, we find that compared with other traditional algorithms, our proposed algorithm exhibits superior performance in terms of gradient magnitude.**

***Index Terms*—quasi-Newton method, symmetric rank-1, superlinear convergence rate, cubic regularization, incremental optimization.**

## I. Introduction

UNCONSTRAINED or constrained optimization problems can be used to formulate many real-world engineering challenges, such as machine learning [32], [38], [45], [46] and wireless communications [18], [19]. The focus of this study is on quasi-Newton approaches for unconstrained optimization problems:

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

where $f(x)$ is the objective function and $x \in \mathbb{R}^n$ is the optimizing variable with dimension $n$.

Using a quadratic model of (1), several quasi-Newton techniques are available to tackle the issue (1). To be more precise, the objective function $f(x)$ is approximated at current iterate $x_k$ for each iteration $k$ by applying the second-order Taylor series, but using an approximate Hessian matrix instead of the real Hessian matrix. Here is an example of the quasi-Newton method in its classical form:

$$x_{k+1} = x_k - \lambda_k B_k^{-1}\nabla f(x_k), \tag{2}$$

where $B_k$ is the approximated Hessian matrix and $\lambda_k$ is the stepsize chosen by standard line search algorithm.

It is necessary to solve the associated linear system at each iteration in order to derive a descent direction of the objective function in terms of the approximated quadratic model. When using the true Hessian matrix, it is necessary to evaluate the second-order derivatives in addition to solving the linear system and inverting the matrix. This can be computationally costly, particularly if the objective function is difficult to evaluate—for example, through a physical situation—and vice versa. In order to reduce the heavy computation, many recursive update strategies for the approximated Hessian matrix have been proposed: These comprise rank-two variations such as the Davidon-Fletcher-Powell (DFP) scheme [8], [9] and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update scheme [6], [10], as well as symmetric rank-one (SR1) update [3], [5], [7]. The DFP update has been demonstrated (under mild conditions) to generate a sequence $\{x_k\}$ that will converge to a local optimal point $x^*$ at Q-superlinear convergence rate when implemented with stepsize selected by standard line search criterion for the rank-two methods, in which $B_{k+1}$ is obtained by adding a matrix of rank at most two. Furthermore, the Hessian approximation matrix sequence $\{B_k\}$ produced by the DFP update lacks the conventional way for establishing a method with Q-superlinear convergence rate: it does not have the property of converging to the genuine Hessian matrix. Actually, it is sufficient but not required for $\{B_k\}$ to converge to $\nabla^2 f(x^*)$. Due to its effectiveness, the DFP updating formula has been used recently to train complex-valued neural networks for signal processing and pattern recognition [11]. Nevertheless, the BFGS formula quickly eclipsed the DFP technique and is now thought to be the most widely used quasi-Newton approach. One advantage of the BFGS approach is that, in strongly convex situations, the generated sequence $\{B_k\}$ remains positive definite, ensuring a direction of descent. [6] has convergence properties that are comparable to the DFP approach.

For rank-one update, the SR1 scheme requires less computation at each iteration. The recursion formula is given by:

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{s_k^T(y_k - B_k s_k)}, \tag{3}$$

where $s_k = x_{k+1} - x_k$ and $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.According to computational experiments, the SR1 is significantly more efficient in the trust region framework than any other investigated quasi-Newton methods [7], and it is highly competitive when compared to the BFGS method [5]. To the best of our knowledge, though, not many previ-

Corresponding author: Heric Tsang (e-mail: 18601024657@163.com).

ous research have examined the SR1 method's convergence analysis. Conversely, [7] investigated the convergence of SR1 using the frameworks of line search and trust region. The method of proof relies on the sequence $\{s_k\}$ being uniformly independent. The outcome demonstrates that the SR1 method-generated sequence $\{B_k\}$ converges to an explicit Hessian matrix. However, in many real-world scenarios, the requirement of uniform linear independence of $\{s_k\}$ is too stringent to be met. Therefore, the conclusion that $\{B_k\}$converges to the actual Hessian matrix might not apply in all cases. Without assuming uniform independence of the sequence $\{s_k\}$, [5] investigates the convergence features of SR1 further. The result demonstrates that the SR1 update with conventional line search framework exhibits $(n+1)$-step Q-superlinear and $2n$-step quadratic convergence rate. One of the shortcomings of the SR1 update, in contrast to the BFGS update, is that it can produce an indefinite matrix $B_k$ even in highly convex problems, which could lead to a non-descent direction.

Big data will lead to an increase in the popularity of large-scale optimization problems with many measurements and variables. It can be very expensive to compute because the gradient evaluation is based on the amount of measurements. Furthermore, it may take a significant amount of memory to store the loss function gradients. Developing effective stochastic optimization methods is therefore becoming more and more important. Finding a straightforward and objective estimator of the whole gradient is the primary driving force [33].Many large-scale issues have been solved using stochastic BFGS methods, which are among the most widely used stochastic quasi-Newton (SQN) methods [15]–[17], [34]. Stochastic BFGS approach ensures positive definiteness of the sequence $\{B_k\}$ in severely convex problems.

A SQN approach for handling large-scale strongly convex issues is presented in [17]. An effective subsampled Hessian-vector product is suggested based on the limited memory BFGS (LBFGS) technique, which skillfully avoids double evaluating the gradients, since the accuracy of the curvature estimate might be challenging to regulate under the stochastic regime. A general framework of SQN approaches for solving nonconvex optimization problems is presented in [15]. A stochastic damped BFGS, which is based on [21], has been offered as a solution to the difficult challenge of maintaining the positive definiteness of the Hessian approximation in nonconvex problems. Furthermore, the norm of the Hessian inverse approximation is not uniformly bounded, which impairs convergence, as it has been demonstrated that the Hessian approximation matrix may be singular or nearly singular.

A regularized stochastic BFGS (RES) approach is suggested in [14] to mitigate the issue. In order to guarantee that the norm of the Hessian approximation is over a given threshold and hence uniformly bounded, it alters the proximity requirement of BFGS. In order to reduce the high computing cost at each iteration for the class of problems where the objective function is represented in a huge sum of highly convex functions, an incremental quasi-Newton methodology (IQN) [16] is proposed. Using incremental approaches, a single function is selected at random and then efficiently implemented using the BFGS method and iteratively updated in a cyclic manner.As a result, each iteration's processing cost is significantly decreased, but at the cost of a slower rate of convergence. The IQN technique achieves local convergence at a superlinear rate because the combined gradients of all functions can lessen the noise of the gradient approximation.

The majority of research on quasi-Newton approaches to large-scale problem solving relies on the BFGS approach and its effective variations. To the best of our knowledge, nevertheless, not many works have been created specifically for the SR1 way of tackling large-scale issues. This could be the cause of the SR1 update formula's two main shortcomings: (i) The SR1 recursion formula's denominator in (3) may disappear; (ii) Even in strongly convex situations, the ensuing Hessian approximation matrix may be infinite, which results in the step's non-descent direction. One common method to fix issue (i) is to forego the SR1 update if

$$|s_k^T(y_k - B_k s_k)| < \epsilon \|s_k\| \|y_k - B_k s_k\|, \tag{4}$$

where the value of the constant $\epsilon$ is normally chosen as $10^{-8}$. In reality, missing the SR1 update makes sense as long as the Hessian approximation matrix is positive definite. This is because the present positive definite $B_k$ still leads to a sufficient reduction in the objective function. [4] has presented a cubic regularized SR1 (CuREG-SR1) approach to mitigate problem (ii). The primary method is to compute the gradient difference using the objective function's cubic approximation. The positive definite Hessian approximation matrix can be obtained using the SR1 recursion formula by selecting an appropriate cubic parameter that meets the above requirement. Moreover, [12], [13] has researched the cubic regularization technique. The cubic regularized technique performs better than the traditional line search criterion because it avoids the issue of an indefinite Hessian approximation matrix. Nonetheless, there has been little research done to date on the cubic regularized SR1 method's convergence characteristics. In addition, creating effective algorithms to tackle large-scale problems with vast amounts of data has garnered a lot of interest lately in the big data age. In order to handle large-scale problems, we primarily design efficient algorithms based on SR1 in this study. One of our primary contributions is the proposal of a unique incremental optimization technique based on the SR1 and CuREG-SR1. Each iteration's primary goal is to update the data for a chosen subset of the objective function's individual functions, which include a massive sum of component functions, while leaving the rest of the functions unchanged from the previous iteration. Comparing our suggested technique to other evaluated traditional algorithms, numerical studies demonstrate that it performs better in terms of gradient magnitude.

The remainder of the paper is structured as follows: The general CuREG-SR1 algorithm, which is based on the line search framework, is reviewed in Section II. We suggest a brand-new ICuREG-SR1 method in Section III. In Section IV, the performance of the suggested ICuREG-SR1 method

is assessed using numerical tests. Section V contains the conclusion. Additionally, we have provided extra material that compares our suggested strategy to the cutting-edge techniques Adam [43] and RMSProp [44].

*Mathematical Notation*: we use $\|a\|$ to denote the Euclidean norm of vector $a$ and $\|A\| := \max\{\frac{\|Ax\|}{\|x\|}\}$ to denote the matrix norm of a matrix $A$. $A \succeq B$ indicates the matrix $A-B$ is positive semidefinite. The identity matrix with appropriate dimension is signified as $I$.

## II. Algorithm Definition

The unconstrained optimization issues listed below are examined in this paper: $x^* = argmin_{x\in\mathbb{R}^d} f(x) : \mathbb{R}^d \to \mathbb{R}$ is the objective function. The variable $x$ has a dimension of $d$. The traditional quasi-Newton techniques utilizing the line search framework are initially reviewed in this section. Our focus will be on presenting the cubically regularized SR1 approach and discussing how the SR1 update is derived. Following is a quick introduction to the cubic regularization approach. The SR1 approach can be made into CuREG-SR1 by including the cubic regularization technique. The requirement for the Hessian approximation update obtained from CuREG-SR1 to be positive definite will also be covered.

### A. Line Search Framework

In the classical line search method, the following update is used at each iteration to generate a search direction $p_k \in \mathbb{R}^d$ that can sufficiently lower the objective function:

$$x_{k+1} = x_k + \lambda_k p_k, \tag{5}$$

where the stepsize $\lambda_k$ determines how far along the search direction to proceed. Choosing the stepsize to solve the following subproblem is ideal: $\lambda_k^* = \operatorname{argmin}_{\lambda\in\mathbb{R}} \varphi(\lambda) := f(x_k + \lambda p_k)$: Seeking the precise one-dimensional minimum, however, is computationally costly in general since it could call for numerous evaluations of the objective function $f$ and the gradient $\nabla f$ at each iteration. Various criteria have been given for inexact line search termination in reality, with the goal of reducing the target function to a tolerable degree while minimizing computation work. A popular method is the inexact line search based on the following Wolfe condition:

$$f(x_k + \lambda p_k) < f(x_k) + c_1\lambda\nabla f(x_k)^T p_k, \tag{6}$$

$$\nabla f(x_k + \lambda p_k)^T p_k \geq c_2\nabla f(x_k)^T p_k, \tag{7}$$

where $c_1 \in (0,1)$ and $c_2 \in (c_1,1)$ are constants. The inequality in (6) is also known as Armijo condition. Intuitively, inequality (6) means that the updated objective function $f(x_k + \lambda p_k)$ lies below the linear function $l(\lambda) := f(x_k) + c_1\lambda\nabla f(x_k)^T p_k$, which has a negative slope $c_1\nabla f(x_k)^T p_k$ with respect to $\lambda$ to ensure that the function will decrease at least at a certain rate (since $p_k$ is a decrease direction and $c_1$ is a positive constant, $c_1\nabla f(x_k)^T p_k < 0$ holds). Moreover, since $f(x_k + \lambda p_k) < f(x_k)$, it indicates that for small $\lambda$, the inequality $\varphi(\lambda) < l(\lambda)$ holds, i.e., the Armijo condition is always satisfied at small stepsize, this may cause insufficient reduction when the resultant stepsize is small. The Armijo condition is thus insufficient to guarantee a reasonable progress. Therefore, the curvature condition (7) is introduced, which is called curvature condition. Note the left hand side of (7) is the derivative of the function $\varphi(\cdot)$ at $\lambda$, i.e., $\varphi'(\lambda) = \nabla f(x_k + \lambda p_k)^T p_k$. Similarly for the right hand side, $\varphi'(0) = \nabla f(x_k)^T p_k$. Consequently, for a stepsize chosen to ensure significant decrease along the search direction, it implies that the corresponding slope $\varphi'(\lambda)$ should be intuitively less slightly negative than $\varphi'(0)$. It makes sense since we have the intuition that at a point that results in sufficient reduction, the related slope is negatively flatter. Thus, combining the Armijo condition and curvature condition, the algorithm can make reasonable progress.

### B. SR1 Quasi-Newton Methods

For Newton's method, the search direction $p_k$ is obtained by solving the system: $\nabla^2 f(x_k)p_k = -\nabla f(x_k)$. However, two drawbacks make Newton's method impractical: (i). evaluation of second-order derivative at each iteration is generally too expensive; (ii). solving the system takes much time and effort, and matrix inversion or factorization increase the arithmetic complexity, which can be computationally huge for large scale problem. Therefore, we consider the quasi-Newton methods, which seek for an approximate Hessian matrix $B_k$. We now turn our attention to SR1 update (3). It has been shown that if the second-order derivative of the objective function is Lipschitz continuous and the sequence $\{\frac{s_k}{\|s_k\|}\}$ (recall $s_k$ is defined as $s_k := x_{k+1} - x_k$) is uniformly linearly independent, the sequence $\{B_k\}$ generated by SR1 formula converges to the true Hessian matrix at optimal point, i.e., $\lim_{k\to\infty} \left\|B_k - \nabla^2 f(x^*)\right\| = 0$ [5], [7]. The convergence results generally involve the following assumptions.

*Assumption 1*: The objective function $f(x)$ is twice continuously differentiable.

*Assumption 2*: The first-order derivative $\nabla f(x)$ is Lipschitz continuous with $L' > 0$, i.e., for all $x, y \in \mathbb{R}^d$, the following inequality holds:

$$\|\nabla f(x) - \nabla f(y)\| < L' \|x - y\|. \tag{8}$$

*Assumption 3*: The second-order derivative $\nabla^2 f(x)$ is Lipschitz continuous with $L'' > 0$, i.e., for all $x, y \in \mathbb{R}^d$, the following inequality holds:

$$\left\|\nabla^2 f(x) - \nabla^2 f(y)\right\| < L'' \|x - y\|. \tag{9}$$

*Assumption 4*: A sequence $\{s_k\}$ in $\mathbb{R}^d$ is defined to be uniformly linearly independent, if there exists a positive constant $\tau > 0$, an integer $k_0$ and $m \geq d$ such that for each $k \geq k_0$, one can choose $d$ distinct indices between $k$ and $k+m$, namely $k \leq k_1 < \cdots < k_d \leq k+m$, such that $\sigma_{\min}(S_k) \geq \tau$, where $\sigma_{\min}(S_k)$ is the minimum singular value of the matrix

$$S_k = [\frac{s_{k_1}}{\|s_{k_1}\|}, \cdots, \frac{s_{k_d}}{\|s_{k_d}\|}]. \tag{10}$$

Next, we first follow the development in [22] to briefly review the derivation of the SR1 formula, which motivates the CuREG-SR1 formula. Recall $y_k$ is defined as:

$$y_k := \nabla f(x_{k+1}) - \nabla f(x_k), \tag{11}$$

By using first-order Taylor expansion, we can further obtain:

$$y_k = \nabla^2 f(x_k) s_k + o(\|s_k\|). \tag{12}$$

For a quadratic function $f(x) = \frac{1}{2}x^T Ax + b^T x + c$, where $A$ is positive definite, $b, c \in \mathbb{R}^d$, it can be verified straightforwardly that the higher order term is zero. In this case, we can set $B_k = A$ and obtain the secant equation $y_k \approx B_k s_k$. Though the secant equation does not hold true for the general nonlinear functions due to the higher order term, it serves as a useful condition for updating the Hessian approximation matrix given $s_k$ and $y_k$ at each iteration:

$$y_k = B_{k+1} s_k. \tag{13}$$

For rank one update, the recursion formula has the following form, and the resulting matrix is symmetric:

$$B_{k+1} = B_k + \rho v v^T, \tag{14}$$

Combining the secant equation in (13), it follows that:

$$y_k = B_k s_k + \rho v v^T s_k \tag{15}$$

Note from the right hand side in (15) that $v^T s_k$ is a scalar, hence the vector $v$ has the same direction with $(y_k - B_k s_k)$ for some scalar $\vartheta$. Therefore, we can simply set $v = \vartheta(y_k - B_k s_k)$. It subsequently leads to the following equation:

$$y_k - B_k s_k = \rho \vartheta^2 [s_k^T (y_k - B_k s_k)](y_k - B_k s_k), \tag{16}$$

where $s_k^T(y_k - B_k s_k) \neq 0$. Comparing the LHS and RHS, there only exists two possible scenarios: (i). $s_k^T(y_k - B_k s_k) < 0$, then $\rho = -1$ and $\vartheta^2 = [-s_k^T(y_k - B_k s_k)]^{-1}$; (ii). $s_k^T(y_k - B_k s_k) > 0$, then $\rho = 1$ and $\vartheta^2 = [s_k^T(y_k - B_k s_k)]^{-1}$. Substituting the results, we obtain the SR1 formula as follows:

$$B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}. \tag{17}$$

Subsequently, the search direction $p_k$ can be calculated by solving the system $B_k p_k = -\nabla f(x_k)$. Moreover, if the inverse Hessian approximation matrix denoted by $H_k$ is positive definite, it can also be updated by employing the Sherman-Morrison formula

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}. \tag{18}$$

Therefore, instead of solving the system $B_k p_k = -\nabla f(x_k)$ by matrix inverse operation, we can simply obtain the search direction $p_k = -H_k \nabla f(x_k)$ through efficient matrix-vector product, making the algorithm computationally very attractive. As mentioned above, when the sequence $\{x_k\}$ converges to the optimal point with Assumption 3, the Hessian approximation sequence generated by (17) will converge to the true Hessian matrix. However, the denominator tends to zero if $\{x_k\}$ converges to $x^*$, which implies that all the future updates will be dominated by the update matrix and violate the uniform linearly independent assumption [4]. Hence, to alleviate both the theoretical and practical problems, the Hessian approximation matrix skips the update whenever the denominator is too small:

$$|(y_k - B_k s_k)^T s_k| < \epsilon \|y_k - B_k s_k\| \|s_k\|, \tag{19}$$

i.e., at each iteration, set $B_{k+1} = B_k$ whenever (19) is satisfied while otherwise, $B_{k+1}$ is calculated via (17). The skipping scenario has been shown to be effective since we can still get the descent direction if $B_k$ is positive definite.

### C. Cubic Regularization Technique

It should be noted that even if $B_k$ is positive definite, $B_{k+1}$ can still be indefinite since the denominator in (17) can be negative. In [4], the cubic regularized technique has been proposed to address this issue. We shall first review the cubic regularization technique [12], [13].

Recall that the search direction in quasi-Newton method for problem (1) is obtained by solving $p_k = \operatorname{argmin}_{p \in \mathbb{R}^d} f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T B_k p$. Similarly in cubic model for the objective function $f(x)$ with Lipschitz continuous $\nabla^2 f(x)$, the search direction $p$ is calculated via minimizing the cubic model $m_k^c(p)$:

$$m_k^c(p) := f(x_k) + \nabla f(x_k)^T p + \frac{1}{2} p^T B_k p + \frac{M_k}{6} \|p\|^3. \tag{20}$$

In general, $M_k$ is chosen to satisfy $M_k \leq L''$. It should be noted that minimizing the cubic model is a non-convex problem and it can have local minima [13]. Here, the search direction $p_k \in \operatorname{Argmin}_{p \in \mathbb{R}^d} m_k^c(p)$ means that $p_k$ is a global minimizer of the cubic model $m_k^c(p)$. It has been shown in [12] that $p_k^*$ is a global minimizer of the problem if and only if $\nabla m_k^c(p_k^*) = 0$ and $B_k + \frac{1}{2} M_k \|p_k^*\| \succeq 0$. The necessary and sufficient condition has provided us a way to compute a global minimizer of the cubic model. However, from a computational point of view, doing so can be prohibitively sophisticated. In fact, an approximate solution to the global minimizer can make the algorithm progress well with less computational complexity. Specifically, for the framework of adaptive regularization using cubics (ARC) proposed in [12], the search direction $p_k$ is only required to ensure the decrease in the cubic model at least as good as that produced by the corresponding Cauchy point. Because in this way and with the condition that $\nabla f(x)$ is uniformly continuous on the sequence $\{x_k\}$, the ARC algorithm has been shown to converge to the first-order critical point, i.e., $\lim_{k \to \infty} \|\nabla f(x_k)\| = 0$. Hence, the more efficient Krylov method can be applied to well approximate the global solution.

## III. Incremental CuREG-SR1

Massive data may cause standard methods to become computationally expensive in the big data era, as was previously indicated. Consequently, it is quite inefficient to apply the methods directly to issues with huge sample sizes. We present

a new and effective approach to solving large-scale problems in this part, which is based on CuREG-SR1 and SR1. In particular, the goal of our suggested approach is to tackle a class of issues that have a sum of functions format. These kind of challenges are common in many domains, including machine learning problems [26], [27], [29], [30], [32] and source localization in sensor networks [25].

Now let us consider the problem mentioned above, which has the form:

$$x^* = \text{argmin}_{x\in\mathbb{R}^d}\ f(x) := \frac{1}{N}\sum_{i=1}^{N} f_i(x), \tag{21}$$

where $N$ is the sample size and $f_i$ is an array of $N$ distinct functions. It goes without saying that when using the classic quasi-Newton approach carelessly, we must compute the big sample summation and assess the gradient $N$ times, both of which are computationally demanding. A stochastic technique that samples a tiny batch of big samples to estimate the gradient—which is utilized in place of the exact gradient—has been widely adopted to reduce such complexity [20], [35]–[37]. This concept has been applied to the stochastic quasi-Newton approach based on BFGS and has significantly decreased the computing cost [15], [17]. In this part, we include the incremental technique into an efficient algorithm based on SR1 and CuREG-SR1.

Specifically, we approximate each individual function $f_i$ by second-order Taylor expansion around its current iterate $z_i^k$. Subsequently, we obtain an approximation to the function $f$:

$$\begin{aligned} f(x) \approx \frac{1}{N}\sum_{i=1}^{N}\{f_i(z_i^k) + \nabla f(z_i^k)^T(x - z_i^k) \\ + \frac{1}{2}(x - z_i^k)^T B_i^k (x - z_i^k)\}, \end{aligned} \tag{22}$$

where the matrix $B_i^k$ is the local Hessian approximation to $\nabla^2 f_i(z_i^k)$. Furthermore, we refer to $\{z_i^k, \nabla f(z_i^k), B_i^k\}$ as the information corresponds to the individual function $f_i$. Using the same strategy with quasi-Newton methods to obtain the step direction for next iteration, we minimize the RHS in (22). It subsequently yields:

$$x_{k+1} = (\tilde{B}^k)^{-1}(\tilde{z}^k - \tilde{g}^k) \tag{23}$$

where $\tilde{B}^k := \sum_{i=1}^N B_i^k$, $\tilde{z}^k = \sum_{i=1}^N B_i^k z_i^k$ and $\tilde{g}^k := \sum_{i=1}^N \nabla f(z_i^k)$. In this paper, we consider update the Hessian approximation matrix by using SR1 and CuREG-SR1.

(23) shows that it requires computationally intensive matrix inversion and summation arising from big samples. In order to streamline calculation, we simply update the data related to one selected individual function per iteration—keeping the data pertaining to the remaining individual functions unaltered. By cyclically iterating through $N$ individual functions, the function is chosen. We begin by choosing the first individual function, $f_1$, without losing generality. At iteration $k$, we then update the data of the $i_k$-th individual function, where $i_k = (k \bmod N) + 1$. Specifically, we have

$$z_{i_k}^{k+1} = x_{k+1}, \quad z_i^{k+1} = z_i^k \quad \text{for } i \neq i_k. \tag{24}$$

Hence, with these settings, while the information of the selected function $f_{i_k}$ is updated, the other terms are kept the same with their previous value. Moreover, it follows that

$$\nabla f_{i_k}(z_{i_k}^{k+1}) = \nabla f_{i_k}(x_{k+1}), \tag{25}$$

$$\nabla f_i(z_i^{k+1}) = \nabla f_i(z_i^k) \quad \text{for } i \neq i_k. \tag{26}$$

According to (24)-(26), we can derive the following for Hessian approximation update:

$$B_i^{k+1} = B_i^k + \frac{(y_i^k - B_i^k s_i^k)(y_i^k - B_i^k s_i^k)^T}{(y_i^k - B_i^k s_i^k)^T s_i^k}, \tag{27}$$

for $i = i_k$ and $B_i^{k+1} = B_i^k$ for $i \neq i_k$, where $y_i^k = \nabla f(z_i^{k+1}) - \nabla f(z_i^k)$ and $s_i^k = z_i^{k+1} - z_i^k$. Obviously, for $i \neq i_k$, it indicates $y_i^k = 0$ and $s_i^k = 0$. This leads to our efficient computation of $\{\tilde{B}^k, \tilde{z}^k, \tilde{g}^k\}$ in (23). To be specific, $\{\tilde{B}^k, \tilde{z}^k, \tilde{g}^k\}$ can be updated as follows:

$$\tilde{B}^{k+1} = \tilde{B}^k - B_{i_k}^k + B_{i_k}^{k+1}, \tag{28}$$

$$\tilde{z}^{k+1} = \tilde{z}^k - B_{i_k}^k z_{i_k}^k + B_{i_k}^{k+1} z_{i_k}^{k+1}, \tag{29}$$

$$\tilde{g}^{k+1} = \tilde{g}^k - \nabla f(z_{i_k}^k) + \nabla f(z_{i_k}^{k+1}). \tag{30}$$

Therefore, we avoid the computation of the large sample summation to update $\{\tilde{B}^k, \tilde{z}^k, \tilde{g}^k\}$. Moreover, since updating the iterate in (23) requires matrix inverse operation, it is desirable to update its inverse to avoid direct matrix inversion. Substituting (27) into (28) leads to:

$$\tilde{B}^{k+1} = \tilde{B}^k + \frac{(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T}{(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T s_{i_k}^k}. \tag{31}$$

By applying the Sherman-Morrison formula to (31), we obtain the following update of the inverse of the approximated Hessian:

$$\tilde{H}^{k+1} = \tilde{H}^k - \frac{\tilde{H}^k(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T \tilde{H}^k}{(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T s_{i_k}^k} \tag{32}$$

where for simplicity we define $\tilde{H}^k := (\tilde{B}^k)^{-1}$. The computational complexity of (32) is $\mathcal{O}(d^2)$ while the cost of direct matrix inversion is $\mathcal{O}(d^3)$, thus substantially reducing the computational complexity. Furthermore, to ensure that each update of the Hessian approximation $B_{i_k}^{k+1}$ is positive definite, we can apply CuREG-SR1 to the resulting update. To be specific, if $(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T s_{i_k}^k < 0$ and Case II happens, we choose $M_{i_k}^k = -b/2a$ where $a$ and $b$ are calculated respectively. For other cases, we skip the update. In this way, $-B_{i_k}^k + B_{i_k}^{k+1}$ is positive definite, and thus $\tilde{B}^{k+1} \succeq 0$ in (28). We summarize the ICuREG-SR1 algorithm in *Algorithm 2*.

***Remark***. In steps 8 and 12, it should be noted from (31) that if the resultant Hessian approximation matrix $\tilde{B}^{k+1}$ is ill-conditioned, an effective regularization technique can be applied. Specifically, we have

$$\tilde{H}^{k+1} \leftarrow (\tilde{B}^{k+1} + rI)^{-1}, \tag{33}$$

with $r := \frac{\rho \text{Tr}(\tilde{B}^{k+1})}{d}$, typical values of $\rho$ are $10^{-2}$ and $10^{-3}$ etc.

**Algorithm 1** ICuREG-SR1

**Input:** set $z_i^0 = x_0$ for randomly generated $x_0$ from uniform distribution $[-1,1]^d$, $B_i^0 = I$ for $i = 1, \cdots, N$, the desired iteration number $KN$.

**Output:** $z_N^{KN} = x^{KN}$.

**for** $k = 0, 1, \ldots, KN-1$ **do**
  Calculate $x^{k+1}$: $x_{k+1} = (\tilde{B}^k)^{-1}(\tilde{z}^k - \tilde{g}^k)$.
  Set $i_k = (k \bmod N) + 1$.
  Compute $y_{i_k}^k$ and $s_{i_k}^k$ according to (24)-(26).
  Set $\tilde{H}_{k+1} = \tilde{H}_k$,
  **if** $|(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T s_{i_k}^k| > \epsilon \left\|y_{i_k}^k - B_{i_k}^k s_{i_k}^k\right\| \left\|s_{i_k}^k\right\|$ **then**
    **if** $(y_{i_k}^k - B_{i_k}^k s_{i_k}^k)^T s_{i_k}^k > 0$ **then**
      Calculate $\tilde{H}_{k+1}$ via (32),
    **else**
      Compute $a$,$b$ and $c$ respectively,
      **if** $b^2 - 4ac > 0$ and $b < 0$ **then**
        Set $M_{i_k}^k = -\frac{b}{2a}$ and calculate $\tilde{H}_{k+1}$ by substituting $\tilde{y}_{i_k}^k := y_{i_k}^k + \frac{M_{i_k}^k}{2}\left\|s_{i_k}^k\right\| s_{i_k}^k$ into (32) in lieu of $y_{i_k}^k$,
      **end if**
    **end if**
  **end if**
  Compute $\tilde{z}^{k+1}$ and $\tilde{g}^{k+1}$ according to (29) and (30) respectively.
**end for**

For $N$ samples with feature dimension $d$, if each iteration only updates one sample, there will be $N$ Hessian approximation matrices for the corresponding individual functions. Without loss of generality, suppose that the incremental algorithm starts by using the first sample (corresponding to first individual function). The memory cost for this scenario will be $O(Nd^2 + Nd)$. Hence, for large scale problems, the incremental method suffers from the problem of large memory cost. However, by grouping $L$ individual functions into a new individual function, the memory cost can be substantially reduced to $O(\frac{Nd^2+Nd}{L})$. Now we illustrate the existing technique for further reducing the memory cost. At the $(KN+1)$, the first variable denoted as $z_1^{KN}$ has been updated $K$ times. Consider limited memory SR1 (chapter 9.16 in [**?**]), it stores the correction pairs $(s_1^k, y_1^k)$ with respect to the first variable $z_1^k$ for $k = (K-m)\cdot N+1, \ldots, (K-1)\cdot N+1$. Similarly, for all components $z_i^k$, $i = 1...N$, the memory cost will be $O(2dmN)$. Moreover, if one groups $L$ individual functions as one new individual function, the memory cost will be further reduced to $O(2dmN/L)$. Here, our main purpose is to propose a framework of incremental method. It can be a future work to further refine the implementation of the above limited memory version of our proposed ICuREG-SR1.

## IV. Numerical Results

Here, we do numerical experiments on our suggested ICuREG-SR1 technique. The approach will be used for logistic regression. For the aforementioned problems, we additionally implement the traditional SGD, SdLBFGS [15], and Sd-REG-LBFGS [39] algorithms as a comparison. Keep in mind that the limited memory BFGS scheme (LBFGS) is the foundation of the subsequent two optimization strategies. The norm of the gradient at each iteration will serve as the basis for evaluating performance. In addition, we employ two datasets for the tests: the *scene* dataset, the *CIFAR-10* dataset. The parameters have been adjusted to provide each stochastic algorithm's best performance in the numerical experiments to allow for fair comparison.

### A. Logistic Regression

We first consider the logistic regression for binary classification [29]. Suppose two classes denoted as $z_n = 0$ and $z_n = 1$ are to be recognized respectively. Logistic regression models the problem as $p(z_n|\theta) = \sigma(\theta^T x_n)^{z_n} \cdot (1 - \sigma(\theta^T x_n))^{1-z_n}$, where $\theta$ is the parameter to be identified, $x_n$ is the feature vector and $\sigma(\cdot)$ is the sigmoid function given by $\sigma(x) = 1/(1 + \exp(-x))$. Given the training data $\{z_n, x_n\}$ with $x_n \in \mathbb{R}^d$ and $n = 1, \cdots, N$, the likelihood function to be maximized is $p(z_{1:N}|\theta) = \prod_{n=1}^N p(z_n|\theta)$. One can maximize the log-likelihood function or equivalently minimize the objective function: $f(\theta) = -\frac{1}{N}\sum_{n=1}^N z_n \log\, \sigma(\theta^T x_n) + (1 - z_n)\log\, \sigma(-\theta^T x_n)$. Moreover, we use the norm of gradient (NOG) for performance evaluation. The NOG for logistic regression is defined as follows:

$$\text{NOG}_{LR} = \left\| \frac{1}{N}\sum_{n=1}^{N}[z_n - \sigma(\theta^T x_n)]x_n \right\|. \tag{34}$$

### B. Numerical results with Scene Dataset

We compare Sd-REG-LBFGS with SdLBFGS and SGD in this subsection using the *scene* dataset [42] as a practical dataset. The training set has 1,211 photos, whereas the testing set has 1,196 images. There are up to 6 scene labels (*beach*, *sunset*, *fall-foliage*, *field*, *mountain*, and *urban*) for each image, out of 294 features total. We merely combine *beach*, *sunset*, and *fall-foliage* into a new category with label $z = 1$ to create the binary classification problem. $z = 0$ is the label used to group the remaining classes. The task of binary classification involves determining if a picture in the testing set belongs to the new category [28]. The numerical experiments conducted on the synthetic dataset have the same parameters as the different optimization strategies.

The performance comparison of the four methods in terms of the NOG is displayed in Figure 1. Since our technique has the smallest NOG value—a sign that it is the closest to the critical point—the result demonstrates that our method performs better than the other ways in most cases. Furthermore, our suggested approach shows a propensity to keep decreasing the gradient magnitude even after the other approaches have converged. We may infer from the subplot that, over the course of one run over the entire *scene* dataset, our suggested technique first displays a lowered gradient magnitude and subsequently an enhanced NOG value.

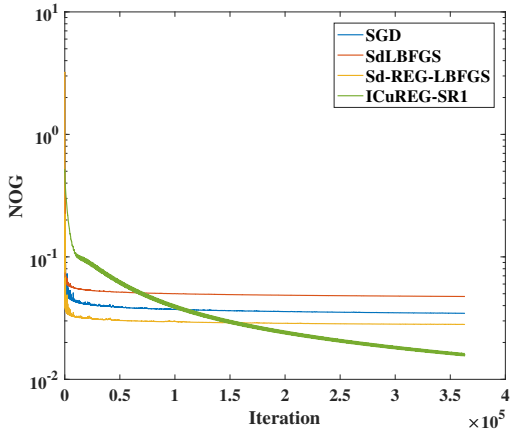


Fig. 1: The NOG performance of different algorithms applied in solving logistic regression using *scene* dataset.

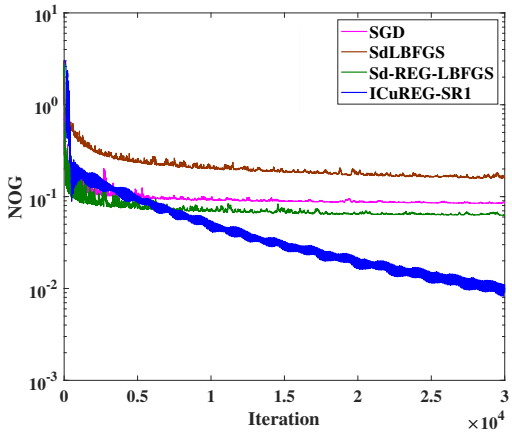


Fig. 2: The NOG performance of different algorithms applied in solving logistic regression using *CIFAR-10* dataset.

### C. Numerical results with CIFAR-10 Dataset

This subsection uses a sizable real-world dataset to test the suggested approach to the Sd-SEG-LBFGS, SdLBFGS, and SGD algorithms: The CIFAR-10 dataset is available at [41]. From the CIFAR-10 dataset, 10,000 data points are selected at random. Each feature vector in the dataset has a dimension of 3072 and a label that can be between 0 and 9. We discuss our suggested approach for binary classification problem optimization in this work. Consequently, we split the data points into two classes: $z_n = 1$ for all other circumstances and $z_n = 0$ if the label is less than or equal to 4. Furthermore, updating a single function on each iteration is wasteful due to the high size of the dataset in terms of both memory cost and arithmetic complexity. As a result, we combine 100 single functions into a single new function. There are currently 100 distinct functions for the regrouped objective function. The algorithm's termination number is set to 300 times the total number of unique functions. This has resulted in significant memory savings and a significant decrease in the number of iterations needed to reach the desired point, which is close enough to the critical point. Furthermore, for numerical stability, we have employed the effective regularization method in (33).

Using the *CIFAR-10* dataset, Figure 2 illustrates how well different strategies work when solving the logistic regression. Our suggested approach often exhibits a steady learning curve with a progressive decline in the NOG. Still, it oscillates within local iterations as well. There are 100 local iterations in a single pass since there are 100 distinct functions. It is evident that NOG typically drops after one run through each of the distinct processes. Furthermore, it has demonstrated that, in contrast to other methods that have reached convergence, our suggested method has the lowest gradient magnitude and keeps falling.

## V. Conclusion

We have thoroughly examined CuREG-SR1's convergence characteristics and have produced some interesting findings. It should be mentioned that there may be issues when directly employing classical methods to assess the convergence qualities of SR1. For instance, the difference between the true and approximated Hessian is bounded in *Theorem 2*. The approximated Hessian converges to the genuine Hessian for SR1, though. This is because the shift is introduced by using the cubic regularized approach. We therefore restrict the regularized parameter, under which we have proven that there are $q - d$ superlinear steps in every $q \geq d + 1$ step for sufficiently large iteration numbers. In addition, we presented a brand-new incremental optimization technique based on SR1 and CuREG-SR1 and demonstrated how to effectively apply it to the resolution of complex issues.

## Acknowledgement

We would like to appreciate Dr. Chen Huiming, who is from Tsinghua University, for the helpful discussions.

## References

[1] H. Chen, et al., "LoSAC: An Efficient Local Stochastic Average Control Method for Federated Optimization". *ACM Trans. Knowl. Discov.* Data 17, 4, Article 58 (May 2023).

[2] H. Chen, et al. "Advancements in Federated Learning: Models, Methods, and Privacy." arXiv preprint arXiv:2302.11466 (2023).

[3] C. G. Broyden, "Quasi-Newton methods and their application to function minimisation,"*Mathematics of Computation*, 21(99), 368 - 381, 1969.

[4] H. Benson and Y. Shanno, "Cubic regularization in symmetric rank-1 quasi-Newton methods. Mathematical Programming Computation,"*Mathematics of Computation*, 10(4), 457 - 486, 2018.

[5] H. Khalfan, R. Byrd and R. Schnabel, "A Theoretical and Experimental Study of the Symmetric Rank-One Update,"*SIAM J. Optim.*, vol. 3, no. 1, pp. 1 - 24, Feb. 1993.

[6] C. G. Broyden, J. E. Dennis and J. Moré, "On the Local and Superlinear Convergence of Quasi-Newton Methods ,"*IMA J. Appl. Math.*, vol. 13, no. 3, pp. 223 - 245, Dec. 1973.

[7] A. Conn, N. Gould and P. Toint, "Convergence of quasi-Newton matrices generated by the symmetric rank one update,"*Mathematical Programming*, vol. 50, no. 1-3, pp. 177 - 195, Mar. 1991.

[8] J. E. Dennis and J. Moré, "A Characterization of Superlinear Convergence and Its Application to Quasi-Newton Methods,"*Mathematics of Computation*, Vol. 28, No. 126, pp. 549 - 560, Apr., 1974.

[9] W. C. Davidon, "Variable metric method for minimization,"*SIAM J. Optim.*, Vol. 1, No. 1, pp. 1 - 17, 1991.

[10] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms 2. the new algorithm ,"*IMA J. Appl. Math.*, vol. 6, no. 1, pp. 222 - 231, 1970.

[11] D. Xu, D. Jian and C. Zhang " Convergence of Quasi-Newton Method for Fully Complex-Valued Neural Networks , "*Neural Processing Letters*, vol. 46, no. 3, pp. 961 - 968, Dec. 2017.

[12] C. Cartis, N. Gould and P. Toint, "Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results, "*Mathematical Programming*, vol. 127, no. 2, pp. 245 - 295, April. 2011.

[13] Y. Nesterov and B. Polyak "Cubic regularization of Newton method and its global performance, "*Mathematical Programming*, vol. 108, no. 1, pp. 177 - 205, Aug. 2006.

[14] A. Mokhtari and A. Ribeiro, "RES: Regularized Stochastic BFGS Algorithm,"*IEEE Trans. Signal Process.*, vol. 62, no. 23, pp. 6089 - 6104, Dec.1, 2014.

[15] X. Wang, S. Ma, D. Goldfarb and W. Liu, "Stochastic Quasi-Newton Methods for Nonconvex Stochastic Optimization,"*SIAM J. Optim.*, vol. 27, no. 2, pp. 927 - 956, 2017.

[16] A. Mokhtari, M. Eisen and A. Ribeiro, "IQN: An Incremental Quasi-Newton Method with Local Superlinear Convergence Rate,"*SIAM J. Optim.*, vol. 28, no. 2, pp. 1670 - 1698, 2018.

[17] R. H. Byrd, S. L. Hansen, Jorge Nocedal, and Y. Singer, "A Stochastic Quasi-Newton Method for Large-Scale Optimization,"*SIAM J. Optim.*, vol. 26, no. 2, pp. 1008 - 1031, 2016.

[18] H. Chen and W. H. Lam, "Training Based Two-Step Channel Estimation in Two-Way MIMO Relay Systems,"*IEEE Trans. on Veh. Tech.*, vol. 67, no. 3, 2193 - 2205, Mar.2018.

[19] H. Chen and W. H. Lam, "Training design and two-stage channel estimation for correlated two-way MIMO relay systems,"*International Conference on Ubiquitous and Future Networks (ICUFN)* , Vol. 67 , Vienna, June. 2016, pp. 307 - 312.

[20] L. Bottou, F. E. Curtis and J. Nocedal, "Optimization Methods for Large-Scale Machine Learning,"*SIAM Rev.*, vol. 60, no. 2, pp. 223 - 311, 2018.

[21] M. J. D. Powell, "Algorithms for nonlinear constraints that use lagrangian functions,"*Math. Programming*, vol. 14, no. 1, pp. 224 - 248, Dec., 1978.

[22] R. Fletcher, "Practical Methods of Optimization,"*Wiley, Chichester*, 1987.

[23] K. Levenberg, "A method for the solution of certain problems in least squares,"*Quart. Appl. Math.*, vol. 2, no. 2, pp. 164 - 168, July, 1944.

[24] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters,"*SIAM J. Appl. Math.*, vol. 11, no. 2, pp. 431 - 441, June, 1963.

[25] D. Blatt and A. Hero, "Energy-based sensor network source localization via projection onto convex sets ,"*IEEE Trans. Signal Process.*, vol. 54, no. 9, pp. 3614 - 3619, Aug., 2006.

[26] S. Babacan, S. Nakajima and M. Do, "Bayesian Group-Sparse Modeling and Variational Inference ,"*IEEE Trans. Signal Process.*, vol. 62, no. 11, pp. 2906 - 2921, June, 2014.

[27] J. Taghia and A. Leijon, "Variational Inference for Watson Mixture Model,"in *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1886 - 1900, Sept., 2016.

[28] C. Wang and D. M. Blei, "Variational Inference in Nonconjugate Models,"*J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 1005 - 1031, Jan., 2013.

[29] C. Bishop, *Pattern Recognition and Machine Learning*, Springer New York., 2006.

[30] D. M. Blei, A. Kucukelbir and J. D. McAuliffe, "Variational Inference: A Review for Statisticians,"*J. Am. Statist. Assoc.*, vol. 112, no. 518, pp. 859 - 877, 2017.

[31] J. Paisley, D. M. Blei and M. I. Jordan, "Variational Bayesian Inference with Stochastic Search,"in *Proc. Int. Conf. Mach. Learn*, vol. 14, pp. 1363 - 1370, 2012.

[32] M. D. Hoffman, D. M. Blei, C. Wang and J. Paisley, "Stochastic Variational Inference,"*J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 1303 - 1347, Jan., 2013.

[33] H. Robbins and S. Monro, "A Stochastic Approximation Method,"*Ann. Math. Statist.*, vol. 22, no. 3, pp. 400 - 407, 1951.

[34] A. Mokhtari and A. Ribeiro, "Global Convergence of Online Limited Memory BFGS,"*J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 3151 - 3181, Jan., 2015.

[35] S. Ghadimi and G. Lan, "Stochastic First- and Zeroth-Order Methods for Nonconvex Stochastic Programming,"*SIAM J. Optim.*, vol. 23, no. 4, pp. 2341 - 2368, 2013.

[36] A. Nemirovski, A. Juditsky, G. Lan and A. Shapiro, "Robust Stochastic Approximation Approach to Stochastic Programming,"*SIAM J. Optim.*, vol. 19, no. 4, pp. 1574 - 1609, 2009.

[37] C. Dang and G. Lan, "Stochastic Block Mirror Descent Methods for Nonsmooth and Stochastic Optimization,"*SIAM J. Optim.*, vol. 25, no. 2, pp. 856 - 881, 2015.

[38] R. Johnson and T. Zhang, "Accelerating Stochastic Gradient Descent Using Predictive Variance Reduction,"in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 1, pp. 315 - 323, 2013.

[39] H.Chen, etc, "A Stochastic Quasi-Newton Method for Large-Scale Nonconvex Optimization with Applications, "submitted to *IEEE Trans. Neural Netw. Learn. Syst*.

[40]

[41] Alex Krizhevsky, *Learning Multiple Layers of Features from Tiny Images*, 2009.

[42] M.R. Boutell, J. Luo, X. Shen and C.M. Brown, "Learning multi-label scene classiffication,"*Pattern Recognition*, vol. 37, no. 9, pp. 1757-1771, 2004.

[43] D. Kingma and J. Ba, "Adam: A method for stochastic optimization", in *3rd International Conference for Learning Representations, 2015*.

[44] T. Tieleman and G. Hinton, "Lecture 6.5 - RMSProp, COURSERA: Neural Networks for Machine Learning", Technical report, 2012.

[45] Huiming Chen, Huandong Wang, Quanming Yao, Yong Li, Depeng Jin, and Qiang Yang. "LoSAC: An Efficient Local Stochastic Average Control Method for Federated Optimization". *ACM Trans. Knowl. Discov.* Data 17, 4, Article 58 (May 2023).

[46] Chen, Huiming, et al. "Advancements in Federated Learning: Models, Methods, and Privacy." arXiv preprint arXiv:2302.11466 (2023).