

Gradient Coding through Iterative Block Leverage Score Sampling

Neophytos Charalambides^μ, Mert Pilanci^σ, and Alfred O. Hero III^μ

^μEECS Department University of Michigan ^σEE Department Stanford University

Email: neochara@umich.edu, pilanci@stanford.edu, hero@umich.edu

Abstract

We generalize the leverage score sampling sketch for ℓ_2 -subspace embeddings, to accommodate sampling subsets of the transformed data, so that the sketching approach is appropriate for distributed settings. This is then used to derive an approximate coded computing approach for first-order methods; known as gradient coding, to accelerate linear regression in the presence of failures in distributed computational networks, *i.e.* stragglers. We replicate the data across the distributed network, to attain the approximation guarantees through the induced sampling distribution. The significance and main contribution of this work, is that it unifies randomized numerical linear algebra with approximate coded computing, while attaining an induced ℓ_2 -subspace embedding through uniform sampling. The transition to uniform sampling is done without applying a random projection, as in the case of the subsampled randomized Hadamard transform. Furthermore, by incorporating this technique to coded computing, our scheme is an iterative sketching approach to approximately solving linear regression. We also propose weighting when sketching takes place through sampling with replacement, for further compression.

Index Terms

Leverage Score Sampling, Low-rank Approximations, Least-squares Regression, Sampling, Randomized Algorithms, Coded Computing, Stragglers, Erasure-Coding, Replication-Coding.

I. INTRODUCTION

In this work we bridge two disjoint areas, to accelerate first-order methods distributively, while focusing on linear regression. Specifically, we propose a framework in which *Randomized Numerical Linear Algebra* (RandNLA) sampling algorithms can be used to devise *Coded Computing* (CC) schemes. We focus on the task of ℓ_2 -subspace embedding (ℓ_2 -s.e.); through leverage score sampling, and distributed gradient computation; which is referred to as *gradient coding* (GC).

Traditional numerical linear algebra algorithms are deterministic. For example, inverting a full-rank matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ requires $O(N^3)$ arithmetic operations by performing Gaussian elimination, as does naive matrix multiplication. The fastest known algorithm which multiplies two $N \times N$ matrices, requires $O(N^\omega)$ operations; for $\omega < 2.373$ [?], [?]. Other important problems are computing the determinant, singular and eigenvalue decompositions, SVD, QR and Cholesky factorizations.

Although these deterministic algorithms run in polynomial time and are numerically stable, their exponents make them prohibitive for many applications in scientific computing and machine learning, when N is in the order of millions or billions [?], [?]. To circumvent this issue, one can perform these algorithms on a significantly smaller approximation. Specifically, for a matrix $\mathbf{S} \in \mathbb{R}^{r \times N}$ with $r \ll N$, we apply the deterministic algorithm on the surrogate $\hat{\mathbf{A}} = \mathbf{S}\mathbf{A} \in \mathbb{R}^{r \times d}$. The matrix \mathbf{S} is referred to as a “dimension-reduction” or a “sketching” matrix, and $\hat{\mathbf{A}}$ is a “sketch” of \mathbf{A} , which contains as much information about \mathbf{A} as possible. For instance, when multiplying $\mathbf{A} \in \mathbb{R}^{L \times N}$ and $\mathbf{B} \in \mathbb{R}^{N \times M}$, we apply a carefully chosen $\mathbf{S} \in \mathbb{R}^{r \times N}$ on each to get

$$\overbrace{\begin{pmatrix} \mathbf{A} \end{pmatrix}}^{L \times N} \cdot \overbrace{\begin{pmatrix} \mathbf{B} \end{pmatrix}}^{N \times M} \approx \overbrace{\begin{pmatrix} \hat{\mathbf{A}} \end{pmatrix}}^{L \times r} \cdot \overbrace{\begin{pmatrix} \hat{\mathbf{B}} \end{pmatrix}}^{r \times M}$$

for $\hat{\mathbf{A}} = \mathbf{A}\mathbf{S}^\top$ and $\hat{\mathbf{B}} = \mathbf{S}\mathbf{B}$. Thus, naive matrix multiplication now requires $O(LMr)$ operations; instead of $O(LMN)$. Such approaches have been motivated by the Johnson-Lindenstrauss lemma [?], and require low complexity.

A multitude of other problems, such as k -means clustering [?], [?], [?] and computing the SVD of a matrix [?], [?], [?], [?], make use of this idea; in order to accelerate computing accurate approximate solutions. We refer the reader to the following monographs and comprehensive surveys on the rich development of RandNLA [?], [?], [?], [?], [?], [?], [?], an interdisciplinary field that exploits randomization as a computational resource; to develop improved algorithms for large-scale linear algebra problems.

The problem of ℓ_2 -s.e.; a form of spectral approximation of a matrix, has been extensively studied in RandNLA. The main techniques for constructing appropriate sketching ℓ_2 -s.e. matrices, are performing a random projection or row-sampling. Well-known choices of \mathbf{S} for reducing the effective dimension N to r include: i) *Gaussian projection*; for a matrix $\Theta \sim \mathcal{N}(0, 1)$ define $\mathbf{S} = \frac{1}{\sqrt{r}}\Theta$, ii) *leverage score sampling*; sample with replacement r rows from the matrix according to its normalized leverage score distribution and rescale them appropriately, iii) *Subsampled Hadamard Transform (SRHT)*; apply a Hadamard transform and a random signature matrix to judiciously make the leverage scores approximately uniform and then follow similar steps to the leverage score sampling procedure.

In this paper, we first generalize ii) to appropriately sample *submatrices* instead of rows to attain a ℓ_2 -s.e guarantee. We refer to such approaches as *block sampling*. Throughout this paper, sampling is done with replacement (w.r.). Sampling blocks has been explored in “block-iterative methods” for solving systems of linear equations [?], [?], [?], [?]. Our motivation in dealing with blocks rather than individual vectors, is the availability to invoke results that can be used to characterize the approximations of distributed computing networks, to speed up first-order methods, as sampling individual rows/columns is prohibitive in real-world environments. This in turn leads to an *iterative sketching* approach, which has been well studied in terms of second-order methods [?], [?], [?]. By iterative sketching, we refer to an iterative algorithm which uses a new sketch $\mathbf{S}_{[s]}$ at each iteration. The scenario where a single sketch \mathbf{S} is applied before the iterative process, is referred to as the “sketch-and-solve paradigm” [?].

Second, we propose a general framework which incorporates our sketching algorithm into a CC approach. This framework accommodates a central class of sketching algorithms, that of importance (block) sampling algorithms (e.g. CUR decomposition [?], CR-multiplication [?]). Coded computing is a novel computing paradigm that utilizes coding theory to effectively inject and leverage data and computation redundancy to mitigate errors and slow or non-responsive servers; known as *stragglers*, among other fundamental bottlenecks, in large-scale distributed computing. In our setting, the straggling effect is due to computations being communicated over *erasure channels*, whose erasure probability follows a probability distribution which is central to the CC probabilistic model. The seminal work of [?] which first introduced CC, focused on exact matrix-vector multiplication and data shuffling. More recent works deal with recovering good approximations, while some have utilized techniques from RandNLA; e.g. [?], [?], [?], [?], [?], [?]. Our results are presented in terms of the standard CC probabilistic model proposed in [?], though they extend to any computing network comprised of a central server and computing nodes, referred to as *servers*.

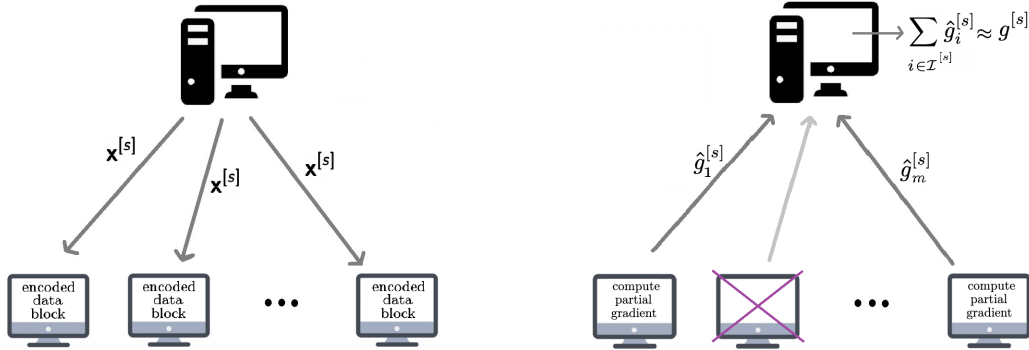


Fig. 1: Schematic of our approximate GC scheme, at iteration s . Each server has an encoded block of data, of which they compute the gradient once they receive the updated parameters $\mathbf{x}^{[s]}$. The central server then aggregates a subset of all the gradients $\{\hat{g}_j^{[s]}\}_{j=1}^m$, indexed by $\mathcal{I}^{[s]}$, to approximate the gradient $g^{[s]}$. At each iteration we expect a different index set $\mathcal{I}^{[s]}$, which leads to iterative sketching.

To mitigate stragglers, we appropriately encode and replicate the data blocks, which leads to accurate CC estimates. In contrast to previous works which simply replicate each computational task or data block the same number of times [?], [?], [?], [?], we replicate blocks according to their *block leverage scores*. Consequently, this induces a non-uniform sampling distribution in the aforementioned CC model; which is an approximation to the normalized block leverage scores. A drawback of using RandNLA techniques is that exact computations are not recoverable, though our method does not require a decoding step, a task of high complexity and a prevalent bottleneck in CC. For more details on the various directions of CC, the reader is referred to the monographs [?], [?].

The central idea of our approach is that non-uniform importance sampling can be emulated, by replicating tasks across the network’s servers, who communicate through an erasure channel. The tasks’ computation times follow a runtime distribution [?], which along with a prespecified gradient transmission “ending time” T , determine the number of replications per task across the network. Though similar ideas have been proposed [?], [?], [?], [?]; where sketching has been incorporated into CC, this is the first time redundancy is introduced through RandNLA; as opposed to compression, to obtain approximation guarantees. In terms of CC, though uniform replication is a very powerful technique, it does not capture the relevance between the information of the dataset. We capture such information through replication and rescaling according to the block leverage scores. By then allowing *uniform* sampling of these blocks, we attain a spectral approximation. In the CC setting this then corresponds to an iterative

ℓ_2 -s.e. sketching method. The shortcoming of this approach, which is the cost we pay for guaranteeing a spectral approximation through uniform sampling, is that we expect to require a large amount of servers; when the underlying sampling distribution is non-uniform.

In Appendix ?? we discuss how further compression can be attained by introducing *weighting*, while guaranteeing the same results when first and second order methods are used for linear regression in the sketch-and-solve paradigm (Proposition ?? and Corollary ??). We also show that in terms of the expected reduced dimension, we have minimal benefit when the block leverage scores distribution is uniform (Theorem ??). This further justifies the fact that sharper decays in leverage scores lead to more accurate algorithms [?].

All completed jobs that are received by the central server are aggregated to get the final gradient approximation, at each iteration of the descent method being carried out. Thus, unlike most CC schemes, ours does not store completed jobs which will not be accounted for. Our method sacrifices accuracy for speed of computation, and the inaccuracy is quantified in terms of the resulting ℓ_2 -s.e. (Theorem ??). Specifically, the computations of the responsive servers will correspond to sampled *block* computation tasks of our proposed generalization to leverage score sampling, summarized in Algorithm ?. Approximate coded computations is a current interest in information-theory, as it is conceivable that data dependent approximation schemes could lead to faster inexact but accurate solutions, at a lower computational cost [?].

To summarize, our main contributions are: 1) propose *block* leverage score sketching, to accommodate block sampling for ℓ_2 -s.e., 2) provide theoretical guarantees for the algorithm's performance, 3) show the significance of weighting; when our *weighted* sketching algorithms are applied in iterative first and second order methods, 4) propose *expansion networks*; which use the sampling distribution to determine how to replicate and distribute blocks in the CC framework — this unifies the disciplines of RandNLA and CC — where replication and uniform sampling (without a random projection) result in a spectral approximation, 5) show how expansion networks are used for approximate distributed *steepest descent* (SD); and approach the optimal solution with unbiased gradient estimators in a similar manner to *batch stochastic steepest descent* (SSD), 6) experimental justification on the performance of our algorithm on artificial datasets with non-uniform induced sampling distributions.

The paper is organized as follows. In ?? we present the notation which will be used, and review necessary background. In ?? we present related works, in terms of CC. In ?? we present our sketching algorithm and its theoretical approximation guarantees. In ?? and ?? we give a framework for which our algorithm; as well as potentially other importance sampling algorithms, can be used to devise CC schemes. This is where we introduce redundancy through RandNLA, which has not been done before. In ?? we summarize our GC scheme, and in ?? we give a brief synopsis of our main results and tie everything together. In ?? we show how our scheme relates to *approximate GC*. We conclude with experimental evaluations in ?? on fabricated data with highly non-uniform underlying sampling distributions, to convey the maximum benefit of what we propose.

II. NOTATION AND BACKGROUND

We denote $\mathbb{N}_n := \{1, 2, \dots, n\}$, and $X_{\{n\}} = \{X_i\}_{i=1}^n$; where X could be replaced by any variable. We use \mathbf{A}, \mathbf{B} to denote real matrices, \mathbf{b}, \mathbf{x} real column vectors, \mathbf{I}_n the $n \times n$ identity matrix, $\mathbf{0}_{n \times m}$ and $\mathbf{1}_{n \times m}$ respectively the $n \times m$ all zeros and all ones matrices, and by \mathbf{e}_i the standard basis column vector whose dimension will be clear from the context. The largest eigenvalue of a matrix \mathbf{M} , is denoted by $\lambda_1(\mathbf{M})$. By $\mathbf{A}_{(i)}$ we denote the i^{th} row of \mathbf{A} , by $\mathbf{A}^{(j)}$ its j^{th} column, by \mathbf{A}_{ij} the value of \mathbf{A} 's entry in position (i, j) , and by x_i the i^{th} element of \mathbf{x} . The rounding function to the nearest integer is expressed by $\lfloor \cdot \rfloor$, i.e. $\lfloor a \rfloor = \lfloor a + 1/2 \rfloor$ for $a \in \mathbb{R}$. Disjoint unions are represented by \sqcup ; e.g. $\mathbb{Z} = \{j : j \text{ is odd}\} \sqcup \{j : j \text{ is even}\}$, and we define \uplus as the addition of multisets; e.g. $\{1, 2, 3\} \uplus \{3, 4\} = \{1, 2, 3, 3, 4\}$. The diagonal matrix with real entries $a_{\{n\}}$ is expressed as $\text{diag}(a_{\{n\}})$.

We partition vectors and matrices across their rows into K submatrices, in a way that no submatrix differs from another by more than one row. For simplicity, we assume that K divides the number of rows N . That is, for a ℓ_2 -s.e. of $\mathbf{A} \in \mathbb{R}^{N \times d}$ with target $\mathbf{b} \in \mathbb{R}^N$, we assume $K \mid N$ and the “size of each partition is $\tau = N/K$.”¹ We partition \mathbf{A}, \mathbf{b} across their rows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1^\top & \dots & \mathbf{A}_K^\top \end{bmatrix}^\top \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} \mathbf{b}_1^\top & \dots & \mathbf{b}_K^\top \end{bmatrix}^\top \quad (1)$$

where $\mathbf{A}_i \in \mathbb{R}^{\tau \times d}$ and $\mathbf{b}_i \in \mathbb{R}^\tau$ for all $i \in \mathbb{N}_K$. Partitions, are referred to as *blocks*. Throughout the paper we consider the case where $N \gg d$. For \mathbf{A} full-rank, its SVD is $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^\top$, where $\mathbf{U} \in \mathbb{R}^{N \times d}$ is its reduced left orthonormal basis.

Matrix \mathbf{A} represents a dataset \mathcal{D} of N samples with d features, and \mathbf{b} the corresponding labels of the data points. The partitioning (??) corresponds to K sub-datasets $\mathcal{D}_{\{K\}}$, i.e. $\mathcal{D} = \bigsqcup_{j=1}^K \mathcal{D}_j$. Our results are presented in terms of an arbitrary partition $\mathbb{N}_N = \bigsqcup_{\ell=1}^K \mathcal{K}_\ell$, for \mathbb{N}_N the index set of the rows of \mathbf{A} and \mathbf{b} . The index subsets $\mathcal{K}_{\{K\}}$ indicate which data samples are in each sub-dataset. By $\mathbf{A}_{(\mathcal{K}_\ell)}$, we denote the submatrix of \mathbf{A} comprised of the rows indexed by \mathcal{K}_ℓ . That is, for $\mathbf{I}_{(\mathcal{K}_\ell)}$ the restriction of \mathbf{I}_N to only include its rows corresponding to \mathcal{K}_ℓ , we have $\mathbf{A}_{(\mathcal{K}_\ell)} = \mathbf{I}_{(\mathcal{K}_\ell)} \cdot \mathbf{A}$. By \mathcal{K}_ℓ^i , we indicate that the ℓ^{th} block was sampled at trial i , i.e. the superscript i indicates the sampling trial and the subscript $\ell \in \mathbb{N}_K$ which block was sampled at that trial. Also, by $j(i)$ we denote the index of the submatrix which was sampled at the i^{th} sampling trial, i.e. $\mathcal{K}_{j(i)} = \mathcal{K}_{j(i)}^i$. The complement of \mathcal{K}_ℓ is denoted by $\bar{\mathcal{K}}_\ell$; i.e. $\bar{\mathcal{K}}_\ell = \mathbb{N}_N \setminus \mathcal{K}_\ell$, for which $\mathbf{U}_{(\bar{\mathcal{K}}_\ell)}^\top \mathbf{U}_{(\mathcal{K}_\ell)} = \mathbf{I}_d - \mathbf{U}_{(\mathcal{K}_\ell)}^\top \mathbf{U}_{(\bar{\mathcal{K}}_\ell)}$.

¹If $K \nmid N$, we appropriately append zero vectors/entries until this is met. It is not required that all blocks have the same size, though we discuss this case to simplify the presentation. One can easily extend our results to blocks of varying sizes, and use the analysis from [?] to determine the optimal size of each partition.

Sketching matrices are represented by \mathbf{S} and $\tilde{\mathbf{S}}_{[s]}$. The script $[s]$ indexes an iteration $s = 0, 1, 2, \dots$ which we drop when it is clear from the context. We will be reducing dimension N to r , i.e. $\mathbf{S} \in \mathbb{R}^{r \times N}$. Sampling matrices are denoted by $\mathbf{\Omega} \in \{0, 1\}^{r \times N}$, and diagonal rescaling matrices by $\mathbf{D} \in \mathbb{R}^{N \times N}$.

Approximate block sampling distributions to $\Pi_{\{K\}}$ are denoted by $\tilde{\Pi}_{\{K\}}$, and the distributions induced through expansion networks by $\bar{\Pi}_{\{K\}}$. We quantify the difference between distributions $\Pi_{\{K\}}$ and $\tilde{\Pi}_{\{K\}}$ by the distortion metric $d_{\Pi, \tilde{\Pi}} := \frac{1}{K} \sum_{i=1}^K |\Pi_i - \tilde{\Pi}_i|$, which is the ℓ_1 distortion between $\Pi_{\{K\}}$ and $\tilde{\Pi}_{\{K\}}$, e.g. [?].

A. Least Squares Approximation

Least squares approximation is a technique to find an approximate solution to a system of linear equations that has no exact solution, and has found applications in many fields [?]. Consider the system $\mathbf{A}\mathbf{x} = \mathbf{b}$, for which we want to find an approximation to the best-fitted

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^d} \left\{ L_{ls}(\mathbf{A}, \mathbf{b}; \mathbf{x}) := \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2 \right\}, \quad (2)$$

which objective function L_{ls} has gradient

$$g^{[s]} = \nabla_{\mathbf{x}} L_{ls}(\mathbf{A}, \mathbf{b}; \mathbf{x}^{[s]}) = 2\mathbf{A}^\top (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}). \quad (3)$$

We refer to the gradient of the block pair $(\mathbf{A}_i, \mathbf{b}_i)$ from (??) as the i^{th} *partial gradient*; $g_i^{[s]} = \nabla_{\mathbf{x}} L_{ls}(\mathbf{A}_i, \mathbf{b}_i; \mathbf{x}^{[s]})$. Existing exact methods find a solution vector \mathbf{x}^* in $O(Nd^2)$ time, where $\mathbf{x}^* = \mathbf{A}^\dagger \mathbf{b}$. In Subsection ?? we focus on approximating the optimal solution \mathbf{x}^* by using our methods, via distributive SD/SSD and iterative sketching. What we present also accommodates regularizers of the form $\lambda \|\mathbf{x}\|_2^2$, though to simplify our expressions, we only consider (??).

B. Steepest Descent

When considering a minimization problem with a convex differentiable objective function $L: \mathbb{R}^d \rightarrow \mathbb{R}$, we select an initial $\mathbf{x}^{[0]} \in \mathbb{R}^d$ and repeat at iteration $s + 1$: $\mathbf{x}^{[s+1]} \leftarrow \mathbf{x}^{[s]} - \xi_s \cdot \nabla_{\mathbf{x}} L(\mathbf{x}^{[s]})$; for $s = 0, 1, 2, \dots$, until a prespecified termination criterion is met. The parameter $\xi_s > 0$ is the corresponding step-size, which may be adaptive or fixed. To guarantee convergence of L_{ls} , one can select $\xi_s = 2/\sigma_{\max}(\mathbf{A})^2$ for all iterations, though this is too conservative.

C. Leverage Scores

Many sampling algorithms select data points according to their *leverage scores* [?], [?]. The leverage scores of \mathbf{A} measure the extent to which the vectors of its orthonormal basis \mathbf{U} are correlated with the standard basis, and define the key structural non-uniformity that must be dealt with when developing fast randomized matrix algorithms; as they characterize the importance of the data points. Leverage scores defined as $\ell_i := \|\mathbf{U}_{(i)}\|_2^2$, and are agnostic to any particular basis, as they are equal to the diagonal entries of the projection matrix $P_{\mathbf{A}} = \mathbf{A}\mathbf{A}^\dagger = \mathbf{U}\mathbf{U}^\top$. The *normalized leverage scores* of \mathbf{A} are

$$\pi_i := \|\mathbf{U}_{(i)}\|_2^2 / \|\mathbf{U}\|_F^2 = \|\mathbf{U}_{(i)}\|_2^2 / d \quad \text{for each } i \in \mathbb{N}_N, \quad (4)$$

and $\pi_{\{N\}}$ form a sampling probability distribution; as $\sum_{i=1}^N \pi_i = 1$ and $\pi_i \geq 0$ for all i . This induced distribution has proven to be useful in linear regression [?], [?], [?], [?].

The *normalized block leverage scores*, introduced independently in [?], [?], are the sum of the normalized leverage scores of the subset of rows constituting the block. Analogous to (??), considering the partitioning of \mathcal{D} according to $\mathcal{K}_{\{K\}}$, the *normalized block leverage scores* of \mathbf{A} are defined as

$$\Pi_l := \|\mathbf{U}_{(\mathcal{K}_l)}\|_F^2 / \|\mathbf{U}\|_F^2 = \|\mathbf{U}_{(\mathcal{K}_l)}\|_F^2 / d = \sum_{j \in \mathcal{K}_l} \pi_j \quad \text{for each } l \in \mathbb{N}_K. \quad (5)$$

A related notion is that of the *Frobenius block scores*, which in the case of a partitioning as in (??); are $\|\mathbf{A}_\iota\|_F^2$ for each $\iota \in \mathbb{N}_K$, which scores have been used for *CR*-multiplication [?], [?]. In our context, the block leverage scores of \mathbf{A} ; are the Frobenius block scores of \mathbf{U} .

A drawback of using leverage scores, is that calculating them requires $O(Nd^2)$ time. To alleviate this, one can instead settle for relative-error approximations which can be approximated much faster, e.g. [?] does so in $O(Nd \log N)$ time. In particular, we can consider approximate normalized scores $\tilde{\Pi}_{\{K\}}$ where $\tilde{\Pi}_i \geq \beta \Pi_i$ for all i , for some misestimation factor $\beta \in (0, 1]$. Since $\Pi_{\{K\}}$ and $\tilde{\Pi}_{\{K\}}$ are identical if and only if $\beta = 1$, a higher β implies the approximate distribution is more accurate. When we want to specify that a misestimation factor is for a specific distribution, we accompany it by a corresponding subscript; e.g. $\beta_{\tilde{\Pi}}$.

D. Subspace Embedding

Our approach to approximating (??), is to apply a ℓ_2 -s.e sketching matrix $\mathbf{S} \in \mathbb{R}^{r \times N}$ on \mathbf{A} . Recall that $\mathbf{S} \in \mathbb{R}^{r \times N}$ is a $(1 \pm \epsilon)$ ℓ_2 -subspace embedding of the column-space of \mathbf{A} , if

$$(1 - \epsilon)\|\mathbf{Ax}\|_2^2 \leq \|\mathbf{SAx}\|_2^2 \leq (1 + \epsilon)\|\mathbf{Ax}\|_2^2 \quad (6)$$

for all $\mathbf{x} \in \mathbb{R}^d$, w.h.p. [?]. Notice that such an \mathbf{S} , is also a $(1 \pm \epsilon)$ ℓ_2 -s.e. of \mathbf{U} , as $\{\mathbf{Ax} : \mathbf{x} \in \mathbb{R}^d\} = \{\mathbf{Uy} : \mathbf{y} \in \mathbb{R}^d\}$. This implies that (??) is equivalent to

$$(1 - \epsilon)\|\mathbf{y}\|_2^2 = (1 - \epsilon)\|\mathbf{Uy}\|_2^2 \leq \|\mathbf{SUy}\|_2^2 \leq (1 + \epsilon)\|\mathbf{Uy}\|_2^2 = (1 + \epsilon)\|\mathbf{y}\|_2^2 \quad (7)$$

for all $\mathbf{y} \in \mathbb{R}^d$. The upper and lower bounds on $\|\mathbf{SUy}\|_2^2$ respectively imply

$$\mathbf{y}^\top ((\mathbf{SU})^\top \mathbf{SU} - \mathbf{I}_d) \mathbf{y} \leq \epsilon \|\mathbf{y}\|_2^2 \quad \text{and} \quad \mathbf{y}^\top (\mathbf{I}_d - (\mathbf{SU})^\top \mathbf{SU}) \mathbf{y} \leq \epsilon \|\mathbf{y}\|_2^2$$

thus, a simplified condition for a ℓ_2 -s.e. of \mathbf{A} is

$$\Pr [\|\mathbf{I}_d - \mathbf{U}^\top \mathbf{S}^\top \mathbf{SU}\|_2 \leq \epsilon] \geq 1 - \delta \quad (8)$$

for a small $\delta \geq 0$.

For the overdetermined system $\mathbf{Ax} = \mathbf{b}$, we require $r > d$, and in the sketch-and-solve paradigm the objective is to determine an $\hat{\mathbf{x}}$ which satisfies

$$(1 - \epsilon)\|\mathbf{Ax}^* - \mathbf{b}\|_2 \leq \|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2 \leq (1 + \epsilon)\|\mathbf{Ax}^* - \mathbf{b}\|_2, \quad (9)$$

where $\hat{\mathbf{x}}$ is an approximate solution to the modified least squares problem

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathbb{R}^d} \left\{ L_{\mathbf{S}}(\mathbf{S}, \mathbf{A}, \mathbf{b}; \mathbf{x}) := \|\mathbf{S}(\mathbf{Ax} - \mathbf{b})\|_2^2 \right\}. \quad (10)$$

If (??) is met, we get w.h.p. the approximation characterizations:

- 1) $\|\mathbf{A}\hat{\mathbf{x}} - \mathbf{b}\|_2 \leq \frac{1+\epsilon}{1-\epsilon} \|\mathbf{Ax}^* - \mathbf{b}\|_2 \leq (1 + O(\epsilon)) \|\mathbf{Ax}^* - \mathbf{b}\|_2$
- 2) $\|\mathbf{A}(\mathbf{x}^* - \hat{\mathbf{x}})\|_2 \leq \epsilon \|(\mathbf{I}_N - \mathbf{UU}^\top) \mathbf{b}\|_2 = \epsilon \|\mathbf{b}^\perp\|_2$

where $\mathbf{b}^\perp = \mathbf{b} - \mathbf{Ax}^*$ is orthogonal to the column span of \mathbf{A} , i.e. $\mathbf{A}^\top \mathbf{b}^\perp = \mathbf{0}_{d \times 1}$.

E. Coded Computing Probabilistic Model

In GC (Figure ??), there is a central server who shares the K disjoint subsets $\mathcal{D}_{\{K\}}$ of \mathcal{D} among m homogeneous² servers, to facilitate computing the solution of minimization problems with differentiable additively separable objective functions, e.g. (??):

$$\mathbf{x}^* = \arg \min_{\mathbf{x} \in \mathbb{R}^d} \left\{ \|\mathbf{Ax} - \mathbf{b}\|_2^2 = \sum_{j=1}^K L_{l_s}(\mathcal{D}_j; \mathbf{x}) \right\}. \quad (11)$$

Since the objective function $L_{l_s}(\mathbf{A}, \mathbf{b}; \mathbf{x})$ is additively separable, it follows that $g^{[s]} = \sum_{j=1}^K g_j^{[s]}$. The objective function's gradient is updated in a distributed manner; while only requiring q servers to respond, i.e. it is robust to $m - q$ stragglers. This is achieved through an encoding of the computed partial gradients by the servers, and a decoding step once q servers have sent back their encoded computation.

We consider the probabilistic computational model introduced in [?], which is the standard CC paradigm; and is central to our framework. This model assumes the existence of a *mother runtime distribution* $F(t)$, with a corresponding probability density function $f(t)$. Let T_0 be the time it takes a single machine to complete its computation, and define $F(t) := \Pr[T_0 \leq t]$. We further assume that the runtime distribution of the subtasks, with random amount of completion time T^i , are a scaled distribution of $F(t)$. That is, when all servers have a computational task of size τ , computing a τ/N -fraction of the overall computation; follows the runtime distribution $\tilde{F}(t) := F(t\tau/N) = \Pr[T^i \leq t]$. In this work, we view the computations as being communicated to the central server over erasure channels, where the l^{th} server W_l has an erasure probability³

$$\phi(t) := 1 - \tilde{F}(t) = 1 - \Pr[W_l \text{ responds by time } t], \quad (12)$$

i.e. the probability that W_l is a straggler at time t is $\phi(t)$. All servers have the same erasure probability, as we are assuming they are homogeneous.

In our setting, there are two hyperparameters required for determining an expansion network. First, one needs to determine a time instance $t \leftarrow T$ after which the central server will stop receiving servers' computations.⁴ This may be decided by factors

²This means that they have the same computational power, independent and identically distributed statistics for the computing time of similar tasks; and expected response time.

³This is also known as the *survival function*: $\phi(t) = \int_t^\infty (1 - \tilde{f}(u)) du = 1 - \tilde{F}(t) = \Pr[T^i > t]$, for $\tilde{f}(t)$ the PDF corresponding to $\tilde{F}(t)$. The function $\phi(t)$ is monotonically decreasing.

⁴By $t \leftarrow T$, we mean T is a realization of the time variable t .

such as the system's limitations, number of servers, or an upper bound on the desired waiting time for the servers to respond. At time T , according to $\tilde{F}(t)$, the central server receives roughly $q(T) := \lfloor \tilde{F}(T) \cdot m \rfloor$ server computations. We refer to the prespecified time instance T after which the central server stops receiving computations; as the “ending time”. If the sketching procedure of the proposed sketching algorithm were to be carried out by a single server, there would be no benefit in setting T such that $q(T)\tau > N$, as the exact calculation could have taken place in the same amount of time. In distributed networks though there is no control over which servers respond, and it is not a major concern if $q(T)\tau$ is slightly over N ; as we still accelerate the computation. The trade-off between accuracy and waiting time t is captured in Theorem ??, for $q \leftarrow q(t)$ sampling trials. The second hyperparameter we need in order to design an expansion network, is the block size τ ; which is determined by K the number of partitions (??). Together, $q(T)$ and τ determine the ideal number of servers needed for our framework to perfectly emulate sampling according to the datas' block leverage scores $\Pi_{\{K\}}$.

III. CODED COMPUTING FROM RANDNLA

In this section, we first present our *block* leverage score sampling algorithm, which is more practical and can be carried out more efficiently than its vector-wise counterpart. Our ℓ_2 -s.e. result is presented in Theorem ?. By setting $\tau = 1$ and for $\beta = 1$, we get a known result for (exact) leverage score sampling.

In Subsection ?? we incorporate our block sampling algorithm into the CC probabilistic model described above, in which we leverage task redundancy to mitigate stragglers. Specifically, we show how to replicate computational tasks among the servers, under the integer constraints imposed by the physical system and the desired waiting time; to approximate the gradient at each iteration, in a way that emulates the sampling procedure of the sketch presented in Algorithm ?. In Subsection ?? we further elaborate on when a perfect emulation is possible, and how emulated block leverage score sampling can be improved when it cannot be done perfectly; through the proposed networks. In Subsection ?? we present our GC approach, and relate it to SD and SSD; which in turn implies convergence guarantees with appropriate step-sizes. Furthermore, at each iteration we have a different induced sketch, hence our procedure lies under the framework of iterative sketching. Specifically, we obtain gradients of multiple sketches of the data $(\tilde{\mathbf{S}}_{[1]}\mathbf{A}, \tilde{\mathbf{S}}_{[2]}\mathbf{A}, \dots, \tilde{\mathbf{S}}_{[n]}\mathbf{A})$ and iteratively refine the solution, where n can be chosen logarithmic in N . A schematic of our approach is provided in Figure ??, and in Appendix ?? we provide a concrete example of the induced sketching matrices resulting from the iterative process.

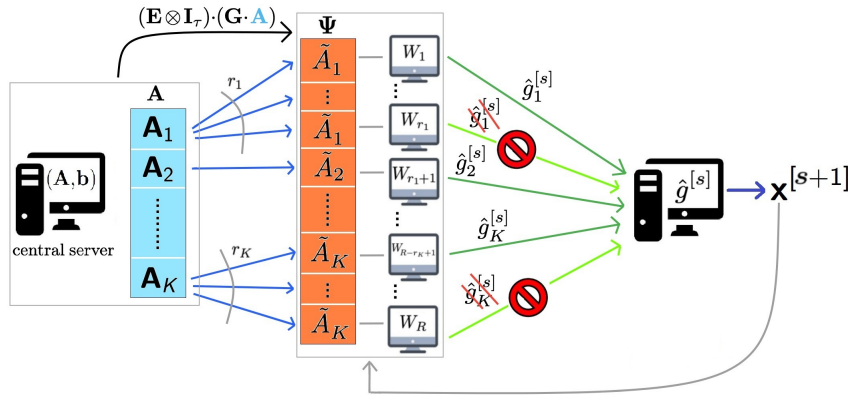


Fig. 2: Illustration of our GC approach, at iteration $s + 1$. The blocks of \mathbf{A} (and \mathbf{b}) are encoded through \mathbf{G} and then replicated through $\mathbf{E} \otimes \mathbf{I}_\tau$, where each block of the resulting Ψ is given to a single server. At this iteration, servers W_{r_1} and W_R are stragglers, and their computations are not received. The central server determines the estimate $\hat{g}^{[s]}$, and then shares $\mathbf{x}^{[s+1]}$ with all the servers. The resulting estimate is the gradient of the induced sketch, i.e. $\hat{g}^{[s]} = \nabla_{\mathbf{x}} L_S(\tilde{\mathbf{S}}_{[s]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[s]})$.

A. Related Work

Related works [?], [?], [?], [?] have utilized similar ideas to the GC approach we present. The paper titled “Anytime Coding for Distributed Computation” [?] proposes replicating subtasks according to the job, while [?] and [?] incorporate sketching into CC. It is worth noting that even though we focus on gradient methods in this paper; our approach also applies to second-order methods, as well as approximate matrix products through the *CR*-multiplication algorithm [?], [?], [?]. We briefly discuss this in Section ??.

The work of [?] deals with matrix-vector multiplication. Similar to our work, they also replicate the computational tasks a certain number of times; and stop the process at a prespecified instance. Here, the computation $\mathbf{A}\mathbf{x}$ for $\mathbf{A} \in \mathbb{R}^{N \times N}$ and $\mathbf{x} \in \mathbb{R}^N$ is broken up into C different tasks; prioritizing the smaller computations. The m servers are split up into c groups, which are asked to compute one of the tasks $\mathbf{y}_j = (\sum_{i \in \mathcal{J}_j} \sigma_i \mathbf{u}_i \mathbf{v}_i^\top) \mathbf{x}$, for $\mathbf{A} = \sum_{l=1}^N \sigma_l \mathbf{u}_l \mathbf{v}_l^\top$ the SVD representation of \mathbf{A} . Each task \mathbf{y}_j

is computed by the servers of the respective group, and $\mathbb{N}_N = \bigsqcup_{j=1}^s \mathcal{J}_j$ is a disjoint partitioning of the rank-1 outer-products of the SVD representation. The size of the j^{th} task is $|\mathcal{J}_j| = p_j$, which in our work is determined by the normalized block scores. The scores in our proposed schemes are motivated and justified by RandNLA, in contrast to the selection of the sizes p_j which is not discussed in [?]. Furthermore, the scheme of [?] requires a separate maximum distance separable code for each job y_j ; thus requiring multiple decodings, while we do not require a decoding step. Another drawback of [?] is that an integer program is set up to determine the optimal ending time, which the authors do not solve, while we determine a scheme for any desired ending time. Lastly, we note that the ℓ_2 -s.e. approximation guarantee of our method, depends on the ending time T .

In terms of sketching and RandNLA, the works of [?] and [?] utilize the *Count-Sketch* [?]; which relies on hashing. In “CodedSketch” [?], Count-Sketches are incorporated into the design of a variant of the improved “Entangled Polynomial Code” [?], to combine approximate matrix multiplication with straggler tolerance. The code approximates the submatrix blocks $\{\mathbf{C}_{i,j} : (i,j) \in \mathbb{N}_{k_1} \times \mathbb{N}_{k_2}\}$ of the final product matrix $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$ (matrix \mathbf{C} is partitioned k_1 times across its rows, and k_2 times across its columns), with an accuracy that depends on $\|\mathbf{C}_{i',j'}\|_F$ for all $(i',j') \in \mathbb{N}_{k_1} \times \mathbb{N}_{k_2}$. This prevents it from being applicable to applications that require accuracy guarantees without oracle knowledge of the outcome of each submatrix of the matrix product \mathbf{C} . This approach permits each block of \mathbf{C} to be approximately recovered, if a subset of the servers complete their tasks.

In “OverSketch” [?], redundancy is introduced through additional Count-Sketches, to mitigate the effect of stragglers in distributed matrix multiplication. In particular, the count-sketches $\tilde{\mathbf{A}} = \mathbf{A}\mathbf{S}$ and $\tilde{\mathbf{B}} = \mathbf{S}^\top \mathbf{B}$ of the two inputs \mathbf{A} and \mathbf{B} are computed, and are partitioned into submatrices of size $b \times b$. The $b \times b$ submatrices of the final product $\mathbf{C} = \mathbf{A} \cdot \mathbf{B}$ are then approximately calculated, by multiplying the corresponding row-block of $\tilde{\mathbf{A}}$ and column-block of $\tilde{\mathbf{B}}$; each of which is done by one server. The “OverSketch” idea has also been extended to distributed Newton Sketching [?] for convex optimization problems.

B. Block Leverage Score Sampling

In the leverage score sketch [?], [?], [?], [?], [?] we sample w.r. r rows according to $\pi_{\{N\}}$ (??), and then rescale each sampled row by $1/\sqrt{r\pi_i}$. Instead, we sample w.r. q blocks from (??) according to $\Pi_{\{K\}}$ (??), and rescale them by $1/\sqrt{q\Pi_i}$. The pseudocode of the *block leverage score sketch* is given in Algorithm ??, where we consider an approximate distribution $\tilde{\Pi}_{\{K\}}$ such that $\tilde{\Pi}_i \geq \beta\Pi_i$ for all i , for $\beta \in (0, 1]$ a dependent loss in accuracy [?], [?], [?]. The spectral guarantee of the sketching matrix $\tilde{\mathbf{S}}$ of Algorithm ?? is presented in Theorem ??. Iterative sketching in our distributed GC approach through Algorithm ??, corresponds to selecting a new sampling matrix $\tilde{\Omega}^{[s]}$ at each iteration through the servers’ responses, i.e. $\tilde{\mathbf{S}}_{[s]} = \tilde{\mathbf{D}} \cdot \tilde{\Omega}^{[s]}$ for each s .

Algorithm 1: Block Leverage Score Sketch

Input: $\mathbf{A} \in \mathbb{R}^{N \times d}$, $\tau = \frac{N}{K}$, $q = \frac{r}{\tau} > \frac{d}{\tau}$

Output: $\tilde{\mathbf{S}} \in \mathbb{R}^{r \times N}$, $\tilde{\mathbf{A}} \in \mathbb{R}^{r \times d}$

Initialize: $\Omega = \mathbf{0}_{q \times K}$, $\mathbf{D} = \mathbf{0}_{q \times q}$

Compute: (approximate) distribution $\tilde{\Pi}_{\{K\}}$ (??)

for $j = 1$ **to** q **do**

 sample w.r. i_j from \mathbb{N}_K , according to $\tilde{\Pi}_{\{K\}}$

$\Omega_{j,i_j} = 1$

$\mathbf{D}_{j,j} = \sqrt{\frac{\tau}{r\Pi_{i_j}}} = \sqrt{\frac{1}{q\Pi_{i_j}}}$

▷ equivalently $\Omega_{(j)} = \mathbf{e}_{i_j}^\top$

end

$\tilde{\Omega} \leftarrow \Omega \otimes \mathbf{I}_\tau$

$\tilde{\mathbf{D}} \leftarrow \mathbf{D} \otimes \mathbf{I}_\tau$

$\tilde{\mathbf{S}} \leftarrow \tilde{\mathbf{D}} \cdot \tilde{\Omega}$

▷ $\tilde{\mathbf{S}} = (\mathbf{D} \cdot \Omega) \otimes \mathbf{I}_\tau$

$\tilde{\mathbf{A}} \leftarrow \tilde{\mathbf{S}} \cdot \mathbf{A}$

Theorem 1. The sketching matrix $\tilde{\mathbf{S}}$ of Algorithm ?? is a $(1 \pm \epsilon)$ ℓ_2 -s.e of \mathbf{A} , according to (??). Specifically, for $\delta > 0$ and $q = \Theta\left(\frac{d}{\tau} \log(2d/\delta)/(\beta\epsilon^2)\right)$ we get

$$\Pr[\|\mathbf{I}_d - \mathbf{U}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{U}\|_2 \leq \epsilon] \geq 1 - \delta.$$

Proof. The main tool is a matrix Chernoff bound [?, Fact 1]. We define random matrices corresponding to the sampling process and bound their norm and variance, in order to apply the aforementioned Chernoff bound. The complete proof can be found in Appendix ??. \square

The importance of Theorem ?? extends beyond leverage score sampling. Specifically, one can apply a random projection to “flatten” the block leverage scores; i.e. they are all approximately equal, and then sample w.r. uniformly at random. This is the main idea behind the analysis of the SRHT [?], [?]. The trade-off between such algorithms and Algorithm ??, is computing the leverage scores explicitly vs. applying a random projection. Such sketching approaches which do not directly utilize the data, are referred to as “data oblivious sketches”, and are better positioned for handling high velocity streams; as well as highly unstructured and arbitrarily distributed data [?]. Multiplying the data by random matrix spreads the information in the rows of the matrix, such

that all rows are of equal importance; and the new matrix is “incoherent”. In Appendix ??, we show when Algorithm ?? and the corresponding block sampling counterpart of the SRHT [?] achieve the same asymptotic guarantees, for the same number of sampling trials q .

Next, we provide a sub-optimality result for non-iterative sketching of the block leverage score sketch for ordinary least squares:

$$\tilde{\mathbf{x}} = \arg \min_{\mathbf{x} \in \mathbb{R}^d} \left\{ L_{\mathbf{S}}(\tilde{\mathbf{S}}, \mathbf{A}, \mathbf{b}; \mathbf{x}) := \|\tilde{\mathbf{S}}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2 \right\}. \quad (13)$$

which follows from the results of [?]. Specifically, following the proof of [?, Theorem 1]; which is based on a reduction from statistical minimax theory combined with information-theoretic bounds and an application of Fanos inequality, we simply need to upper bound $\|\mathbb{E}[\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}}\tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}}]\|_2$.

Corollary 1. *For any full-rank data matrix $\mathbf{A} \in \mathbb{R}^{N \times d}$ with a noisy observation model $\mathbf{b} = \mathbf{A}\mathbf{x}^\bullet + \mathbf{w}$ where $\mathbf{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$, the optimal least squares solution \mathbf{x}^\star of (??); has prediction error $\mathbb{E}[\|\mathbf{A}(\mathbf{x}^\bullet - \mathbf{x}^\star)\|_2^2] \lesssim \frac{\sigma^2 d}{N}$. On the other hand, [?, Theorem 1] implies that any estimate $\tilde{\mathbf{x}}$ based on the sketched system $(\tilde{\mathbf{S}}\mathbf{A}, \tilde{\mathbf{S}}\mathbf{b})$ produced in Algorithm ?? with sampling probabilities $\Pi_{\{K\}}$, has a prediction error lower bound of*

$$\mathbb{E}[\|\mathbf{A}(\mathbf{x}^\bullet - \tilde{\mathbf{x}})\|_2^2] \gtrsim \frac{\sigma^2 d}{\min\{r, N\}}. \quad (14)$$

Even though Corollary ?? considers sampling according to the exact block leverage scores, its proof can be modified to accommodate approximate sampling also. Additionally, the above corollary holds for constrained least squares, though we do not explicitly state it; as it is not a focus of the work presented in this paper. From (??), it is clear that for a smaller r with $r < N$; we get a less efficient sketch and approximation $\tilde{\mathbf{x}}$, though when considering a higher r which approaches N ; we get an improvement in the accuracy of $\tilde{\mathbf{x}}$ at the cost of a higher computation and computational load in our resulting GC scheme.

C. Expansion Networks

The framework we propose emulates the sampling w.r. procedure of Algorithm ??, in distributed CC environments. Even though we focus on ℓ_2 -s.e. and descent methods in this paper, the proposed framework applies to any matrix algorithm which utilizes importance sampling with replacement. In contrast to other CC schemes in which RandNLA was used to *compress* the network; e.g. [?], [?], [?], here the networks are *expanded* according to $\Pi_{\{K\}}$ — the computations corresponding to the blocks are replicated through the servers; proportional to $\Pi_{\{K\}}$. It is unlikely that we can exactly emulate this distribution, as the number of replications per task need to be integers. Instead, we mimic the exact probabilities with an *induced* distribution $\bar{\Pi}_{\{K\}}$ through *expansion networks*, which are determined by $\tilde{F}(t)$ at a prespecified $t \leftarrow T$; after which the central server stops receiving computations for that iteration.

We propose the minimization problem (??), whose approximate solution $\hat{r}_{\{K\}}$ (??) suggests the number of replicas r_i of each block in our expansion network. We note that (??) is a surrogate to the integer program (??), whose solution can achieve an accurate realizable distribution $\bar{\Pi}_{\{K\}}$ to $\Pi_{\{K\}}$ through the distributed network, by appropriately replicating the blocks. Unfortunately, the integer program (??) is not always solvable. Nonetheless, when we have an approximation to (??) or (??), through *uniform* sampling we can minimize w.h.p. the ℓ_2 -s.e. condition for \mathbf{A} (??), up to a small error, given the integer constraints imposed by the physical system — $r_i \in \mathbb{Z}_+$ for all i and $R = \sum_{l=1}^K r_l$ such that $R \approx m$. In the CC context, we want $m = R$; i.e. the total number of replicated blocks is equal to the number of servers. Next, we describe the desired induced distribution $\bar{\Pi}_{\{K\}}$, in order to set up (??).

Assume w.l.o.g. that $\Pi_j \leq \Pi_{j+1}$ for all $j \in \mathbb{N}_{K-1}$, thus $r_j \leq r_{j+1}$. The sampling distribution through the expansion network translates to

$$\bar{\Pi}_i := \Pr[\text{the } i^{\text{th}} \text{ block is sampled}] = r_i / R \approx \Pi_i \quad (15)$$

for all $i \in \mathbb{N}_K$ and $R = \sum_{l=1}^K r_l$. Our objective is to determine $r_{\{K\}}$ such that $\bar{\Pi}_i \approx \Pi_i$ for all i . Furthermore, for an erasure probability determined by $\phi(t)$ at a specified time t (??), the probability that the computation corresponding to the i^{th} block is sampled w.r. through the erasure channels at time t is

$$\Pr[\text{sample the } i^{\text{th}} \text{ block through the channels}] = 1 - \phi(t)^{\rho_i(t)}, \quad (16)$$

for some $\rho_i(t) \in \mathbb{R}_{>0}$,⁵ and the network emulates the sampling distribution $\Pi_{\{K\}}$ exactly when

$$\Pi_i = 1 - \phi(t)^{\rho_i(t)} \quad \text{for all } i. \quad (17)$$

The replications which take place can be interpreted as the task allocation through a directed bipartite graph $G = (\mathcal{L}, \mathcal{R}, \mathcal{E})$, where \mathcal{L} and \mathcal{R} correspond to the K encoded partitions $\tilde{A}_{\{K\}}$ and m servers respectively, where $\deg(x_i) = r_i$ for all $x_i \in \mathcal{L}$ and $\deg(y_j) = 1$ for all $y_j \in \mathcal{R}$; with $\{x_i, y_j\} \in \mathcal{E}$ only if the j^{th} server W_j is assigned \tilde{A}_i .

⁵To be realizable, through replications, we need $\rho_i(t) \in \mathbb{Z}_+$.

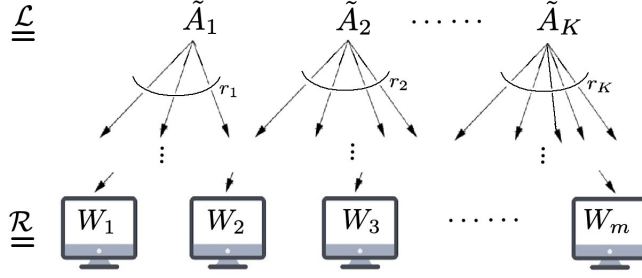


Fig. 3: Depiction of an expansion network, as a bipartite graph, for $m = \sum_{l=1}^K r_l$.

Our goal is to determine $r_{\{K\}}$, which minimize the error in the emulated distribution $\bar{\Pi}_{\{K\}}$. Under the assumption that we have an integer number of replicas per block, from (??) and (??) we deduce that $\Pi_i \approx 1 - \phi(t)^{r_i}$ for $r_i \in \mathbb{Z}_+$, which lead us to the minimization problem⁶

$$\arg \min_{r_{\{K\}} \subseteq \mathbb{Z}_+} \left\{ \Delta_{\Pi, \bar{\Pi}} := \frac{1}{K} \sum_{i=1}^K |\Pi_i - (1 - \phi(t)^{r_i})| \right\} = \arg \left\{ \sum_{i=1}^K \min_{r_i \in \mathbb{Z}_+} \left\{ |\Pi_i - (1 - \phi(t)^{r_i})| \right\} \right\}. \quad (18)$$

By combining (??) and (??), we then solve for the approximate replications $\hat{r}_{\{K\}}$ at time t :

$$\bar{\Pi}_i \approx \Pi_i = 1 - \phi(t)^{\rho_i(t)} \implies \hat{r}_i = \left\lfloor \frac{\log(1 - \Pi_i)}{\log(\phi(t))} \right\rfloor = \lfloor \rho_i(t) \rfloor, \quad (19)$$

which result in the induced distribution $\bar{\Pi}_i = \hat{r}_i / \hat{R}$, for $\hat{R} := \sum_{l=1}^K \hat{r}_l$. In our context, we also require that $\hat{R} \approx m$.

Ideally, the above procedure would result in replication numbers $\hat{r}_{\{K\}}$ for which $\hat{R} = m$. This though is unlikely to occur, as $\Pi_{\{K\}}$ and R are determined by the data, and m is a physical limitation. There are several practical ways to work around this. One approach is to redefine $\hat{r}_{\{K\}}$ to $\tilde{r}_{\{K\}}$ by $\tilde{r}_i = \hat{r}_i \pm \alpha_i$ for α_i small integers such that $\sum_{l=1}^K \tilde{r}_l = m$ and $\sum_{l=1}^K |\Pi_l - \tilde{r}_l/m|$ is minimal. If $m \gg \hat{R}$ for a large enough τ , we can set the number of replicas to be $\tilde{r}_i \approx \left\lfloor m/\hat{R} \right\rfloor \cdot \hat{r}_i$. Furthermore, the block size τ can be selected such that \hat{R} is approximately equal to the system's parameter m . We focus on the issue of having $\hat{R} \approx m$ in Subsection ??.

Lemma 1. *The approximation $\hat{r}_{\{K\}}$ according to (??) of the minimization problem (??) at time t , satisfies*

$$\Delta_{\Pi, \bar{\Pi}} \leq \left(1 - \sqrt{\phi(t)}\right) \cdot \left(\sum_{l=1}^K \phi(t)^{\min_{i \in \mathbb{N}_K} \{\hat{r}_i, \rho_i(t)\}} \right).$$

Proof. We break the proof into the cases where we round $\rho_i(t)$ to both the closest integers above and below. In either case, we know that $(\rho_i(t) - \hat{r}_i(t)) \in [-1/2, 1/2]$, for each $i \in \mathbb{N}_K$. Denote the respective individual summands of $\Delta_{\Pi, \bar{\Pi}}$ by Δ_i . In the case where $r_i = \lfloor \rho_i(t) \rfloor$, we have $\rho_i(t) = \hat{r}_i + \eta$ for $\eta \in [0, 1/2]$, hence

$$\begin{aligned} \Delta_i &= \left| (1 - \phi(t)^{\rho_i(t)}) - (1 - \phi(t)^{\hat{r}_i}) \right| \\ &= \left| \phi(t)^{\hat{r}_i} - \phi(t)^{\rho_i(t)} \right| \\ &= \left| \phi(t)^{\hat{r}_i} - \phi(t)^{\hat{r}_i + \eta} \right| \\ &= \left| \phi(t)^{\hat{r}_i} \cdot (1 - \phi(t)^\eta) \right| \\ &\leq \left| \phi(t)^{\hat{r}_i} \cdot (1 - \phi(t)^{1/2}) \right| \\ &= \phi(t)^{\hat{r}_i} \cdot \left(1 - \sqrt{\phi(t)}\right). \end{aligned}$$

Similarly, in the case where $r_i = \lceil \rho_i(t) \rceil$, we have $\hat{r}_i = \rho_i(t) + \eta$ for $\eta \in [0, 1/2]$; and

$$\Delta_i \leq \phi(t)^{\rho_i(t)} \cdot \left(1 - \sqrt{\phi(t)}\right).$$

Considering all summands, it follows that

$$\Delta_{\Pi, \bar{\Pi}} = \sum_{l=1}^K \Delta_l \leq \sum_{l=1}^K \left(\phi(t)^{\min_{i \in \mathbb{N}_K} \{\hat{r}_i, \rho_i(t)\}} \cdot \left(1 - \sqrt{\phi(t)}\right) \right).$$

⁶Note that $\Delta_{\Pi, \bar{\Pi}} \equiv d_{\Pi, \bar{\Pi}}$, where $\tilde{\Pi}_i = 1 - \phi(t)^{r_i}$ for all $i \in \mathbb{N}_K$. For our proposed distribution $\bar{\Pi}_{\{K\}}$, we may have $d_{\Pi, \bar{\Pi}} \neq \Delta_{\Pi, \bar{\Pi}}$.

□

We note that all terms involved in the upper bound of Lemma ?? are positive and strictly less than one. Furthermore, for a larger t we have a smaller $\phi(t)$, while for a smaller t we have a smaller \hat{r}_i for each i . This bound further corroborates the importance of the hyperparameter t and the distribution $F(t)$, in designing expansion networks.

The replication of blocks which takes place, can be described through a corresponding “*expansion matrix*”:

$$\tilde{\mathbf{E}} = \mathbf{E} \otimes \mathbf{I}_\tau = \begin{pmatrix} \mathbf{1}_{r_1 \times 1} & & & \\ & \mathbf{1}_{r_2 \times 1} & & \\ & & \ddots & \\ & & & \mathbf{1}_{r_K \times 1} \end{pmatrix} \otimes \mathbf{I}_\tau \in \{0, 1\}^{R\tau \times K\tau} \quad (20)$$

where $\mathbf{E} \in \{0, 1\}^{R \times K}$ is the adjacency matrix of the bipartite graph G (up to a permutation of the rows/server indices). It follows that $(\tilde{\mathbf{E}} \cdot \mathbf{A}, \tilde{\mathbf{E}} \cdot \mathbf{b})$ are comprised of replicated blocks of the partitioning in (??), with replications according to $r_{\{K\}}$.

For the proposed networks, the multiplicative misestimation factor in Theorem ?? is $\beta_{\Pi} = \min_{i \in \mathbb{N}_K} \{\Pi_i / \tilde{\Pi}_i\} \leq 1$. In the case where $\tilde{R} = \sum_{l=1}^K \tilde{r}_l > m$ and $\tilde{\Pi}_i := \tilde{r}_i / \tilde{R}$, Algorithm ?? takes $\tilde{r}_{\{K\}}$ as an input and determines $r_{\{K\}}$ such that $R = \sum_{l=1}^K r_l = m$. The updated distribution $\tilde{\Pi}_{\{K\}}$ where $\tilde{\Pi}_i = r_i / R$ for each i , also has a more accurate misestimation factor; i.e. $\beta_{\tilde{\Pi}} > \beta_{\Pi}$. To establish sampling guarantees in relation to $d_{\Pi, \tilde{\Pi}}$, one would need to invoke an additive approximation error to the scores, i.e. $\tilde{\Pi}_i \leq \Pi_i + \epsilon$ for all i where $\epsilon \geq 0$ is a small constant [?], [?]. In our distributed networks, the additive error would be $\epsilon_{\Pi} = \max_{i \in \mathbb{N}_K} \{|\Pi_i - \tilde{\Pi}_i|\}$.

D. Optimal Induced Distributions

Recall that (??) is a surrogate to the integer program

$$r_{\{K\}}^* = \arg \min_{\substack{r_1, \dots, r_K \in \mathbb{Z}_+ \\ R = \sum_{l=1}^K r_l}} \left\{ d_{\Pi, \Pi^*} = \frac{1}{K} \sum_{i=1}^K |\Pi_i - \overbrace{r_i/R}^{\Pi_i^*}| \right\} \quad (21)$$

for $R \approx m$ the total number of servers. Potential solutions $r_{\{K\}}^*$ can achieve the closest realizable distribution to $\Pi_{\{K\}}$ through expansion networks. Similar to $\Delta_{\Pi, \tilde{\Pi}}$ from (??), the distortion metric d_{Π, Π^*} is a measure of closeness between the distributions $\Pi_{\{K\}}$ and $\tilde{\Pi}_{\{K\}}$; under the network imposed constraints. In the case where $\Pi_{\{K\}} \subsetneq (0, 1) \setminus \mathbb{Q}_+$; i.e. $\Pi_{\{K\}}$ are not necessarily all rational, the integer constraints of the physical network may deem exactly emulating $\Pi_{\{K\}}$ impossible. The integer program (??) cannot be solved exactly when $\Pi_{\{K\}} \subsetneq (0, 1) \setminus \mathbb{Q}_+$; as we can always get finer approximations, e.g. through a continued fraction approximation. This is specific to the ending time T when considering erasures over the communication channels according to (??), which T we do not include in (??); in order to simplify notation. Furthermore, (??) can also be considered for centralized distributed settings which differ from the system model proposed in [?]. We note that solvers to (??) exist, when R is fixed and we remove the constraint $R = \sum_{l=1}^K r_l$. The proof of Corollary ?? is a constructive solution of (??) when $\Pi_{\{K\}} \subsetneq [0, 1] \cap \mathbb{Q}_+$, in which case perfect emulation is possible.

Proposition 1. *A perfect emulation occurs when $d_{\Pi, \Pi^*} = 0$. This is possible if and only if $\Pi_i \in [0, 1] \cap \mathbb{Q}_+$ for all i and the denominators of $\Pi_{\{K\}}$ in reduced form are factors of R ; i.e. $R \cdot \Pi_i \in \mathbb{Z}_+$.*

Proof. If $d_{\Pi, \Pi^*} = 0$, then $\Pi_i = \Pi_i^*$ for all $i \in \mathbb{N}_K$, thus $\Pi_{\{K\}}$ and $\Pi_{\{K\}}^*$ are the same sampling distributions.

For the reverse direction, assume that for all i we have $\Pi_i = a_i/b_i$ for coprime integers $a_i, b_i \in \mathbb{Z}_+$, and that $R = \mu_i b_i$ for some $\mu_i \in \mathbb{Z}_+$; thus $R \cdot \Pi_i = \mu_i a_i \in \mathbb{Z}_+$. Let $r_i = \mu_i a_i$. It follows that $\Pi_i^* = \frac{r_i}{R} = \frac{\mu_i a_i}{\mu_i b_i} = \frac{a_i}{b_i} = \Pi_i$ for all i , hence $d_{\Pi, \Pi^*} = 0$. Now, assume for a contradiction that there is a $j \in \mathbb{N}_K$ for which $\Pi_j \in (0, 1) \setminus \mathbb{Q}_+$. Then, by definition, Π_j cannot be expressed as a fraction r_j/R for $r_j, R \in \mathbb{Z}_+$, thus $d_{\Pi, \Pi^*} \geq |\Pi_j - \Pi_j^*| > 0$. □

Corollary 2. *When $\Pi_{\{K\}} \subsetneq [0, 1] \cap \mathbb{Q}_+$, we can solve (??), so that $d_{\Pi, \Pi^*} = 0$.*

Proof. From Proposition ??, the smallest R in order to attain $r_{\{K\}}$ for which $d_{\Pi, \Pi^*} = 0$ when considering $\Pi_i = a_i/b_i$ in reduced form, is the *least common multiple* $R = \text{lcm}(b_1, \dots, b_K)$. For each $i \in \mathbb{N}_K$, we then have $R = \mu_i b_i$ for $\mu_i \in \mathbb{Z}_+$, and $r_i = \mu_i a_i$. Hence $\Pi_i^* = \frac{r_i}{R} = \frac{\mu_i a_i}{\mu_i b_i} = \frac{a_i}{b_i} = \Pi_i$, for which $d_{\Pi, \Pi^*} = 0$. □

Lemma 2. *If for a set of integers $\tilde{r}_{\{K\}}$; we have $\tilde{R} = \sum_{l=1}^K \tilde{r}_l$, $m = \tilde{R}$, and $\tilde{\Pi}_i = \tilde{r}_i / \tilde{R}$ for all $i \in \mathbb{N}_K$, then:*

$$\frac{1}{m} \cdot \min_{i \in \mathbb{N}_K} \{ \lfloor |m \cdot \Pi_i - \tilde{r}_i| \rfloor \} \leq d_{\Pi, \tilde{\Pi}} \leq \frac{1}{m} \cdot \max_{i \in \mathbb{N}_K} \{ \lceil |m \cdot \Pi_i - \tilde{r}_i| \rceil \}. \quad (22)$$

Proof. Let $\tilde{d}_i = |\Pi_i - \tilde{\Pi}_i| = |\Pi_i - \tilde{r}_i/m|$ for each i , hence

$$\tilde{r}_L := \frac{1}{m} \cdot \min_{i \in \mathbb{N}_K} \{ \lfloor |m \cdot \Pi_i - \tilde{r}_i| \rfloor \} \leq \tilde{d}_i \leq \frac{1}{m} \cdot \max_{i \in \mathbb{N}_K} \{ \lceil |m \cdot \Pi_i - \tilde{r}_i| \rceil \} =: \tilde{r}_U$$

for all $i \in \mathbb{N}_K$. By rescaling the sum over all $\tilde{d}_{\{K\}}$ by $1/K$, we get

$$\tilde{r}_L = \frac{K \cdot \tilde{r}_L}{K} \leq \frac{1}{K} \cdot \sum_{i=1}^K \tilde{d}_i \leq \frac{K \cdot \tilde{r}_U}{K} = \tilde{r}_U$$

which completes the proof. \square

Next, we give a simple approximation to (??), for when we do not consider the erasure channel characterization through (??); nor an ending time T . Given $\Pi_{\{K\}}$, the replication numbers are $\tilde{r}_i = \lfloor \Pi_i / \Pi_1 \rfloor$ and $\tilde{R} = \sum_{l=1}^K \tilde{r}_l$. Further note that for more accurate approximations, we can select an integer $\nu > 1$ and take $\tilde{r}_i = \lfloor \nu \cdot \Pi_i / \Pi_1 \rfloor$. From the proof of Corollary ??, it follows that if $\Pi_{\{K\}} \subsetneq [0, 1] \cap \mathbb{Q}_+$ and $\nu = \mu_1 a_1$, we get $\tilde{r}_i = \Pi_i / \Pi_1 = \mu_i a_i \in \mathbb{Z}_+$, which solves (??). The drawback of designing an expansion network with this solution, is that as ν increases; \tilde{R} also increases.

To drop the constraint $R = \sum_{l=1}^K r_l$ of (??) and the assumption that $m = R$, we give a procedure in Algorithm ?? for determining $r_{\{K\}}$ from a given set $\tilde{r}_{\{K\}}$ (e.g. those proposed in (??)) to get the induced distribution $\{\bar{\Pi}_i = r_i/m\}_{i=1}^K$; where $m = \sum_{l=1}^K r_l$. In Algorithm ??, $\chi = 1$ and $\chi = 0$ indicate whether $\tilde{R} > m$ or $\tilde{R} < m$ respectively.

Algorithm 2: Determine $r_{\{K\}}$ from $\tilde{r}_{\{K\}}$

Input: $m, \Pi_{\{K\}}, \tilde{r}_{\{K\}}, \tilde{R} = \sum_{i=1}^K \tilde{r}_i$

Output: replication numbers $r_{\{K\}}$

Initialize: $\tilde{d}_{\{K\}} = \{\tilde{d}_i := \Pi_i - \tilde{r}_i/m\}_{i=1}^K, r_{\{K\}} = \tilde{r}_{\{K\}}, R = \tilde{R}, \chi = 1, \tilde{j} = 0$

if $R < m$ **then**

$\chi \leftarrow 0$

$\tilde{d}_{\{K\}} \leftarrow \{-\tilde{d}_i\}_{i=1}^K$

end

while $R \neq m$ **do**

$j \leftarrow \arg \min_{i \in \mathbb{N}_K} \{\tilde{d}_{\{K\}}\} \quad (\diamond)$

if $(-1)^{\chi+1} \cdot (\Pi_j - \frac{1}{m}(r_j + (-1)^\chi)) > 0$ **and** $\tilde{j} = j$ **then**

$\tilde{d}_j \leftarrow 1$

end

else

$r_j \leftarrow r_j + (-1)^\chi$

$R \leftarrow R + (-1)^\chi$

$\tilde{d}_j \leftarrow (-1)^{\chi+1} \cdot (\Pi_j - r_j/m)$

end

$\tilde{j} \leftarrow j$

end

$\triangleright \triangleright \chi$ indicates: $\tilde{R} \geq m$ or $\tilde{R} < m$

Remark 1. The objective of Algorithm ?? is to reduce the upper bound of (??) when $m < \tilde{R}$, while guaranteeing that $\sum_{l=1}^K r_l = m$. In practice, the more concerning and limiting case is when $m < \tilde{R}$. The bottleneck of Algorithm ?? is retrieving the index j in the while loop, which takes $O(K)$ time. In order to reduce the number of instances we solve (\diamond), we ensure that we only reduce the replica numbers $\hat{r}_{\{K\}}$ in the case where $R > m$; and increase them when $R < m$, by the inner **if** statement. Moreover, this is carried out once before sharing the replicated blocks. The more practical and realistic case is when $R > m$, as we can get a closer approximation with a greater R ; and $\text{lcm}(b_1, \dots, b_K)$ will likely be large when $\Pi_{\{K\}} \subsetneq [0, 1] \cap \mathbb{Q}_+$. We note that the integers $\hat{r}_{\{K\}}$ of (??) and their sum \hat{R} are a byproduct of the prespecified ending time $t \leftarrow T$, the mother runtime distribution $F(t)$, and the block size τ , which can be selected so that $\hat{R} > m$.

Proposition 2. Assume we are given $\tilde{r}_{\{K\}}$ (not necessarily according to (??)), for which $m < \tilde{R} = \sum_{l=1}^K \tilde{r}_l$. Denote by $\tilde{\Pi}_{\{K\}}$ the corresponding sampling distribution $\{\tilde{\Pi}_i = \tilde{r}_i/\tilde{R}\}_{i=1}^K$, for which $d_{\Pi, \tilde{\Pi}} \leq \tilde{\epsilon}$. Then, the output $r_{\{K\}}$ of Algorithm ?? produces an induced distribution $\{\bar{\Pi}_i = r_i/m\}_{i=1}^K$ which satisfies $d_{\Pi, \bar{\Pi}} \leq d_{\Pi, \tilde{\Pi}} \leq \tilde{\epsilon}$.

Proof. The case where $\tilde{R} > m$, corresponds to $\chi = 1$, in which case we Algorithm ?? returns $r_{\{K\}}$ for which $r_i \leq \tilde{r}_i$ for all $i \in \mathbb{N}_K$. In this case, the optimization problem (\diamond) assigns to j a partition index for which $\Pi_j < r_j/m$.⁷ The **if** statement guarantees that we did not produce an r_j for which $\Pi_j > r_j/m$; when we previously previously $\Pi_j < \tilde{r}_j/\tilde{R}$ (or $\Pi_j < r_j/R$ after a reassignment of \tilde{r}_j). Along with the fact that $\frac{r_j-1}{R-1} < \frac{r_j}{R}$, it follows that for the updated difference d'_j :

$$|d'_j| = \left| \Pi_j - \frac{r_j-1}{R-1} \right| = \frac{r_j-1}{R-1} - \Pi_j \leq \frac{r_j}{R} - \Pi_j = \left| \Pi_j - \frac{r_j}{R} \right| = |\tilde{d}_j|,$$

⁷Since $\tilde{d}_j = \Pi_j - r_j/m < 0$, we have $\Pi_j < r_j/m$.

i.e. at each iteration of the **else** statement, we decrease the distortion metric, thus $d_{\Pi, \tilde{\Pi}} \leq d_{\Pi, \tilde{\Pi}} \leq \tilde{\epsilon}$.

The **else** statement is carried out $\tilde{R} - m$ times, producing $r_{\{K\}}$ for which $\sum_{l=1}^K r_l = \tilde{R} - (\tilde{R} - m) = m$, hence the normalizing integer for the distribution $\tilde{\Pi}_{\{K\}}$ is $R = m$. \square

Remark 2. To summarize, the objective is to determine replicas (??), in order to emulate block leverage score sampling of Algorithm ?? through erasure channels; at an ending time T . By Proposition ??, this is not always possible. Instead, we give an estimate solution to (??) through $\hat{r}_{\{K\}}$ in (??). To get a valid sampling distribution in the CC framework, we normalize the replica numbers $\hat{r}_{\{K\}}$ by their sum \tilde{R} . Furthermore, we propose Algorithm ?? which takes estimates $\hat{r}_{\{K\}}$ and modifies them to return $r_{\{K\}}$ for which $m = \sum_{l=1}^K r_l$, and improves the induced approximate block leverage score distribution when $\tilde{R} > m$.

E. GC through Leverage Score Sampling

Next, we derive the server computations of our GC scheme, so that the central server retrieves the gradient of $\|\tilde{\mathbf{S}}_{[s]}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$; for $\tilde{\mathbf{S}}_{[s]}$ according to Algorithm ?? at iteration s , to iteratively approximate (??). Furthermore, we show that with a diminishing step-size, our updates $\mathbf{x}^{[s]}$ converge in expectation to the optimal solution \mathbf{x}^* .

The blocks of our leverage score sampling procedure are those of the encoded data $\tilde{\mathbf{A}} := \mathbf{G} \cdot \mathbf{A}$ and $\tilde{\mathbf{b}} := \mathbf{G} \cdot \mathbf{b}$, for

$$\mathbf{G} = \text{diag}\left(\left\{1/\sqrt{q\tilde{\Pi}_i}\right\}_{i=1}^K\right) \otimes \mathbf{I}_\tau \in \mathbb{R}_{\geq 0}^{N \times N}. \quad (23)$$

Specifically, the encoding carried out by the central server corresponds to the rescaling through \mathbf{G} . We partition both $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{b}}$ across their rows analogous to (??):

$$\tilde{\mathbf{A}} = \mathbf{G} \cdot \mathbf{A} = \begin{bmatrix} \tilde{A}_1^\top & \cdots & \tilde{A}_K^\top \end{bmatrix}^\top \quad \text{and} \quad \tilde{\mathbf{b}} = \mathbf{G} \cdot \mathbf{b} = \begin{bmatrix} \tilde{b}_1^\top & \cdots & \tilde{b}_K^\top \end{bmatrix}^\top \quad (24)$$

where $\tilde{A}_i \in \mathbb{R}^{\tau \times d}$ and $\tilde{b}_i \in \mathbb{R}^\tau$ for all $i \in \mathbb{N}_K$. Furthermore, all the data across the expansion network after the scalar encoding, is contained in aggregated ‘expanded and encoded matrix-vector pairs’ $(\Psi, \vec{\psi}) := (\tilde{\mathbf{E}} \cdot \tilde{\mathbf{A}}, \tilde{\mathbf{E}} \cdot \tilde{\mathbf{b}}) \in \mathbb{R}^{R\tau \times d} \times \mathbb{R}^{R\tau}$ (Figure ??). For the encoded objective function $L_{\mathbf{G}}(\mathbf{x}) := \|\mathbf{G}(\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$, we have:

$$\begin{aligned} \text{(i)} \quad L_{\mathbf{G}}(\mathbf{x}) &= \mathbf{x}^\top \left(\sum_{i=1}^K \tilde{A}_i^\top \tilde{A}_i \right) \mathbf{x} + \left(\sum_{i=1}^K (\tilde{b}_i^\top - 2\mathbf{x}^\top \tilde{A}_i^\top) \tilde{b}_i \right) \\ \text{(ii)} \quad \nabla_{\mathbf{x}} L_{\mathbf{G}}(\mathbf{x}) &= 2 \sum_{i=1}^K \tilde{A}_i^\top (\tilde{A}_i \mathbf{x} - \tilde{b}_i) \\ \text{(iii)} \quad \nabla_{\mathbf{x}} L_{\mathbf{G}}(\mathbf{x}_{\mathbf{G}}^*) &= 0 \implies \mathbf{x}_{\mathbf{G}}^* = \left(\sum_{i=1}^K \tilde{A}_i^\top \tilde{A}_i \right)^{-1} \cdot \left(\sum_{i=1}^K \tilde{A}_i^\top \tilde{b}_i \right). \end{aligned}$$

We make use of (ii) to approximate the gradient distributively. Each server is provided with a partition $\tilde{\mathcal{D}}_j = (\tilde{A}_j, \tilde{b}_j)$, and computes the respective summand of the gradient from (ii), which is the encoded partial gradient on $\mathcal{D}_j = (\mathbf{A}_j, \mathbf{b}_j)$:

$$\hat{g}_i^{[s]} := \nabla_{\mathbf{x}} L_{ls}(\tilde{\mathcal{D}}_i; \mathbf{x}^{[s]}) = \nabla_{\mathbf{x}} L_{ls} \left(1/\sqrt{q\tilde{\Pi}_i} \cdot \mathbf{A}_i, 1/\sqrt{q\tilde{\Pi}_i} \cdot \mathbf{b}_i; \mathbf{x}^{[s]} \right) = \frac{1}{q\tilde{\Pi}_i} \cdot g_i^{[s]}. \quad (25)$$

Once a server computes its assigned partial gradient, it sends it back to the central server. When the central server receives q responses, it sums them in order to obtain the approximate gradient $\hat{g}^{[s]}$.

Denote the index multiset corresponding to the encoded pairs $(\tilde{A}_j, \tilde{b}_j)$ of the received computations at iteration s with $\mathcal{S}^{[s]}$, for which $|\mathcal{S}^{[s]}| = q$. The vector parameter’s update is then $\mathbf{x}^{[s+1]} = \mathbf{x}^{[s]} - \xi_s \cdot \hat{g}^{[s]}$, where

$$\hat{g}^{[s]} = \sum_{i \in \mathcal{S}^{[s]}} \nabla_{\mathbf{x}} L_{ls}(\tilde{\mathcal{D}}_j; \mathbf{x}^{[s]}) = 2 \sum_{i \in \mathcal{S}^{[s]}} \tilde{A}_i^\top (\tilde{A}_i \mathbf{x}^{[s]} - \tilde{b}_i) \quad (26)$$

and ξ_s is an appropriate step-size. In the case where q is not determined a priori or varies at each iteration, the scaling corresponding to $1/\sqrt{q}$ in the encoding through \mathbf{G} could be done by the central server; after that iteration’s computations are aggregated. We consider the case where q is the same for all iterations.

Next, we present the guarantees of our proposed GC scheme, which rely on Algorithm ?. The procedure we outlined above along with the following results, show how RandNLA can be utilized to devise efficient approximate GC schemes.

Remark 3. By sampling $q = q(T)$ blocks at each iteration, and performing the approximate gradient update (??), one obtains a SSD version of the encoded linear system $\mathbf{G} \cdot (\mathbf{A}\mathbf{x}) = \mathbf{G} \cdot \mathbf{b}$. This follows from the fact that different servers are expected to respond faster at each iteration, as we assume that they are homogeneous and have the same expected response time.

In Remark ??, the application of $\tilde{\Omega}^{[s]}$ (in Algorithm ??) has a direct correspondence to the index set $\mathcal{I}^{[s]}$ of the $q(T)$ non-stragglers of iteration s , which can be viewed as drawing $\mathcal{I}^{[s]}$ uniformly at random from $\{\mathcal{I} \subseteq \mathbb{N}_m : |\mathcal{I}| = q(T)\}$. This is what induces a first-order stochastic iterative sketching method for (??) through the proposed GC scheme, and removes the bias towards the samples which would be selected through the single $\tilde{\Omega}^{[1]}$; in the sketch-and-solve paradigm. Specifically, we do not solve the

modified problem (??) which only accounts for a reduced dimension determined at the beginning of the iterative process, whose approximate solution is $\mathbf{x}_G^* \approx (\tilde{\mathbf{S}}\mathbf{A})^\dagger(\tilde{\mathbf{S}}\mathbf{b})$,⁸ Instead, we consider a different reduced linear system $\tilde{\mathbf{S}}_{[s]} \cdot (\mathbf{A}\mathbf{x}^{[s]}) = \tilde{\mathbf{S}}_{[s]} \cdot \mathbf{b}$ at each iteration. This further justifies the result of Corollary ?? . This benefit of iterative sketching is validated numerically in Section ??.

Theorem 2. *The proposed GC scheme based on the block leverage score sketch results in a SSD procedure for $L_{ls}(\Psi, \vec{\psi}; \mathbf{x})$. Furthermore, at each iteration it produces an unbiased estimator of (??), i.e. $\mathbb{E}[\hat{g}^{[s]}] = g^{[s]}$.*

Proof. The computations of the q fastest servers indexed by $\mathcal{I}^{[s]}$ (which corresponds to $\tilde{\Omega}^{[s]}$), are added to produce $\hat{g}^{[s]}$, and the sampling of Algorithm ?? is according to $\bar{\Pi}_{\{K\}}$. By Remark ??, it follows that each $\mathcal{I}^{[s]}$ has equal chance of occurring, which is precisely the stochastic step of SSD, i.e. each group of q encoded block pairs has an equal chance of being selected.

Since the servers are homogeneous and respond independently of each other, it follows that at iteration s ; each \hat{g}_i is received with probability $\bar{\Pi}_i$. Therefore

$$\begin{aligned} \mathbb{E}[\hat{g}^{[s]}] &= \mathbb{E}\left[\sum_{i \in \mathcal{I}^{[s]}} \hat{g}_i^{[s]}\right] = \sum_{i \in \mathcal{I}^{[s]}} \mathbb{E}[\hat{g}_i^{[s]}] = \sum_{i \in \mathcal{I}^{[s]}} \sum_{j=1}^K \bar{\Pi}_j \cdot \hat{g}_j^{[s]} \\ &= q \cdot \sum_{j=1}^K \bar{\Pi}_j \cdot \hat{g}_j^{[s]} \stackrel{\text{b}}{=} q \cdot \sum_{j=1}^K \bar{\Pi}_j \cdot \frac{1}{q\bar{\Pi}_j} \cdot g_j^{[s]} = \sum_{j=1}^K g_j^{[s]} = g^{[s]} \end{aligned}$$

where in b we invoked (??). □

Lemma 3. *The optimal solution of the modified least squares problem $L_{ls}(\Psi, \vec{\psi}; \mathbf{x})$, is equal to the optimal solution \mathbf{x}^* of (??).*

Proof. Note that the modified objective function $L_{ls}(\Psi, \vec{\psi}; \mathbf{x})$ is $\|\tilde{\mathbf{E}}\mathbf{G} \cdot (\mathbf{A}\mathbf{x} - \mathbf{b})\|_2^2$. Denote its optimal solution by $\mathbf{x}^* \in \mathbb{R}^d$. Further note that $\tilde{\mathbf{E}}$ is comprised of $\tau \times \tau$ identity matrices in such a way that it is full-rank, and \mathbf{G} corresponds to a rescaling of these \mathbf{I}_τ matrices, thus $\tilde{\mathbf{E}}\mathbf{G}$ is also full-rank. It then follows that

$$\mathbf{x}^* = ((\mathbf{E}\mathbf{G}) \cdot \mathbf{A})^\dagger \cdot ((\mathbf{E}\mathbf{G}) \cdot \mathbf{b}) = \mathbf{A}^\dagger \cdot ((\mathbf{E}\mathbf{G})^\dagger \cdot (\mathbf{E}\mathbf{G})) \cdot \mathbf{b} = \mathbf{A}^\dagger \cdot \mathbf{I}_N \cdot \mathbf{b} = \mathbf{x}^*.$$
□

The crucial aspect of our expansion network (incorporated in Theorem ??), which allowed us to use block leverage score sampling in the proposed GC scheme, is that uniform sampling of $L_{ls}(\Psi, \vec{\psi}; \mathbf{x}^{[s]})$ is $\beta_{\bar{\Pi}}$ -approximately equivalent to block sampling of $L_{ls}(\tilde{\mathbf{A}}, \tilde{\mathbf{b}}; \mathbf{x}^{[s]})$ according to the block leverage scores of $\tilde{\mathbf{A}}$. Since the two objective functions are differentiable and additively separable, the resulting gradients are equal, under the assumption that we use the same $\mathbf{x}^{[s]}$ and sampled index set $\mathcal{S}^{[s]}$.⁹ As previously mentioned, the main drawback is that in certain cases we need significantly more servers to accurately emulate $\Pi_{\{K\}}$.

The significance of Theorem ??, is that our distributed approach guarantees well-known established SD and SSD results which assume that the approximate gradient is an unbiased estimator, e.g. [?, Chapter 14]. Even though we are not guaranteed a descent at every iteration (i.e. we could have $L_{ls}(\mathcal{D}; \mathbf{x}^{[s+1]}) > L_{ls}(\mathcal{D}; \mathbf{x}^{[s]})$ or $\|\mathbf{x}^{[s+1]} - \mathbf{x}^*\|_2^2 > \|\mathbf{x}^{[s]} - \mathbf{x}^*\|_2^2$), stochastic descent methods are more common in practice when dealing with large datasets, as empirically they outperform regular SD. This is also confirmed in our experiments.

F. Convergence to \mathbf{x}^*

Next, we give a summary of our main results thus far, and explain how together they imply convergence of our approach in expectation, to iteratively solves $L_{ls}(\Psi, \vec{\psi}; \mathbf{x}^{[s]})$. Moreover, the contraction of our method is quantified in Appendix ??.

Firstly, as summarized in Remark ??, we mimic block leverage score sampling w.r. of (\mathbf{A}, \mathbf{b}) (from $L_{ls}(\mathbf{A}, \mathbf{b}; \mathbf{x}^{[s]})$) through uniform sampling, by approximately solving (??) through the implication of (??) (Lemma ??). This is done implicitly by communicating computations over erasure channels. Secondly, by Theorem ?? we know that the proposed block leverage score sketching matrices satisfy (??); where the approximate sampling distribution $\bar{\Pi}_{\{K\}}$ is determined through the proposed expansion network associated with $\Pi_{\{K\}}$. Hence, at each iteration, we approach a solution $\hat{\mathbf{x}}^{[s]}$ of the induced sketched system $\tilde{\mathbf{S}}_{[s]} \cdot (\mathbf{A}\mathbf{x}^{[s]}) = \tilde{\mathbf{S}}_{[s]} \cdot \mathbf{b}$, which $\tilde{\mathbf{S}}_{[s]}$ satisfies (??) with overwhelming probability. Thirdly, by Theorem ?? and Lemma ??, with a diminishing step-size ξ_s , our updates $\mathbf{x}^{[s]}$ converge to \mathbf{x}^* in expectation, at a rate of $O(1/\sqrt{s} + r/s)$ [?], [?]. A synopsis is given below:

$$\left\{ L_{\mathbf{S}}(\tilde{\mathbf{S}}_{[s]}, \mathbf{A}, \mathbf{b}; \mathbf{x}) \text{ sol'ns} \right\} \xleftarrow{?,?} \left\{ \text{Solve } L_{ls}(\Psi, \vec{\psi}; \mathbf{x}^{[s]}) \text{ through 'sketched-GC'} \right\} \xrightarrow{?,?} \left\{ \text{With a diminishing } \xi_s: \lim_{\mathbf{x}^{[s]} \rightarrow \mathbf{x}^*} \mathbb{E}[\mathbf{x}^{[s]}] \right\}.$$

⁸Since $\tilde{\mathbf{S}} \in \mathbb{R}^{r \times N}$ for $r < N$; we have $\tilde{\mathbf{S}}^\dagger \tilde{\mathbf{S}} \neq \mathbf{I}_N$, hence $(\tilde{\mathbf{S}}\mathbf{A})^\dagger(\tilde{\mathbf{S}}\mathbf{b}) \neq \mathbf{A}^\dagger \mathbf{b}$.

⁹The index set of the sampled blocks from $L_{ls}(\Psi, \vec{\psi}; \mathbf{x}^{[s]})$, corresponds to an index *multiset* of the sampled blocks from $L_{ls}(\tilde{\mathbf{A}}, \tilde{\mathbf{b}}; \mathbf{x}^{[s]})$, as in the latter we are considering sampling *with replacement*.

G. Approximate GC from ℓ_2 -s.e.

In conventional GC, the objective is to construct an encoding matrix $\mathbf{G} \in \mathbb{R}^{m \times K}$ and decoding vectors $\mathbf{a}_{\mathcal{I}} \in \mathbb{R}^{1 \times q}$, such that $\mathbf{a}_{\mathcal{I}} \mathbf{G}_{(\mathcal{I})} = \vec{\mathbf{I}}$ for any set of non-straggling servers \mathcal{I} . It follows that the optimal decoding vector for a set \mathcal{I} of size q in approximate GC [?], [?], [?] is

$$\mathbf{a}_{\mathcal{I}}^* = \arg \min_{\mathbf{a} \in \mathbb{R}^{1 \times q}} \{ \|\mathbf{a} \mathbf{G}_{(\mathcal{I})} - \vec{\mathbf{I}}\|_2^2 \} \implies \mathbf{a}_{\mathcal{I}}^* = \vec{\mathbf{I}} \mathbf{G}_{(\mathcal{I})}^\dagger \quad (27)$$

for $\mathbf{G}_{(\mathcal{I})}^\dagger = (\mathbf{G}_{(\mathcal{I})}^\top \mathbf{G}_{(\mathcal{I})})^{-1} \mathbf{G}_{(\mathcal{I})}^\top \in \mathbb{R}^{K \times q}$.

Proposition 3. The error in the approximated gradient $\dot{g}^{[s]}$ of an optimal approximate linear regression GC scheme $(\mathbf{G}, \mathbf{a}_{\mathcal{I}}^*)$, satisfies

$$\|g^{[s]} - \dot{g}^{[s]}\|_2 \leq 2\sqrt{K} \cdot \text{err}(\mathbf{G}_{(\mathcal{I})}) \cdot \|\mathbf{A}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2, \quad (28)$$

for $\text{err}(\mathbf{G}_{(\mathcal{I})}) := \|\mathbf{I}_K - \mathbf{G}_{(\mathcal{I})}^\dagger \mathbf{G}_{(\mathcal{I})}\|_2$.

Proof. Consider the optimal decoding vector of an approximate GC scheme $\mathbf{a}_{\mathcal{I}}^*$ (??). In the case where $q \geq K$, it follows that $\mathbf{a}_{\mathcal{I}}^* = \vec{\mathbf{I}} \mathbf{G}_{(\mathcal{I})}^\dagger$.

Let $\mathbf{g}^{[s]}$ be the matrix comprised of the transposed exact partial gradients at iteration s , i.e.

$$\mathbf{g}^{[s]} := \begin{pmatrix} g_1^{[s]} & g_2^{[s]} & \dots & g_K^{[s]} \end{pmatrix}^\top \in \mathbb{R}^{K \times d}.$$

Then, for a GC encoding-decoding pair $(\mathbf{G}, \mathbf{a}_{\mathcal{I}})$ satisfying $\mathbf{a}_{\mathcal{I}} \mathbf{G}_{(\mathcal{I})} = \vec{\mathbf{I}}$ for any \mathcal{I} , it follows that

$$\mathbf{a}_{\mathcal{I}} \left(\mathbf{G}_{(\mathcal{I})} \mathbf{g}^{[s]} \right) = \vec{\mathbf{I}} \mathbf{g}^{[s]} = \sum_{j=1}^K (g_j^{[s]})^\top = (g^{[s]})^\top.$$

Hence, the gradient can be recovered exactly. Considering an optimal approximate scheme $(\mathbf{G}, \mathbf{a}_{\mathcal{I}}^*)$ which recovers the gradient estimate $\dot{g}^{[s]} = (\mathbf{a}_{\mathcal{I}}^* \mathbf{G}_{(\mathcal{I})}) \mathbf{g}^{[s]}$, the error in the gradient approximation is

$$\begin{aligned} \|g^{[s]} - \dot{g}^{[s]}\|_2 &= \left\| (\vec{\mathbf{I}} - \mathbf{a}_{\mathcal{I}}^* \mathbf{G}_{(\mathcal{I})}) \mathbf{g}^{[s]} \right\|_2 \\ &= \left\| \vec{\mathbf{I}} (\mathbf{I}_K - \mathbf{G}_{(\mathcal{I})}^\dagger \mathbf{G}_{(\mathcal{I})}) \mathbf{g}^{[s]} \right\|_2 \\ &\leq \|\vec{\mathbf{I}}\|_2 \cdot \left\| \mathbf{I}_K - \mathbf{G}_{(\mathcal{I})}^\dagger \mathbf{G}_{(\mathcal{I})} \right\|_2 \cdot \|\mathbf{g}^{[s]}\|_2 \\ &\stackrel{\mathcal{L}}{\leq} \sqrt{K} \cdot \left\| \mathbf{I}_K - \mathbf{G}_{(\mathcal{I})}^\dagger \mathbf{G}_{(\mathcal{I})} \right\|_2 \cdot \|g^{[s]}\|_2 \\ &\stackrel{\$}{\leq} 2\sqrt{K} \cdot \underbrace{\left\| \mathbf{I}_K - \mathbf{G}_{(\mathcal{I})}^\dagger \mathbf{G}_{(\mathcal{I})} \right\|_2}_{\text{err}(\mathbf{G}_{(\mathcal{I})})} \cdot \|\mathbf{A}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2 \end{aligned}$$

where \mathcal{L} follows from the facts that $\|\mathbf{g}^{[s]}\|_2 \leq \|g^{[s]}\|_2$ and $\|\vec{\mathbf{I}}\|_2 = \sqrt{K}$, and $\$$ from (??) and sub-multiplicativity of matrix norms. \square

In Theorem ??, we show the accuracy of the approximate gradient of iterative GC approaches based on sketching techniques that satisfy the ℓ_2 -s.e. property $\|\mathbf{I}_d - \mathbf{U}^\top \mathbf{S}^\top \mathbf{S} \mathbf{U}\|_2 \leq \epsilon$ from (??) (w.h.p.). This then holds true for our approach through expansion networks, by Theorem ?? and Remark ??.

Theorem 3. Assume that the induced sketching matrix \mathbf{S} from a GC scheme satisfies $\|\mathbf{I}_d - \mathbf{U}^\top \mathbf{S}^\top \mathbf{S} \mathbf{U}\|_2 \leq \epsilon$ (w.h.p.). Then, the updated approximate gradient estimate $\hat{g}^{[s]}$ at any iteration, satisfies (w.h.p.):

$$\|g^{[s]} - \hat{g}^{[s]}\|_2 \leq 2\epsilon \cdot \|\mathbf{A}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2. \quad (29)$$

Specifically, it satisfies (??) with $\text{err}(\mathbf{G}_{(\mathcal{I})}) = \epsilon/\sqrt{K}$.

Proof. Now, consider $\hat{g}^{[s]}$ the approximated gradient of our scheme for linear regression with gradient (??). It follows that

$$\begin{aligned} \|g^{[s]} - \hat{g}^{[s]}\|_2 &= \|2\mathbf{A}^\top (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}) - 2\mathbf{A}^\top (\mathbf{S}^\top \mathbf{S}) (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b})\|_2 \\ &= 2\|\mathbf{A}^\top (\mathbf{I}_N - \mathbf{S}^\top \mathbf{S}) (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b})\|_2 \\ &\leq 2\|\mathbf{A}\|_2 \cdot \|\mathbf{I}_N - \mathbf{S}^\top \mathbf{S}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2 \\ &= 2\|\mathbf{A}\|_2 \cdot \|\mathbf{U}^\top (\mathbf{I}_N - \mathbf{S}^\top \mathbf{S}) \mathbf{U}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2 \\ &= 2\|\mathbf{A}\|_2 \cdot \|\mathbf{I}_d - \mathbf{U}^\top \mathbf{S}^\top \mathbf{S} \mathbf{U}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2 \\ &\stackrel{b}{\leq} 2\epsilon \cdot \|\mathbf{A}\|_2 \cdot \|\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}\|_2 \end{aligned}$$

where in \mathfrak{b} we make use of the assumption that \mathbf{S} satisfies (??). Our approximate GC approach therefore (w.h.p.) satisfies (??), with $\text{err}(\mathbf{G}_{(\mathcal{I})}) = \epsilon/\sqrt{K}$. \square

IV. EXPERIMENTS

In this section, we corroborate our theoretical results, and show their benefits on fabricated datasets. The minimum benefit of Algorithm ?? occurs when $\Pi_{\{K\}}$ is close to uniform. For this reason, and the fact that our expansion approach depends on the implicit distribution through $r_{\{K\}}$, we construct dataset matrices whose resulting sampling distributions and block leverage scores are non-uniform.

For the first experiment, we considered $\mathbf{A} \in \mathbb{R}^{2000 \times 40}$ following a t -distribution, and standard Gaussian noise was added to an arbitrary vector from $\text{im}(\mathbf{A})$ to define \mathbf{b} . We considered $K = 100$ blocks, thus $\tau = 20$. The effective dimension $N = 2000$ was reduced to $r = 1000$, i.e. $q = 20$. We compared the iterative approach with exact block leverage scores (i.e. $\beta = 1$), against analogous approaches using the block-SRHT and Gaussian sketches, and uncoded regular SD.

In Figure ?? we ran 600 iterations on six different instances for each approach, and varied ξ for each experiment by logarithmic factors of $\xi^\times = 2/\sigma_{\max}(\mathbf{A})^2$. The average log residual errors $\log_{10}(\|\mathbf{x}_{l_s}^* - \hat{\mathbf{x}}\|_2/\sqrt{N})$ are depicted in Figure ??, and reported in Table ?. In Figure ?? we observe the convergence of the different approaches, in the case where $\xi \approx 0.42$. In this case, our method (block-lvg) outperforms the Gaussian sketching approach and regular SD. The fact that the performance of the block-SRHT is similar to our proposed algorithm, reflects the result of Proposition ??.

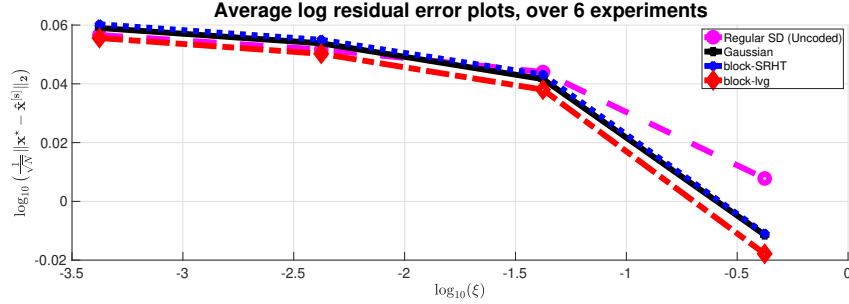


Fig. 4: residual error for varying ξ_s

Average log residual error $\log_{10}(\ \mathbf{x}_{l_s}^* - \hat{\mathbf{x}}\ _2/\sqrt{N})$				
$\log_{10}(\xi)$	0.0004	0.0042	0.0421	0.4207
Regular SD	0.0566	0.0517	0.0440	0.0078
Gaussian	0.0590	0.0538	0.0416	-0.0114
block-SRHT	0.0603	0.0550	0.0431	-0.0110
block-lvg	0.0556	0.0502	0.0380	-0.0178

TABLE I: Average log residual errors, for six instances of SD with fixed steps, when performing Gaussian sketching with updated sketches, iterative block-SRHT and iterative block leverage score sketching, and uncoded SD.

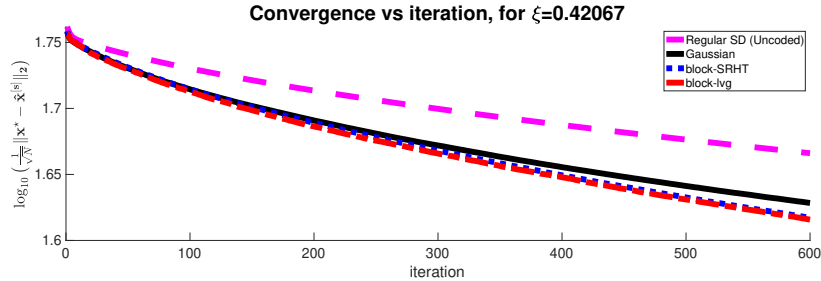
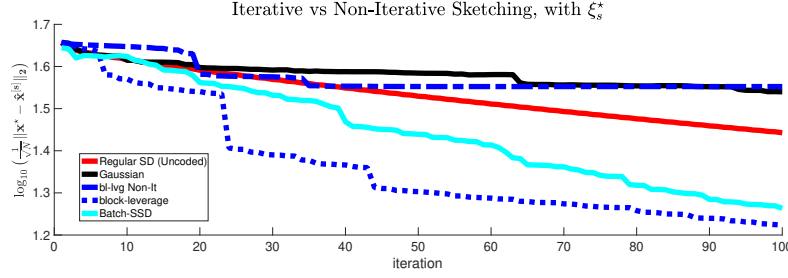


Fig. 5: log residual error convergence

We also considered the same experiment with \mathbf{A} drawn from a t -distribution, with and optimal step-size $\xi_s^* = \langle \mathbf{A}g^{[s]}, \mathbf{A}\mathbf{x}^{[s]} - \mathbf{b} \rangle / \|\mathbf{A}g^{[s]}\|_2^2$ at each iteration. From Figure ??, we observe that our iterative sketching approach outperforms Gaussian sketching with updated sketches; and iterative sketching is superior to non-iterative. Furthermore, we validate Lemma ?? and Theorem ??, as our iterative sketching approach and SSD have similar convergence. Furthermore, it was observed that in some case cases when our iterative sketching method would outperform regular SD (and SSD). We also compared our method to iterative and non-iterative approaches according to the block leverage score sampling, block-SRHT, and Rademacher sketching methods, in which our corresponding approach again produced more accurate final approximations.

Fig. 6: log convergence with ξ_s^*

V. CONCLUSION AND FUTURE WORK

In this paper, we showed how one can exploit results from RandNLA to distributed CC, in the context of GC. By taking enough samples, or equivalently; waiting long enough, the approximation errors can be made arbitrarily small. In terms of CC, the advantages are that *encodings* correspond to a scalar multiplication, and *no* decoding step is required. By utilizing these techniques, we are also advantageous over other CC approximation schemes [?], [?], [?], [?], [?], [?], [?], [?], [?], [?]; by incorporating information from our dataset into our scheme.

Our methods were validated numerically through various experiments, presented in Section ???. Further experiments were performed on various distributions for \mathbf{A} , in which similar results were obtained. We also considered the empirical distribution from real-server completion times taken from 500 AWS-servers [?], and emulated the proposed CC scheme. In this experiment, we obtained the expected results in terms of ℓ_2 -s.e., misestimation factors β_{Π} , and metrics $\Delta_{\Pi, \tilde{\Pi}}$, $d_{\Pi, \tilde{\Pi}}$.

Even though we focused on leverage score sampling for linear regression, other sampling algorithms and problems could benefit by designing analogous replication schemes. One such problem is the *column subset selection problem*, which can be used to compute partial SVD, QR decompositions, as well as low-rank approximations [?]. As for the sampling technique we studied, one can judiciously define a sampling distribution to approximate solutions to such problems [?], which are known to be NP-hard under the Unique Games Conjecture assumption [?].

Furthermore, existing block sampling algorithms can also benefit from the proposed expansion networks, *e.g.* *CR*-multiplication [?] and *CUR* decomposition [?]. For instance, a coded matrix multiplication algorithm of minimum variance can be designed, where the sampling distribution proposed in [?] is used to determine the replication numbers of the expansion network. In terms of matrix decompositions, the block leverage score algorithm of [?] can be used to distributively determine an additive ϵ -error decomposition of \mathbf{A} , in the CC setting. Another future direction is generalizing existing tensor product and factorization algorithms to block sampling, according to both approximate and exact sampling distributions, in order to make them practical for distributed environments.

APPENDIX A

PROOFS OF SUBSECTION ??

In this appendix, we provide the proof of Theorem ?? and Corollary ?. We first recall the following matrix Chernoff bound [?, Fact 1], and prove Proposition ?.

Theorem 4 (Matrix Chernoff Bound, [?, Fact 1]). *Let $\mathbf{X}_1, \dots, \mathbf{X}_q$ be independent copies of a symmetric random matrix $\mathbf{X} \in \mathbb{R}^{d \times d}$, with $\mathbb{E}[\mathbf{X}] = 0$, $\|\mathbf{X}\|_2 \leq \gamma$, $\|\mathbb{E}[\mathbf{X}^\top \mathbf{X}]\|_2 \leq \sigma^2$. Let $\mathbf{Z} = \frac{1}{q} \sum_{i=1}^q \mathbf{X}_i$. Then $\forall \epsilon > 0$:*

$$\Pr \left[\|\mathbf{Z}\|_2 > \epsilon \right] \leq 2d \cdot \exp \left(-\frac{q\epsilon^2}{\sigma^2 + \gamma\epsilon/3} \right).$$

Proposition 4. *The sketching matrix $\tilde{\mathbf{S}}$ of Algorithm ?? with $\tilde{\Pi}_i \geq \beta \Pi_i$ for all i and $\beta \in (0, 1]$, guarantees*

$$\Pr \left[\|\mathbf{I}_d - \mathbf{U}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{U}\|_2 > \epsilon \right] \leq 2d \cdot e^{-q\epsilon^2 \Theta(\beta/d)} \quad (30)$$

for any $\epsilon > 0$, and $q = r/\tau > d/\tau$.

Proof. [Proposition ?] In order to use Theorem ??, we first need to define an appropriate symmetric random matrix \mathbf{X} , whose realizations $\mathbf{X}_{\{q\}}$ correspond to the sampling procedure of Algorithm ??, and $\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} = \frac{1}{q} \sum_{i=1}^q \mathbf{X}_i$. The realization of the matrix random variable are

$$\mathbf{X}_i = \mathbf{I}_d - \left(\frac{\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)}}{\tilde{\Pi}_i} \right) = \mathbf{I}_d - \left(\frac{\sum_{l \in \mathcal{K}_i} \mathbf{U}_{(l)}^\top \mathbf{U}_{(l)}}{\tilde{\Pi}_i} \right)$$

where \mathcal{K}_ι^i indicates the ι^{th} block of $\mathbf{A} \in \mathbb{R}^{N \times d}$, which was sampled at trial i . This holds for all $\iota \in \mathbb{N}_K$. The expectation of the symmetric random matrix \mathbf{X} is

$$\begin{aligned}\mathbb{E}[\mathbf{X}] &\stackrel{\ddagger}{=} \mathbf{I}_d - \left(\sum_{j=1}^K \tilde{\Pi}_j \cdot \frac{\mathbf{U}_{(\mathcal{K}_j)}^\top \mathbf{U}_{(\mathcal{K}_j)}}{\tilde{\Pi}_j} \right) \\ &= \mathbf{I}_d - \sum_{j=1}^K \mathbf{U}_{(\mathcal{K}_j)}^\top \mathbf{U}_{(\mathcal{K}_j)} \\ &\stackrel{\#}{=} \mathbf{I}_d - \sum_{l=1}^N \mathbf{U}_{(l)}^\top \mathbf{U}_{(l)} \\ &= \mathbf{I}_d - \mathbf{I}_d \\ &= \mathbf{0}_{d \times d}\end{aligned}$$

where \ddagger follows from the fact that each realization \mathbf{X}_i corresponding to each $\{\mathcal{K}_j^i\}_{j=1}^K$ of the random matrix is sampled with probability $\tilde{\Pi}_j$, and in $\#$ we simplify the expression in terms of rank-1 outer-product matrices. Furthermore, for $\{\tilde{\ell}_l\}_{l=1}^N$ the corresponding *approximate* leverage scores of \mathbf{A}

$$\|\mathbf{X}_i\|_2 \stackrel{\natural}{\leq} \|\mathbf{I}_d\|_2 + \frac{\|\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)}\|_2}{\tilde{\Pi}_i} \stackrel{\diamond}{\leq} 1 + \frac{\sum_{l \in \mathcal{K}_i} \ell_l}{\left(\sum_{l \in \mathcal{K}_i} \tilde{\ell}_l \right) / d} = 1 + \frac{d \cdot \Pi_i}{\tilde{\Pi}_i} = 1 + \frac{d}{\beta}$$

where in \natural we use the triangle inequality on the definition of \mathbf{X}_i , and in \diamond we use it on the sum of outer-products (the numerator of second summand).

We now upper bound the variance of the copies of \mathbf{X} :

$$\begin{aligned}\|\mathbb{E}[\mathbf{X}_i^\top \mathbf{X}_i]\|_2 &= \left\| \mathbb{E} \left[\left(\mathbf{I}_d - \left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} / \tilde{\Pi}_i \right) \right)^\top \left(\mathbf{I}_d - \left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} / \tilde{\Pi}_i \right) \right) \right] \right\|_2 \\ &= \left\| \mathbf{I}_d - 2 \cdot \mathbb{E} \left[\left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} / \tilde{\Pi}_i \right) \right] + \mathbb{E} \left[\left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} \right)^2 / \tilde{\Pi}_i^2 \right] \right\|_2 \\ &= \left\| 2 \left(\mathbf{I}_d - \overbrace{\mathbb{E} \left[\left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} / \tilde{\Pi}_i \right) \right]}^{=\mathbf{I}_d} \right) - \mathbf{I}_d + \mathbb{E} \left[\left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} \right)^2 / \tilde{\Pi}_i^2 \right] \right\|_2 \\ &= \left\| \mathbb{E} \left[\left(\mathbf{U}_{(\mathcal{K}_i)}^\top \mathbf{U}_{(\mathcal{K}_i)} \right)^2 / \tilde{\Pi}_i^2 \right] - \mathbf{I}_d \right\|_2 \\ &= \left\| \left(\sum_{l=1}^K \Pi_l \cdot \left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right)^2 / \tilde{\Pi}_l^2 \right) - \mathbf{I}_d \right\|_2 \\ &\stackrel{\#}{\leq} \left\| \left(\sum_{l=1}^K \frac{1}{\beta} \left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right)^2 / \tilde{\Pi}_l \right) - \mathbf{I}_d \right\|_2 \\ &= \left\| \left(\sum_{l=1}^K \frac{d}{\beta} \cdot \frac{\left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right)^2}{\|\mathbf{U}_{(\mathcal{K}_l)}\|_F^2} \right) - \mathbf{I}_d \right\|_2 \\ &\leq \left\| \left(\sum_{l=1}^K \frac{d}{\beta} \cdot \frac{\left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right)^2}{\|\mathbf{U}_{(\mathcal{K}_l)}\|_2^2} \right) - \mathbf{I}_d \right\|_2 \\ &\stackrel{\flat}{=} \left\| \frac{d}{\beta} \cdot \left(\sum_{l=1}^K \left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right)^2 \right) - \mathbf{I}_d \right\|_2 \\ &\stackrel{\natural}{\leq} \left\| \frac{d}{\beta} \left(\sum_{l=1}^K \left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right) \left(\mathbf{I}_d - \mathbf{U}_{(\bar{\mathcal{K}}_l)}^\top \mathbf{U}_{(\bar{\mathcal{K}}_l)} \right) \right) - \mathbf{I}_d \right\|_2 \\ &= \left\| \frac{d}{\beta} \left(\overbrace{\sum_{l=1}^K \mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)}}^{=\mathbf{I}_d} \right) - \frac{d}{\beta} \left(\sum_{l=1}^K \left(\mathbf{U}_{(\mathcal{K}_l)}^\top \mathbf{U}_{(\mathcal{K}_l)} \right) \left(\mathbf{U}_{(\bar{\mathcal{K}}_l)}^\top \mathbf{U}_{(\bar{\mathcal{K}}_l)} \right) \right) - \mathbf{I}_d \right\|_2\end{aligned}$$

$$\begin{aligned}
&= \left\| (d/\beta - 1) \cdot \mathbf{I}_d - \frac{d}{\beta} \left(\sum_{\iota=1}^K \mathbf{U}_{(\mathcal{K}_\iota)}^\top \overbrace{(\mathbf{U}_{(\mathcal{K}_\iota)} \mathbf{U}_{(\bar{\mathcal{K}}_\iota)}^\top)}^{\mathbf{0}_{d \times d}} \mathbf{U}_{(\bar{\mathcal{K}}_\iota)} \right) \right\|_2 \\
&= \|(d/\beta - 1) \cdot \mathbf{I}_d\|_2 \\
&= d/\beta - 1
\end{aligned}$$

where in \sharp we used the fact $\Pi_\iota/\tilde{\Pi}_\iota \leq 1/\beta$, in \flat that $\|\mathbf{U}_{(\mathcal{K}_\iota)}\|_2^2 = 1$, and in \natural that $\mathbf{U}_{(\mathcal{K}_\iota)}^\top \mathbf{U}_{(\mathcal{K}_\iota)} = \mathbf{I}_d - \mathbf{U}_{(\bar{\mathcal{K}}_\iota)}^\top \mathbf{U}_{(\bar{\mathcal{K}}_\iota)}$ for each ι .

According to Theorem ??, we substitute $\gamma = 1 + d$ and $\sigma^2 = d/\beta - 1$ to get

$$\begin{aligned}
\frac{1}{q} \sum_{i=1}^q \mathbf{X}_i &= \frac{1}{q} \sum_{i=1}^q \left(\mathbf{I}_d - \frac{\mathbf{U}_{(\mathcal{K}_{j(i)})}^\top \mathbf{U}_{(\mathcal{K}_{j(i)})}}{\tilde{\Pi}_{j(i)}} \right) \\
&= \mathbf{I}_d - \frac{1}{q} \left(\sum_{i=1}^q \frac{\mathbf{U}_{(\mathcal{K}_{j(i)})}^\top \mathbf{U}_{(\mathcal{K}_{j(i)})}}{\tilde{\Pi}_{j(i)}} \right) \\
&= \mathbf{I}_d - \mathbf{U}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{U}
\end{aligned}$$

where the last equality follows from the definition of $\tilde{\mathbf{S}}$. Putting everything together into Theorem ??, we get that

$$\Pr [\|\mathbf{I}_d - \mathbf{U}^\top \tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} \mathbf{U}\|_2 > \epsilon] \leq 2d \cdot e^{-q\epsilon^2 \Theta(\beta/d)}.$$

□

Proof. [Theorem ??] By substituting $q = \Theta\left(\frac{d}{\tau} \log(2d/\delta)/(\beta\epsilon^2)\right)$ in (??) and taking the complementary event, we attain the desired statement. □

Before we prove Corollary ??, we introduce the notion of *block α -balanced*, which is a generalization of α -balanced from [?]. The sampling distribution $\Pi_{\{K\}}$ is said to be *block α -balanced*, if

$$\max_{i \in \mathbb{N}_K} \{\Pi_i\} \leq \frac{\alpha}{N/\tau} = \frac{\alpha}{K} \quad (31)$$

for some constant α independent of K and q . Furthermore, in our context, if the individual leverage scores $\pi_{\{N\}}$ are α -balanced for α independent of N and r , then the block leverage scores $\Pi_{\{K\}}$ are block α -balanced, as

$$\max_{i \in \mathbb{N}_K} \{\Pi_i\} \leq \tau \cdot \max_{j \in \mathbb{N}_N} \{\pi_j\} \leq \tau \cdot \frac{\alpha}{N} = \frac{\alpha}{N/\tau} = \frac{\alpha}{K}. \quad (32)$$

Proof. [Corollary ??] From the proof of [?, Theorem 1], we simply need to show that

$$\|\mathbb{E}[\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}}]\|_2 \leq \eta \cdot \frac{r}{N}$$

for $\tilde{\mathbf{S}}$ a single sketch produced in Algorithm ??, and an appropriate constant η independent of N and r . We assume that the individual leverage scores $\pi_{\{N\}}$ are α -balanced, where α is a constant independent of N and r . By (??), it follows that the block leverage scores $\Pi_{\{K\}}$ are block α -balanced; i.e. $\Pi_i \leq \Pi_K \leq \frac{\alpha}{K}$ for all $i \in \mathbb{N}_{K-1}$.

A direct computation shows that

$$\begin{aligned}
(\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} &= \left((\mathbf{D} \cdot \boldsymbol{\Omega} \otimes \mathbf{I}_\tau) \cdot (\boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top \otimes \mathbf{I}_\tau) \right)^{-1} \\
&= \left((\mathbf{D} \cdot \boldsymbol{\Omega} \cdot \boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top) \otimes \mathbf{I}_\tau \right)^{-1} \\
&= (\mathbf{D} \cdot \boldsymbol{\Omega} \cdot \boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top)^{-1} \otimes \mathbf{I}_\tau
\end{aligned}$$

and consequently

$$\begin{aligned}
\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}} &= (\boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top \otimes \mathbf{I}_\tau) \cdot \left((\mathbf{D} \cdot \boldsymbol{\Omega} \cdot \boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top)^{-1} \otimes \mathbf{I}_\tau \right) \cdot (\mathbf{D} \cdot \boldsymbol{\Omega} \otimes \mathbf{I}_\tau) \\
&= \left(\boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top \cdot (\mathbf{D} \cdot \boldsymbol{\Omega} \cdot \boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top)^{-1} \otimes \mathbf{I}_\tau \right) \cdot (\mathbf{D} \cdot \boldsymbol{\Omega} \otimes \mathbf{I}_\tau) \\
&= \underbrace{\left(\boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top \cdot (\mathbf{D} \cdot \boldsymbol{\Omega} \cdot \boldsymbol{\Omega}^\top \cdot \mathbf{D}^\top)^{-1} \cdot \mathbf{D} \cdot \boldsymbol{\Omega} \right)}_{:= Z \in \mathbb{R}_{\geq 0}^{K \times K}} \otimes \mathbf{I}_\tau
\end{aligned}$$

where $Z = \sum_{i=1}^q Z_i$, for $\{Z_i\}_{i=1}^q$ rank-1 outer-product matrices of size $K \times K$ corresponding to each sampling trial, through Ω . For each sampling trial, we know that the i^{th} block is sampled with probability Π_i . Furthermore, if the i^{th} block is sampled at iteration ι , it follows that

$$\begin{aligned} Z_i &= \mathbf{e}_i \cdot \sqrt{\frac{1}{q\Pi_i}} \cdot \left(\mathbf{e}_i^\top \cdot \left(\sqrt{\frac{1}{q\Pi_i}} \right)^2 \cdot \mathbf{e}_i \right)^{-1} \cdot \sqrt{\frac{1}{q\Pi_i}} \cdot \mathbf{e}_i^\top \\ &= \mathbf{e}_i \cdot \sqrt{\frac{1}{q\Pi_i}} \cdot \left(\sqrt{q\Pi_i} \right)^2 \cdot \sqrt{\frac{1}{q\Pi_i}} \cdot \mathbf{e}_i^\top \\ &= \mathbf{e}_i \cdot \mathbf{e}_i^\top \end{aligned}$$

hence

$$\mathbb{E} \left[\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}} \right] = \mathbb{E}_\Omega \left[\sum_{j=1}^K \mathbf{e}_j \cdot \mathbf{e}_j^\top \right] \otimes \mathbf{I}_\tau = \left(\sum_{j=1}^K \mathbb{E}_\Omega [\mathbf{e}_j \cdot \mathbf{e}_j^\top] \right) \otimes \mathbf{I}_\tau = \mathbf{Q} \otimes \mathbf{I}_\tau$$

where $\mathbf{Q} = \text{diag}(\{h_j\}_{j=1}^q)$, for $h_i = 1 - (1 - \Pi_i)^q$ the probability that the i^{th} block was sampled at least once. Since we assume that the blocks $\mathbf{A}_{\{K\}}$ are indexed in ascending order according their block leverage scores; *i.e.* $\Pi_i \leq \Pi_{i+1}$ for all $i \in \mathbb{N}_{K-1}$, it follows that $h_i \leq h_K$ for all i ; thus

$$\mathbb{E} \left[\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}} \right] = \text{diag}(\{h_j\}_{j=1}^q) \otimes \mathbf{I}_\tau \preceq h_K \cdot \mathbf{I}_N = (1 - (1 - \Pi_K)^q) \cdot \mathbf{I}_N \preceq q\Pi_K \cdot \mathbf{I}_N.$$

Consequently, since $\Pi_{\{K\}}$ are block α -balanced; we have

$$\left\| \mathbb{E} \left[\tilde{\mathbf{S}}^\top (\tilde{\mathbf{S}} \tilde{\mathbf{S}}^\top)^{-1} \tilde{\mathbf{S}} \right] \right\|_2 \leq q \cdot \Pi_K \leq \alpha \cdot \frac{q}{K} = \alpha \cdot \frac{r}{N}.$$

This completes the proof, as we can take $\eta = \alpha$. \square

APPENDIX B CONCRETE EXAMPLE OF INDUCED SKETCHING

In this appendix, we give a simple demonstration of the induced sketching matrices, through our proposed GC approach. For simplicity, we will consider exact sampling, with $\Pi_{\{K\}} = \{3/20, 3/20, 4/20, 5/20, 5/20\}$ for $K = 5$, an arbitrary large block size of τ , and $m = 20$. The optimal replication numbers resulting from this distribution are $r_{\{K\}}^* = \{3, 3, 4, 5, 5\}$, hence $m = R$. In order to obtain a reduced dimension r which is 60% of the original N , we carry out $q = 3$ sampling trials at each iteration.

Let the resulting index multisets corresponding to the encoded pairs $(\tilde{A}_j, \tilde{b}_j)$ of the first four iterations; along with the resulting estimated gradients, be:

$$\begin{aligned} 1) \mathcal{S}^{[1]} &= \{1, 4, 5\} \implies \hat{g}^{[1]} = \nabla_{\mathbf{x}} L_{\mathbf{S}} \left(\tilde{\mathbf{S}}_{[1]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[1]} \right) = \hat{g}_1^{[1]} + \hat{g}_4^{[1]} + \hat{g}_5^{[1]} \\ 2) \mathcal{S}^{[2]} &= \{3, 5, 5\} \implies \hat{g}^{[2]} = \nabla_{\mathbf{x}} L_{\mathbf{S}} \left(\tilde{\mathbf{S}}_{[2]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[2]} \right) = \hat{g}_3^{[2]} + \hat{g}_5^{[2]} + \hat{g}_5^{[2]} \\ 3) \mathcal{S}^{[3]} &= \{2, 4, 5\} \implies \hat{g}^{[3]} = \nabla_{\mathbf{x}} L_{\mathbf{S}} \left(\tilde{\mathbf{S}}_{[3]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[3]} \right) = \hat{g}_2^{[3]} + \hat{g}_4^{[3]} + \hat{g}_5^{[3]} \\ 4) \mathcal{S}^{[4]} &= \{4, 1, 4\} \implies \hat{g}^{[4]} = \nabla_{\mathbf{x}} L_{\mathbf{S}} \left(\tilde{\mathbf{S}}_{[4]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[4]} \right) = \hat{g}_4^{[4]} + \hat{g}_1^{[4]} + \hat{g}_4^{[4]}. \end{aligned}$$

Then, the corresponding *induced* block leverage score sketching matrices of Algorithm ??, are:

$$\begin{aligned} \tilde{\mathbf{S}}_{[1]} &= \begin{pmatrix} 1/\sqrt{3\Pi_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/\sqrt{3\Pi_4} & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{3\Pi_5} \end{pmatrix} \otimes \mathbf{I}_\tau & \tilde{\mathbf{S}}_{[2]} &= \begin{pmatrix} 0 & 0 & 1/\sqrt{3\Pi_3} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{3\Pi_5} \\ 0 & 0 & 0 & 0 & 1/\sqrt{3\Pi_5} \end{pmatrix} \otimes \mathbf{I}_\tau \\ \tilde{\mathbf{S}}_{[3]} &= \begin{pmatrix} 0 & 1/\sqrt{3\Pi_2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/\sqrt{3\Pi_4} & 0 \\ 0 & 0 & 0 & 0 & 1/\sqrt{3\Pi_5} \end{pmatrix} \otimes \mathbf{I}_\tau & \tilde{\mathbf{S}}_{[4]} &= \begin{pmatrix} 0 & 0 & 0 & 1/\sqrt{3\Pi_4} & 0 \\ 1/\sqrt{3\Pi_1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/\sqrt{3\Pi_4} & 0 \end{pmatrix} \otimes \mathbf{I}_\tau \end{aligned}$$

each of which are of size $(3\tau) \times (5\tau)$.

APPENDIX C COMPARISON TO THE BLOCK-SRHT

An alternative view point is that the matrix $\tilde{\mathbf{U}}_{\text{exp}} := \Psi \cdot (\Sigma \mathbf{V}^\top)^{-1}$ has uniform Frobenius block scores, which further justifies the proposed GC approach for homogeneous servers. This resembles the main idea behind the SRHT [?], [?] and different variants known as *block-SRHT* [?], [?], where a random projection is applied to the data matrix to “flatten” its leverage scores, *i.e.* make them close to uniform. These two techniques for flattening the corresponding block scores are vastly different, and the proximity of the corresponding block scores are measured and quantified differently. In contrast to the block-SRHT, the quality of the approximation in our case depends on the misestimation factor β_{Π} ; and is not quantified probabilistically.

We note that since $\tilde{\Psi}$ has repeated blocks from the expansion, the scores we consider in Lemma ?? are not the block leverage scores of $\tilde{\Psi}$. The Frobenius block scores of $\tilde{\mathbf{U}}_{\text{exp}}$, are in fact the corresponding block leverage scores of $\tilde{\mathbf{A}}$, which are replicated in the expansion. Moreover, note that the closer $\beta_{\tilde{\Pi}}$ is to 1, the closer the sampling distribution according to the Frobenius block scores of $\tilde{\mathbf{U}}_{\text{exp}}$; which we denote by $\tilde{Q}_{\{R\}}$, is to being exactly uniform. We denote the uniform sampling distribution by $\mathcal{U}_{\{R\}}$.

Lemma 4. *When $\tilde{\Pi}_{\{K\}} = \Pi_{\{K\}}$, the sampling distribution $\tilde{Q}_{\{R\}}$ is uniform. When $\tilde{\Pi}_\iota \geq \beta_{\tilde{\Pi}} \Pi_\iota$ for $\beta_{\tilde{\Pi}} = \min_{i \in \mathbb{N}_K} \{\Pi_i / \tilde{\Pi}_i\} \in (0, 1)$ and all $\iota \in \mathbb{N}_K$, the resulting distribution $\tilde{Q}_{\{R\}}$ is approximately uniform, and satisfies $d_{\mathcal{U}, \tilde{Q}} \leq 1/(R\beta_{\tilde{\Pi}})$.*

Proof. Let $\mathbf{U} = [\mathbf{U}_1^\top \cdots \mathbf{U}_K^\top]^\top$ denote the corresponding blocks of \mathbf{U} according the partitioning of \mathcal{D} . Without loss of generality, assume that the data points within each partition are consecutive rows of \mathbf{A} , and $\mathbf{U}_\iota \in \mathbb{R}^{\tau \times d}$ for all $\iota \in \mathbb{N}_K$.

From (??) and (??), it follows that

$$\begin{aligned} \Psi &= \tilde{\mathbf{E}} \cdot \tilde{\mathbf{A}} = (\mathbf{E} \otimes \mathbf{I}_\tau) \cdot (\mathbf{G} \cdot \mathbf{U} \Sigma \mathbf{V}^\top) \\ &= (\mathbf{E} \otimes \mathbf{I}_\tau) \cdot \left[\mathbf{U}_1^\top / \sqrt{q\tilde{\Pi}_1} \cdots \mathbf{U}_K^\top / \sqrt{q\tilde{\Pi}_K} \right]^\top \cdot \Sigma \mathbf{V}^\top \\ &=: (\mathbf{E} \otimes \mathbf{I}_\tau) \cdot \left[\tilde{\mathbf{U}}_1^\top \cdots \tilde{\mathbf{U}}_K^\top \right]^\top \cdot \Sigma \mathbf{V}^\top \\ &=: \overbrace{\left[\tilde{\mathbf{U}}_1^\top \cdots \tilde{\mathbf{U}}_1^\top \quad \tilde{\mathbf{U}}_2^\top \cdots \tilde{\mathbf{U}}_2^\top \cdots \tilde{\mathbf{U}}_K^\top \cdots \tilde{\mathbf{U}}_K^\top \right]^\top}_{\tilde{\mathbf{U}}_{\text{exp}} \in \mathbb{R}^{R\tau \times d}} \cdot \Sigma \mathbf{V}^\top. \end{aligned}$$

Note that $\tilde{\mathbf{U}}_{\text{exp}} \Sigma \mathbf{V}^\top$ is not the SVD of Ψ . For the normalizing factor of $\frac{q}{Rd}$:

$$\tilde{Q}_\iota = \frac{q}{Rd} \cdot \|\tilde{\mathbf{U}}_\iota\|_F^2 = \frac{q}{Rd} \cdot \frac{\|\mathbf{U}_\iota\|_F^2}{q\tilde{\Pi}_\iota} = \frac{\Pi_\iota}{R\tilde{\Pi}_\iota} \leq \frac{1}{R\beta_{\tilde{\Pi}}} \implies \sum_{i=1}^R |\tilde{Q}_i - 1/R| \triangleq \frac{R}{R\beta_{\tilde{\Pi}}} = \frac{1}{\beta_{\tilde{\Pi}}},$$

where \triangle follows from the fact that $|\tilde{Q}_i - 1/R| \leq 1/(R\beta_{\tilde{\Pi}})$ for each $i \in \mathbb{N}_R$. After normalizing by $1/R$ according to the distortion metric, we deduce that $d_{\mathcal{U}, \tilde{Q}} \leq 1/(R\beta_{\tilde{\Pi}})$.

In the case where $\tilde{\Pi}_{\{K\}} = \Pi_{\{K\}}$, we have

$$\tilde{Q}_\iota = \frac{q}{Rd} \cdot \|\tilde{\mathbf{U}}_\iota\|_F^2 = \frac{q}{Rd} \cdot \frac{\|\mathbf{U}_\iota\|_F^2}{q\Pi_\iota} = \frac{\Pi_\iota}{R\Pi_\iota} = \frac{1}{R}$$

for all $\iota \in \mathbb{N}_K$, thus $\tilde{Q}_{\{K\}} = \mathcal{U}_{\{K\}}$. □

Finally, in Proposition ?? we show when the block leverage score sampling sketch of Algorithm ?? and the block-SRHT of [?] have the same ℓ_2 -s.e. guarantee. We first recall the corresponding result to Theorem ??, of the block-SRHT.

Theorem 5 ([?, Theorem 7]). *The block-SRHT $\mathbf{S}_{\tilde{\mathbf{H}}}$ is a ℓ_2 -s.e. of \mathbf{A} . For $\delta > 0$ and $q = \Theta(\frac{d}{\tau} \log(Nd/\delta) \cdot \log(2d/\delta)/\epsilon^2)$:*

$$\Pr [\|\mathbf{I}_d - \mathbf{U}^\top \mathbf{S}_{\tilde{\mathbf{H}}}^\top \mathbf{S}_{\tilde{\mathbf{H}}} \mathbf{U}\|_2 \leq \epsilon] \geq 1 - \delta.$$

Proposition 5. *Let $\beta = 1$. For $\delta = e^{\Theta(1)}/(Nd)$, the sketches of Algorithm ?? and the block-SRHT of [?] achieve the same asymptotic ℓ_2 -s.e. guarantee, for the same number of sampling trials q .*

Proof. For $\delta = e^{\Theta(1)}/(Nd)$, the two sketching methods have the same q ; and both satisfy the ℓ_2 -s.e. property with error probability $1 - \delta$. □

APPENDIX D

CONTRACTION RATE OF BLOCK LEVERAGE SCORE SAMPLING

In this appendix we quantify the contraction rate of our method on the error term $\mathbf{x}^{[s]} - \mathbf{x}^*$, which further characterizes the convergence of SD after applying our method. The contraction rate is compared to that of regular SD.

Recall that the contraction rate of an iterative process given by a function $h(x^{[s]})$ is the constant $\gamma \in (0, 1)$ for which at each iteration we are guaranteed that $h(x^{[s+1]}) \leq \gamma \cdot h(x^{[s]})$, therefore $h(x^{[s]}) \leq \gamma_s \cdot h(x^{[0]})$. Let ξ be a fixed step-size, $\tilde{\mathbf{S}}_{[s]}$ the induced sketching matrix of Algorithm ?? at iteration s , and define $\mathbf{B}_{SD} = (\mathbf{I}_d - 2\xi \cdot \mathbf{A}^\top \mathbf{A})$ and $\mathbf{B}_s = (\mathbf{I}_d - 2\xi \cdot \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{A})$. We note that the contraction rates could be further improved if one also incorporates an optimal step-size. It is also worth noting that when weighting from Appendix ?? is introduced, we have the same contraction rate and straggler ratio.

Lemma 5. *For $\tilde{\mathbf{S}}$ the sketching matrix of Algorithm ??, we have $\mathbb{E} [\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}}] = \mathbf{I}_N$.*

Proof. Similar to the proof of Proposition ??, we define a symmetric random matrix \mathbf{Y} , whose realizations correspond to the sampled and rescaled submatrices of Algorithm 1. The realizations are

$$\mathbf{Y}_i = \frac{\mathbf{I}_{(\mathcal{K}_i^i)}^\top \mathbf{I}_{(\mathcal{K}_i^i)}}{q\tilde{\Pi}_i} = \frac{\sum_{l \in \mathcal{K}_i^i} \mathbf{e}_l \mathbf{e}_l^\top}{q\tilde{\Pi}_i}.$$

After q sampling trials, we have $\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}} = \sum_{i=1}^q \mathbf{Y}_i$. It follows that

$$\mathbb{E} [\tilde{\mathbf{S}}^\top \tilde{\mathbf{S}}] = \mathbb{E} \left[\sum_{i=1}^q \mathbf{Y}_i \right] = \sum_{i=1}^q \mathbb{E} [\mathbf{Y}_i] = q \cdot \left(\sum_{j=1}^K \tilde{\Pi}_j \cdot \frac{\mathbf{I}_{(\mathcal{K}_j)}^\top \mathbf{I}_{(\mathcal{K}_j)}}{q\tilde{\Pi}_j} \right) = \sum_{j=1}^K \mathbf{I}_{(\mathcal{K}_j)}^\top \mathbf{I}_{(\mathcal{K}_j)} = \sum_{l=1}^N \mathbf{e}_{(l)} \mathbf{e}_{(l)}^\top = \mathbf{I}_N.$$

□

Theorem 6. *The contraction rate of our GC approach through the expected sketch $\tilde{\mathbf{S}}_{[s]}$ at each iteration, is equal to the contraction rate of regular SD. Specifically, for $e_s := \mathbf{x}^{[s]} - \mathbf{x}^*$ the error at iteration s and $\gamma_{SD} = \lambda_1(\mathbf{B}_{SD})$ the contraction rate of regular SD, we have $\|\mathbb{E}[e_{s+1}]\|_2 \leq \gamma_{SD} \cdot \|e_s\|_2$.*

Proof. For a fixed step-size 2ξ , our SD parameter update at iteration $s+1$ is

$$\mathbf{x}^{[s+1]} \leftarrow \mathbf{x}^{[s]} - 2\xi \cdot \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}),$$

where for regular SD we have $\tilde{\mathbf{S}}_{[s]} \leftarrow \mathbf{I}_N$. At iteration $s+1$, the error e_s of the previous iteration is not random, hence $\mathbb{E}[e_s] = e_s$. By substituting the expression of \mathbf{B}_s , it follows that

$$\begin{aligned} e_{s+1} &= \mathbf{x}^{[s+1]} - \mathbf{x}^* \\ &= \left(\mathbf{x}^{[s]} - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} (\mathbf{A}\mathbf{x}^{[s]} - \mathbf{b}) \right) - \mathbf{x}^* \\ &= \mathbf{x}^{[s]} - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{A}\mathbf{x}^{[s]} + 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b} - \mathbf{x}^* \\ &= \mathbf{B}_s \mathbf{x}^{[s]} - \left(\mathbf{x}^* - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b} \right) \\ &= \mathbf{B}_s \mathbf{x}^{[s]} - \left(\mathbf{x}^* - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} (\mathbf{A}\mathbf{x}^* + \mathbf{b}^\perp) \right) \\ &= \mathbf{B}_s \left(\mathbf{x}^{[s]} - \mathbf{x}^* \right) - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b}^\perp \\ &= \mathbf{B}_s e_s - 2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b}^\perp \end{aligned} \quad (33)$$

and by Lemma ??

$$\mathbb{E} \left[2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b}^\perp \right] = 2\xi \mathbf{A}^\top \mathbb{E} \left[\tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \right] \mathbf{b}^\perp = 2\xi \mathbf{A}^\top \mathbf{b}^\perp = \mathbf{0}_{d \times 1} \quad (34)$$

as \mathbf{b}^\perp lies in the kernel of \mathbf{A}^\top , and

$$\mathbb{E} [\mathbf{B}_s] = \mathbf{I}_d - 2\xi \mathbf{A}^\top \mathbb{E} \left[\tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \right] \mathbf{A} = \mathbf{I}_d - 2\xi \mathbf{A}^\top \mathbf{A} = \mathbf{B}_{SD}. \quad (35)$$

From (??), (??) and (??), it follows that

$$\mathbb{E} [e_{s+1}] = \mathbb{E} [\mathbf{B}_s e_s] - \mathbb{E} \left[2\xi \mathbf{A}^\top \tilde{\mathbf{S}}_{[s]}^\top \tilde{\mathbf{S}}_{[s]} \mathbf{b}^\perp \right] = \mathbb{E} [\mathbf{B}_s] \cdot e_s = \mathbf{B}_{SD} \cdot e_s. \quad (36)$$

This gives us the contraction rate of the expected sketch through Algorithm ??

$$\|\mathbb{E}[e_{s+1}]\|_2 \leq \lambda_1(\mathbb{E}[\mathbf{B}_s]) \cdot \|e_s\|_2 = \lambda_1(\mathbf{B}_{SD}) \cdot \|e_s\|_2 \implies \gamma_{s+1} = \lambda_1(\mathbf{B}_{SD}). \quad (37)$$

By replacing \mathbf{B}_s with \mathbf{B}_{SD} in (??) and $\tilde{\mathbf{S}}_{[s]} \leftarrow \mathbf{I}_N$, we conclude that the contraction rate of SD is $\gamma_{SD} = \lambda_1(\mathbf{B}_{SD})$. □

We conclude this appendix by stating the expected ratio of dimensions r to N , and stragglers to servers.

Remark 4. *The expected ratio of the reduced dimension r to the original dimension N is $q(T)\tau/N$, and the expected straggler to servers ratio is $(m-q)/m$. Since we stop receiving computations at a time instance T ; we expect that $q \leftarrow q(T)$ computations are received, hence there are $m-q$ stragglers. Thus, the straggler to servers ratio is $(m-q)/m$. The expected ratio between the two dimensions is immediate from the fact that $r = q\tau$ is the new reduced dimension, in the case where $q \leq K$.*

APPENDIX E

WEIGHTED BLOCK LEVERAGE SCORE SKETCH

So far, we have considered sampling w.r. according to the normalized block leverage scores, to reduce the effective dimension N of \mathbf{A} and \mathbf{b} to $r = q\tau$. In this appendix, we show that by *weighting* the sampled blocks according to the sampling which has taken place through Ω for the construction of the sketching matrix $\tilde{\mathbf{S}}$, we can further compress the data matrix \mathbf{A} ; and get the same results when first and second order optimization methods are used to solve (??). The weighting we propose is more beneficial with non-uniform distributions, as we expect the sampling w.r. to capture the importance of the more influence blocks. The *weighted sketching matrix* $\tilde{\mathbf{S}}_{\mathbf{w}}$ we propose, is a simple extension to $\tilde{\mathbf{S}}$ of Algorithm ???. For simplicity, we do not consider iterative sketching, though similar arguments and guarantees can be derived.

The main idea is to not keep repetitions of blocks which were sampled multiple times, but rather weigh each block by the number of times it was sampled. By doing so, we retain the *weighted sketch* $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$ of size $\bar{q}\tau \times d$; for \bar{q} the number of distinct blocks that were sampled.¹⁰ Additionally, the gradient and Hessian of $L_{\mathbf{S}}(\tilde{\mathbf{S}}_{\mathbf{w}}, \mathbf{A}, \mathbf{b}; \mathbf{x})$ are respectively equal to those of $L_{\mathbf{S}}(\tilde{\mathbf{S}}, \mathbf{A}, \mathbf{b}; \mathbf{x})$; and are unbiased estimators of the gradient and Hessian of $L_{\mathbf{S}}(\mathbf{A}, \mathbf{b}; \mathbf{x})$.

To achieve the weighting algorithmically, we count how many times each of the distinct \bar{q} blocks were sampled, and at the end of the sampling procedure we multiply the blocks by their corresponding “weight”. We initialize a weight vector $\mathbf{w} = \mathbf{0}_{1 \times K}$, and in the sampling procedure whenever the i^{th} partition is drawn, we update its corresponding weight: $\mathbf{w}_i \leftarrow \mathbf{w}_i + 1$. It is clear that once q trials are been carried out, we have $\|\mathbf{w}\|_1 = q$ for $\mathbf{w} \in \mathbb{N}_0^{1 \times K}$.

Let \mathcal{S} denote the index multiset observed after the sampling procedure of Algorithm ??, and $\bar{\mathcal{S}}$ the set of indices comprising \mathcal{S} . That is, \mathcal{S} has cardinality q and may have repetitions, while $\bar{\mathcal{S}} = \mathbb{N}_K \cap \mathcal{S}$ has cardinality $\bar{q} = |\bar{\mathcal{S}}| \leq q$ with no repetitions. We denote the ratio of the two sets by $\zeta := q/\bar{q} = \|\mathbf{w}\|_1/\|\mathbf{w}\|_0 \geq 1$, which indicates how much further compression we get by utilizing the fact that blocks may be sampled multiple times. The corresponding weighted sketching matrix $\tilde{\mathbf{S}}_{\mathbf{w}}$ of $\tilde{\mathbf{S}}$ is then

$$\tilde{\mathbf{S}}_{\mathbf{w}} = \overbrace{\text{diag} \left(\left\{ \sqrt{\mathbf{w}_j / (q\Pi_j)} \right\}_{j \in \bar{\mathcal{S}}} \right)}^{\bar{\mathbf{W}}_{1/2} \in \mathbb{R}_{\geq 0}^{\bar{q} \times \bar{q}}} \cdot \mathbf{I}_{(\bar{\mathcal{S}})} \otimes \mathbf{I}_{\tau} \in \mathbb{R}^{\bar{q}\tau \times N} \quad (38)$$

where $\mathbf{I}_{(\bar{\mathcal{S}})} \in \{0, 1\}^{\bar{q} \times K}$ is the restriction of \mathbf{I}_K to the rows indexed by $\bar{\mathcal{S}}$.

For simplicity, assume that the sampling matrices which are devised for $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{S}}_{\mathbf{w}}$ follow the ordering of the sampled blocks in order of the samples, i.e. if $(\Omega_{(i)})_j = 1$ then $(\Omega_{(i+1)})_l = 1$ for $l \geq j$; and equivalently $\mathcal{S}_l \leq \mathcal{S}_{l+1}$ and $\bar{\mathcal{S}}_l < \bar{\mathcal{S}}_{l+1}$ for all valid l . We restrict the sampling matrix $\Omega \in \{0, 1\}^{q \times K}$ to its unique rows, by applying $\bar{\Omega} \in \{0, 1\}^{\bar{q} \times K}$:

$$\bar{\Omega}_{ij} = \begin{cases} 1 & \text{for } i = 1 \text{ and } j = \mathcal{S}_1 \\ 1 & \text{if } \mathcal{S}_j > \mathcal{S}_{j-1} \text{ for } j \in \mathbb{N}_q \setminus \{1\} \\ 0 & \text{otherwise} \end{cases}$$

to the left of the sampling matrix Ω of Algorithm ??, i.e. $\Omega_{\mathbf{w}} := \bar{\Omega} \cdot \Omega \in \{0, 1\}^{\bar{q} \times K}$. This sampling matrix then satisfies

$$(\Omega_{\mathbf{w}})_{ij} = \begin{cases} 1 & \text{if } j = \bar{\mathcal{S}}_j \\ 0 & \text{otherwise} \end{cases} \quad \text{for } j \in \mathbb{N}_{\bar{q}}.$$

Let $\tilde{\mathbf{w}} = \mathbf{w}_{|\bar{\mathcal{S}}} \in \mathbb{Z}_+^{1 \times \bar{q}}$ be the restriction of \mathbf{w} to its nonzero elements; hence $\|\tilde{\mathbf{w}}\|_1 = \|\mathbf{w}\|_1 = |\mathcal{S}| = q$, and define the rescaling diagonal matrix $\tilde{\mathbf{W}}_{1/2} = \text{diag} \left(\left\{ \sqrt{\tilde{\mathbf{w}}_i} \right\}_{i=1}^{\bar{q}} \right)$. We then have the following relationship

$$\tilde{\mathbf{S}}_{\mathbf{w}} = \overbrace{\left(\tilde{\mathbf{W}}_{1/2} \cdot \bar{\Omega} \cdot \mathbf{D} \cdot \bar{\Omega}^{\top} \right)}^{=\bar{\mathbf{W}}_{1/2}} \cdot \Omega_{\mathbf{w}} \otimes \mathbf{I}_{\tau} = (\tilde{\mathbf{W}}_{1/2} \otimes \mathbf{I}_{\tau}) \cdot (\Omega_{\mathbf{w}} \otimes \mathbf{I}_{\tau}) \quad (39)$$

when $\bar{\mathcal{S}} = \mathbb{N}_K \cap \mathcal{S}$.

As previously noted, $\tilde{\mathbf{S}}_{\mathbf{w}}$ has ζ times less rows than $\tilde{\mathbf{S}}$. Hence, the required storage space for the sketch $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$ drops by a multiplicatively factor of ζ , and the required operations are reduced analogously; according to the computation. The weighted sketching matrix $\tilde{\mathbf{S}}_{\mathbf{w}}$ has the following guarantees, which imply that the proposed weighting will not affect first or second order iterative methods which are used to approximate (??).

Proposition 6. *The resulting gradient and Hessian of the modified least squares problem (??) when sketching with $\tilde{\mathbf{S}}$ of Algorithm ??, are respectively identical to the resulting gradient and Hessian when sketching with $\tilde{\mathbf{S}}_{\mathbf{w}}$ presented in (??) and (??).*

¹⁰In practice, for highly non-uniform $\Pi_{\{K\}}$ we expect $\bar{q}\tau \ll r = q\tau$. The sketch $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$ could therefore be stored in much less space than $\tilde{\mathbf{S}}\mathbf{A}$, and the system of equations $\tilde{\mathbf{S}}_{\mathbf{w}}(\mathbf{A} - \mathbf{b}) = \mathbf{0}_{\bar{q}\tau \times 1}$ could have significantly fewer equations than $\tilde{\mathbf{S}}(\mathbf{A} - \mathbf{b}) = \mathbf{0}_{q\tau \times 1}$.

Proof. From Algorithm ??, the assumption on the ordering of the elements in \mathcal{S} and $\bar{\mathcal{S}}$, and the construction of $\tilde{\mathbf{S}}$, we have

$$\begin{aligned}\tilde{\mathbf{S}}_{\mathbf{w}}^{\top} \cdot \tilde{\mathbf{S}}_{\mathbf{w}} &= \left((\boldsymbol{\Omega}_{\mathbf{w}}^{\top} \cdot \tilde{\mathbf{W}}_{1/2}^{\top}) \otimes \mathbf{I}_{\tau} \right) \cdot \left((\tilde{\mathbf{W}}_{1/2} \cdot \boldsymbol{\Omega}_{\mathbf{w}}) \otimes \mathbf{I}_{\tau} \right) \\ &= \left((\boldsymbol{\Omega}^{\top} \cdot \mathbf{D}^{\top}) \otimes \mathbf{I}_{\tau} \right) \cdot \left((\mathbf{D} \cdot \boldsymbol{\Omega}) \otimes \mathbf{I}_{\tau} \right) \\ &= \tilde{\mathbf{S}}^{\top} \cdot \tilde{\mathbf{S}}.\end{aligned}$$

Let $\mathcal{T} = \biguplus_{j \in \mathcal{S}} \mathcal{K}_j$ and $\bar{\mathcal{T}} = \bigsqcup_{j \in \bar{\mathcal{S}}} \mathcal{K}_j$, thus $\bar{\mathcal{T}}$ is contained in \mathcal{T} when both are viewed as multisets. Considering the objective function $L_{\mathbf{S}}(\tilde{\mathbf{S}}, \mathbf{A}, \mathbf{b}; \mathbf{x})$ of (??), the equivalence of gradients is observed through the following computation

$$\begin{aligned}\nabla_{\mathbf{x}} L_{\mathbf{S}}(\tilde{\mathbf{S}}, \mathbf{A}, \mathbf{b}; \mathbf{x}) &= 2\mathbf{A}^{\top} \left(\tilde{\mathbf{S}}^{\top} \tilde{\mathbf{S}} \right) (\mathbf{A}\mathbf{x} - \mathbf{b}) \\ &= 2 \sum_{l \in \mathcal{T}} \mathbf{A}_{(l)}^{\top} \cdot \mathbf{D}_{ll}^2 \cdot (\mathbf{A}_{(l)}\mathbf{x} - \mathbf{b}_l) \\ &= 2 \sum_{j \in \bar{\mathcal{T}}} \tilde{\mathbf{w}}_j \cdot \mathbf{A}_{(j)}^{\top} \cdot \mathbf{D}_{jj}^2 \cdot (\mathbf{A}_{(j)}\mathbf{x} - \mathbf{b}_j) \\ &= 2 \sum_{j \in \bar{\mathcal{T}}} \mathbf{A}_{(j)}^{\top} \cdot (\tilde{\mathbf{W}}_{1/2})_{jj}^2 \cdot (\mathbf{A}_{(j)}\mathbf{x} - \mathbf{b}_j) \\ &= 2\mathbf{A}^{\top} \left(\tilde{\mathbf{S}}_{\mathbf{w}}^{\top} \tilde{\mathbf{S}}_{\mathbf{w}} \right) (\mathbf{A}\mathbf{x} - \mathbf{b}) \\ &= \nabla_{\mathbf{x}} L_{\mathbf{S}}(\tilde{\mathbf{S}}_{\mathbf{w}}, \mathbf{A}, \mathbf{b}; \mathbf{x}).\end{aligned}$$

Recall that the Hessian of the least squares objective function (??) is $\nabla_{\mathbf{x}}^2 L_{ls}(\mathbf{A}, \mathbf{b}; \mathbf{x}) = 2\mathbf{A}^{\top} \mathbf{A}$. Considering the modified objective function (??) and our sketching matrices, it follows that

$$\begin{aligned}\nabla_{\mathbf{x}}^2 L_{\mathbf{S}}(\tilde{\mathbf{S}}, \mathbf{A}, \mathbf{b}; \mathbf{x}) &= 2\mathbf{A}^{\top} \left(\tilde{\mathbf{S}}^{\top} \tilde{\mathbf{S}} \right) \mathbf{A} \\ &= 2 \sum_{l \in \mathcal{T}} \mathbf{A}_{(l)}^{\top} \cdot \mathbf{D}_{ll}^2 \cdot \mathbf{A}_{(l)} \\ &= 2 \sum_{j \in \bar{\mathcal{T}}} \tilde{\mathbf{w}}_j \cdot \mathbf{A}_{(j)}^{\top} \cdot \mathbf{D}_{jj}^2 \cdot \mathbf{A}_{(j)} \\ &= 2 \sum_{j \in \bar{\mathcal{T}}} \mathbf{A}_{(j)}^{\top} \cdot (\tilde{\mathbf{W}}_{1/2})_{jj}^2 \cdot \mathbf{A}_{(j)} \\ &= 2\mathbf{A}^{\top} \left(\tilde{\mathbf{S}}_{\mathbf{w}}^{\top} \tilde{\mathbf{S}}_{\mathbf{w}} \right) \mathbf{A} \\ &= \nabla_{\mathbf{x}}^2 L_{\mathbf{S}}(\tilde{\mathbf{S}}_{\mathbf{w}}, \mathbf{A}, \mathbf{b}; \mathbf{x})\end{aligned}$$

which completes the proof. \square

Corollary 3. *At each iteration, the gradient and Hessian of the weighted sketch system of equations $\tilde{\mathbf{S}}_{\mathbf{w}}(\mathbf{A} - \mathbf{b}) = \mathbf{0}_{\bar{q}\tau \times 1}$, are unbiased estimators of the gradient and Hessian of the original system $(\mathbf{A} - \mathbf{b}) = \mathbf{0}_{N \times 1}$.*

Proof. Denote the gradient and Hessian of the weighted sketch at iteration s by $\hat{g}_{\mathbf{w}}^{[s]}$ and $\hat{H}_{\mathbf{w}}^{[s]}$ respectively. By Proposition ?? we know that $\hat{g}_{\mathbf{w}}^{[s]} = \hat{g}^{[s]}$, and by Theorem ?? that $\mathbb{E}[\hat{g}^{[s]}] = g^{[s]}$. Hence $\mathbb{E}[\hat{g}_{\mathbf{w}}^{[s]}] = g^{[s]}$.

Following the same notation as in the proof of Theorem ??, the Hessian $\hat{H}^{[s]} = \nabla_{\mathbf{x}}^2 L_{\mathbf{S}}(\tilde{\mathbf{S}}^{[s]}, \mathbf{A}, \mathbf{b}; \mathbf{x}^{[s]})$ is

$$\hat{H}^{[s]} = 2 \sum_{i \in \mathcal{I}^{[s]}} \frac{1}{q\tilde{\Pi}_i} \mathbf{A}_i^{\top} \mathbf{A}_i$$

thus

$$\mathbb{E}[\hat{H}^{[s]}] = 2\mathbb{E} \left[\sum_{i \in \mathcal{I}^{[s]}} \frac{1}{q\tilde{\Pi}_i} \mathbf{A}_i^{\top} \mathbf{A}_i \right] = 2 \sum_{i \in \mathcal{I}^{[s]}} \sum_{j=1}^K \tilde{\Pi}_j \frac{1}{q\tilde{\Pi}_j} \mathbf{A}_j^{\top} \mathbf{A}_j = 2q \cdot \sum_{j=1}^K \frac{1}{q} \mathbf{A}_j^{\top} \mathbf{A}_j = 2\mathbf{A}^{\top} \mathbf{A}$$

which is precisely the Hessian of (??). By Proposition ??, it follows that $\mathbb{E}[\hat{H}_{\mathbf{w}}^{[s]}] = 2\mathbf{A}^{\top} \mathbf{A}$, which completes the proof. \square

Geometrically, from the point of view of adding vectors, the partial gradients of the partitions sampled will be scaled accordingly to their weights. Therefore, the partial gradients \hat{g}_i with higher weights have a greater influence in the direction of the resulting gradient \hat{g} . This was also the fundamental idea behind our sketching and GC techniques, as the partitions sampled multiple times are of greater importance.

Next, we quantify the expected dimension of the weighted sketch $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$. This shows the dependence on $\tilde{\Pi}_{\{K\}}$, and further justifies that we attain a higher compression factor ζ when the block leverage scores are non-uniform.

Theorem 7. *The expected reduced dimension of $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$ is $\left(K - \sum_{i=1}^K (1 - \tilde{\Pi}_i)^q\right) \cdot \tau$, which is maximal when $\tilde{\Pi}_{\{K\}}$ is uniform.*

Proof. It suffices to determine the expected number of distinct blocks \mathbf{A}_i which are sampled after q trials when carrying out Algorithm ???. The probability of not sampling \mathbf{A}_i at a given trial is $(1 - \tilde{\Pi}_i)$, hence not sampling \mathbf{A}_i at any trial occurs with probability $(1 - \tilde{\Pi}_i)^q$; since the trials are identical and independent. Thus, the expected number of distinct blocks being sampled is

$$\begin{aligned} \mathbb{E}[\bar{q}] &= \sum_{i=1}^K 1 \cdot \Pr[\mathbf{A}_i \text{ was sampled at least once}] \\ &= \sum_{i=1}^K (1 - \Pr[\mathbf{A}_i \text{ was not sampled at any trial}]) \\ &= \sum_{i=1}^K \left(1 - (1 - \tilde{\Pi}_i)^q\right) \\ &= K - \sum_{i=1}^K (1 - \tilde{\Pi}_i)^q. \end{aligned}$$

Thus, the expected reduced dimension is $\tau \cdot \mathbb{E}[\bar{q}]$.

Let $Q(\tilde{\Pi}_{\{K\}}) := \sum_{i=1}^K (1 - \tilde{\Pi}_i)^q$, and introduce the Lagrange multiplier $\lambda > 0$ to the constraint $R(\tilde{\Pi}_{\{K\}}) = \left(\sum_{i=1}^K \tilde{\Pi}_i - 1\right)$, to get the Lagrange function

$$\mathcal{L}(\tilde{\Pi}_{\{K\}}, \lambda) := Q(\tilde{\Pi}_{\{K\}}) + \lambda \cdot R(\tilde{\Pi}_{\{K\}}) = \sum_{i=1}^K \left(\lambda \cdot \tilde{\Pi}_i + (1 - \tilde{\Pi}_i)^q\right) - \lambda \quad (40)$$

for which

$$\frac{\partial \mathcal{L}(\tilde{\Pi}_{\{K\}}, \lambda)}{\partial \tilde{\Pi}_i} = \lambda - q(1 - \tilde{\Pi}_i)^{q-1} = 0 \quad (41)$$

$$\implies \tilde{\Pi}_i = 1 - (\lambda/q)^{1/(q-1)} \quad \text{and} \quad \lambda = q(1 - \tilde{\Pi}_i)^{q-1} \quad (42)$$

for all $i \in \mathbb{N}_K$, and

$$\frac{\partial \mathcal{L}(\tilde{\Pi}_{\{K\}}, \lambda)}{\partial \lambda} = \sum_{i=1}^K \tilde{\Pi}_i - 1 = 0. \quad (43)$$

Note that the uniform distribution $\mathcal{U}_{\{K\}} = \{\tilde{\Pi}_i = 1/K\}_{i=1}^K$ is a solution to (??). We will now verify that $\mathcal{U}_{\{K\}}$ satisfies (??). From (??); for $\tilde{\Pi}_{\{K\}} \leftarrow \mathcal{U}_{\{K\}}$, we have $\lambda = q(1 - 1/K)^{q-1} > 0$, which we substitute into (??):

$$\lambda - q(1 - \tilde{\Pi}_i)^{q-1} = q(1 - 1/K)^{q-1} - q(1 - 1/K)^{q-1} = 0. \quad (44)$$

Hence, $\mathcal{U}_{\{K\}}$ is the solution to both (??) and (??).

By the second derivative test; since $\partial^2 \mathcal{L}(\tilde{\Pi}_{\{K\}})/\partial \tilde{\Pi}_i^2 = q(q-1)(1 - \tilde{\Pi}_i)^{q-2}$ is positive for $\tilde{\Pi}_i = 1/K$, we conclude that $Q(\mathcal{U}_{\{K\}}) \leq Q(\tilde{\Pi}_{\{K\}})$ for any $\tilde{\Pi}_{\{K\}} \neq \mathcal{U}_{\{K\}}$. This implies that $\mathbb{E}[\bar{q}]$ is maximal when $\tilde{\Pi}_{\{K\}} = \mathcal{U}_{\{K\}}$, and so is the expected reduced dimension of $\tilde{\mathbf{S}}_{\mathbf{w}}\mathbf{A}$. \square

We further note that $\mathbb{E}[\bar{q}]$ from the proof of Theorem ??, is trivially minimal in the degenerate case where $\tilde{\Pi}_\ell = 1$ for a single $\ell \in \mathbb{N}_K$, and $\tilde{\Pi}_j = 0$ for every $j \in \mathbb{N}_K \setminus \{\ell\}$. This occurs in the case where $\mathbf{A}_j = \mathbf{0}_{\tau \times d}$ for each j , and \bar{q} is therefore exactly

$$K - \sum_{j \neq \ell} (1 - \tilde{\Pi}_j)^q = K - \sum_{j \neq \ell} 1^q = K - (K-1) = 1.$$