A   Additional Notation

Definition A.1. *Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$ then the Kronecker product, $A \otimes B \in \mathbb{R}^{mp \times nq}$ is defined by*

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \ldots & a_{1n}B \\ a_{21}B & a_{22}B & \ldots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \ldots & a_{mn}B \end{bmatrix}$$

Definition A.2. *Let $A \in \mathbb{R}^{m \times R}$ and $B \in \mathbb{R}^{n \times R}$ then the Khatri-Rao product, $A \odot B \in \mathbb{R}^{mn \times R}$ is defined by*

$$A \odot B = \begin{bmatrix} | & | & & | \\ a_1 \otimes b_1 & a_2 \otimes b_2 & \ldots & a_R \otimes b_R \\ | & | & & | \end{bmatrix}$$

where $a_1, \ldots, a_R \in \mathbb{R}^m$ are the columns of A, $b_1, \ldots, b_R \in \mathbb{R}^n$ are the columns of B and the columns of $A \odot B$ is the subset of the Kronecker product. In the corresponding tensor network diagram, the copy tensor captures the fact that the second indices are the same.

A.1   Details about Orthogonalization of the TT Decomposition

Figure 6 illustrates the single-site TT-ALS method, which begins with a TT decomposition in canonical form initialized by a crude guess. Core $\mathcal{A}_1$ of the decomposition is non-orthogonal; in sweeps from left-to-right and right-to-left, the algorithm holds all but one core constant and solves for the optimal value for the remaining core. After updating each core, by a QR decomposition the non-orthonormal part is merged to the left or right (depending on the direction of the sweep), a step which is called *core orthogonalization*.
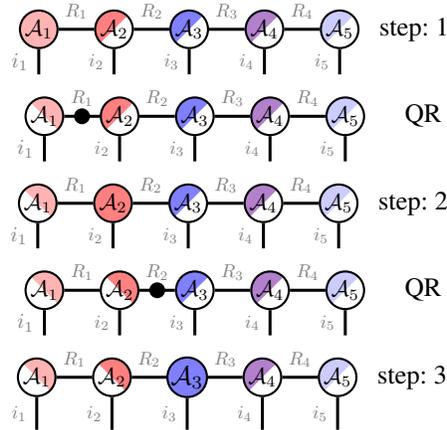
Figure 6: Half-sweep of TT-ALS. In each non-QR step the fully colored core is optimized and in each QR step the non-orthogonal component (depicted by black circle) is absorbed to the next core. This procedure repeats until reaching the right side of the decomposition then the same procedure is repeated from right until reaching to the left side (not demonstrated in this figure.)

16

B Proofs

B.1 Proof of Lemma 4.1

Noting that $A_{\leq j}[s_1 \ldots s_j, :]$ is a **row vector**, we write

$$
\begin{aligned}
& p(\hat{s}_k = s_k \mid \hat{s}_{>k} = s_{>k}) \\
&= \sum_{s_1,\ldots,s_{k-1}} p(\hat{s}_1 = s_1 \wedge \ldots \wedge \hat{s}_j = s_j) \\
&= \sum_{s_1,\ldots,s_{k-1}} \frac{1}{R_j} \left(A_{\leq j}[s_1 \ldots s_j, :] \cdot A_{\leq j}[s_1 \ldots s_j, :]^\top \right) \\
&= \sum_{s_1,\ldots,s_{k-1}} \frac{1}{R_j} \mathrm{Tr} \left[A_{\leq j}[s_1 \ldots s_j, :]^\top \cdot A_{\leq j}[s_1 \ldots s_j, :] \right] \\
&= \frac{1}{R_j} \sum_{s_1,\ldots,s_{k-1}} \mathrm{Tr} \left[\mathcal{A}_j[:, s_j, :]^\top \cdot \ldots \cdot \mathcal{A}_1[:, s_1, :]^\top \cdot \mathcal{A}_1[:, s_1, :] \cdot \ldots \cdot \mathcal{A}_j[:, s_j, :] \right] \\
&= \frac{1}{R_j} \sum_{s_2,\ldots,s_{k-1}} \mathrm{Tr} \left[\mathcal{A}_j[:, s_j, :]^\top \cdot \ldots \cdot \left(\sum_{s_1} \mathcal{A}_1[:, s_1, :]^\top \cdot \mathcal{A}_1[:, s_1, :] \right) \cdot \ldots \cdot \mathcal{A}_j[:, s_j, :] \right] \\
&= \frac{1}{R_j} \sum_{s_2,\ldots,s_{k-1}} \mathrm{Tr} \left[\mathcal{A}_j[:, s_j, :]^\top \cdot \ldots \cdot \mathcal{A}_2[:, s_2, :]^\top \cdot I \cdot \mathcal{A}_2[:, s_2, :] \cdot \ldots \cdot \mathcal{A}_j[:, s_j, :] \right].
\end{aligned}
\tag{7}
$$

In the expressions above, the summation over each variable s_t, $1 \leq t \leq k$, is taken over the range $[I_t]$. The first step follows by marginalizing over random variables $\hat{s}_1, \ldots, \hat{s}_{k-1}$. The second step follows from Equation (4). The third step rewrites an inner product of two vectors as the trace of their outer product. The fourth step follows from the definition of $A_{\leq j}$. The fifth step follows from the linearity of the trace by moving the summation over s_1 into the product expression. The last step follows from the definition of the left-orthonormality property on $\mathcal{A}_1$; that is, $\sum_{s_1} \mathcal{A}_1[:, s_1, :]^\top \cdot \mathcal{A}[:, s_1, :] = A_1^{L\top} A_1^L = I$. By successively moving summation operators into the product expression to repeat the last step (exploiting the left-orthonormality of each core in the process), we find

$$
\begin{aligned}
& p(\hat{s}_k = s_k \mid \hat{s}_{>k} = s_{>k}) \\
&= \frac{1}{R_j} \mathrm{Tr} \left[\mathcal{A}_j[:, s_j, :]^\top \cdot \ldots \cdot \mathcal{A}_k[:, s_k, :]^\top \cdot \mathcal{A}_k[:, s_k, :] \cdot \ldots \cdot \mathcal{A}_j[:, s_j, :] \right] \\
&= \frac{1}{R_j} \mathrm{Tr} \left[H_{>k}^\top \cdot \mathcal{A}_k[:, s_k, :]^\top \cdot \mathcal{A}_k[:, s_k, :] \cdot H_{>k} \right],
\end{aligned}
\tag{8}
$$

where the last line follows from the definition of $H_{>k}$. $\square$

B.2 Proof of Lemma 4.2

We write

$$
\begin{aligned}
p(\hat{s}_k = s_k \mid \hat{s}_{>k} = s_{>k}) &= \frac{1}{R_j} \mathrm{Tr} \left[H_{>k}^\top \cdot \mathcal{A}_k[:, s_k, :]^\top \mathcal{A}_k[:, s_k, :] \cdot H_{>k} \right] \\
&= \frac{1}{R_j} \sum_{r=1}^{R_j} \left(e_r^\top \cdot H_{>k}^\top \cdot \mathcal{A}_k[:, s_k, :]^\top \cdot \mathcal{A}_k[:, s_k, :] \cdot H_{>k} \cdot e_r \right) \\
&= \frac{1}{R_j} \sum_{r=1}^{R_j} \left(h_{>k}^\top \cdot \mathcal{A}_k[:, s_k, :]^\top \cdot \mathcal{A}_k[:, s_k, :] \cdot h_{>k} \right) \\
&= \frac{1}{R_j} \sum_{r=1}^{R_j} p(\hat{t}_k = t_k \mid \hat{t}_{>k} = t_{>k}, \hat{r} = r).
\end{aligned}
\tag{9}
$$

The first step follows from Lemma 4.1. The second step follows from the definition of the trace. The third step follows from the definitions of $h_{>k}$ and $H_{>k}$. The fourth step follows from the definition of the variables $\hat{t}_1, \ldots, \hat{t}_j$. Now observe that $p(\hat{r} = r) = 1/R_j$ for $1 \leq r \leq R_j$, so we can write

17

$$p(\hat{s}_k = s_k \mid \hat{s}_{>k} = s_{>k}) = \sum_{r=1}^{R_j} p(\hat{t}_k = t_k \mid \hat{t}_{>k} = t_{>k}, \hat{r} = r) p(\hat{r} = r) \tag{10}$$
$$= p(\hat{t}_k = t_k \mid \hat{t}_{>k} = t_{>k}),$$

which completes the proof. $\qquad\square$

B.3 Efficient Sampling Data Structure

Lemma 4.3 first appeared as Lemma 3.2 in the original work by Bharadwaj et al. [2023]. We state a condensed form of the original claim below:

Lemma B.1 (Bharadwaj et al. [2023], Original). *Given $U \in \mathbb{R}^{M \times R}$, $Y \in \mathbb{R}^{R \times R}$ with Y p.s.d., there exists a data structure parameterized by positive integer F that requires $O(MR^2)$ time to construct and additional space space $O(R^2 \lceil M/F \rceil)$. After construction, the data structure can draw a sample from the distribution defined elementwise by*

$$q_{h,U,Y}[s] := C^{-1} U[s,:] \left(Y \circledast hh^\top \right) U[s,:]^\top$$

in time $O(R^2 \log \lceil M/F \rceil + FR^2)$. When Y is a rank-1 matrix, the runtime drops to $O(R^2 \log \lceil M/F \rceil + FR)$.

In the statement above, C is an appropriate normalization constant. To prove our adapted lemma, take $Y = [1]$, a matrix of all ones that is rank-1, and set $F = R$. Then

$$q_{h,U,Y}[s] = C^{-1} U[s,:] \left(hh^\top \right) U[s,:]^\top = C^{-1}(U[s,:] \cdot h)^2$$

This is the target probability distribution of Lemma 4.3, and the runtime to draw each sample is $O(R^2 \log(M/R) + R^2) = O(R^2 \log(M/R))$. The choice $F = R$ also induces space usage $O(MR)$, linear in the size of the input. Our modified claim follows. $\qquad\square$

B.4 Proof of Theorem 1.1

We provide a short end-to-end proof that shows that Algorithms 1 and 2 correctly draw samples from $A_{\leq j}$ (the matricization of the left-orthogonal core chain) according to the distribution of its squared row norms while meeting the runtime and space guarantees of Theorem 1.1.

Construction Complexity: The cost of Algorithm 1 follows from 4.3 with $M = IR_{k-1}$, the row count of A_k^L for $1 \leq k \leq j$. Using this lemma, construction of each sampling data structure Z_k requires time $O(I_k R_{k-1} R_k^2)$. The space required by sampler Z_k is $O(I_k R_{k-1} R_k)$; summing over all indices k gives the construction claim in Theorem 1.1.

Sampling Complexity: The complexity to draw samples in Algorithm 2 is dominated by calls to the RowSample procedure, which as discussed in Section 4 is $O(R_k^2 \log(I_k R_{k-1}/R_k))$ Summing the complexity over indices $1 \leq k < j$ yields the cost claimed by Theorem 1.1 to draw a single sample. The complexity of calling the RowSample procedure repeatedly dominates the complexity to update the history vector h over all loop iterations, which is $O\left(\sum_{k=1}^{j} R_{k-1} R_k\right)$ for each sample.

Correctness: Our task is to show that Algorithm 2 each sample t_d, $1 \leq d \leq J$, is a multi-index that follows the squared row norm distribution on the rows of $A_{\leq j}$. To do this, we rely on lemmas proven earlier. For each sample, the variable $\hat{r}$ is a uniform random draw from $[R_j]$, and h is initialized to the corresponding basis vector. By Equation (6) and Lemma 4.3, Line 5 from Algorithm 2 draws each index $\hat{t}_k$ correctly according to the probability distribution specified by Equation (5). The history vector is updated by Line 6 of the algorithm so that subsequent draws past iteration k of the loop are also drawn correctly according to Equation (5). Lemma 4.2 (relying on Lemma 4.1) shows that the multi-index $\hat{t}_1 \ldots \hat{t}_j$ drawn according to Equation (5) follows the same distribution as $\hat{s}_1 \ldots \hat{s}_j$, which was defined to follow the squared norm distribution on the rows of $A_{\leq j}$. This completes the proof. $\qquad\square$

18

B.5 Proof of Corollary 4.4

Since $A^{\neq j} \in \mathbb{R}^{\prod_{k\neq j}^{N} I_k \times R_{j-1}R_j}$ and $X_{(j)} \in \mathbb{R}^{\prod_{k\neq j}^{N} I_k \times I_j}$, we draw $\tilde{O}(R^2/\varepsilon\delta)$ samples to achieve the error bound $(1+\varepsilon)$ with probability $(1-\delta)$ for each least squares solve in the down-sampled problem (3.5). By Theorem 1.1, the complexity of drawing J samples with our data structure is

$$O\left(\sum_{k\neq j} J \log I_k R^2\right) = \tilde{O}\left(\sum_{k\neq j} R^4/(\varepsilon\delta) \log I_k\right)$$

where we suppose that $R_1 = R_2 = \cdots = R_{N-1}$ and $I_1 = \cdots = I_N$. The cost of sampling a corresponding subset of $X_{(j)}$ is $O(JI_j) = \tilde{O}\left(R^2/(\varepsilon\delta)I_j\right)$. Solving the downsampled least squares problem also costs $O(JR^2I_j) = \tilde{O}\left(I_j R^4/(\varepsilon\delta)\right)$. Summing them all together for $1 \leq j \leq N$ gives

$$\tilde{O}\left(1/\varepsilon\delta\left(\sum_{j=1}^{N}\left(\sum_{k\neq j} R^4 \log I_k\right) + R^4 I_j\right)\right)$$

$$= \tilde{O}\left(R^4/\varepsilon\delta \cdot \sum_{j=1}^{N}(N-1)\log I_j + I_j\right)$$

$$= \tilde{O}\left(R^4/\varepsilon\delta \cdot \sum_{j=1}^{N} N \log I_j + I_j\right)$$

where we wrote the last equation considering the fact that N dominates $(N-1)$.

C Details about Datasets & Experiments

C.1 Datasets

For the real dense datasets experiment, we truncated and reshaped the original data tensors in to the fourth order tensors as follows.

- **Pavia University dataset**: The original has dimensions $(610, 340, 103)$. We truncate it to $(600, 320, 100)$, permute the modes to dimensions $(100, 320, 600)$ tensor and reshape it into a tensor of dimensions $(100, 320, 24, 25)$. It is available at
  http://lesun.weebly.com/hyperspectral-data-set.html
- **Tabby Cat dataset** is permuted to $(286, 720, 1280)$ and reshaped to a tensor of size $(286, 720, 40, 32)$. The video is in color and converted to grayscale by averaging the three color channels. It is available at
  https://www.pexels.com/video/video-of-a-tabby-cat-854982/.
- **The MNIST dataset** was reshaped into a tensor of size $(280, 600, 28, 10)$ and is available at https://www.kaggle.com/datasets/hojjatk/mnist-dataset
- **The Washington DC Mall dataset** was truncated to dimensions $(1280, 306, 190)$ before reshaping into a tensor of size $(1280, 306, 10, 19)$. It is available at
  https://engineering.purdue.edu/Ëœbiehl/MultiSpec/hyperspectral.html.

The sparse tensors **Uber**, **Enron**, and **NELL-2** were downloaded from the FROSTT collection [Smith et al., 2017]. The dimensions of these tensors were unchanged from the versions available online. Consistent with established practice [Larsen and Kolda, 2022], we computed the logarithm of the tensor values in the Enron and NELL-2 datasets before performing our experiments.

C.2 Computing Resources

The dense data experiments were conducted on MILA cluster nodes with 4 CPUs and 16GB of RAM each. Sparse tensor decomposition experiments were conducted on NERSC Perlmutter nodes with 2 CPUs and 512 GB of RAM each.

19