# Adaptive Sketching for Fast and Convergent Canonical Polyadic Decomposition

**Kareem S. Aggour** [*] [1] **Alex Gittens** [*] [2] **Bülent Yener** [2]

## Abstract

This work considers the canonical polyadic decomposition (CPD) of tensors using proximally regularized sketched alternating least squares algorithms. First, it establishes a sublinear rate of convergence for proximally regularized sketched CPD algorithms under two natural conditions that are known to be satisfied by many popular forms of sketching. Second, it demonstrates that the iterative nature of CPD algorithms can be exploited algorithmically to choose more performant sketching rates. This is accomplished by introducing CPD-MWU, a proximally-regularized sketched alternating least squares algorithm that adaptively selects the sketching rate at each iteration. On both synthetic and real data we observe that for noisy tensors CPD-MWU produces decompositions of comparable accuracy to the standard CPD decomposition in less time, often half the time; for ill-conditioned tensors, given the same time budget, CPD-MWU produces decompositions with an order-of-magnitude lower relative error. For a representative real-world dataset CPD-MWU produces residual errors on average 20% lower than CPRAND-MIX and 44% lower than SPALS, two recent sketched CPD algorithms.

## 1. Introduction

Tensors of ever larger sizes appear with growing frequency in many applications including data mining (Papalexakis et al., 2017), signal processing (Cichocki et al., 2015), video analysis (Sobral et al., 2015), and more (Fanaee-T & Gama, 2016). Many of these applications use low-rank decompositions as a fundamental primitive in extracting latent factors from these tensorial datasets. A recent body of work from the machine learning, data mining, and applied mathematics communities has arisen that attempts to increase the scalability of tensor decomposition algorithms to match the scale of modern datasets.

There are multiple, inequivalent forms of low-rank tensor decompositions. In this work, we focus on the CANDECOMP/PARAFAC/canonical polyadic decomposition (CPD) (Bro, 1997), a generalization of the matrix singular value decomposition that uncovers a small set of latent factors describing each mode, or independent dimension, of the tensor. A CPD comprises a collection of latent factor matrices, one for each mode. Given the three-mode tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$, we denote its factor matrices by $\mathbf{A} \in \mathbb{R}^{I \times R}, \mathbf{B} \in \mathbb{R}^{J \times R}$, and $\mathbf{C} \in \mathbb{R}^{K \times R}$. The corresponding rank-$R$ CPD decomposition is (Kolda & Bader, 2009)

$$\boldsymbol{\mathcal{X}} = \sum_{i=1}^{R} \mathbf{a}_i \circ \mathbf{b}_i \circ \mathbf{c}_i + \boldsymbol{\mathcal{E}} := [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!] + \boldsymbol{\mathcal{E}}, \qquad (1)$$

where $\circ$ denotes the vector outer product, $\mathbf{a}_i, \mathbf{b}_i$ and $\mathbf{c}_i$ are columns of the factor matrices, and $\boldsymbol{\mathcal{E}}$ denotes the residual error. The factor matrices are obtained by minimizing the reconstruction error

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_F^2, \qquad (2)$$

where $\|\boldsymbol{\mathcal{X}}\|_F = \sqrt{\sum_{ijk} x_{ijk}^2}$ is the Frobenius norm of $\boldsymbol{\mathcal{X}}$. This is a difficult optimization problem both because $\boldsymbol{\mathcal{X}}$ is often large, and because the problem is non-convex. Attractive algorithms for CPD decomposition are computationally efficient and converge to critical points of the CPD objective.

The alternating least squares algorithm (ALS) is the standard workhorse algorithm for computing CP decompositions (Kolda & Bader, 2009). CPD-ALS uses an alternating minimization approach that updates one factor matrix to minimize the objective while fixing all the others. Each update requires the solution of a large least squares problem: if the tensor has size $n \times n \times n$, then the cost of each iteration is $O(n^3 R)$. One popular and empirically effective technique for accelerating CPD-ALS is the application of randomization: by using a randomly selected subset of the constraints of the linear systems at each iteration, i.e. by sketching the linear systems, the computational cost of the optimization can be greatly decreased. Both the choice of the sampling distribution and the fraction of the constraints

[*] Equal contribution [1] GE Global Research, Niskayuna, New York, USA [2] Rensselaer Polytechnic Institute, Troy, New York, USA. Correspondence to: Alex Gittens <gittea@rpi.edu>.

*Proceedings of the 37th International Conference on Machine Learning*, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

that are sampled, a.k.a. the sketching rate, are crucial to the empirical performance of sketching-based algorithms.

Several approaches to accelerate the decomposition of large tensors have used different forms of sketching (Battaglino et al., 2018; Cheng et al., 2016; Sidiropoulos et al., 2014; Tsourakakis, 2010) and recent research has demonstrated that regularization synergises with sketching to further accelerate CPD (Aggour et al., 2018) on large, dense tensors. This line of works demonstrates empirically that *if* crucial hyperparameters such as the sketching rate are chosen appropriately, then accurate decompositions are obtained efficiently. Despite the promising empirical results, little is guaranteed about the performance of these algorithms. Do they converge? Does the error decrease between iterations? The answers to these questions are largely unknown. Also their performance is sensitive to the choice of these parameters and the optimal choices vary significantly between tensors (Aggour et al., 2018).

In this work, we focus on sketched CPD-ALS algorithms in particular, for which no prior works address the important question of hyperparameter selection, and no prior works establish that these algorithms converge or even that the approximation error decreases. This work addresses both of these questions; the main contributions are summarized as follows.

1. We provide the most complete theory available in the sketched CPD-ALS literature. Theorem 1 states intuitive conditions under which the objective is guaranteed to decrease between iterations of sketched ALS, and is leveraged in Theorem 2 to provide the first guarantee on the convergence of sketched CPD-ALS. It quantifies an analog of the phenomenon that one sees with SGD—that one must increase the accuracy of the gradient estimate in order to guarantee convergence—and identifies exactly what should be preserved by sketching to ensure convergence.

2. Theorem 2 implies that convergence is guaranteed if the sketching rates are large enough, but finding sketching rates which satisfy these conditions requires expensive computations. We introduce a novel heuristic, CPD-MWU, which dynamically adjusts the sketching rate to ensure convergence with much less computation. CPD-MWU also greatly ameliorates the problem of hyperparameter selection: instead of requiring the user to select one sketching rate that is appropriate over all iterations of ALS, they can provide a set of hyperparameters varying over multiple orders of magnitude, and CPD-MWU will adapt to use an appropriate sketching rate over the course of the optimization.

3. We experimentally show, using real and synthetic data sets, that CPD-MWU has superior runtime-vs-accuracy tradeoffs to prior sketched CPD-ALS algorithms even when the sketching rates of the latter are selected using *a priori* knowledge of the best fixed sketching rate.

We review the proximally regularized sketched CPD-ALS algorithm in § 2, and present our results on the convergence of this algorithm; proofs are provided in the supplementary material. The CPD-MWU algorithm is presented in § 3. In § 4 we discuss the related works. In § 5, we empirically evaluate CPD-MWU on synthetic and real datasets. We discuss and conclude our work in § 6.

### 1.1. Notation

We adopt the notation of (Kolda & Bader, 2009), in which cursive, bold capital letters (e..g, $\boldsymbol{\mathcal{X}}$) denote tensors, roman bold capital letters (e.g., $\mathbf{A}$) denote matrices, and capital letters with a subscript (e.g., $\mathbf{X}_{(i)}$) denote an unfolding or matricization of a tensor in that mode (Kolda & Bader, 2009). E.g., $\mathbf{X}_{(1)} \in \mathbb{R}^{I \times JK}$ represents the unfolding of $\boldsymbol{\mathcal{X}}$ in the first mode. The Khatri-Rao product is represented by the symbol $\odot$ (Kolda & Bader, 2009). Let $\mathbf{P_A}$ denote the orthogonal projector onto the column span of the matrix $\mathbf{A}$. For notational convenience we work with a third-order tensor $\mathbf{X} \in \mathbb{R}^{I \times J \times K}$, but the results generalize in a straightforward manner to higher-order tensors. Throughout we let $F$ denote the squared reconstruction error, $F(\mathbf{A}, \mathbf{B}, \mathbf{C}) = \|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_F^2$.

## 2. Proximally Regularized Sketched CPD-ALS

Alternating least squares (ALS) approaches are the most widely-used class of algorithms for obtaining CPDs, due to their simplicity and performance in practice (Bro, 1997; Kolda & Bader, 2009): these methods are so named because when two factor matrices are fixed, the third can be updated to decrease $F$ by solving a linear system. The canonical CPD-ALS algorithm repeated solves the following series of linear systems until convergence:

$$\begin{aligned}
\mathbf{A}_{t+1} &= \operatorname{argmin}_{\mathbf{A}} F(\mathbf{A}, \mathbf{B}_t, \mathbf{C}_t) \\
&= \operatorname{argmin}_{\mathbf{A}} \|\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C}_t \odot \mathbf{B}_t)^T\|_F \\
\mathbf{B}_{t+1} &= \operatorname{argmin}_{\mathbf{B}} F(\mathbf{A}_{t+1}, \mathbf{B}, \mathbf{C}_t) \\
&= \operatorname{argmin}_{\mathbf{B}} \|\mathbf{X}_{(2)} - \mathbf{B}(\mathbf{C}_t \odot \mathbf{A}_{t+1})^T\|_F \\
\mathbf{C}_{t+1} &= \operatorname{argmin}_{\mathbf{C}} F(\mathbf{A}_{t+1}, \mathbf{B}_{t+1}, \mathbf{C}) \\
&= \operatorname{argmin}_{\mathbf{C}} \|\mathbf{X}_{(3)} - \mathbf{C}(\mathbf{B}_{t+1} \odot \mathbf{A}_{t+1})^T\|_F.
\end{aligned} \tag{3}$$

This algorithm is simple and has the property that $F$ is non-increasing throughout, but has the drawback that each iteration requires solving large, potentially ill-conditioned linear systems.

Ill-conditioning can be mitigated by adding proximal regularization to each inner solve in (3); prior work has shown that this modification does not change the local optima of the optimization process (Li et al., 2013), and that proximal regularization is particularly effective at accelerating convergence when decomposing ill-conditioned tensors (Aggour et al., 2018).

The cost of the inner solves can be reduced by observing that the CPD systems (3) are often overconstrained: the matricizations of $\boldsymbol{\mathcal{X}}$ often have many more columns than rows. For example, $\mathbf{X}_{(1)} \in \mathbb{R}^{I \times JK}$, while $\mathbf{A} \in \mathbb{R}^{I \times R}$, so each row of $\mathbf{A}$ has only $R$ degrees of freedom but is constrained by $JK \gg R$ equations. Because of this fact, one can obtain approximate solutions by updating the factor matrices using smaller systems comprising subsets of the original constraints. We model this sampling procedure by multiplication from the right with a random *sketching matrix* $\mathbf{S}_{t+1}$ that selects $s_{t+1}JK$ columns randomly without replacement then scales them by $\sqrt{s_{t+1}}$. The scalar $s_{t+1}$ is in $(0, 1]$ and is called the *sketching rate*.

Combined with proximal regularization, we obtain the regularized sketched ALS algorithm proposed in (Aggour et al., 2018), which takes $\mathbf{A}_{t+1}$ to be the minimizer of

$$\| \left(\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C}_t \odot \mathbf{B}_t)^T\right) \mathbf{S}_{t+1}\|_F^2 + \lambda \|\mathbf{A} - \mathbf{A}_t\|_F^2 \quad (4)$$

and updates $\mathbf{B}$ and $\mathbf{C}$ analogously. The distribution of the random sketching matrix $\mathbf{S}_{t+1}$ may vary; depending on the application, users may employ standard choices such as the uniform column sampling scheme described previously, sampling according to an importance distribution over the columns of $(\mathbf{C}_t \odot \mathbf{B}_t)^T$, or CountSketch sampling (Wang et al., 2017). The sketching rates $s_t$ may change between iterations of regularized sketched CPD-ALS; in this paper we consider $\lambda$ to be fixed.

### 2.1. Convergence of Proximally Regularized Sketched CPD-ALS

Recent works that introduce sketched CPD-ALS algorithms provide per-iteration guarantees on the quality of the factor matrices that state, with high probability,

$$F(\mathbf{A}_{t+1}, \mathbf{B}_t, \mathbf{C}_t) \leq (1 + \varepsilon) \min_{\mathbf{A}} F(\mathbf{A}, \mathbf{B}_t, \mathbf{C}_t),$$

when the sketching rate is sufficiently high (Cheng et al., 2016; Battaglino et al., 2018). Analogous guarantees hold for the other two factor matrices.

Such guarantees are non-informative in the context of CPD-ALS algorithms, because it is not clear that convergence will occur even if guarantees of this form hold for $\varepsilon$ very small. Even if the goal is relaxed to simply ensuring that the objective decreases at each iteration, these guarantees are weak: if $\min_{\mathbf{A}} F(\mathbf{A}, \mathbf{B}_t, \mathbf{C}_t)$ is not much smaller than $F(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t)$, then the factor matrix $\mathbf{A}_{t+1}$ obtained using sketching can in fact increase $F$. Unfortunately, this can occur even when the factor matrices are far from converging to a local minimum of $F$.

Our first result guarantees that, in fact, sketched CPD-ALS *will* achieve a decrease in the objective with high probability when the sketching rate is sufficiently high.

**Theorem 1 (Sufficient Decrease)** *Fix a failure probability $\delta \in (0, 1)$ and a precision $\varepsilon_{t+1} \in (0, 1)$, a regularization parameter $\lambda = O(\sigma_{\mathit{min}}(\mathbf{C}_t \odot \mathbf{B}_t))$, and let $\boldsymbol{\mathcal{R}} = \boldsymbol{\mathcal{X}} - [\![\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]\!]$ be the residual tensor after step $t$. If $\mathbf{S}_{t+1}$ samples columns uniformly at random with or without replacement so that $s_{t+1}IJK = \Omega\left(\frac{\mu R}{\nu \varepsilon^2 \delta} \log \frac{R}{2\delta}\right)$, then $\mathbf{A}_{t+1}$, the solution to the sketched ridge regression problem (4), satisfies*

$$\begin{aligned} F(\mathbf{A}_{t+1}, \mathbf{B}_t, \mathbf{C}_t) &\leq F(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t) \\ &\quad - (1 - \varepsilon_{t+1})\|\mathbf{C}_t \odot \mathbf{B}_t\|_2^{-2} \|\nabla_{\mathbf{A}} F(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t)\|_F^2 \end{aligned} \quad (5)$$

*with probability at least $1-\delta$ with respect to the randomness in the sketching matrix $\mathbf{S}_{t+1}$. Here, $\mu$ is the row coherence of $\mathbf{C}_t \odot \mathbf{B}_t$ and $\nu \in (0, 1]$ is the relative squared Frobenius norm of the projection of the residual tensor onto the span of $\mathbf{C}_t \odot \mathbf{B}_t$ (see the supplementary material for precise definitions of $\mu$ and $\nu$).*

Analogous results hold for the updates of the $\mathbf{B}$ and $\mathbf{C}$ factor matrices. Conditions under which sufficient decrease is guaranteed for several other common forms of sketching are given in the supplementary material. Sufficient decrease is guaranteed even if no proximal regularization is used, i.e. $\lambda = 0$. Thus Theorem 1 implies that prior sketched ALS algorithms do decrease the objective function when the sketching rates are sufficiently high.

The second goal in the analysis of CPD-ALS algorithms is to guarantee convergence to an approximate critical point of the objective (2). To facilitate this, we make the following assumptions. The first is standard. The second simply requires that the sketching rates are selected to ensure that the amount of decrease is bounded below at each iteration. This assumption holds when the sketching rate is selected to satisfy the sample complexity given by Theorem 1.

**Assumption 1 (Bounded Iterates)** *The gradients of $F$ with respect to (w.r.t.) $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are $L$-Lipschitz along the solution path, e.g.*

$$\begin{aligned} \|\nabla_{\mathbf{A}} F(\mathbf{A}_{t+1}, \mathbf{B}_t, \mathbf{C}_t) - \nabla_{\mathbf{A}} F(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t)\|_F \\ \leq L\|\mathbf{A}_{t+1} - \mathbf{A}_t\|_F, \end{aligned}$$

*and similarly for the gradients w.r.t. $\mathbf{B}$ and $\mathbf{C}$.*

**Assumption 2 (Sufficient Decrease)** *For each value of $t$ from $1$ to $T$, the sketching and regularization parameters*

*$s_{t+1}$ and $\lambda$ are selected so that sufficient decrease (5) is achieved during the updates from $(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t)$ to $(\mathbf{A}_{t+1}, \mathbf{B}_{t+1}, \mathbf{C}_{t+1})$ with probability at least $1-\delta$. Also, the constants of sufficient decrease,*

$$c_{t+1} = (1-\varepsilon_{t+1}) \min \Big\{ \|\mathbf{C}_t \odot \mathbf{B}_t\|_2^{-2}, \\ \|\mathbf{C}_t \odot \mathbf{A}_{t+1}\|_2^{-2}, \|\mathbf{B}_{t+1} \odot \mathbf{A}_{t+1}\|_2^{-2} \Big\}$$

*in (5) satisfy $\frac{1}{\lambda} + \max_{t=1}^{T} \frac{1}{c_t} \leq R$.*

Standard results imply that proximal gradient algorithms converge to approximate critical points of non-convex functions at a sublinear rate because of the sufficient decrease condition (Beck, 2017). Theorem 1 guarantees sufficient decrease, so Assumption 2 is satisfiable. By adapting the standard arguments slightly to account for the fact that CPD-ALS is a Gauss-Seidel algorithm instead of a gradient method, and using the two above assumptions, we obtain that sketched CPD-ALS converges with at least the same sublinear rate.

**Theorem 2** *If Assumptions 1 and 2 are satisfied, then after $T$ iterations, proximally regularized sketched CPD-ALS reaches an $O(T^{-1/2})$-approximate critical point with probability at least $(1-\delta)^T$:*

$$\min_{1 \leq t \leq T} \|\nabla F(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t)\|_F \leq \sqrt{\frac{C' F(\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_0)}{T}},$$

*where $C'$ is at most $12RL^2$.*

Theorem 2 is quite general, as it applies to *all* forms of sketching as long as the sketching and regularization parameters selected at each iteration satisfy the sufficient decrease condition. In the analysis of even deterministic tensor decomposition algorithms, it is common practice to assume a-priori that the sequence of factor matrices is bounded (Xu & Yin, 2013); we do the same, and because $F$ is a smooth function, this implies that its gradients are indeed Lipschitz along the solution path.

These two results ensure that *there exist* choices of sketching rates $s_t$ such that sketched regularized CPD-ALS will converge sublinearly, but in practice estimating the relevant properties to determine appropriate $s_{t+1}$ at each iterate is costly, and the theory provides pessimistic estimates. It is preferable to choose the sketching parameters at each iteration in a way that adapts to the given tensor. The remaining portion of this work addresses this problem.

**Algorithm 1** CPD-MULTIPLICATIVE WEIGHTS UPDATE

```
Inputs: $\boldsymbol{\mathcal{X}}$, sketching rates $\{s_i\}_{i=1}^N$, regularization $\lambda$, update probability $\varepsilon$, momentum $\eta$
Outputs: $\mathbf{A}, \mathbf{B}, \mathbf{C}$
$w_{i,0} \leftarrow 1, \quad i = 1, \ldots, N$
Random initialization of $\mathbf{A}_0, \mathbf{B}_0, \mathbf{C}_0$
for $t \leftarrow 0, \ldots, \infty$ do
    $s \leftarrow s_i$ with probability proportional to $w_{i,t}$
    $\mathbf{A}_{t+1} \leftarrow \mathrm{RS\_LS}(\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t, \mathbf{X}_{(1)}, \lambda, s)$   ▷ solve the ridge regression problem (4)
    $\mathbf{B}_{t+1} \leftarrow \mathrm{RS\_LS}(\mathbf{B}_t, \mathbf{A}_{t+1}, \mathbf{C}_t, \mathbf{X}_{(2)}, \lambda, s)$
    $\mathbf{C}_{t+1} \leftarrow \mathrm{RS\_LS}(\mathbf{C}_t, \mathbf{B}_{t+1}, \mathbf{A}_{t+1}, \mathbf{X}_{(3)}, \lambda, s)$
    if Bern$(\varepsilon) = 1$ then
        $w_{i,t+1} \leftarrow$ computed as in (7), $i = 1, \ldots, N$
    else
        $w_{i,t+1} \leftarrow w_{i,t}, \quad i = 1, \ldots, N$
    end if
    if convergence criterion is satisfied then
        $\mathbf{A} \leftarrow \mathbf{A}_{t+1}, \mathbf{B} \leftarrow \mathbf{B}_{t+1}, \mathbf{C} \leftarrow \mathbf{C}_{t+1}$
    end if
end for
```

# 3. Adaptive Sketching Rate Selection

As suggested by Theorem 1, the performance of sketched ALS algorithms is sensitive to the sketching rate $s_{t+1}$: overly aggressive sketching can lead to poor convergence due to under-sampling of the tensor, while overly conservative sketching can increase the runtime. Further, the range of sketching rates that guarantee decrease in the CPD objective are determined by the type of sketching and the properties of the linear system at each iteration, so may change between iterations. These properties, e.g. coherence or conditioning, are expensive to compute at each iteration.

## 3.1. Sketching Rate Selection via Multiplicative Weights Updates

To address the challenge of inexpensively selecting sketching rates appropriately at each iteration, observe that the aim of sketching CPD-ALS is to maximize the rate of decrease in the objective while minimizing the computational cost of each iteration. Accordingly, we define the loss of a sketching rate $s$ at iteration $t$ to be

$$\ell_t(s) = \frac{\|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}_{t+1}, \mathbf{B}_{t+1}, \mathbf{C}_{t+1}]\!]\|_F - \|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}_t, \mathbf{B}_t, \mathbf{C}_t]\!]\|_F}{\mathrm{runtime}(t) \|\boldsymbol{\mathcal{X}}\|_F}, \tag{6}$$

where runtime$(t)$ is the time in seconds that it takes to finish the iteration, and the factor matrices at $t+1$ are computed using regularized sketched ALS with sketching rate $s$. If progress was made towards convergence and the runtime was small, $\ell_t(s)$ is a large negative number, indicating that $s$ performed well during this iteration. Intuitively, knowing which sketching rates performed well during earlier iterations gives us knowledge of which sketching rates we can expect to perform well on the next iteration.

In this vein, the CPD-MWU algorithm presented in Listing 1 employs a label efficient forecasting scheme (Cesa-Bianchi & Gábor, 2006) to dynamically select an appropriate sketching rate at each iteration.

The algorithm takes as input a set of $N$ possible values for the sketching rate: this collection is assumed to be sufficiently diverse to contain well-performing sketching rates at each iteration of ALS. A weight $w_{i,t}$ is associated with each sketching rate $s_i$, initialized to one, and is updated as the algorithm progresses to track the performance of that rate as determined by $\ell_t(s_i)$. At each iteration, the sketching rate used to update the factor parameters is sampled from the $N$ choices with probability proportional to its weight.

It would be inefficient to update the weights of all sketching rates at each iteration, as evaluating their losses requires $N$ iterations of regularized sketched ALS. Hence we infrequently update the weights using the label efficient updating scheme of (Cesa-Bianchi & Gábor, 2006): at each iteration, with probability $\varepsilon$ all weights are multiplicatively updated,

$$w_{i,t+1} = w_{i,t} \exp\left(-\frac{\eta \ell_t(s_i)}{\varepsilon}\right) \quad \text{for } i = 1, \dots, N, \quad (7)$$

and with probability $1 - \varepsilon$ the weights are unchanged, $w_{i,t+1} = w_{i,t}$ for all $i$. Here $0 < \varepsilon < 1$ determines the update frequency, and $\eta > 0$ determines the aggressiveness of the weight updates.

The computational cost of CPD-MWU using $N$ sketching rates, amortized over $N$ iterations, is $\mathcal{O}((\hat{s} + \varepsilon N \overline{s})IJKR)$, where $\hat{s}$ is the average of the $s_i$ selected during those $N$ iterations and $\overline{s}$ is the mean of the $s_i$. In practice we choose $\varepsilon < \frac{1}{N}$ so the cost is $\mathcal{O}((\hat{s} + \overline{s})IJKR)$. The complexity of a single iteration of traditional or regularized CPD-ALS is $\mathcal{O}(IJKR)$, so the CPD-MWU algorithm is significantly more computationally efficient per iteration when the average of the sketching rates is small *and* $\hat{s}$ is also small. The factor $\hat{s}$ implicitly captures the unavoidable fact that the ranges of sketching rates that deliver good performance changes over time: in particular, we can expect that near convergence, $\hat{s}$ approaches unity.

The experimental evaluations conducted in § 5 show that CPD-MWU provides a desirable accuracy-time trade-off in practice. The supplementary material provides a regret bound guarantee for a modified, more expensive, version of CPD-MWU.

## 4. Related Work

Early work on fast randomized tensor decomposition focused on entry-wise sparsification (Tsourakakis, 2010; Nguyen et al., 2015), then several groups investigated sketched ALS algorithms (Bhojanapalli & Sanghavi, 2015; Reynolds et al., 2016; Wang et al., 2015; Yu et al., 2015; Vervliet & De Lathauwer, 2016; Song et al., 2016; Cheng et al., 2016). More recently, two groups within the data mining community (Gujral et al., 2018; Yang et al., 2018) refined the earlier ParCube system (Papalexakis et al., 2012) that uses a block-sampling approach to enhance the scalability of CP decomposition, and (Battaglino et al., 2018) proposed two sketching approaches for ALS. Most of these works do not provide guarantees on the convergence to critical points of the CPD objective. One exception, (Wang et al., 2015), provides strong convergence guarantees for a sketched tensor power method, which (Cheng et al., 2016) argues is less efficient than sketched ALS.

Of these works, the most closely related to our approach are (Cheng et al., 2016; Battaglino et al., 2018; Aggour et al., 2018). In (Cheng et al., 2016), Cheng et al. introduce the SPALS algorithm, which accelerates ALS by sampling rows of the Khatri-Rao product with probability proportional to their statistical leverage scores (Cheng et al., 2016). In (Battaglino et al., 2018), Battaglino et al. propose two sketching approaches for CPD-ALS, CPRAND and CPRAND-MIX. CPRAND samples rows of the Khatri-Rao product uniformly at random; the CPRAND-MIX algorithm first mixes the modes of an input tensor to make the tensor incoherent, before applying CPRAND. In (Aggour et al., 2018), Aggour et al. demonstrated that regularization works with sketching to further accelerate the convergence of ALS.

Although these prior works applied sketching to interpolate between accuracy and efficiency at each step of ALS, they propose using a fixed sketching rate throughout the optimization process (Wang et al., 2015; Cheng et al., 2016; Battaglino et al., 2018; Song et al., 2018; Aggour et al., 2018). To our knowledge, CPD-MWU is the first algorithm that adaptively chooses the sketching rates to increase the speed of convergence of the ALS procedure.

## 5. Experimental Evaluation

Experiments were conducted on both synthetic and real, small and large datasets to illustrate that: (1) adaptive selection of the sketching rate improves performance over prior sketched ALS algorithms by decreasing the runtime and/or reducing the final residual error, and (2) the decompositions obtained perform on par with those obtained via conventional ALS when used in downstream data mining tasks.

Large-scale synthetic tensors were generated to evaluate the impact of adaptive sketching rate selection on CPD runtime and residual error, and the impact of the quantity of the sketching rates. To demonstrate performance on real datasets, a moderately-sized video was decomposed for background subtraction, and a small knowledge base was decomposed. The large synthetic and moderately-sized video datasets were decomposed in a distributed setting, while the small knowledge base was decomposed in shared memory.

For the distributed evaluations, each algorithm was implemented in Python using Apache Spark (Zaharia et al., 2010) to ensure consistency across the comparisons, as Spark is the de facto standard for implementing Big Data analytics on commodity hardware. For the small dataset, each algorithm was implemented in Python to also ensure consistency of comparison.

Synthetic tensors were generated to gauge the influence of noise and conditioning of the factor matrices on the performance of CPD-MWU. We used the methodology described in (Tomasi & Bro, 2006; Acar et al., 2011) to synthesize tensors with hetero- and/or homoskedastic noise. We further synthesized ill-conditioned tensors (tensors with collinear factor matrices) following the methodology described in (Kiers et al., 1999). Each synthetic tensor is rank-5 and in $\mathbb{R}^{366\times 366\times 100{,}000}$, split evenly in the $3^{rd}$ mode into 20K slices for distributed processing in Spark.

### 5.1. Setup

We compare CPD-MWU to a traditional CPD-ALS implementation (denoted by 'CPD-ALS') (Aggour & Yener, 2016) and a regularized, sketched CPD-ALS using a manually tuned static sketching rate (denoted by 'Sketched CPD') (Aggour et al., 2018). We also compare to a heuristic that selects from the same $N$ sketching rates as CPD-MWU, but randomly; this proved much less effective than CPD-MWU (the results are available in the supplementary material). Uniform random sampling was used to select rows of the Khatri-Rao product for both Sketched CPD and CPD-MWU. Row norm-weighted sampling was also investigated, but proved less effective than uniform random sampling. We quantify the performance of CPD-MWU in terms of both runtime and final relative residual error, defined as the normalized Frobenius norm error of the factorization,

$$\epsilon = \frac{\|\boldsymbol{\mathcal{X}} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_F}{\|\boldsymbol{\mathcal{X}}\|_F}$$

Five sketching rates were used for the CPD-MWU experiments, with four rates linearly spaced in the interval $[10^{-6}, 10^{-4}]$. The fifth sketching rate was set to 1 so that CPD-MWU could use the full tensor if it determined that to be advantageous. We observed that the performance is robust to the choice of $\eta$ and used the value of 2 for our experiments; we set $\varepsilon$ to 0.15, which is smaller than $\frac{1}{N}$, to amortize the costs of the solves as described in the discussion following Listing 1.

For each synthetic experiment we generated 10 distinct tensors with the relevant properties and evaluated each 3 times with different initial conditions (using the same initial conditions for the same run of each algorithm). Results are averaged across the 30 runs for each tensor type. Each of the real datasets was decomposed 10 times using different initial conditions in each run.

### 5.2. Results

#### 5.2.1. Improved Runtime

Although CPD-MWU as set forth in Algorithm 1 uses proximal regularization, previous research has shown that Tikhonov regularization is effective for accelerating the decomposition of noisy tensors when used with sketching (Aggour et al., 2018). Hence we also investigated the performance of CPD-MWU when Tikhonov regularization is substituted for proximal regularization, comparing to the performance of standard ALS and Sketched CPD (with $\lambda$=0.001 and sketching rate $s = 10^{-4}$). These regularization and sketching parameters were chosen for Sketched CPD by conducting an expensive grid search to identify a pairing that gives the fastest runtime and lowest residual error. Table 1 shows the means and standard deviations of the residual error and runtimes across 30 noisy tensor experiments.

*Table 1.* Average and standard deviation of runtime (in seconds) and residual error $\epsilon$ across 30 noisy tensor decompositions per data point.

| | $\epsilon$ | Std($\epsilon$) | Time | Std(Time) |
|---|---|---|---|---|
| CPD-ALS | 0.3817 | 1.79*$10^{-3}$ | 526.94 | 133.05 |
| Sketched CPD | 0.3808 | 5.96*$10^{-7}$ | 214.51 | 23.44 |
| CPD-MWU | 0.3808 | 8.59*$10^{-7}$ | **185.67** | 34.42 |

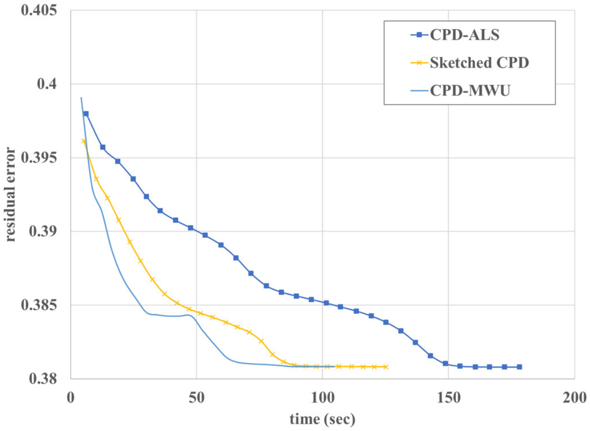

*Figure 1.* Residual error $\epsilon$ over time when decomposing a noisy tensor. CPD-MWU converges fastest to final residual error compared to the other algorithms.

While there is no significant difference in the final residual error when decomposing noisy tensors, there are significant differences in the runtimes. CPD-MWU with adaptive sketching is the fastest algorithm, followed by CPD with hand-tuned static sketching. Thus, adaptively selecting the sketching rate in CPD produces faster decompositions than a manually optimized static sketching rate, which are both considerably faster than traditional CPD-ALS for noisy tensors. Figure 1 shows the residual error over time for a representative noisy tensor decomposition.

### 5.2.2. Improved Relative Residual Error

For decomposing ill-conditioned tensors, we use proximal regularization ($\lambda$=0.001) for CPD-MWU and Sketched CPD (Aggour et al., 2018). The Sketched CPD static sketching rate selection ($s = 10^{-4}$) again required hand-tuning. Table 2 shows the means and standard deviations of the residual error across the 30 ill-conditioned tensor experiments. On average CPD-MWU produces the lowest error given the same time allotment. CPD-MWU produces a 10.3x lower final residual error compared to CPD-ALS, over an order of magnitude improvement.

*Table 2.* Average residual error $\epsilon$ across 30 ill-conditioned tensor decompositions per data point, in which the runtime was fixed to 10 minutes.

| | $\epsilon$ | Std($\epsilon$) |
|---|---|---|
| CPD-ALS | 0.0309 | 4.70*$10^{-3}$ |
| Sketched CPD | 0.0149 | 1.33*$10^{-2}$ |
| CPD-MWU | **0.0030** | 7.87*$10^{-4}$ |

Figure 2 shows the residual error over time for a representative ill-conditioned tensor decomposition.

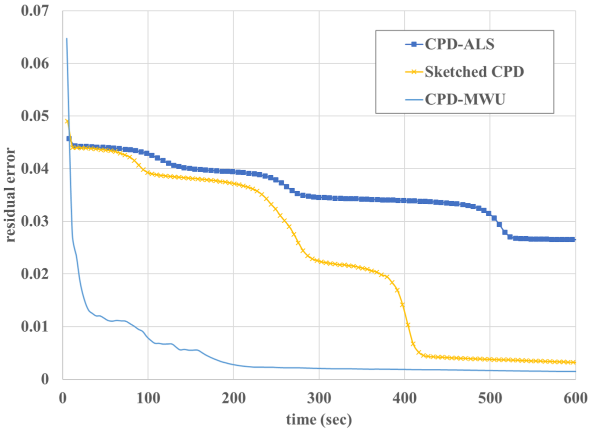

*Figure 2.* Residual error $\epsilon$ over time when decomposing an extremely ill-conditioned tensor (factor matrices each have column collinearity of 0.9). CPD-MWU converges to a significantly lower residual error, significantly faster, as compared to traditional CPD-ALS.

### 5.2.3. Impact of Quantity of Sketching Rates

We next explored the impact of the quantity of sketching rates $N$ on the overall performance. Every sketching rate must be evaluated when an update is performed (recall that the update frequency is determined by $\varepsilon$). Thus, given a fixed time allocation, the larger $N$ the more time the algorithm must spend updating the weights for the rates and the less time it can spend making progress on the decomposition.

Figure 3 shows traditional CPD-ALS for an ill-conditioned tensor compared to CPD-MWU with increasing values of $N \in [5, 1000]$. As expected, CPD-MWU performs best with fewer sketching rates to draw from. $N$ = 5 and 10 perform similarly, and significantly outperform the same algorithm with more sketching rates.

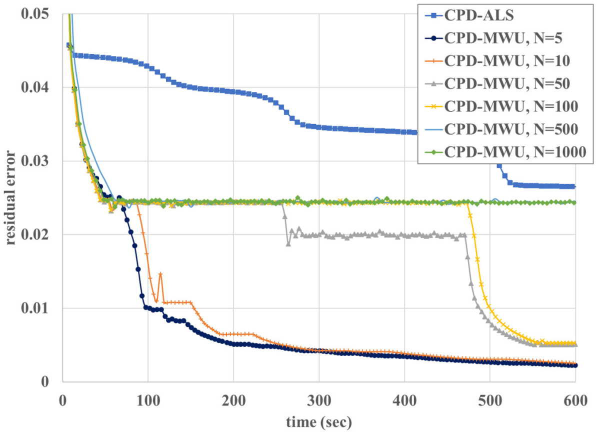

*Figure 3.* Residual error $\epsilon$ over time when decomposing an ill-conditioned tensor using CPD-MWU with increasing numbers of sketching rates to select from $N$=[5,10,50,100,500,1000].

### 5.2.4. Downstream Application - Video Analysis

Tensor decomposition is used in video analysis for, among other tasks, background subtraction (Sobral et al., 2015). Thus, as an illustrative example, we decomposed a black-and-white video of a person sitting on a park bench overlooking a city (Unknown, 2018). The tensor is in $\mathbb{R}^{1,080\times1,920\times363}$. Table 3 shows the means and standard deviations of a subset of both the runtimes and residual errors for the decompositions of the park bench video across 10 experiments of a rank-250 decomposition. A rank-250 decomposition for this tensor represents only 0.064% of the largest possible true rank of the tensor (namely, $min(IJ, JK, IK)$ (Kolda & Bader, 2009)). Results are shown for CPD-ALS, Sketched CPD (with hand-tuned $s = 10^{-3}$, $\lambda = 0.001$), and CPD-MWU. CPD-MWU produced a decomposition of similar or lower residual error in comparable or less time than the other algorithms.

*Table 3.* Average and standard deviation of runtime (in seconds) and residual error $\epsilon$ across 10 decompositions of the park bench video, in which the algorithms ran until convergence.

| | $\epsilon$ | Std($\epsilon$) | Time | Std(Time) |
|---|---|---|---|---|
| CPD-ALS | 0.4999 | 6.17*$10^{-4}$ | 621.12 | 78.35 |
| Sketched CPD | 0.5042 | 2.57*$10^{-4}$ | 527.98 | 62.83 |
| CPD-MWU | 0.5020 | 5.94*$10^{-3}$ | **465.83** | 167.16 |

### 5.3. Downstream Application - Knowledge Base Mining

The Carnegie Mellon Never Ending Language Learning (NELL) repository (Carlson et al., 2010) has previously been analyzed using CPD (Kang et al., 2012). We developed single-server shared memory Python implementations of CPD-MWU, CPRAND, CPRAND-MIX (Battaglino et al., 2018) and SPALS (Cheng et al., 2016), to compare the performance of CPD-MWU to these ALS baselines all on the same platform. We also compared to an implementation of the first-order BrasCPD algorithm (Fu et al., 2019), a recent block-randomized sketched gradient algorithm for CPD, for comparison. From NELL, we created a tensor in $\mathbb{R}^{120\times 918\times 2{,}881}$ with a density of $2.7 \times 10^{-4}$. We assumed that one cannot obtain a rank 30 approximation with error below what is achievable by running traditional CPD-ALS to convergence. Therefore, we set out to determine which approach would achieve a target relative residual error (less than 0.060, as determined by running CPD-ALS to convergence) in the least amount of time. Each implementation was configured to stop after achieving an error below the target threshold or after 30 minutes.

We compared the performance of each algorithm with and without pre-mixing the tensor, and report the results for the best version of each. Here, mixing refers to a preprocessing procedure that attempts to reduce the coherence of the tensor so that sketching is more accurate (Battaglino et al., 2018). Each algorithm performed best with pre-mixing except SPALS. Table 4 shows the results comparing CPD-ALS, CPD-MWU, CPRAND-MIX, SPALS, and BrasCPD-MIX, all implemented in Python. On average, CPD-MWU is 2.9x faster than traditional CPD-ALS with pre-mixing. Neither CPRAND-MIX, SPALS, or BrasCPD-MIX reach the target decomposition error, and thus end when the maximum execution time is reached.

*Table 4.* Average and standard deviation of runtime (in seconds) and residual error $\epsilon$ across 10 decompositions of the NELL knowledge base extract after up to 30 minutes.

| | $\epsilon$ | Std($\epsilon$) | Time | Std(Time) |
|---|---|---|---|---|
| SPALS | 0.104 | 0.0061 | 1829.36 | 14.84 |
| CPRAND-MIX | 0.072 | 0.0046 | 1806.70 | 3.50 |
| CPD-ALS + MIX | 0.060 | 0.0002 | 1044.75 | 386.03 |
| CPD-MWU + MIX | 0.058 | 0.0015 | **354.55** | 224.59 |
| BrasCPD + MIX | 0.349 | 0.0427 | 1829.99 | 8.70 |

We next ran the same experiments for considerably longer durations to demonstrate that the statically-sketched algorithms are unable to converge, even after a substantial amount of time. Table 5 shows the results when the algorithms run up to two hours. Note that we re-ran all of the algorithms, hence we report slightly different (though consistent) results for CPD-ALS and CPD-MWU, both of which converge well before the two hour limit. This is experimental evidence of the fact that the sketching rates must be non-static for sketched ALS algorithms to converge.

*Table 5.* Average and standard deviation of runtime (in seconds) and residual error $\epsilon$ across 10 decompositions of the NELL knowledge base extract after up to 2 hours.

| | $\epsilon$ | Std($\epsilon$) | Time | Std(Time) |
|---|---|---|---|---|
| SPALS | 0.098 | 0.0045 | 7224.28 | 18.50 |
| CPRAND-MIX | 0.066 | 0.0039 | 7205.37 | 4.03 |
| CPD-ALS + MIX | 0.060 | 0.0002 | 1007.48 | 372.58 |
| CPD-MWU + MIX | 0.058 | 0.0015 | **337.16** | 204.28 |
| BrasCPD + MIX | 0.285 | 0.0337 | 7209.90 | 2.40 |

For each of these synthetic and real data experiments, CPD-MWU outperformed the baselines in terms of runtime and/or final decomposition accuracy.

## 6. Conclusions & Future Work

This work establishes the sublinear convergence rate of sketched CPD-ALS algorithms, and introduces CPD-MWU, a regularized, sketched CPD-ALS algorithm that dynamically selects the sketching rate to balance computational efficiency and decomposition accuracy.

Experiments on both synthetic and real datasets demonstrate that CPD-MWU produces lower error decompositions in less time than traditional CPD-ALS and prior sketched CPD-ALS algorithms. We believe that future investigations will unearth more of the potential in dynamically adjusting the sketching rate. For example, alternative approaches could modify the proximal regularization at each iteration while using aggressive sampling throughout, or extend results on continuous bandit optimization problems to remove the requirement that the algorithm be provided with a fixed, finite set of sketching rates.

CPD-MWU is available at `https://github.com/kaggour/CPD-MWU`.

## References

Acar, E., Dunlavy, D. M., and Kolda, T. G. A scalable optimization approach for fitting canonical tensor decompositions. *J. Chemometrics*, 25(2):67–86, 2011.

Aggour, K. S. and Yener, B. Adapting to data sparsity for efficient parallel PARAFAC tensor decomposition in

Hadoop. In *IEEE Int. Conf. Big Data (Big Data)*, pp. 294–301, December 2016. doi: 10.1109/BigData.2016.7840615.

Aggour, K. S., Gittens, A. A. T., and Yener, B. Accelerating a distributed CPD algorithm for large dense, skewed tensors. In *IEEE Int. Conf. Big Data (Big Data)*, pp. 408–417, December 2018.

Battaglino, C., Ballard, G., and Kolda, T. G. A practical randomized CP tensor decomposition. *SIAM J. Matrix Anal. Appl.*, 39(2):876–901, 2018. doi: 10.1137/17m1112303.

Beck, A. *First-Order Methods in Optimization*. Society for Industrial and Applied Mathematics, Philadephia, PA, 2017.

Bhojanapalli, S. and Sanghavi, S. A new sampling technique for tensors. 2015. URL arXiv:1502.05023[stat.ML].

Bro, R. PARAFAC. Tutorial and applications. *Chemometrics Intell. Laboratory Syst.*, 38(2):149–171, October 1997. doi: 10.1016/s0169-7439(97)00032-4.

Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, Jr., E. R., and Mitchell, T. M. Toward an architecture for never-ending language learning. In *Proc. 24th AAAI Conf. Artificial Intell.*, AAAI'10, pp. 1306–1313, July 2010.

Cesa-Bianchi, N. and Gábor, L. *Prediction, Learning, and Games*. Cambridge Univ. Press, Cambridge, UK, 2006.

Cheng, D., Peng, R., Liu, Y., and Perros, I. SPALS: Fast alternating least squares via implicit leverage scores sampling. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Proc. 29th Adv. Neural Info. Process. Syst. (NIPS)*, pp. 721–729. Curran Associates, Inc., 2016.

Cichocki, A., Mandic, D., De Lathauwer, L., Zhou, G., Zhao, Q., Caiafa, C., and Phan, A.-H. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag.*, 32(2):145–163, March 2015. doi: 10.1109/msp.2013.2297439.

Fanaee-T, H. and Gama, J. Tensor-based anomaly detection: An interdisciplinary survey. *Knowl.-Based Syst.*, 98: 130–147, 2016. doi: 10.1016/j.knosys.2016.01.027.

Fu, X., Gao, C., Wai, H.-T., and Huang, K. Block-randomized stochastic proximal gradient for constrained low-rank tensor factorization. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7485–7489. IEEE, 2019.

Gujral, E., Pasricha, R., and Papalexakis, E. E. SamBaTen: Sampling-based batch incremental tensor decomposition. In *Proc. SIAM Int. Conf. Data Mining (SDM)*, pp. 387–395, May 2018.

Kang, U., Papalexakis, E. E., Harpale, A., and Faloutsos, C. GigaTensor: Scaling tensor analysis up by 100 times - algorithms and discoveries. In *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, pp. 316–324, 2012.

Kiers, H. A. L., ten Berge, J. M. F., and Bro, R. PARAFAC2 - Part I. a direct fitting algorithm for the PARAFAC2 model. *J. Chemometrics*, 13(3-4):275—-294, 1999.

Kolda, T. G. and Bader, B. W. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, August 2009. ISSN 0036-1445. doi: 10.1137/07070111X.

Li, N., Kindermann, S., and Navasca, C. Some convergence results on the regularized alternating least-squares method for tensor decomposition. *Linear Algebra Appl.*, 438(2): 796–812, January 2013. doi: 10.1016/j.laa.2011.12.002.

Nguyen, N. H., Drineas, P., and Tran, T. D. Tensor sparsification via a bound on the spectral norm of random tensors. *Information and Inference: A Journal of the IMA*, 4(3):195–229, 2015.

Papalexakis, E. E., Faloutsos, C., and Sidiropoulos, N. D. ParCube: Sparse parallelizable tensor decompositions. In *Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases (ECMLKDD)*, pp. 521–536, 2012.

Papalexakis, E. E., Faloutsos, C., and Sidiropoulos, N. D. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Trans. Intell. Syst. Technol. (TIST)*, 8(2):16:1–16:44, January 2017.

Reynolds, M. J., Doostan, A., and Beylkin, G. Randomized alternating least squares for canonical tensor decompositions: Application to a PDE with random data. *SIAM J. Scientific Comput.*, 38(5):2634–2664, 2016. doi: 10.1137/15m1042802.

Sidiropoulos, N. D., Papalexakis, E. E., and Faloutsos, C. A parallel algorithm for big tensor decomposition using randomly compressed cubes (PARACOMP). In *IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, pp. 1–5, May 2014. doi: 10.1109/ICASSP.2014.6853546.

Sobral, A., Javed, S., Jung, S. K., Bouwmans, T., and hadi Zahzah, E. Online stochastic tensor decomposition for background subtraction in multispectral video sequences. In *IEEE Int. Conf. Comput. Vision Workshop (ICCVW)*, pp. 946–953, December 2015. doi: 10.1109/ICCVW.2015.125.

Song, Z., Woodruff, D., and Zhang, H. Sublinear time orthogonal tensor decomposition. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Proc. 29th Adv. Neural Info. Process. Syst. (NIPS)*, pp. 793–801. Curran Associates, Inc., 2016.

Song, Z., Woodruff, D. P., and Zhong, P. Relative error tensor low rank approximation. *Electron. Colloq. Comput. Complexity (ECCC)*, 25:103, 2018.

Tomasi, G. and Bro, R. A comparison of algorithms for fitting the PARAFAC model. *Comput. Statist. Data Anal.*, 50(7):1700–1734, April 2006.

Tsourakakis, C. E. MACH: Fast randomized tensor decompositions. *Proc. SIAM Int. Conf. Data Mining (SDM)*, pp. 689–700, 2010. doi: 10.1137/1.9781611972801.60.

Unknown. Man sitting on a bench, 2018. URL https://videos.pexels.com/videos/man-sitting-on-a-bench-853751. Accessed on Jan. 10, 2019.

Vervliet, N. and De Lathauwer, L. A randomized block sampling approach to canonical polyadic decomposition of large-scale tensors. *IEEE J. Sel. Topics Signal Process.*, 10(2):284–295, March 2016. doi: 10.1109/jstsp.2015.2503260.

Wang, S., Gittens, A., and Mahoney, M. W. Sketched ridge regression: optimization perspective, statistical perspective, and model averaging. *The Journal of Machine Learning Research*, 18(1):8039–8088, 2017.

Wang, Y., Tung, H.-Y., Smola, A. J., and Anandkumar, A. Fast and guaranteed tensor decomposition via sketching. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Proc. 28th Adv. Neural Info. Process. Syst. (NIPS)*, pp. 991–999. Curran Associates, Inc., 2015.

Xu, Y. and Yin, W. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789, 2013.

Yang, B., Zamzam, A., and Sidiropoulos, N. D. ParaSketch: Parallel tensor factorization via sketching. In *Proc. SIAM Int. Conf. Data Mining (SDM)*, pp. 396–404, May 2018.

Yu, R., Purushotham, S., and Liu, Y. Efficient spatiotemporal sampling via low-rank tensor sketching. In *Proc. Time Series Workshop at Conf. Neural Info. Process. Syst. (NIPS)*, pp. 5, 2015.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., and Stoica, I. Spark: Cluster computing with working sets. In *Proc. 2nd USENIX Conf. Hot Topics Cloud Comput. (HotCloud)*, pp. 10–10, 2010.