Random Projection Streams for (Weighted) Nonnegative Matrix Factorization

Farouk Yahaya, Matthieu Puigt, Gilles Delmaire, Gilles Roussel

RANDOM PROJECTION STREAMS FOR (WEIGHTED) NONNEGATIVE MATRIX FACTORIZATION

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

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

ABSTRACT

Random projections recently became popular tools to process big data. When applied to Nonnegative Matrix Factorization (NMF), it was shown that, in practice, with the same compression level, structured random projections were more efficient than classical strategies based on, *e.g.*, Gaussian compression. However, as they are data-dependent, they remain costly and might not fully benefit from recent very fast random projection techniques. In this paper, we thus investigate an alternative framework to structured random projections—named *random projection streams* (RPS)—which *(i)* are based on classical random compression strategies only—and are thus data-independent—and *(ii)* can benefit from the above fast techniques. We experimentally show that, under some mild conditions, RPS allow the same NMF performance as structured random projection along iterations. We also show that even a CPU implementation of Gaussian Compression Streams allows a faster convergence than structured random projections when applied to weighted NMF.

***Index Terms*—** Random projections, Big data, Nonnegative Matrix Factorization, Weighted Nonnegative Matrix Factorization, Compressive learning

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 dimensional data while providing the benefit of mapping onto a much lower dimensional space, due to their computational intricacies and geometric properties. Among the numerous existing techniques, those based on randomized linear algebra [1, 2] were shown to be particularly efficient.

In particular, they were successfully combined with Nonnegative Matrix Factorization (NMF) [3–8]—and its weighted extension [9]—under the name of *compressive* or *compressed* NMF. More precisely, it was experimentally shown in [5] that *structured* random compression—based on Randomized Power Iteration (RPI)—was far more efficient than classical Gaussian Compression (GC) when applied to NMF, for a similar compression level. Indeed, RPIs enhance the low-rank structure of the data matrix, which is a major assumption behind NMF, while GC is data-independent. However, while several strategies have been proposed to speed-up GC—*e.g.*, CountGauss [10] or specific hardware [11]—RPIs still suffer from a high computational cost because of its data-dependent nature. This analysis remains true when considering a stable extension of RPIs named Randomized Subspace Iterations (RSIs) [1] that were combined with NMF in [7]. In this paper, we propose a new paradigm named Random Projection Streams (RPS) in which we assume the data-independent random projection matrices to be of infinite size and to be processed as streams where only a subset of the random projection matrices are processed[1]. In practice, RPS allow a similar performance as RPIs/RSIs under some mild conditions.

F. Yahaya gratefully acknowledges the Région Hauts-de-France to partly fund his Ph.D. fellowship. Experiments presented in this paper were carried out using the CALCULCO computing platform, supported by SCoSI/ULCO.

2. ANALYSIS OF COMPRESSED (WEIGHTED) NMF

2.1. Principles of Compressed NMF

NMF is a popular signal and image processing / machine learning tool which was successfully applied to many fields, *e.g.*, hyperspectral unmixing [13], or environmental data processing [14]. NMF 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 [15, 16]

$$X \simeq G \cdot F. \tag{1}$$

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.*,

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

$$\hat{F} = \arg\min_{F \geq 0} ||X - G \cdot F||_{\mathcal{F}}, \tag{3}$$

When X is large, several strategies have been proposed to speed-up the updates, *e.g.*, distributed [17] or online [12] computations, fast solvers [18], or randomized techniques [5].

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 (2) by resp. replacing X and F by X_R and F_R
    Define $G_L \triangleq L \cdot G$
    Solve (3) by resp. replacing X and G by X_L and G_L
  until a stopping criterion

Actually, several randomized strategies were proposed in the literature. In [3], 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 [4], the authors introduced the concept of *dual random compression* described in Algorithm 1. The key idea consists of noticing that compressing X by a projection on the left or

[1] RPS thus significantly differ from classical streaming data processing, *e.g.*, [12]. Indeed, the latter assumes to see a subset of the data matrix at each iteration—*i.e.*, the data to process evolve with time—while this not necessarily the case for the former.

the right side still allows to estimate the full matrix F or G, respectively. Compressing X on the left side (resp. right side) is done by left-multiplying it by a $(p+\nu)\times n$ matrix L (resp. right-multiplying it by a $m\times(p+\nu)$ matrix R), where ν is an oversampling parameter such that $p+\nu \ll \min\{n,m\}$. The difficulty then lies in designing efficient matrices L and R: the authors in [4] used scaled Gaussian realizations as tentative compression matrices, thus following the general proof of the Johnson-Lidenstrauss Lemma (JLL) [19] on which is built the theory of random projections. The authors in [5,6] 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 read

$$L \triangleq \text{QR}\left((XX^T)^q \cdot X \cdot \Omega_L\right)^T, \tag{4}$$

where $\Omega_L \in \mathbb{R}^{m\times(p+\nu)}$ is a scaled Gaussian random matrix—with ν set to a small value (*e.g.*, $\nu = 10$ in [5])—and q is a small integer (*e.g.*, $q = 4$ in [5]). L captures the range of the columns of X. Indeed, when $q = 0$, L is an orthogonal matrix obtained by a randomized QR decomposition of X. However, when X is a noisy low-rank data matrix, its singular values may slowly decay and computing $(XX^T)^q$ with $q > 0$ allows to significantly increase their decay, hence enforcing the low-rank structure of X. Combining NMF with RPIs was further extended in [7] where the authors used RSIs, *i.e.*, a round-off-error stable alternative to RPIs [1]. Lastly, the authors in [8] 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.

2.2. Analysis of Major Random Compression Techniques

At this stage, it should be noticed that computing random projections is costly. Indeed, deriving X_L in Algorithm 1 requires $m(p+\nu)(2n-1)$ operations. Several alternatives to GC have thus been proposed. For example, (Very) Sparse Random Projections—(V)SRPs—were proposed in [20] and [21], respectively. They replace the Gaussian matrix by a random matrix containing three possible values, *i.e.*, in the case when the compression matrix is L, each entry l_{ij} of L is defined as

$$l_{ij} = \sqrt{s}\cdot\begin{cases} 1 & \text{with prob. } 1/(2s),\\ 0 & \text{with prob. } (s-1)/s,\\ -1 & \text{with prob. } 1/(2s),\end{cases} \tag{5}$$

where s is set to $s = 1, 3$ [20] or $s \gg 3$ [21]. As most of the entries of X are multiplied by zero when $s \geq 3$ and as the product by $\sqrt{s}$ may be delayed, computing (V)SRP is much less expensive than GC while being asymptotically equivalent to GC when the dimension of the data is large [21]. However, to the best of our knowledge, these projections techniques were never applied to NMF.

Another popular randomized dimension reduction technique is named CountSketch [22]. Applied to design L in an NMF problem, it consists of generating a $(p+\nu)\times n$ matrix S with only one randomly-chosen nonzero entry per row, whose value is either $+1$ or -1 with equal probability. The product $S\cdot X$ provides a sketch of X which is inexpensive to compute. However, CountSketch requires more samples than GC to reach the same approximation accuracy. It was then combined with GC in [10]—under the name of CountGauss—to leverage 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 $\mu \geq \nu$ and $(p+\mu) \leq n$.

Another faster way to compute GC consists of using a dedicated hardware, *e.g.*, Optical Process Unit (OPU) [11]. OPUs optically perform random projections, so that they can process very large matrices in a very short time. Still, all these alternatives are data-independent techniques and provide a similar performance to GC. As a consequence, their use in NMF should be less accurate than using RPIs/RSIs which are—on the contrary to the above methods—data-dependent compression techniques.

When RPIs are used in Compressed NMF, computing L in Eq. (4) requires—using the Householder QR decomposition [23]—$(2q+1)nm(p+\nu) + 2n(p+\nu)^2 - 2/3(p+\nu)^3$ operations. In practice, the computation of $(XX^T)^q$ in Eq. (4)—and of $(X^TX)^q$ to derive R in RPIs—are done in a loop. RSIs follow the same procedure, except that they add intermediate QR decompositions. As a consequence, both randomized methods are equivalent in theory but RSIs are less sensitive to round-off errors [1] and are more computational demanding. In particular—assuming the use of the Householder QR decomposition—deriving L requires $(2q+1)nm(p+\nu)+2(p+\nu)^2\left((q+1)n+qm\right)-\frac{2}{3}(2q+1)(p+\nu)^3$ operations. It should be noticed that (V)SRPs, CountSketch, CountGauss, or an OPU only allow to speed-up the computation of $X\cdot\Omega_L$ and $\Omega_R\cdot X$. If one aims to use these alternatives to GC in the RPI/RSI computation, this has a limited impact on the global computational cost of the latter. This motivates the need to propose new paradigms for random projections.

2.3. Case of Random Projections Applied to Weighted NMF

Weighted NMF (WNMF) is an extension of NMF in which a specific confidence measure is associated to each entry of X. WNMF thus allows to process the case of missing entries in X. The weighted extension of Eq. (1) then reads

$$W \circ X \simeq W \circ (G\cdot F), \tag{6}$$

where W is the above matrix of weights and $\circ$ denotes the Hadamard product. As for NMF, WNMF aims to alternatingly solve both weighted extensions of Subproblems (2) and (3). WNMF is usually solved using one of the following strategies: *(i)* keeping W in the update rules [24], *(ii)* using a stochastic gradient descend if W is binary [25], or *(iii)* removing the Hadamard product in Eq. (6) within an Expectation-Maximization (EM) technique [26]. The latter assumes the entries of W to be between[2] 0 and 1. Noticing that the best estimate of the unknown entries of X are obtained from the product $G\cdot F$ and denoting $\mathbb{1}_{n,m}$ as the $n\times m$ matrix of ones, the E-step consists of computing

$$X^{\text{comp}} = W\circ X + (\mathbb{1}_{n,m} - W)\circ(G\cdot F). \tag{7}$$

Then, one may apply any standard NMF update rules to X^{comp} in order to derive G and F in the M-step. Once NMF converged to a given solution [26] or after a given number $\text{Max}_{\text{OutIter}}$ of iterations [27], X^{comp} is updated in another E-step using the last estimates of G and F in Eq. (7). In practice, the EM strategy was shown to be well-suited to fast NMF solvers such as Nesterov gradient [27]. In [9], we proposed a compressed extension of WNMF which is based on the same EM strategy [26] and which consists of applying Algorithm 1 at each M-step during $\text{Max}_{\text{OutIter}}$ iterations.

It should be noticed that the above computational analysis of Compressed NMF is slightly different in the weighted setting. Indeed, as X^{comp} is updated at each E-step, its compressed versions

[2] Such an assumption is not an issue, as it is possible to scale any non-null matrix W so that its maximum value is 1.

X_L^{comp} and X_R^{comp} must be regularly recomputed. Even worse, if RPIs or RSIs are applied, the matrices L and R must also be reestimated, which is particularly costly. This was shown to be the bottleneck of the proposed technique in [9]. However, by choosing a large-enough number $\text{Max}_{\text{OutIter}}$ of iterations in the M-step, randomized WNMF was shown to still outperform its uncompressed version.

3. RANDOM PROJECTION STREAMS

We now introduce our proposed RPS concept, that we firstly illustrate with GC, hence its name GC Stream (GCS). Let us first recall the JLL which states that [19] 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 $\forall u, v \in X$, $(1-\varepsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\varepsilon)\|u-v\|^2$. 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 [5], 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, at the price of a reduced computation speed-up.

Our proposed strategy thus reads as follows. We assume that ν is extremely large (or even infinite), so that L and R—which are drawn according to a scaled Gaussian distribution in GC—cannot fit in memory. We thus assume these matrices to be observed in a streaming fashion, *i.e.*, during an 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, along the NMF iterations, the updates of G and F are done using different compressed matrices $X_R^{(i)}$ and $X_L^{(i)}$, respectively. In practice, $L^{(i)}$ and $R^{(i)}$ are updated every ω iterations, where ω is the user-defined number of passes of the NMF algorithm using the same compression matrices in the streams. Using such a simple framework, one can use any of the data-independent random projection techniques described in Subsect. 2.2—*i.e.*, (V)SRPs, CountSketch, or CountGauss—to derive their Streaming extension, hence their respective names or (V)SRPS, CountSketchS, and CountGaussS.

RPS can also be applied to WNMF: in that case, we assume to observe new compression submatrices $L^{(i)}$ and $R^{(i)}$ every ω E-steps. This technique is also investigated in the next section.

Algorithm 2 Proposed compressed NMF strategy with RPS.

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** ω **do**
      Define $F_R^{(i)} \triangleq F \cdot R^{(i)}$ and $G_L^{(i)} \triangleq L^{(i)} \cdot G$
      Solve (2) by resp. replacing X and F by $X_R^{(i)}$ and $F_L^{(i)}$
      Solve (3) by resp. replacing X and G by $X_L^{(i)}$ and $G_L^{(i)}$
    end for
  until a stopping criterion

4. EXPERIMENTS

We empirically validate the enhancement provided by our proposed method for both NMF and WNMF. In both cases, we consider two different NMF solvers, *i.e.*, Active Set (AS-NMF) [28] and Nesterov gradient (NeNMF) [18]. Further, we consider two state-of-the-art compression strategies—*i.e.*, RSIs and GC—through which the NMF performance is assessed when compared with our proposed RPS and their vanilla (*i.e.*, uncompressed) versions. In both experimental settings, we consider 15 simulations where we draw random nonnegative matrices G^{theo} and F^{theo} such that $n = m = 10000$ and $p = 5$. As a consequence, their product X^{theo} is a 10000×10000 rank-5 matrix. The performance criterion used in this paper is a Relative Reconstruction Error (RRE) defined as

$$\text{RRE} \triangleq \left|\left|X^{\text{theo}} - G \cdot F\right|\right|_{\mathcal{F}}^2 / \left|\left|X^{\text{theo}}\right|\right|_{\mathcal{F}}^2. \quad (8)$$

In each simulation, we consider the same random initialization for each tested method. All the experiments are conducted using Matlab R2018b on a computer equipped with 2.5 GHz Intel Xeon E5-2620.

4.1. Standard NMF

We firstly investigate the performance achieved by our proposed GCS strategy when combined to NMF. Figure 1 provides the median performance reached by both AS-NMF and NeNMF when combined with GCS for different values of the parameters used in Algorithm 1, *i.e.*, $\nu_i = 10$, 50, 100, or 150, and $\omega = 1$, 2, 5, 10, or ∞ (in the last case, GCS reduces to GC). These plots show several interesting results. First of all, GC is not stable when combined with AS-NMF or NeNMF: the RRE is not always decreasing along iterations. This is particularly visible when $\nu_i = 10$ and 100. However, the global NMF performance reached with GCS after 100 iterations significantly decreases when ν_i increases. Such a result was expected as GC follows the proof of the JLL. Then, GCS always outperforms GC, even for high values of ν_i. When $\nu_i = 10$, the plotted RREs are not always decreasing along iterations, which means that the methods are not always stable. However, this effect is reduced (or cancelled) by increasing ν_i. Lastly, we can see that over all the considered values of ν_i, setting ω to 1 appears to be a good trade-off. A similar behavior—not shown for space consideration—was also found with the other data-independent random projection techniques considered in this paper.

Figure 2 shows the evolution of the median RREs with no compression, RSIs, and RPS—*i.e.*, GCS, CountSketchS, CountGaussS, SRPS with $s = 3$—when $\omega = 1$ and $\nu_i = 150$. The RPS techniques provide a similar or a better enhancement than the other strategies, which shows the relevance of the proposed approach.

However, it should be emphasized that in these experiments, the tested RPS implementations need more CPU time than RSIs—even if SRPS seems faster than the other techniques—because of the too high number of NMF iterations. Let us recall that such an issue might be solved by efficient implementations or a specific hardware dedicated to random projections [11].

4.2. Weighted NMF

We now investigate the enhancement provided by GCS in WNMF. We consider 15 simulations with random 10000×10000 rank-5 matrices X^{theo} that we randomly sample with a sampling rate varying from 10 to 90% with a step-size of 20%. As explained above, we consider two solvers, *i.e.*, AS-NMF and NeNMF that we eventually combine with RSIs, GC, or GCS and that we compare with their vanilla counterparts. However, as we showed above that GCS allowed a similar or better enhancement than RSIs along NMF iterations, we consider a slightly different experimental setting in this

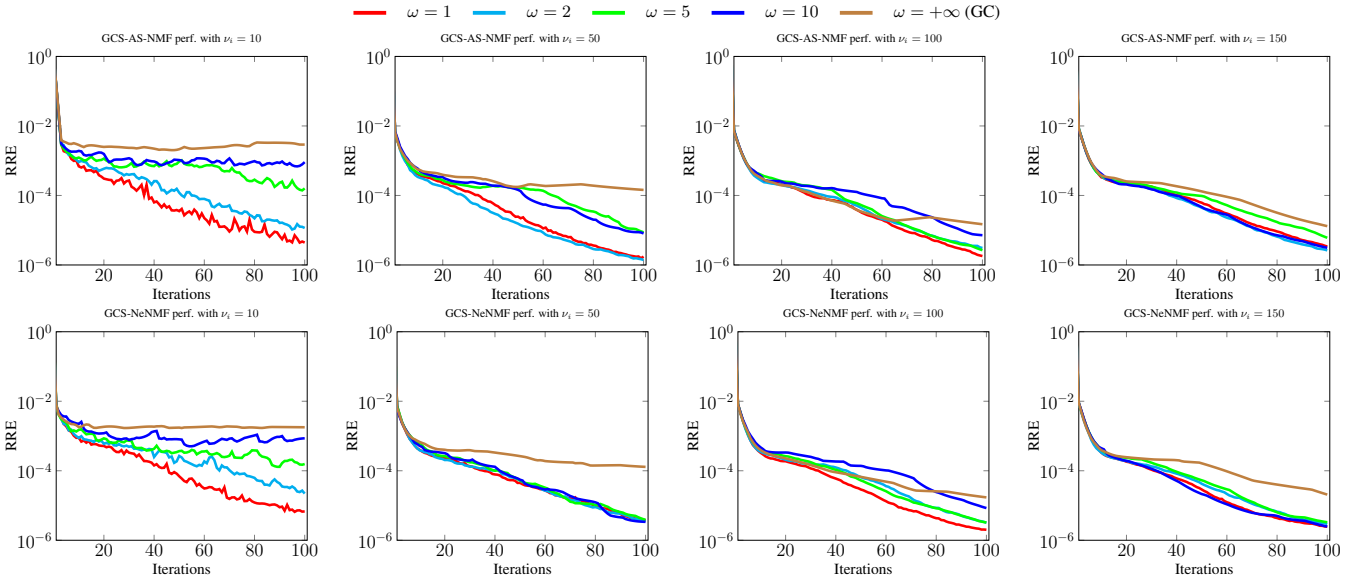

Fig. 1. NMF performance for different parameters of the GCS strategy.

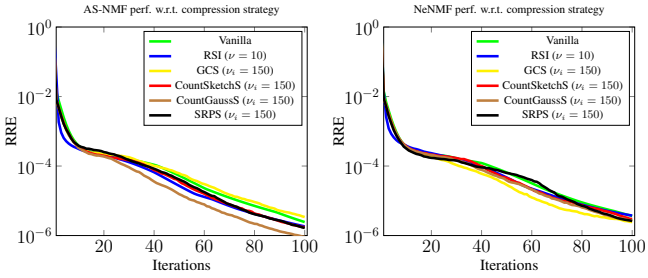

Fig. 2. NMF performance with respect to compression techniques.

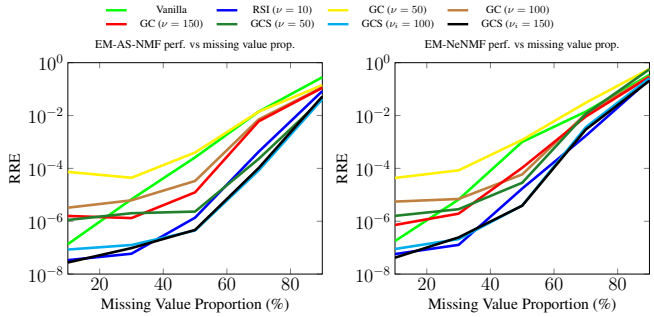

Fig. 3. WNMF performance vs the missing value proportion.

subsection. Indeed and as in [9], we let the methods run during 60 s and we only compute the final RREs obtained after this limit. This setting is harder than above as we also take into consideration the computational cost of each method. One key parameter to select in the approach in [9] is the number $\text{Max}_{\text{OutIter}}$ of iterations in the M-step. We found in [9] that this value should be set to 50 and we keep it in this paper. We also set the value of ω in GCS to $\omega = 1$. Lastly, we found in preliminary results that a high value of ν_i allowed a better enhancement than a moderate one. As a consequence and for the sake of readability on the plots, we only show the RREs reached when ν (for GC) or ν_i (for GCS) are equal to 50, 100, or 150.

Figure 3 shows the RREs obtained in these different conditions. First of all and as for standard NMF, *(i)* increasing ν for GC provides a better performance and *(ii)* GCS always outperforms GC. However, the behaviour of GCS is different from the previous results. When the proportion of missing values in X is high, we find in these experiments that the value of ν_i has a very limited influence on the WNMF performance. However, when this proportion is low, then a higher value of ν_i allows a better WNMF performance. In particular, when $\nu_i = 100$ or 150, the performance reached with GCS is quite similar to the one reached with RSI (and even slightly better when the missing value proportion is between 40% and 70%). Let us recall that such results are obtained while GCS is computed with $\omega = 1$, which is the most computational demanding scenario. Still, GCS allows a similar performance as RSIs within a limited amount of time. Moreover and as explained above, using another RPS method should speed-up WNMF.

5. CONCLUSION

In this paper, we proposed an alternative to structured random projections which is only based on data-independent random projections. Our strategy is built on the Johnson-Lidenstrauss Lemma and can be seen as a streamed random projection. RPS then allow a similar NMF or WNMF performance when compared to data-dependent RPIs/RSIs. Even if its computational cost may remain expensive on a CPU implementation—as compression matrices are updated each ω iterations—RPS should significantly benefit from new strategies to compute random projections—*e.g.*, from specific hardwares—while structured random projections techniques should not. This is what we aim to investigate in the future, for compressed NMF and other compressive learning methods. We will also extend these techniques to an informed NMF framework in order to apply them to, *e.g.*, air quality monitoring [29] or mobile sensor calibration [30, 31].

6. 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] 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.

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

[5] 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.

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

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

[8] 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.

[9] 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.

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

[11] 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.

[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] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 5, no. 2, pp. 354–379, 2012.

[14] M. Puigt, G. Delmaire, and G. Roussel, "Environmental signal processing: new trends and applications," in *Proc. ESANN'17*, 2017, pp. 205–214.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

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

[20] D. Achlioptas, "Database-friendly random projections," in *Proc. ACM SIGMOD-SIGACT-SIGART'01*. ACM, 2001, pp. 274–281.

[21] P. Li, Trevor J Hastie, and Kenneth W Church, "Very sparse random projections," in *Proc. ACM SIGKDD'06*, 2006, pp. 287–296.

[22] K. L. Clarkson and D. P. Woodruff, "Low-rank approximation and regression in input sparsity time," *Journal of the ACM*, vol. 63, no. 6, pp. 1–45, 2017.

[23] L. Vandenberghe, "Applied numerical computing – QR factorizations," Lectures notes available at `http://www.seas.ucla.edu/~vandenbe/ee133a.html`.

[24] N.-D. Ho, *Non negative matrix factorization algorithms and applications*, Phd thesis, Université Catholique de Louvain, 2008.

[25] H.-F. Yu, C.-J. Hsieh, S. Si, and I. Dhillon, "Scalable coordinate descent approaches to parallel matrix factorization for recommender systems," in *Proc. IEEE ICDM'12*. IEEE, 2012, pp. 765–774.

[26] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proc. SIAM ICDM'06*. SIAM, 2006, pp. 549–553.

[27] C. Dorffer, M. Puigt, G. Delmaire, and G. Roussel, "Fast nonnegative matrix factorization and completion using Nesterov iterations," in *Proc. LVA/ICA'17*, 2017, vol. LNCS 10179, pp. 26–35.

[28] 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.

[29] A. Limem, G. Delmaire, M. Puigt, G. Roussel, and D. Courcot, "Non-negative matrix factorization under equality constraints—a study of industrial source identification," *Applied Numerical Mathematics*, vol. 85, pp. 1–15, Nov. 2014.

[30] 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.

[31] O. Vu thanh, M. Puigt, F. Yahaya, G. Delmaire, and G. Roussel, "*In situ* calibration of cross-sensitive sensors in mobile sensor arrays using fast informed non-negative matrix factorization," in *Proc. ICASSP'21*, 2021, accepted.