A Sampling-Based Method for Tensor Ring Decomposition: Supplementary Material

Osman Asif Malik* Stephen Becker†

S1 Missing Proofs

In this section we give proofs for Lemma 6 and Theorem 7 in the main manuscript. We first state some results that we will use in these proofs.

Lemma S1 is a variant of Lemma 1 by Drineas et al. (2011) but for multiple right hand sides. The proof by Drineas et al. (2011) remains essentially identical with only minor modifications to account for the multiple right hand sides.

Lemma S1. *Let* $\mathrm{OPT} \overset{\text{def}}{=} \min_{\boldsymbol{X}} \|\boldsymbol{A}\boldsymbol{X} - \boldsymbol{Y}\|_{\mathrm{F}}$ *be a least squares problem where* $\boldsymbol{A} \in \mathbb{R}^{I \times R}$ *and* $I > R$*, and let* $\boldsymbol{U} \in \mathbb{R}^{I \times \mathrm{rank}(\boldsymbol{A})}$ *contain the left singular vectors of* $\boldsymbol{A}$*. Moreover, let* $\boldsymbol{U}^{\perp}$ *be an orthogonal matrix whose columns span the space perpendicular to* $\mathrm{range}(\boldsymbol{U})$ *and define* $\boldsymbol{Y}^{\perp} \overset{\text{def}}{=} \boldsymbol{U}^{\perp}(\boldsymbol{U}^{\perp})^{\top}\boldsymbol{Y}$*. Let* $\boldsymbol{S} \in \mathbb{R}^{J \times I}$ *be a matrix satisfying*

$$\sigma_{\min}^2(\boldsymbol{S}\boldsymbol{U}) \geq \frac{1}{\sqrt{2}},$$

$$\|\boldsymbol{U}^{\top}\boldsymbol{S}^{\top}\boldsymbol{S}\boldsymbol{Y}^{\perp}\|_{\mathrm{F}}^2 \leq \frac{\varepsilon}{2}\mathrm{OPT}^2,$$

for some $\varepsilon \in (0, 1)$*. Then, it follows that*

$$\|\boldsymbol{A}\tilde{\boldsymbol{X}} - \boldsymbol{Y}\|_{\mathrm{F}} \leq (1 + \varepsilon)\mathrm{OPT},$$

where $\tilde{\boldsymbol{X}} \overset{\text{def}}{=} \arg\min_{\boldsymbol{X}} \|\boldsymbol{S}\boldsymbol{A}\boldsymbol{X} - \boldsymbol{S}\boldsymbol{Y}\|_{\mathrm{F}}$*.*

Lemma S2 is a slight restatement of Theorem 2.11 in the monograph by Woodruff (2014). The statement in that monograph has a constant 144 instead of 8/3. However, we found that 8/3 is sufficient under the assumption that $\varepsilon \in (0, 1)$. The proof given by Woodruff (2014) otherwise remains the same.

Lemma S2. *Let* $\boldsymbol{A} \in \mathbb{R}^{I \times R}$*,* $\varepsilon \in (0, 1)$*,* $\eta \in (0, 1)$ *and* $\beta \in (0, 1]$*. Suppose*

$$J > \frac{8}{3}\frac{R\ln(2R/\eta)}{\beta\varepsilon^2}$$

and that $\boldsymbol{S} \sim \mathcal{D}(J, \boldsymbol{q})$ *is a leverage score sampling matrix for* $(\boldsymbol{A}, \beta)$ *(see Definition 5 in the main manuscript). Then, with probability at least* $1 - \eta$*, the following holds:*

$$(\forall i \in [\mathrm{rank}(\boldsymbol{A})]) \quad 1 - \varepsilon \leq \sigma_i^2(\boldsymbol{S}\boldsymbol{U}) \leq 1 + \varepsilon,$$

where $\boldsymbol{U} \in \mathbb{R}^{I \times \mathrm{rank}(\boldsymbol{A})}$ *contains the left singular vectors of* $\boldsymbol{A}$*.*

Lemma S3 is a part of Lemma 8 by Drineas et al. (2006).

*Dept. of Applied Mathematics, University of Colorado Boulder, Boulder, CO, USA. Email: osman.malik@colorado.edu
†Dept. of Applied Mathematics, University of Colorado Boulder, Boulder, CO, USA. Email: stephen.becker@colorado.edu

Lemma S3. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be matrices with I rows, and let $\beta \in (0,1]$. Let $\boldsymbol{q} \in \mathbb{R}^I$ be a probability distribution satisfying*

$$\boldsymbol{q}(i) \geq \beta \frac{\|\boldsymbol{A}(i,:)\|_2^2}{\|\boldsymbol{A}\|_{\mathrm{F}}^2} \quad \text{for all } i \in [I].$$

If $\boldsymbol{S} \sim \mathcal{D}(J, \boldsymbol{q})$, then

$$\mathbb{E}\|\boldsymbol{A}^\top \boldsymbol{B} - \boldsymbol{A}^\top \boldsymbol{S}^\top \boldsymbol{S} \boldsymbol{B}\|_{\mathrm{F}}^2 \leq \frac{1}{\beta J} \|\boldsymbol{A}\|_{\mathrm{F}}^2 \|\boldsymbol{B}\|_{\mathrm{F}}^2.$$

In the following, $\otimes$ denotes the matrix Kronecker product; see e.g. Section 1.3.6 of Golub and Van Loan (2013) for details.

Lemma S4. *For $n \in [N]$, the subchain $\mathcal{G}^{\neq n}$ satisfies*

$$\operatorname{range}(\boldsymbol{G}_{[2]}^{\neq n}) \subseteq \operatorname{range}(\boldsymbol{G}_{(2)}^{(n-1)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)} \otimes \boldsymbol{G}_{(2)}^{(N)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)}). \tag{S1}$$

Proof. We have

$$\boldsymbol{G}_{[2]}^{\neq n}(\overline{i_{n+1} \cdots i_N i_1 \cdots i_{n-1}}, \overline{r_{n-1} r_n}) = \sum_{\substack{r_1,\ldots,r_{n-2} \\ r_{n+1},\ldots,r_N}} \boldsymbol{G}_{(2)}^{(n+1)}(i_{n+1}, \overline{r_n r_{n+1}}) \cdots \boldsymbol{G}_{(2)}^{(N)}(i_N, \overline{r_{N-1} r_N}) \boldsymbol{G}_{(2)}^{(1)}(i_1, \overline{r_N r_1}) \cdots \boldsymbol{G}_{(2)}^{(n-1)}(i_{n-1}, \overline{r_{n-2} r_{n-1}}).$$

From this, it follows that

$$\boldsymbol{G}_{[2]}^{\neq n}(:, \overline{r_{n-1} r_n}) = \sum_{\substack{r_1,\ldots,r_{n-2} \\ r_{n+1},\ldots,r_N}} \boldsymbol{G}_{(2)}^{(n-1)}(:, \overline{r_{n-2} r_{n-1}}) \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)}(:, \overline{r_N r_1}) \otimes \boldsymbol{G}_{(2)}^{(N)}(:, \overline{r_{N-1} r_N}) \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)}(:, \overline{r_n r_{n+1}}).$$

The right hand side of this equation is a sum of columns in

$$\boldsymbol{G}_{(2)}^{(n-1)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)} \otimes \boldsymbol{G}_{(2)}^{(N)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)}. \tag{S2}$$

Consequently, every column of $\boldsymbol{G}_{[2]}^{\neq n}$ is in the range of the matrix in (S2), and the claim in (S1) follows. □

Lemma S5. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be two matrices with I rows such that* $\operatorname{range}(\boldsymbol{A}) \subseteq \operatorname{range}(\boldsymbol{B})$. *Then $\ell_i(\boldsymbol{A}) \leq \ell_i(\boldsymbol{B})$ for all $i \in [I]$.*

Proof. Let $\boldsymbol{Q} \in \mathbb{R}^{I \times \operatorname{rank}(\boldsymbol{B})}$ be an orthogonal matrix containing the left singular vectors of $\boldsymbol{B}$. Then there exists a matrix $\boldsymbol{M}$ such that $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{M}$. Let $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{M}$ be the thin SVD of $\boldsymbol{M}$ (i.e., such that $\boldsymbol{U}$ and $\boldsymbol{V}$ have only $\operatorname{rank}(\boldsymbol{M})$ columns and $\boldsymbol{\Sigma} \in \mathbb{R}^{\operatorname{rank}(\boldsymbol{M}) \times \operatorname{rank}(\boldsymbol{M})}$). Then $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^\top$, where $\boldsymbol{W} \stackrel{\text{def}}{=} \boldsymbol{Q}\boldsymbol{U}$. Since

$$\boldsymbol{W}^\top \boldsymbol{W} = \boldsymbol{U}^\top \boldsymbol{Q}^\top \boldsymbol{Q} \boldsymbol{U} = \boldsymbol{I},$$

$\boldsymbol{W}$ is orthogonal, so $\boldsymbol{W}\boldsymbol{\Sigma}\boldsymbol{V}^\top = \boldsymbol{A}$ is a thin SVD of $\boldsymbol{A}$. It follows that

$$\ell_i(\boldsymbol{A}) = \|\boldsymbol{W}(i,:)\|_2^2 = \|\boldsymbol{Q}(i,:)\boldsymbol{U}\|_2^2 \leq \|\boldsymbol{Q}(i,:)\|_2^2 \|\boldsymbol{U}\|_2^2 = \|\boldsymbol{Q}(i,:)\|_2^2 = \ell_i(\boldsymbol{B}).$$

□

Remark S6. It is reasonable to assume that $\mathcal{G}^{\neq n} \neq \boldsymbol{0}$ for all $n \in [N]$ during the execution of Algorithms 1 and 2. If this was not the case, the least squares problem for the nth iteration of the inner for loop in these algorithms would have a zero system matrix which would make the least squares problem meaningless. The assumption $\mathcal{G}^{\neq n} \neq \boldsymbol{0}$ for all $n \in [N]$ also implies that $\mathcal{G}^{(n)} \neq \boldsymbol{0}$ for all $n \in [N]$. This means that $\operatorname{rank}(\boldsymbol{G}_{(2)}^{(n)}) \geq 1$ and $\operatorname{rank}(\boldsymbol{G}_{[2]}^{\neq n}) \geq 1$ for all $n \in [N]$, so that the various divisions with these quantities in this paper are well-defined.

Lemma S7 is a restatement of Lemma 6 in the main manuscript.

Lemma S7. *Let β_n be defined as*

$$\beta_n \stackrel{\text{def}}{=} \Big(R_{n-1}R_n \prod_{\substack{j=1\\ j\notin\{n-1,n\}}}^{N} R_j^2\Big)^{-1}.$$

For each $n \in [N]$, the vector $\boldsymbol{q}^{\neq n}$ is a probability distribution on $[\prod_{j\neq n} I_j]$, and $\boldsymbol{S} \sim \mathcal{D}(J, \boldsymbol{q}^{\neq n})$ is a leverage score sampling matrix for $(\boldsymbol{G}_{[2]}^{\neq n}, \beta_n)$.

Proof. All $\boldsymbol{q}^{\neq n}(i)$ are clearly nonnegative. Moreover,

$$\sum_{\substack{i_1,\dots,i_{n-1}\\ i_{n+1},\dots,i_N}} \boldsymbol{q}^{\neq n}(\overline{i_{n+1}\cdots i_N i_1 \cdots i_{n-1}}) = \prod_{\substack{j=1\\ j\neq n}}^{N} \sum_{i_j} \boldsymbol{p}^{(j)}(i_j) = \prod_{\substack{j=1\\ j\neq n}}^{N} 1 = 1,$$

since each $\boldsymbol{p}^{(j)}$ is a probability distribution. So $\boldsymbol{q}^{\neq n}$ is clearly also a probability distribution. Next, define the vector $\boldsymbol{p} \in \mathbb{R}^{\prod_{j\neq n} I_n}$ via

$$\boldsymbol{p}(i) \stackrel{\text{def}}{=} \frac{\ell_i(\boldsymbol{G}_{[2]}^{\neq n})}{\operatorname{rank}(\boldsymbol{G}_{[2]}^{\neq n})}.$$

To show that $\boldsymbol{S}$ is a leverage score sampling matrix for $(\boldsymbol{G}_{[2]}^{\neq n}, \beta_n)$, we need to show that $\boldsymbol{q}^{\neq n}(i) \geq \beta_n \boldsymbol{p}(i)$ for all $i = \overline{i_{n+1}\cdots i_N i_1 \cdots i_{n-1}} \in [\prod_{j\neq n} I_n]$. To this end, combine Lemmas S4 and S5 to get

$$\ell_i(\boldsymbol{G}_{[2]}^{\neq n}) \leq \ell_i(\boldsymbol{G}_{(2)}^{(n-1)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)} \otimes \boldsymbol{G}_{(2)}^{(N)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)}). \tag{S3}$$

For each $n \in [N]$, let $\boldsymbol{U}^{(n)} \in \mathbb{R}^{I_n \times \operatorname{rank}(\boldsymbol{G}_{(2)}^{(n)})}$ contain the left singular vectors of $\boldsymbol{G}_{(2)}^{(n)}$. It is well-known (Van Loan, 2000) that the matrix

$$\boldsymbol{U}^{(n-1)} \otimes \cdots \otimes \boldsymbol{U}^{(1)} \otimes \boldsymbol{U}^{(N)} \otimes \cdots \otimes \boldsymbol{U}^{(n+1)}$$

contains the left singular vectors corresponding to nonzero singular values of

$$\boldsymbol{G}_{(2)}^{(n-1)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)} \otimes \boldsymbol{G}_{(2)}^{(N)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)}.$$

Consequently,

$$\begin{aligned}
&\ell_i(\boldsymbol{G}_{(2)}^{(n-1)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(1)} \otimes \boldsymbol{G}_{(2)}^{(N)} \otimes \cdots \otimes \boldsymbol{G}_{(2)}^{(n+1)})\\
&= \big((\boldsymbol{U}^{(n-1)} \otimes \cdots \otimes \boldsymbol{U}^{(1)} \otimes \boldsymbol{U}^{(N)} \otimes \cdots \otimes \boldsymbol{U}^{(n+1)})(\boldsymbol{U}^{(n-1)} \otimes \cdots \otimes \boldsymbol{U}^{(1)} \otimes \boldsymbol{U}^{(N)} \otimes \cdots \otimes \boldsymbol{U}^{(n+1)})^\top\big)_{ii}\\
&= \big((\boldsymbol{U}^{(n-1)}\boldsymbol{U}^{(n-1)\top}) \otimes \cdots \otimes (\boldsymbol{U}^{(1)}\boldsymbol{U}^{(1)\top}) \otimes (\boldsymbol{U}^{(N)}\boldsymbol{U}^{(N)\top}) \otimes \cdots \otimes (\boldsymbol{U}^{(n+1)}\boldsymbol{U}^{(n+1)\top})\big)_{ii}\\
&= (\boldsymbol{U}^{(n-1)}\boldsymbol{U}^{(n-1)\top})_{i_{n-1}i_{n-1}} \cdots (\boldsymbol{U}^{(1)}\boldsymbol{U}^{(1)\top})_{i_1 i_1} (\boldsymbol{U}^{(N)}\boldsymbol{U}^{(N)\top})_{i_N i_N} \cdots (\boldsymbol{U}^{(n+1)}\boldsymbol{U}^{(n+1)\top})_{i_{n+1}i_{n+1}}\\
&= \ell_{i_{n-1}}(\boldsymbol{G}_{(2)}^{(n-1)}) \cdots \ell_{i_1}(\boldsymbol{G}_{(2)}^{(1)}) \ell_{i_N}(\boldsymbol{G}_{(2)}^{(N)}) \cdots \ell_{i_{n+1}}(\boldsymbol{G}_{(2)}^{(n+1)}),
\end{aligned} \tag{S4}$$

where the first and fourth equalities follow from the definition of leverage score and the fact that

$$\|\boldsymbol{M}(i,:)\|_2^2 = (\boldsymbol{M}\boldsymbol{M}^\top)(i,i)$$

for any matrix $\boldsymbol{M}$, and the second equality follows from well-known properties of the Kronecker product (Van Loan, 2000). Combining (S3) and (S4), we have

$$\ell_i(\boldsymbol{G}_{[2]}^{\neq n}) \leq \prod_{\substack{j=1 \\ j\neq n}}^{N} \ell_{i_j}(\boldsymbol{G}_{(2)}^{(j)}). \tag{S5}$$

We therefore have

$$\boldsymbol{q}^{\neq n}(i) = \frac{\prod_{j\neq n} \ell_{i_j}(\boldsymbol{G}_{(2)}^{(j)})}{\prod_{j\neq n} \operatorname{rank}(\boldsymbol{G}_{(2)}^{(j)})} \geq \frac{\ell_i(\boldsymbol{G}_{[2]}^{\neq n})}{R_{n-1}R_n \prod_{j\in[N]\setminus\{n-1,n\}} R_j^2} = \beta_n \ell_i(\boldsymbol{G}_{[2]}^{\neq n}) \geq \beta_n \boldsymbol{p}(i)$$

as desired, where the first inequality follows from (S5) and the fact that $\operatorname{rank}(\boldsymbol{G}_{(2)}^{(j)}) \leq R_{j-1}R_j$. □

The following is a restatement of Theorem 7 in the main manuscript. The proof is similar to that of Theorem 2 by Drineas et al. (2011).

Theorem S8. *Let* $\boldsymbol{S} \sim \mathcal{D}(J, \boldsymbol{q}^{\neq n})$, $\varepsilon \in (0,1)$, $\delta \in (0,1)$ *and* $\tilde{\boldsymbol{Z}} \overset{\text{def}}{=} \arg\min_{\boldsymbol{Z}} \|\boldsymbol{S}\boldsymbol{G}_{[2]}^{\neq n}\boldsymbol{Z}_{(2)}^\top - \boldsymbol{S}\boldsymbol{X}_{[n]}^\top\|_{\mathrm{F}}$. *If*

$$J > \Big(\prod_{j=1}^{N} R_j^2\Big) \max\Big(\frac{16}{3(\sqrt{2}-1)^2} \ln\Big(\frac{4R_{n-1}R_n}{\delta}\Big), \frac{4}{\varepsilon\delta}\Big),$$

then the following holds with probability at least $1-\delta$*:*

$$\|\boldsymbol{G}_{[2]}^{\neq n}\tilde{\boldsymbol{Z}}_{(2)}^\top - \boldsymbol{X}_{[n]}^\top\|_{\mathrm{F}} \leq (1+\varepsilon) \min_{\boldsymbol{Z}} \|\boldsymbol{G}_{[2]}^{\neq n}\boldsymbol{Z}_{(2)}^\top - \boldsymbol{X}_{[n]}^\top\|_{\mathrm{F}}. \tag{S6}$$

Proof. Let $\boldsymbol{U} \in \mathbb{R}^{\prod_{j\neq n} I_j \times \operatorname{rank}(\boldsymbol{G}_{[2]}^{\neq n})}$ contain the left singular vectors of $\boldsymbol{G}_{[2]}^{\neq n}$. According to Lemma S7, $\boldsymbol{S}$ is a leverage score sampling matrix for $(\boldsymbol{G}_{[2]}^{\neq n}, \beta_n)$. Since $\boldsymbol{U}$ has at most $R_{n-1}R_n$ columns and

$$J > \frac{16}{3(\sqrt{2}-1)^2}\Big(\prod_{j=1}^{N} R_j^2\Big) \ln\Big(\frac{4R_{n-1}R_n}{\delta}\Big),$$

choosing $\varepsilon = 1 - 1/\sqrt{2}$ and $\eta = \delta/2$ in Lemma S2 therefore gives that

$$\sigma_{\min}^2(\boldsymbol{S}\boldsymbol{U}) \geq 1/\sqrt{2} \tag{S7}$$

with probability at least $1-\delta/2$. Similarly to Lemma S1, define $(\boldsymbol{X}_{[n]}^\top)^\perp \overset{\text{def}}{=} \boldsymbol{U}^\perp(\boldsymbol{U}^\perp)^\top \boldsymbol{X}_{[n]}^\top$ and $\mathrm{OPT} \overset{\text{def}}{=} \min_{\boldsymbol{Z}} \|\boldsymbol{G}_{[2]}^{\neq n}\boldsymbol{Z}_{(2)}^\top - \boldsymbol{X}_{[n]}^\top\|_{\mathrm{F}} = \|(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}$. Since

$$\boldsymbol{q}^{\neq n}(i) \geq \beta_n \frac{\ell_i(\boldsymbol{G}_{[2]}^{\neq n})}{\operatorname{rank}(\boldsymbol{G}_{[2]}^{\neq n})} = \beta_n \frac{\|\boldsymbol{U}(i,:)\|_2^2}{\|\boldsymbol{U}\|_{\mathrm{F}}^2} \quad \text{for all } i \in [I],$$

and $\boldsymbol{U}^\top(\boldsymbol{X}_{[n]}^\top)^\perp = \boldsymbol{0}$, Lemma S3 gives that

$$\mathbb{E}\|\boldsymbol{U}^\top\boldsymbol{S}^\top\boldsymbol{S}(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}^2 \leq \frac{1}{\beta_n J}\|\boldsymbol{U}\|_{\mathrm{F}}^2\|(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}^2 \leq \frac{R_{n-1}R_n}{\beta_n J}\mathrm{OPT}^2.$$

By Markov's inequality,

$$\mathbb{P}(\|\boldsymbol{U}^\top\boldsymbol{S}^\top\boldsymbol{S}(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}^2 > \varepsilon\mathrm{OPT}^2/2) \leq \frac{\mathbb{E}\|\boldsymbol{U}^\top\boldsymbol{S}^\top\boldsymbol{S}(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}^2}{\varepsilon\mathrm{OPT}^2/2} \leq \frac{2}{\varepsilon J}\Big(\prod_{j\in[N]} R_j^2\Big) < \frac{\delta}{2}$$

since

$$J > \frac{4}{\varepsilon\delta}\Big(\prod_{j=1}^{N} R_j^2\Big).$$

Consequently,

$$\|\boldsymbol{U}^\top \boldsymbol{S}^\top \boldsymbol{S}(\boldsymbol{X}_{[n]}^\top)^\perp\|_{\mathrm{F}}^2 \leq \frac{\varepsilon}{2}\mathrm{OPT}^2 \tag{S8}$$

with probability at least $1 - \delta/2$. By a union bound, it follows that both (S7) and (S8) are true with probability at least $1 - \delta$. From Lemma S1 it therefore follows that (S6) is true with probability at least $1 - \delta$. □

S2 Detailed Complexity Analysis

We provide a detailed complexity analysis in this section to show how we arrived at the numbers in Table 1 in the main manuscript.

S2.1 TR-ALS

In our calculations below, we refer to steps in Algorithm 1 in the main paper, consequently ignoring any cost associated with e.g. normalization and checking termination conditions.

Upfront costs of TR-ALS:

- **Line 1: Initializing cores.** This depends on how the initialization of the cores is done. We assume they are randomly drawn, e.g. from a Gaussian distribution, resulting in a cost $O(NIR^2)$.

Costs per outer loop iteration of TR-ALS:

- **Line 4: Compute unfolded subchain tensor.** If the $N-1$ cores are dense and contracted in sequence, the cost is

  $$R^3(I^2 + I^3 + \cdots + I^{N-1}) \leq R^3(NI^{N-2} + I^{N-1}) \leq 2R^3 I^{N-1} = O(I^{N-1}R^3),$$

  where we use the assumption $N < I$ in the second inequality. Doing this for each of the N cores in the inner loop brings the cost to $O(NI^{N-1}R^3)$.

- **Line 5: Solve least squares problem.** We consider the cost when using the standard QR-based approach described in Section 5.3.3 in the book by Golub and Van Loan (2013). The matrix $\boldsymbol{G}_{[2]}^{\neq n}$ is of size $I^{N-1} \times R^2$. Doing a QR decomposition of this matrix costs $O(I^{N-1}R^4)$. Updating the right hand sides and doing back substitution costs $O(I(I^{N-1}R^2 + R^4)) = O(I^N R^2)$, where we used the assumption that $R^2 < I$. The leading order cost for solving the least squares problem is therefore $O(I^N R^2)$. Doing this for each of the N cores in the inner loop brings the cost to $O(NI^N R^2)$.

It follows that the overall leading order cost of TR-ALS is $NIR^2 + \#\text{iter} \cdot NI^N R^2$.

S2.2 rTR-ALS

In our calculations below, we refer to steps in Algorithm 1 by Yuan et al. (2019) with the TR decomposition step in their algorithm done using TR-ALS.

Cost of initial Tucker compression:

- **Line 4: Draw Gaussian matrix.** Drawing these N matrices costs $O(NI^{N-1}K)$.
- **Line 5: Compute random projection.** Computing N projections costs $O(NI^N K)$.

- **Line 6: QR decomposition.** Computing the QR decomposition of an $I \times K$ matrix N times costs $O(NIK^2)$.
- **Line 7: Compute Tucker core.** The cost of compressing each dimension of the input tensor in sequence is
$$I^N K + I^{N-1}K^2 + I^{N-2}K^3 + \cdots + IK^N = O(NI^N K),$$
where we assume that $K < I$.

Cost of TR decompositions:

- **Line 9: Compute TR decomposition of Tucker core.** Using TR-ALS, this costs $O(NKR^2 + \#\text{iter} \cdot NK^N R^2)$ as discussed in Section S2.1.
- **Line 11: Compute large TR cores.** This costs in total $O(NIKR^2)$ for all cores.

Combining these costs and recalling the assumption $R^2 < I$, we get an overall leading order cost for rTR-ALS of $NI^N K + \#\text{iter} \cdot NK^N R^2$.

S2.3 TR-SVD

In our calculations below, we refer to steps in Algorithm 1 by Mickelin and Karaman (2020). We consider a modified version of this algorithm which takes a target rank $(R_1, \ldots, R_N)$ as input instead of an accuracy upper bound ε. For simplicity, we ignore all permutation and reshaping costs, and focus only on the leading order costs which are made up by the SVD calculations.

- **Line 3: Initial SVD.** This is an economy sized SVD of an $I \times I^{N-1}$ matrix, which costs $O(I^{N+1})$; see the table in Figure 8.6.1 of Golub and Van Loan (2013) for details.
- **Line 10: SVD in for loop.** The size of the matrix being decomposed will be $IR \times I^{N-k}R$. The cost of computing an economy sized SVD of each for $k = 2, \ldots, N-1$ is
$$R^3(I^N + I^{N-1} + \cdots + I^3) \leq 2R^3 I^N,$$
where we use the assumption that $N < I$.

Adding these costs up, we get a total cost of $O(I^{N+1} + I^N R^3)$.

S2.4 TR-SVD-Rand

In our calculations below, we refer to steps in Algorithm 7 by Ahmadi-Asl et al. (2020). For simplicity, we ignore all permutation and reshaping costs, and focus only on the leading order costs. In particular, note that the QR decompositions that come up are of relatively small matrices and therefore relatively cheap. Moreover, we assume that the oversampling parameter P is small enough to ignore.

- **Line 2: Compute projection.** The matrix $\boldsymbol{C}$ is of size $I \times I^{N-1}$ and the matrix $\boldsymbol{\Omega}$ is of size $I^{N-1} \times R^2$. The cost of computing their product is $O(I^N R^2)$.
- **Line 6: Computing tensor-times-matrix (TTM) product.** $\boldsymbol{\mathcal{X}}$ is an N-way tensor of size $I \times \cdots \times I$, and $\boldsymbol{Q}^{(1)}$ is of size $I \times R^2$, so this contraction costs $O(I^N R^2)$.
- **Line 11: Compute projections in for loop.** Each of these costs $I^{N-n+1}R^3$. The total cost for all iterations for $n = 2, \ldots, N-1$ is therefore
$$R^3(I^{N-1} + I^{N-2} + \cdots + I^2) \leq 2R^3 I^{N-1},$$
where we used the assumption that $N < I$.

- **Line 15: Compute TTM product in for loop.** Each of these costs $I^{N-n+1}R^3$. The total cost for all iterations is therefore
$$R^3(I^{N-1} + I^{N-2} + \cdots + I^2) \leq 2R^3 I^{N-1},$$
where we used the assumption that $N < I$.

Adding these costs up, we get a total cost of $O(I^N R^2)$.

S2.5 TR-ALS-Sampled

In our calculations below, we refer to steps in Algorithm 2 in the main paper.

Upfront costs of TR-ALS-Sampled:

- **Line 1: Initializing cores.** This is the same as for TR-ALS, namely $O(NIR^2)$.
- **Line 2: Compute distributions.** For each $n = 2, \ldots, N$, this involves computing the economic SVD of an $I \times R^2$ matrix for a cost $O(IR^4)$, and then computing $\boldsymbol{p}^{(n)}$ from the left singular vectors for a cost of $O(IR)$. This yields a total cost for this line of $O(NIR^4)$.

We ignore the cost of the sampling in Line 6 since it is typically very fast.

Costs per outer loop iteration of TR-ALS-Sampled:

- **Line 7: Compute sampled unfolded subchain.** The main cost for this line is computing the product of a sequence of $N-1$ matrices of size $R \times R$, for each of the J sampled slices (see Figure 2 in the main paper). This costs $O(NR^3J)$ per inner loop iteration, or $O(N^2R^3J)$ per outer loop iteration.
- **Line 8: Sample input tensor.** This step requires copying $O(IJ)$ elements from the input tensor. For one iteration of the outer loop, this is $O(NIJ)$ elements. In practice this step together with the least squares solve are the two most time consuming since it involves sampling from a possibly very large array.
- **Line 9: Solve least squares problem.** The cost is computed in the same way as the least squares solve in TR-ALS, but with the large dimension I^{N-1} replaced by J. The cost per least squares problem is therefore $O(IJR^2)$, or $O(NIJR^2)$ for one iteration of the outer loop.
- **Line 10: Update distributions.** For one iteration of the outer loop this costs $O(NIR^4)$.

Adding these costs and simplifying, we arrive at a cost

$$O(NIR^4 + \#\text{iter} \cdot NIJR^2). \tag{S9}$$

If ε and δ are small enough, then (5) simplifies to $J > 4R^{2N}/(\varepsilon\delta)$. Plugging this into (S9) gives us the complexity in Table 1.

S3 Links to Datasets

- The Pavia University dataset was downloaded from
http://lesun.weebly.com/hyperspectral-data-set.html.
- The Washington DC Mall dataset was downloaded from
https://engineering.purdue.edu/~biehl/MultiSpec/hyperspectral.html.
- The Park Bench video was downloaded from
https://www.pexels.com/video/man-sitting-on-a-bench-853751.
The video, which is in color, was made into grayscale by averaging the three color bands.

- The Tabby Cat video was downloaded from
https://www.pexels.com/video/video-of-a-tabby-cat-854982/.
The video, which is in color, was made into grayscale by averaging the three color bands.
- The Red Truck images are part of the COIL-100 dataset, which was downloaded from
https://www1.cs.columbia.edu/CAVE/software/softlib/coil-100.php.

S4 Additional Experiment Details

Here we provide additional experiment details, including how the number of ALS iterations and appropriate sketch rates are determined for the experiments in Section 5.1 of the main paper.

S4.1 Randomly Generated Data

First Experiment When determining the number of ALS iterations, TR-ALS is run until the change in relative error is below 1e−6 or for a maximum of 500 iterations, whichever is satisfied first. The sample size J for TR-ALS-Sampled is started at 200 and incremented by 100. The embedding dimension K for rTR-ALS is started at $I/10$ and incremented by $I/20$. Incrementation is done until the error for each method is smaller than 1.2 times the TR-ALS error.

Second Experiment The sketch rate J for TR-ALS-Sampled is now incremented by 1000. Incrementation is done until the error for each method is smaller than 1.02 times the TR-ALS error. A smaller factor is used compared to the first experiment since the TR-ALS error is much larger. The other settings remain the same as in the first experiment.

Remark S9 (Performance of SVD-based methods)**.** The SVD-based methods typically require much higher ranks than the ALS-based methods. To achieve a similar error to the ALS-based methods in Figures 3 (a) and 4 (a) (0.0031 and 0.94, respectively) the *original unaltered implementation* of TR-SVD by Mickelin and Karaman (2020) requires average TR ranks (i.e., $(R_1 + R_2 + R_3)/3$) in the range of 82–233 and 35–84, respectively. TR-SVD-Rand does poorly for the same reason.

Remark S10 (Empirical vs. theoretical complexity)**.** As discussed in Section 4.3, the benefit of our proposed method is that it has a lower complexity than the competing methods. In particular, it avoids the I^N factors in the complexity expression. It may therefore seem surprising that our method is not the fastest in the experiments. For example, in Figure 3 (b) our method is always slower than TR-SVD-Rand, and in Figure 4 (b) it is always slower than both TR-SVD and TR-SVD-Rand. The reason for this seeming discrepancy is that we only consider a small range of I values ($I \in [100, 500]$) in those figures and therefore the plots will not necessarily reflect the leading order complexities that are given in Table 1. The main cost in TR-SVD-Rand is matrix multiplication which is very efficient, so the hidden constant in the complexity for TR-SVD-Rand is small, and this helps explain why it is faster than our method for the relatively small I values used in Figures 3 and 4. Similar comments also apply to the other experiments.

S4.2 Highly Oscillatory Functions

To determine the number of iterations for the ALS-based methods, we run TR-ALS until the change in relative error is less than 1e−3 or for a maximum of 100 iterations, whichever is satisfied first. The sample size J for TR-ALS-Sampled is started at $2R^2$ and incremented by 100. The embedding dimension K for rTR-ALS is started at 2 and incremented by 1. If $K \geq I_n$, no compression is applied to the nth dimension. The incrementation is done until the error for each method is smaller than 1.1 times the TR-ALS error.

S4.3 Image and Video Data

To determine the number of iterations for the ALS-based methods, we run TR-ALS until the change in relative error is less than 1e−3 or for a maximum of 100 iterations, whichever is satisfied first. The sample size J for TR-ALS-Sampled is started at $2R^2$ and incremented by 1000. The embedding dimension K for rTR-ALS is started at $\max_{n\in[N]} I_n/10$ and incremented by $\max_{n\in[N]} I_n/20$. If $K \geq I_n$, no compression is applied to the nth dimension. The incrementation is done until the error for each method is smaller than 1.1 times the TR-ALS error.

In the experiment on the reshaped tensors, each mode (except the mode representing color channels in the Red Truck dataset) of the original tensor is split into two new modes. Some datasets are also truncated somewhat to allow for this reshaping. The details are given below.

- **Pavia Uni.** is first truncated to size $600 \times 320 \times 100$. This is done by discarding the last elements in each mode via `X = X(1:600, 1:320, 1:100)` in Matlab. The tensor is then reshaped into a $24 \times 25 \times 16 \times 20 \times 10 \times 10$ tensor. This is done by splitting each original mode into two modes via `X = reshape(X,24,25,16,20,10,10)` in Matlab.
- **DC Mall** is truncated to size $1280 \times 306 \times 190$ and then reshaped into a $32 \times 40 \times 18 \times 17 \times 10 \times 19$ tensor similarly to how the Pavia Uni. dataset is truncated and reshaped.
- **Park Bench** does not require any truncation. The original tensor is reshaped into a $24 \times 45 \times 32 \times 60 \times 28 \times 13$ tensor similarly to how the Pavia Uni. dataset is reshaped.
- **Tabby Cat** does not require any truncation. The original tensor is reshaped into a $16 \times 45 \times 32 \times 40 \times 13 \times 22$ tensor similarly to how the Pavia Uni. dataset is reshaped.
- **Red Truck** does not require any truncation. The original tensor is reshaped into a $8\times16\times8\times16\times3\times8\times9$ tensor. Here, all modes have been split into two modes, except for the mode corresponding to the three color channels which is left as it is. This is done via `X = reshape(X,8,16,8,16,3,8,9)` in Matlab.

Detailed results for the experiments on the reshaped tensors are shown in Table S1.

Table S1: Decomposition results for reshaped real datasets with target rank $R = 10$. The SVD-based methods cannot handle any of these reshaped datasets since they require $R_0R_1 \leq I_1$. The ALS-based methods all fail on the reshaped Park Bench dataset due to Matlab running out of memory. Time is in seconds.

	Pavia Uni.		DC Mall		Park Bench		Tabby Cat		Red Truck	
Method	Error	Time	Error	Time	Error	Time	Error	Time	Error	Time
TR-ALS	0.28	1372.2	0.29	3947.7	✗	✗	0.15	6629.0	0.25	546.3
rTR-ALS	0.31	944.7	0.31	2575.0	✗	✗	0.17	3416.4	0.26	423.5
TR-ALS-S. (proposal)	0.31	3.4	0.31	5.8	✗	✗	0.17	2.3	0.27	5.8
TR-SVD	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
TR-SVD-Rand	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

S4.4 Rapid Feature Extraction for Classification

The ALS-based algorithms all run until the change in relative error is below 1e−4. The embedding dimension K for rTR-ALS is 20 for the first and second modes, and 200 for the fourth mode. No compression is applied to the third mode since it is already so small. TR-ALS-Sampled uses the sketch rate $J = 1000$.

References

Salman Ahmadi-Asl, Andrzej Cichocki, Anh Huy Phan, Maame G. Asante-Mensah, Farid Mousavi, Ivan Oseledets, and Toshihisa Tanaka. Randomized algorithms for fast computation of low-rank tensor ring model. *Machine Learning: Science and Technology*, 2020.

Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

Petros Drineas, Michael W. Mahoney, Shan Muthukrishnan, and Tamás Sarlós. Faster least squares approximation. *Numerische Mathematik*, 117(2):219–249, 2011.

Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, fourth edition, 2013. ISBN 978-1-4214-0794-4.

Oscar Mickelin and Sertac Karaman. On algorithms for and computing with the tensor ring decomposition. *Numerical Linear Algebra with Applications*, 27(3):e2289, 2020.

Charles F. Van Loan. The ubiquitous Kronecker product. *Journal of Computational and Applied Mathematics*, 123(1):85–100, November 2000. ISSN 0377-0427. doi: 10.1016/S0377-0427(00)00393-9.

David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1-2):1–157, 2014.

Longhao Yuan, Chao Li, Jianting Cao, and Qibin Zhao. Randomized tensor ring decomposition and its application to large-scale data reconstruction. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2127–2131. IEEE, 2019.