Gaussian Compression Stream: Principle and Preliminary Results

Farouk Yahaya, Matthieu Puigt, Gilles Delmaire, and Gilles Roussel

Univ. Littoral Côte d'Opale, LISIC – EA 4491, F-62228 Calais, France

Abstract— **Random projections became popular tools to process big data. In particular, when applied to Nonnegative Matrix Factorization (NMF), it was shown that structured random projections were far more efficient than classical strategies based on Gaussian compression. However, they remain costly and might not fully benefit from recent fast random projection techniques. In this paper, we thus investigate an alternative to structured random projections—named *Gaussian compression stream*—which (i) is based on Gaussian compressions only, (ii) can benefit from the above fast techniques, and (iii) is shown to be well-suited to NMF.**

1 Introduction

Dimension reduction techniques are the linchpin for solving problems involving high dimensional data. They can capture most of the important features of the underlying high data while providing the benefit of mapping onto a much lower dimensional space, due to their computational intricacies and geometric properties. Among the numerous techniques proposed in the literature, those based on randomized linear algebra [1, 2] have been shown to be particularly efficient.

In particular, they were successfully combined to Nonnegative Matrix Factorization (NMF) [3–5] and its weighted extension [6] and is sometimes named *compressed NMF*. More precisely, it was shown in [3] that *structured* random compression—based on Randomized Power Iteration (RPI)—was far more efficient than classical Gaussian Compression (GC) when applied to NMF. However, while several strategies have been proposed to speed-up GC—*e.g.*, CountGauss [7] or specific hardware [8]—RPIs still suffer from a high computational cost. This analysis remains true when considering a stable extension of RPIs named Randomized Subspace Iterations (RSIs) [1]. In this paper, we propose an extension of GC nammed GC Stream (GCS). We show that GCS allows a NMF performance which is similar to RPIs/RSIs. We lastly discuss conditions where GCS should outperform RPIs/RSIs when the computational complexity is taken into consideration.

2 Compressed NMF

NMF is a popular signal & image processing / machine learning tool which consists of estimating two $n \times p$ and $p \times m$ nonnegative matrices G and F, respectively, from a $n \times m$ nonnegative matrix X such that $X \approx G \cdot F$ [9, 10]. While several cost functions and additive constraints have been proposed to that end, in its basic form involving the Frobenius norm $||\cdot||_{\mathcal{F}}$, NMF usually consists of solving alternating subproblems, *i.e.*,

$$\hat{G} = \arg\min_{G \geq 0} ||X - G \cdot F||_{\mathcal{F}}, \qquad (1)$$

$$\hat{F} = \arg\min_{F \geq 0} ||X - G \cdot F||_{\mathcal{F}}, \qquad (2)$$

When X is large, several strategies have been proposed to speed-up the updates, *e.g.*, distributed [11] or online [12] computations, fast solvers [13], or randomized techniques [3].

Actually, several randomized strategies were proposed in the literature. In [14], the authors assumed that X is low-rank and can be replaced by a product $A \cdot B$ which helps the NMF factors to be cheaper to update, and which can be efficiently computed using randomized SVD. In [15], the authors introduced the concept of *dual random compression* described in Algorithm 1. The key idea consists in noticing that compressing X by a projection on the left or the right side still allows to estimate the full matrix F or G, respectively. The difficulty then lies in designing efficient matrices L and R: the authors in [15] used scaled Gaussian realizations as tentative compression matrices, thus following the general proof of the Johson-Lidenstrauss Lemma (JLL) [16] on which is built the theory of random projections. The authors in [3,4] then found that adding some structure on the compression matrices allows a much better NMF performance (with different tested solvers). To that end, they used RPIs [1]. To compute L, RPIs are defined as

$$L \triangleq \mathrm{QR}\left((XX^T)^q \cdot X \cdot \Omega_L\right)^T, \qquad (3)$$

where $\Omega_L \in \mathbb{R}^{m \times (p+\nu)}$ is a Gaussian random matrix, $p + \nu \ll n$, and q is a small integer (*e.g.*, $q = 4$ in [3]). This was further extended in [5] where the authors used RSIs, *i.e.*, a round-off-error stable alternative to RPIs [1]. Lastly, the authors in [17] assumed to only observe X_L. F and G_L could then be estimated from X_L. Assuming the columns of G to be sparse w.r.t a known dictionary, they could then be estimated from G_L.

Algorithm 1 Compressed NMF strategy.

Require: initial and compression matrices G, F, L, and R.
    Define $X_L \triangleq L \cdot X$ and $X_R \triangleq X \cdot R$
    repeat
        Define $F_R \triangleq F \cdot R$
        Solve (1) by resp. replacing X and F by X_R and F_R
        Define $G_L \triangleq L \cdot G$
        Solve (2) by resp. replacing X and G by X_L and G_L
    until a stopping criterion

At this stage, it should be noticed that computing random projections is costly. Indeed, deriving X_L in Algorithm 1 requires $nm(p + \nu)$ operations. Even worse, computing L in Eq. (3) requires—using the Householder QR decomposition—$2q(p + \nu)nm + 2n(p + \nu)^2 - 2/3(p + \nu)^3$ operations. As a consequence, the authors in [7] then proposed a cheaper strategy than GC—named CountGauss—which combines the ideas of both the CountSketch method [18] and GC. The former consists of generating a matrix S with only one nonzero entry per row, whose value is either $+1$ or -1 with equal probability. The product $S \cdot X$ then provides a sketch of X which is very cheap to compute. However, it requires more samples than GC

to reach the same approximation accuracy. Combining CountSketch with Gaussian projection thus leverages the CountSketch drawback while still being faster to compute than a standard Gaussian projection. In that case, applied to NMF, the matrix L reads $L = \Omega_L \cdot S$ where Ω_L and S have dimensions of size $(p+\nu) \times (p+\mu)$ and $(p+\mu) \times n$, respectively, with $\mu > \nu$ and $(p+\mu) \le n$. Another faster way to compute GC consists of using a dedicated hardware, *e.g.*, Optical Process Unit (OPU) [8]. OPUs optically perform random projections, so that they can process very large matrices in a very short time. Still, all these alternatives provide a similar performance to GC and should thus be less accurate than RPIs/RSIs when applied to NMF.

3 Gaussian Compression Stream

Principle We now introduce our proposed GCS concept. Let us first recall the JLL which states that [16] given $0 < \varepsilon < 1$, a set X of n points in $\mathbb{R}^m$, and a number $k > 8\log(n)/\varepsilon^2$, there is a linear map $f : \mathbb{R}^m \to \mathbb{R}^k$ such that $\forall u, v \in X$, $(1-\varepsilon)\|u-v\|^2 \le \|f(u)-f(v)\|^2 \le (1+\varepsilon)\|u-v\|^2$. Interestingly, the dimension k of the low-dimensional space only depends on the number n of points in the original high dimensional space and on a distortion parameter ε. Applied to NMF, the linear mapping f is a compression matrix, *i.e.*, L or R. In [3], the authors chose $k \triangleq p + \nu$ where ν was set to a small value, *i.e.*, $\nu = 10$. This led to a poor NMF performance. However, the JLL implies that by increasing k (or ν), we can reduce the distortion parameter ε as we less compress the data. However, this implies a reduced speed up of the computations.

Our proposed strategy thus reads as follows. We assume that ν is extremely large (or even infinite), so that L and R—which are draw according to a scaled Gaussian distribution—cannot fit in memory. We thus assume these matrices to be observed in a streaming fashion, *i.e.*, during the i-th NMF iteration, we only observe two $(p+\nu_i) \times n$ and $m \times (p+\nu_i)$ submatrices of L and R, denoted $L^{(i)}$ and $R^{(i)}$, respectively. As a consequence, at each NMF iteration, the update of G and F is done using different compressed matrices $X_R^{(i)}$ and $X_L^{(i)}$, respectively. In the experiments below, we find our strategy to yield the same accuracy as RSIs or vanilla NMF. Still, the approach is only based on GC so that the faster compression strategies discussed above can be applied. However, if we aim to use CPU only, GCS might not be as computationally efficient as RPIs or RSIs as we must compute $X_R^{(i)}$ and $X_L^{(i)}$ at each iteration. We thus also propose an alternative where $L^{(i)}$ and $R^{(i)}$ are updated every Max_{Iter} iterations, as shown in Algorithm 2.

Algorithm 2 Proposed compressed NMF strategy with GCS.

Require: initial matrices G, F, $i = 0$
  repeat
    Update $i = i + 1$ and get $L^{(i)}$ and $R^{(i)}$
    Define $X_R^{(i)} \triangleq X \cdot R^{(i)}$ and $X_L^{(i)} \triangleq L^{(i)} \cdot X$
    for counter $= 1$ **to** Max_{Iter} **do**
      Define $F_R^{(i)} \triangleq F \cdot R^{(i)}$ and $G_L^{(i)} \triangleq L^{(i)} \cdot G$
      Solve (1) by resp. replacing X and F by $X_R^{(i)}$ and $F_L^{(i)}$
      Solve (2) by resp. replacing X and G by $X_L^{(i)}$ and $G_L^{(i)}$
    end for
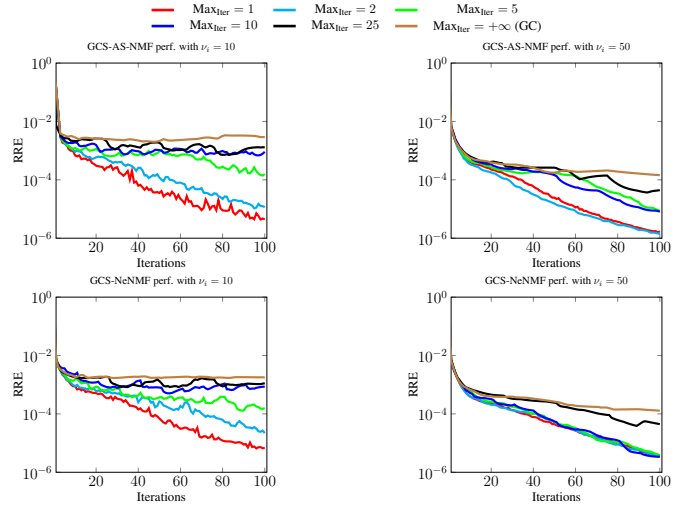 **until** a stopping criterion

---

Figure 1: NMF performance for different parameters of the GCS strategy.

Experiments We now investigate the performance of our proposed method. For that purpose, we consider 15 simulations where we draw random nonnegative matrices G and F such that $n = m = 10000$ and $p = 5$. As a consequence, their product X is a 10000×10000 rank-5 matrix. To assess the performance of the proposed method, we consider two different NMF solvers, *i.e.*, Active Set (AS-NMF) [19] and Nesterov gradient (NeNMF) [13] and three different compression strategies, *i.e.*, RSIs, GC and GCS, that we compare to the performance reached by the vanilla strategy. The performance criterion used in this paper is a Relative Reconstruction Error (RRE), defined as $\text{RRE} \triangleq \|X - G \cdot F\|_{\mathcal{F}}^2 / \|X\|_{\mathcal{F}}^2$. In each simulation, we consider the same random initialization and we compute the RRE performance for each tested method at each NMF iteration. Figure 1 shows the median RREs achieved by AS-NMF and NeNMF combined with GCS with respect to Max_{Iter} and ν_i. We notice that (i) GCS always outperforms GC, (ii) the RREs are not always decreasing along iterations when $\nu_i = 10$, and (iii) the best performance is achieved when $\text{Max}_{\text{Iter}} = 1$ or 2.
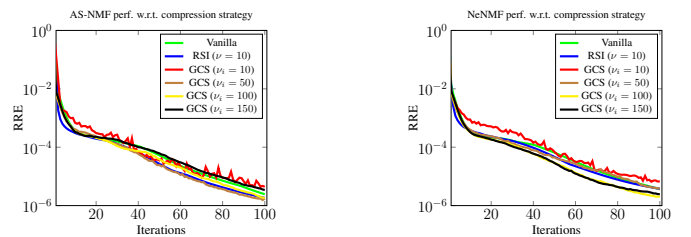

Figure 2: NMF performance with respect to compression techniques.

Figure 2 shows the evolution of the median RREs with no compression, RSIs, and GCS (with $\text{Max}_{\text{Iter}} = 1$ but for different values of ν_i). The latter provides a similar or better enhancement than the former except $\nu_i = 10$.

Conclusion We proposed a new randomized compression technique which is based on GC only and which is shown to be accurate on the considered NMF simulations. However, it requires the computation of the compressed matrices every Max_{Iter} iterations, which might be prohibitive w.r.t. structured compression if the number of NMF iterations is high. This issue might be solved using a dedicated hardware. We will investigate it in our future work and we will investigate the use of GCS for mobile sensor calibration [20, 21].

References

[1] N. Halko, P.-G. Martinsson, and J. A Tropp, "Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions," *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.

[2] P. Drineas and M. W. Mahoney, "RandNLA: randomized numerical linear algebra," *Communications of the ACM*, vol. 59, no. 6, pp. 80–90, 2016.

[3] M. Tepper and G. Sapiro, "Compressed nonnegative matrix factorization is fast and accurate," *IEEE Trans. Signal Process.*, vol. 64, no. 9, pp. 2269–2283, May 2016.

[4] N. B. Erichson, A. Mendible, S. Wihlborn, and J N. Kutz, "Randomized nonnegative matrix factorization," *Pattern Recognition Letters*, 2018.

[5] F. Yahaya, M. Puigt, G. Delmaire, and G. Roussel, "Faster-than-fast NMF using random projections and Nesterov iterations," in *Proc. iTWIST'18*, 2018.

[6] F. Yahaya, M. Puigt, G. Delmaire, and G. Roussel, "How to apply random projections to nonnegative matrix factorization with missing entries?," in *Proc. EUSICPO'19*, 2019.

[7] M. Kapralov, V. Potluru, and D. Woodruff, "How to fake multiply by a gaussian matrix," in *Proc. ICML'16*, 2016, pp. 2101–2110.

[8] A. Saade, F. Caltagirone, I. Carron, L. Daudet, A. Drémeau, S. Gigan, and F. Krzakala, "Random projections through multiple optical scattering: Approximating kernels at the speed of light," in *Proc. ICASSP'16*, 2016, pp. 6215–6219.

[9] Y. X. Wang and Y. J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 6, pp. 1336–1353, June 2013.

[10] N. Gillis, "The why and how of nonnegative matrix factorization," in *Regularization, Optimization, Kernels, and Support Vector Machines*, pp. 257–291. Chapman and Hall/CRC, 2014.

[11] C. Liu, H.-C. Yang, J. Fan, L.-W. He, and Y.-M. Wang, "Distributed nonnegative matrix factorization for web-scale dyadic data analysis on MapReduce," in *Proc. WWW Conf.'10*, April 2010.

[12] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research*, vol. 11, no. Jan, pp. 19–60, 2010.

[13] N. Guan, D.Vin Tao, Z. Luo, and B. Yuan, "NeNMF: An optimal gradient method for nonnegative matrix factorization," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2882–2898, 2012.

[14] G. Zhou, A. Cichocki, and S. Xie, "Fast nonnegative matrix/tensor factorization based on low-rank approximation," *IEEE Trans. Signal Process.*, vol. 60, no. 6, pp. 2928–2940, June 2012.

[15] F. Wang and P. Li, "Efficient nonnegative matrix factorization with random projections," in *Proc. SIAM ICDM'10*. SIAM, 2010, pp. 281–292.

[16] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Contemporary mathematics*, vol. 26, no. 189-206, pp. 1, 1984.

[17] V. Sharan, K. S. Tai, P. Bailis, and G. Valiant, "Compressed factorization: Fast and accurate low-rank factorization of compressively-sensed data," in *Proc. ICML'19*, 2019, pp. 5690–5700.

[18] K. L. Clarkson and D. P. Woodruff, "Low rank approximation and regression in input sparsity time," in *Proc. ACM STOC'13*, New York, NY, USA, 2013, pp. 81–90, Association for Computing Machinery.

[19] H. Kim and H. Park, "Nonnegative matrix factorization based on alternating nonnegativity constrained least squares and active set method," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 2, pp. 713–730, 2008.

[20] C. Dorffer, M. Puigt, G. Delmaire, and G. Roussel, "Informed nonnegative matrix factorization methods for mobile sensor network calibration," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 4, pp. 667–682, Dec 2018.

[21] C. Dorffer, M. Puigt, G. Delmaire, and G. Roussel, "Nonlinear mobile sensor calibration using informed semi-nonnegative matrix factorization with a Vandermonde factor," in *Proc. SAM'16*, 2016.