Parallel Algorithms for Tensor Train Arithmetic

Hussam Al Daas[1], Grey Ballard[2], Peter Benner[3]

SIAM ALA 2021

21.05.2021

[1] Rutherford Appleton Laboratory
[2] Wake Forest University
[3] MPI Magdeburg

Outline

Outline

Motivation

Low rank tensor techniques can deal with high-order data efficiently in many applications such as

- molecular simulations
- parameter dependent PDEs
- uncertainty quantification
- classification
- etc

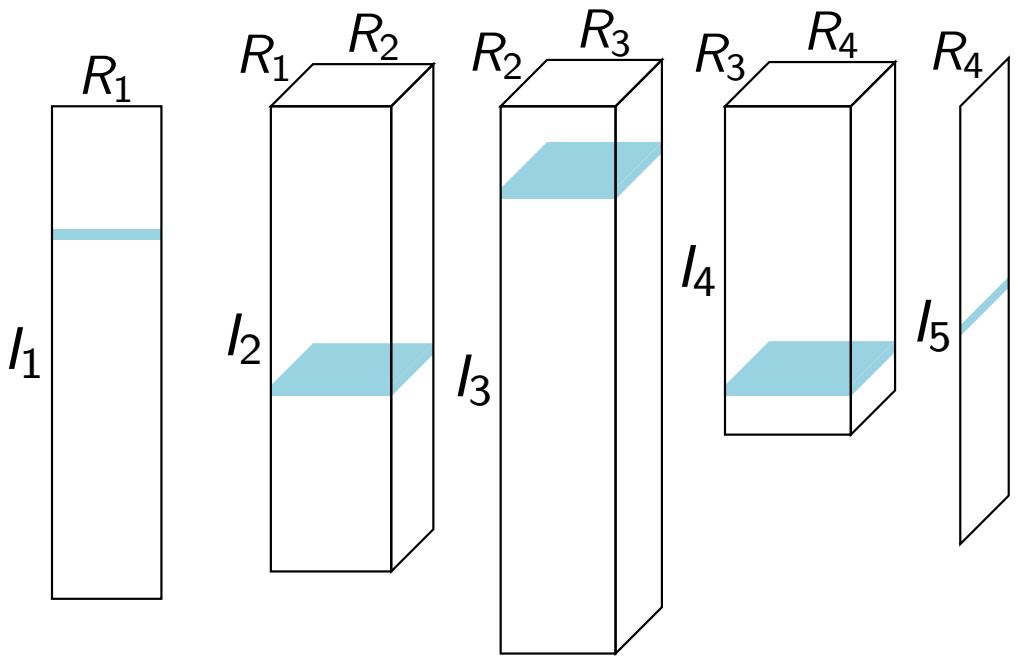
Applications

In molecular simulations (NMR), and UQ (N-point correlations)

- $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$.
- Large number of modes (N can be of order 100).
- Mode size $I_i \approx 10^6$.

Outline

Tensor Train Notation



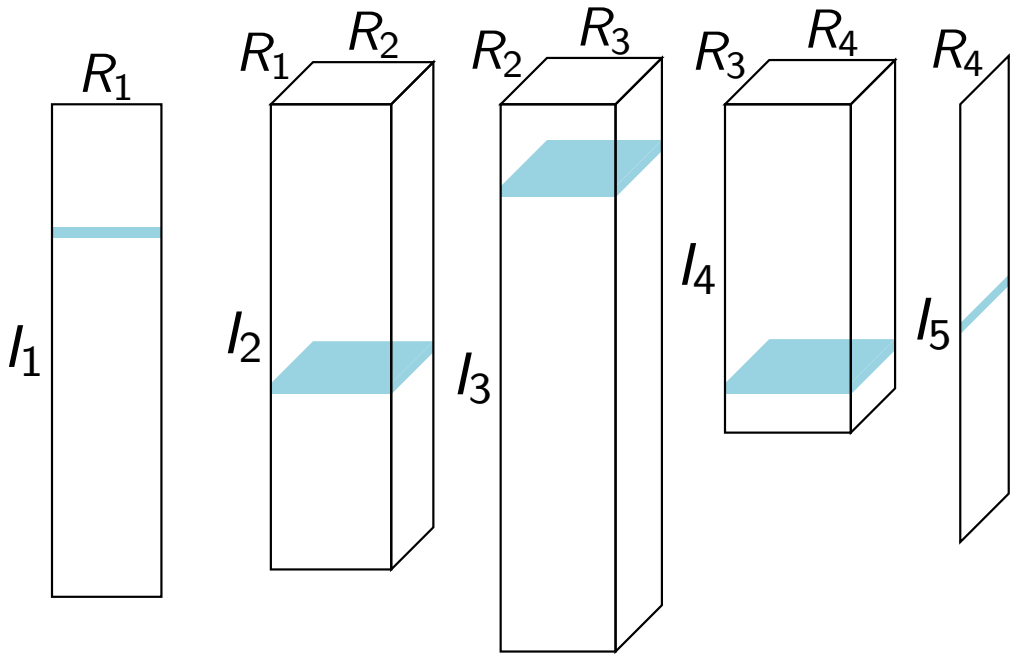
$$\mathcal{X} \approx \{\mathcal{T}_{\mathcal{X},k}\}, \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4 \times I_5}$$

$$\mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$$

are *TT cores*

$$x_{i_1,\ldots,i_N} \approx \sum_{\alpha=1}^{R_2} \sum_{\beta=1}^{R_3} \sum_{\gamma=1}^{R_4} \sum_{\delta=1}^{R_5} \mathcal{T}_{\mathcal{X},1}(i_1,\alpha)\mathcal{T}_{\mathcal{X},2}(\alpha,i_2,\beta)\mathcal{T}_{\mathcal{X},3}(\beta,i_3,\gamma)\mathcal{T}_{\mathcal{X},4}(\gamma,i_4,\delta)\mathcal{T}_{\mathcal{X},5}(\delta,i_5)$$

Tensor Train Notation



$$\mathcal{X} \approx \{\mathcal{T}_{\mathcal{X},k}\}, \mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4 \times I_5}$$

$$\mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$$

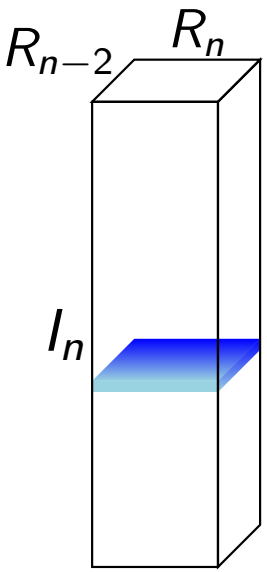
are *TT cores*

$$x_{i_1,\ldots,i_N} \approx$$

$$\sum_{\alpha=1}^{R_2} \sum_{\beta=1}^{R_3} \sum_{\gamma=1}^{R_4} \sum_{\delta=1}^{R_5} \mathcal{T}_{\mathcal{X},1}(i_1, \alpha) \mathcal{T}_{\mathcal{X},2}(\alpha, i_2, \beta) \mathcal{T}_{\mathcal{X},3}(\beta, i_3, \gamma) \mathcal{T}_{\mathcal{X},4}(\gamma, i_4, \delta) \mathcal{T}_{\mathcal{X},5}(\delta, i_5)$$

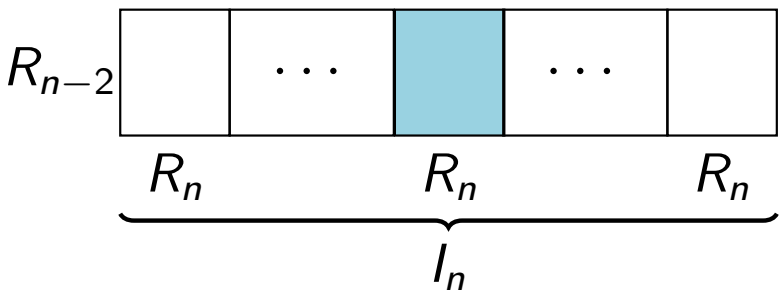
$$x_{i_1,\ldots,i_N} \approx \mathbf{T}_{\mathcal{X},1}(i_1) \cdots \mathbf{T}_{\mathcal{X},N}(i_N)$$

Horizontal and Vertical Unfolding

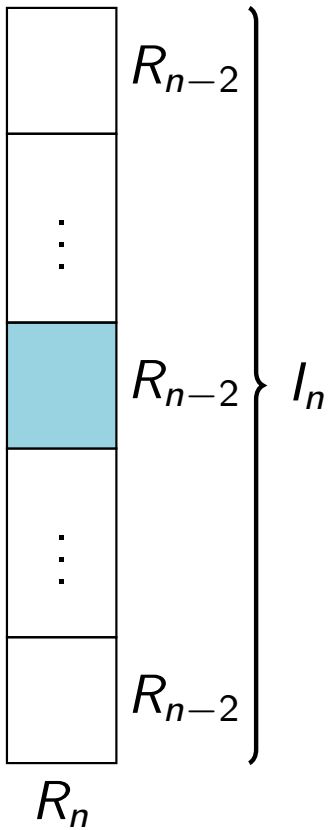


$\mathcal{T}_{\mathfrak{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$

are *TT cores*



$\mathcal{H}(\mathcal{T}_{\mathfrak{X},k}) \in \mathbb{R}^{R_k \times I_k R_{k+1}}$



$\mathcal{V}(\mathcal{T}_{\mathfrak{X},k}) \in \mathbb{R}^{R_k I_k \times R_{k+1}}$

Algebraic Operations

Scaling:
$\mathcal{X} = \{\mathcal{T}_{\mathcal{X},k}\},\ \mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}},\ \mathcal{Z} = \lambda\mathcal{X}$
How to express $\mathcal{Z}$ in TT format?

Algebraic Operations

Scaling:
$\mathcal{X} = \{\mathcal{T}_{\mathcal{X},k}\},\ \mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}},\ \mathcal{Z} = \lambda\mathcal{X}$
How to express $\mathcal{Z}$ in TT format?

$$z_{i_1,\ldots,i_N} = (\lambda \mathbf{T}_{\mathcal{X},1}(i_1)) \cdots \mathbf{T}_{\mathcal{X},k}(i_k) \cdots \mathbf{T}_{\mathcal{X},N}(i_N),$$

then, $\mathcal{T}_{\mathcal{Z},1} = \lambda\mathcal{T}_{\mathcal{X},1}$, $\mathcal{T}_{\mathcal{Z},k} = \mathcal{T}_{\mathcal{X},k}$, for $k \in [\![2; N]\!]$

Algebraic Operations

Summation:

$\mathcal{X} = \{\mathcal{T}_{\mathcal{X},k}\},\ \mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$,

$\mathcal{Y} = \{\mathcal{T}_{\mathcal{Y},k}\},\ \mathcal{T}_{\mathcal{Y},k} \in \mathbb{R}^{L_k \times I_k \times L_{k+1}}$, $\mathcal{Z} = \mathcal{X} + \mathcal{Y}$

How to express $\mathcal{Z}$ in TT format?

Algebraic Operations

Summation:

$\mathcal{X} = \{\mathcal{T}_{\mathcal{X},k}\},\ \mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$,

$\mathcal{Y} = \{\mathcal{T}_{\mathcal{Y},k}\},\ \mathcal{T}_{\mathcal{Y},k} \in \mathbb{R}^{L_k \times I_k \times L_{k+1}}$, $\mathcal{Z} = \mathcal{X} + \mathcal{Y}$

How to express $\mathcal{Z}$ in TT format?

$$z_{i_1,\ldots,i_N} = \mathbf{T}_{\mathcal{X},1}(i_1) \cdots \mathbf{T}_{\mathcal{X},k}(i_k) \cdots \mathbf{T}_{\mathcal{X},N}(i_N) + \mathbf{T}_{\mathcal{Y},1}(i_1) \cdots \mathbf{T}_{\mathcal{Y},k}(i_k) \cdots \mathbf{T}_{\mathcal{Y},N}(i_N),$$

Algebraic Operations

Summation:

$\mathcal{X} = \{\mathcal{T}_{\mathcal{X},k}\},\ \mathcal{T}_{\mathcal{X},k} \in \mathbb{R}^{R_k \times I_k \times R_{k+1}}$,

$\mathcal{Y} = \{\mathcal{T}_{\mathcal{Y},k}\},\ \mathcal{T}_{\mathcal{Y},k} \in \mathbb{R}^{L_k \times I_k \times L_{k+1}}$, $\mathcal{Z} = \mathcal{X} + \mathcal{Y}$

How to express $\mathcal{Z}$ in TT format?

$$z_{i_1,\ldots,i_N} = \mathbf{T}_{\mathcal{X},1}(i_1) \cdots \mathbf{T}_{\mathcal{X},k}(i_k) \cdots \mathbf{T}_{\mathcal{X},N}(i_N) + \mathbf{T}_{\mathcal{Y},1}(i_1) \cdots \mathbf{T}_{\mathcal{Y},k}(i_k) \cdots \mathbf{T}_{\mathcal{Y},N}(i_N),$$

$$z_{i_1,\ldots,i_N} = \begin{pmatrix} \mathbf{T}_{\mathcal{X},1}(i_1) & \mathbf{T}_{\mathcal{Y},1}(i_1) \end{pmatrix} \cdots \begin{pmatrix} \mathbf{T}_{\mathcal{X},k}(i_k) & \\ & \mathbf{T}_{\mathcal{Y},k}(i_k) \end{pmatrix} \cdots \begin{pmatrix} \mathbf{T}_{\mathcal{X},N}(i_N) \\ \mathbf{T}_{\mathcal{Y},N}(i_N) \end{pmatrix}$$

TT-ranks of $\mathcal{Z}$ are formal and can be compressed, e.g., $\mathcal{X} = \mathcal{Y}$, TT-ranks of $\mathcal{Z}$ and $\mathcal{X}$ are equal!

TT Rounding

Given a tensor in TT format, want to compress the ranks

- algebraic operations on TT formats over-extend ranks
- recompression subject to some error threshold

Rounding done in two phases: orthogonalization and truncation

- orthogonalization done core-by-core in sequence
- truncation is done core-by-core (opposite direction)

TT Rounding

Given a tensor in TT format, want to compress the ranks

- algebraic operations on TT formats over-extend ranks
- recompression subject to some error threshold

Rounding done in two phases: orthogonalization and truncation

- orthogonalization done core-by-core in sequence
- truncation is done core-by-core (opposite direction)

for $n = N$ down to 2 **do**
  $L \cdot \mathcal{H}(\mathcal{Q}) = \mathcal{H}(\mathcal{T}_{\mathcal{X},n})$ ▷ LQ factorization of short-fat matrix
  $\mathcal{T}_{\mathcal{X},n} = \mathcal{Q}$
  $\mathcal{V}(\mathcal{T}_{\mathcal{X},n-1}) = \mathcal{V}(\mathcal{T}_{\mathcal{X},n-1})L$
for $n = 1$ to $N-1$ **do**
  $\mathcal{V}(\mathcal{U}_n) \cdot \Sigma_n \cdot V_n^T = \mathcal{V}(\mathcal{T}_{\mathcal{X},n})$ ▷ truncated SVD of tall-skinny matrix
  $\mathcal{T}_{\mathcal{X},n} = \mathcal{U}_n$
  $\mathcal{H}(\mathcal{T}_{\mathcal{X},n+1}) = \Sigma_n V_n^\top \mathcal{H}(\mathcal{T}_{\mathcal{X},n+1})$

TT Rounding – Matrix Case

$$\mathbf{A} = \mathbf{X}\mathbf{Y}^T$$

TT Rounding – Matrix Case

$$\mathbf{A} = \mathbf{X}\mathbf{Y}^T$$

Right orthogonalization:
$\mathbf{Y} = \mathbf{QR}$ QR factorization

TT Rounding – Matrix Case

$$\mathbf{A} = \mathbf{X}\mathbf{Y}^T$$

Right orthogonalization:
$\mathbf{Y} = \mathbf{Q}\mathbf{R}$ QR factorization

$$\mathbf{A} = (\mathbf{X}\mathbf{R}^T)\mathbf{Q}^T,$$

TT Rounding – Matrix Case

$$\mathbf{A} = \mathbf{X}\mathbf{Y}^T$$

Right orthogonalization:
$\mathbf{Y} = \mathbf{Q}\mathbf{R}$ QR factorization

$$\mathbf{A} = (\mathbf{X}\mathbf{R}^T)\mathbf{Q}^T,$$

Left to right truncation:
$\mathbf{X}\mathbf{R}^T = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T + \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T$ is SVD (ε truncation).

TT Rounding – Matrix Case

$$\mathbf{A} = \mathbf{X}\mathbf{Y}^T$$

Right orthogonalization:
$\mathbf{Y} = \mathbf{Q}\mathbf{R}$ QR factorization

$$\mathbf{A} = (\mathbf{X}\mathbf{R}^T)\mathbf{Q}^T,$$

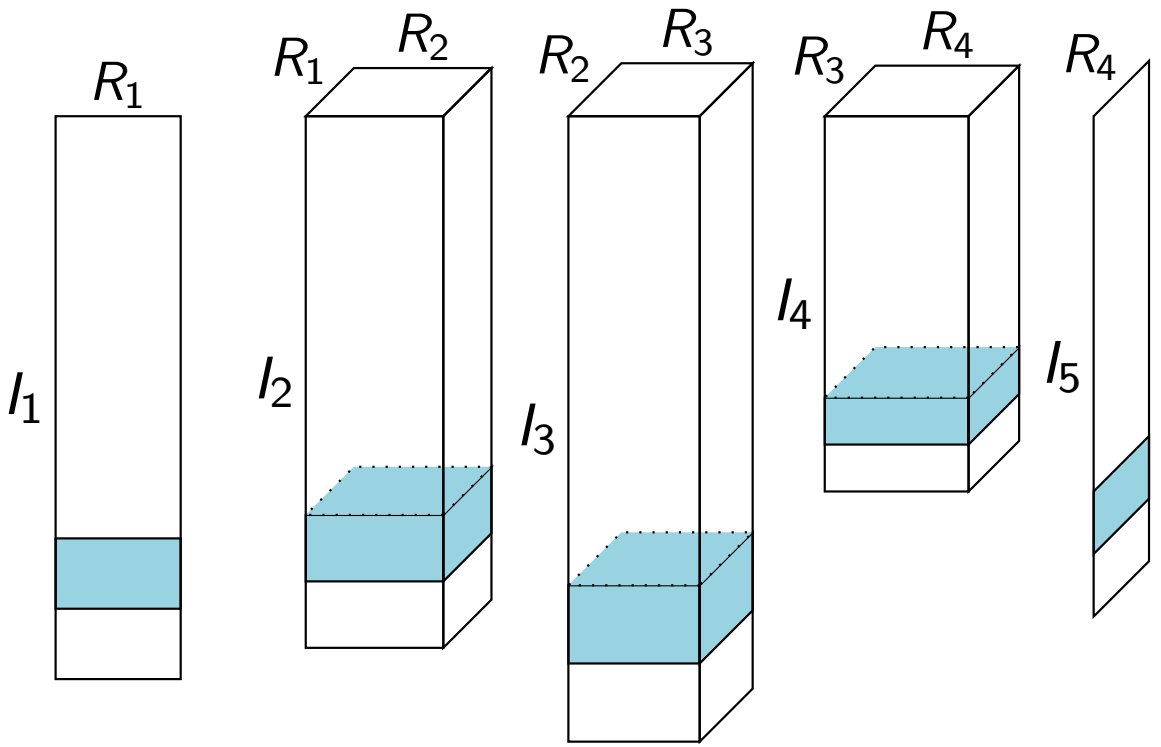
Left to right truncation:
$\mathbf{X}\mathbf{R}^T = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T + \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T$ is SVD (ε truncation).

$$\begin{aligned}\mathbf{A} &= \left(\mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T + \mathbf{U}_2\mathbf{\Sigma}_2\mathbf{V}_2^T\right)\mathbf{Q}^T, \\ &\approx \mathbf{U}_1\left(\mathbf{Q}\mathbf{V}_1\mathbf{\Sigma}_1\right)^T,\end{aligned}$$

More in [Oseledets, 2011]

Outline

Parallel Distribution



- Each core distributed across all P processors
- Local nth core dimensions are $R_n \times \frac{I_n}{P} \times R_{n+1}$
- Vertical and horizontal unfoldings are 1D-distributed

Parallel Arithmetic

Embarassingly parallel:

- Scaling
- Summation

Parallel Arithmetic

Embarassingly parallel:

- Scaling
- Summation

Classic parallel:

- Matrix-vector $((\mathbf{A}_1 \otimes \cdots \otimes \mathbf{A}_N)\,\mathcal{X})$

Parallel Arithmetic

Embarassingly parallel:

- Scaling
- Summation

Classic parallel:

- Matrix-vector $((\mathbf{A}_1 \otimes \cdots \otimes \mathbf{A}_N)\mathcal{X})$

Sequential in modes:

- Inner product
- **Rounding**

More in [HA, G. Ballard, P. Benner, 2020]

Parallel TT Rounding Algorithm

function $\{\mathcal{T}_{\mathcal{Y},n}^{(p)}\} = \textsc{Par-TT-Rounding}(\{\mathcal{T}_{\mathcal{X},n}^{(p)}\})$

  for $n = N$ down to 2 **do**

    $[\{Y_p^{(\ell)}\}_n, R_n] = \textbf{TSQR}(\mathcal{H}(\mathcal{T}_{\mathcal{X},n}^{(p)})^T)$ ▷ QR factorization

    $R^{(p)} = \textbf{Broadcast}(R_n, \text{root})$ ▷ Broadcast R to all procs

    $\mathcal{V}(\mathcal{T}_{\mathcal{X},n-1}^{(p)}) = \textbf{Mult}(\mathcal{V}(\mathcal{T}_{\mathcal{X},n-1}^{(p)}), R^{(p)})$ ▷ Apply R to previous core

  $\mathcal{Y} = \mathcal{X}$

  for $n = 1$ to $N-1$ **do**

    $[\{Y_p^{(\ell)}\}_n, R_n] = \textbf{TSQR}(\mathcal{V}(\mathcal{T}_{\mathcal{Y},n}^{(p)}))$ ▷ QR factorization

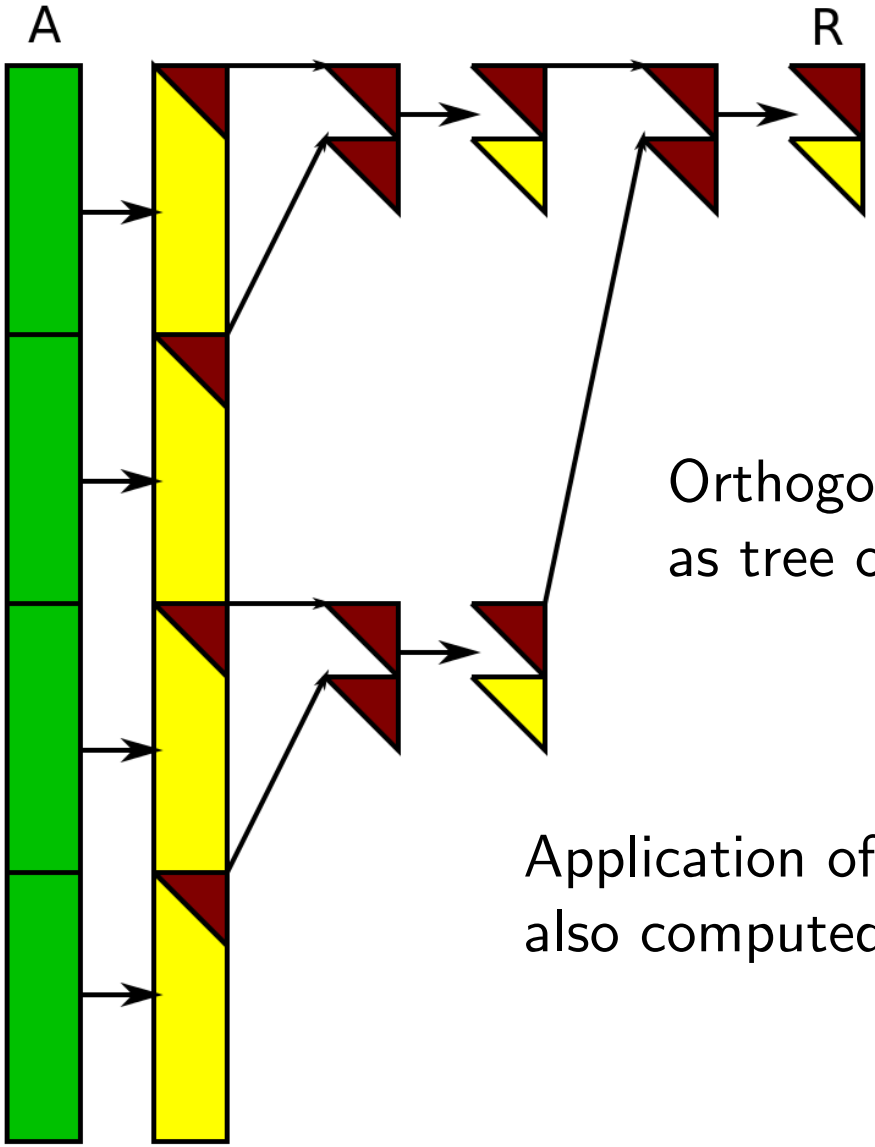
    if $p = \text{root}$ **then**

      $[\hat{U}_R, \hat{\Sigma}, \hat{V}] = \textbf{tSVD}(R_n)$ ▷ Truncated SVD of R

    $\mathcal{V}(\mathcal{T}_{\mathcal{Y},n}^{(p)}) = \textbf{TSQR-Apply-Q}(\{Y_p^{(\ell)}\}_n, \hat{U}_R)$ ▷ Form explicit $\hat{U}$

    $\mathcal{H}(\mathcal{T}_{\mathcal{X},n+1}^{(p)})^T = \textbf{TSQR-Apply-Q}(\{Y_p^{(\ell)}\}_{n+1}, \hat{V})$ ▷ Apply $\hat{V}$ to next core

Tall-Skinny QR (TSQR) Algorithm [Demmel et al., 2012]



Key benefit of TSQR:
one parallel reduction

Orthogonal factor stored implicitly
as tree of Householder vectors

Application of implicit orthogonal factor
also computed via tree (backwards)

Cost Analysis of TT-Rounding

Assuming $I_n = I$ and $R_n = R$, and R is reduced by factor of 2... computational cost is

$$7\frac{NIR^3}{P} + O(NR^3 \log P) \text{ flops}$$

communication cost is

$$O(NR^2 \log P) \text{ words and } O(N \log P) \text{ messages}$$

Cost Analysis of TT-Rounding

Assuming $I_n = I$ and $R_n = R$, and R is reduced by factor of 2... computational cost is

$$7\frac{NIR^3}{P} + O(NR^3 \log P) \text{ flops}$$

communication cost is

$$O(NR^2 \log P) \text{ words and } O(N \log P) \text{ messages}$$

- costs are linear not exponential in N (property of TT)
- communication cost is independent of I (good)
- communication cost increases slightly with P and N (bad)

Numerical Experiments – **MPI-ATTAC**

- Programming language: C
- Dependencies: MKL, (MPI for parallel computation)
- Shared-memory tests run on COBRA (MPCDF), MKL version 2019 and MATLAB 2019
- Parallel tests run on COBRA (MPCDF), MKL version 2019, Intel-MPI 2019
- 40 cores per node

Numerical Experiments – **MPI-ATTAC**

Model	# Modes	Dimensions	Ranks	Memory
1	50	$2K \times \cdots \times 2K$	50	2 GB
2	16	$100M \times 50K \times \cdots \times 50K \times 1M$	30	28 GB
3	30	$2M \times \cdots \times 2M$	30	385 GB

Table: Synthetic TT models used for performance experiments. In each case the formal ranks are all the same and are cut in half by the TT rounding procedure.

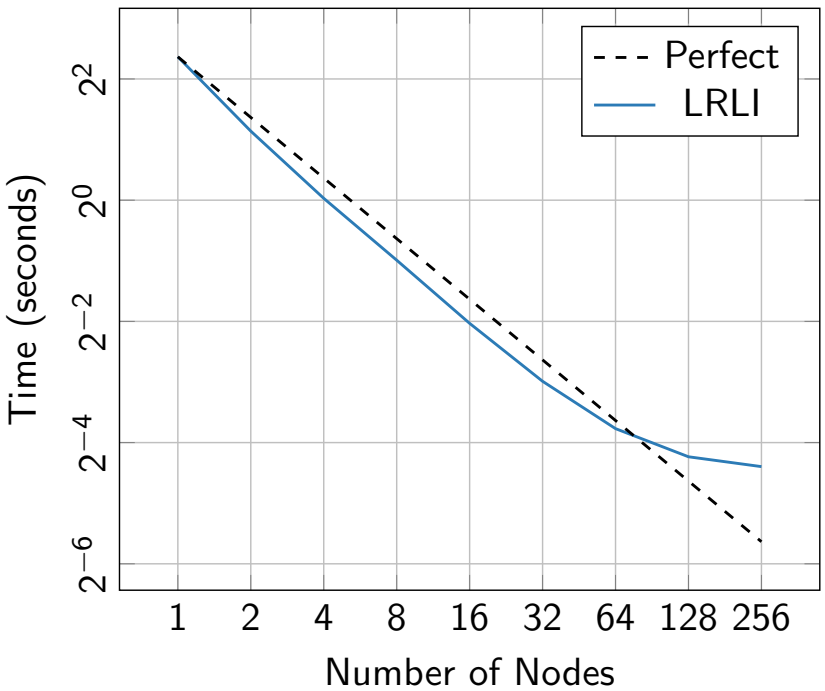
Numerical Experiments – **MPI-ATTAC**

Comparison: TT-Toolbox (MATLAB 2019) vs MPI-ATTAC (shared-memory).

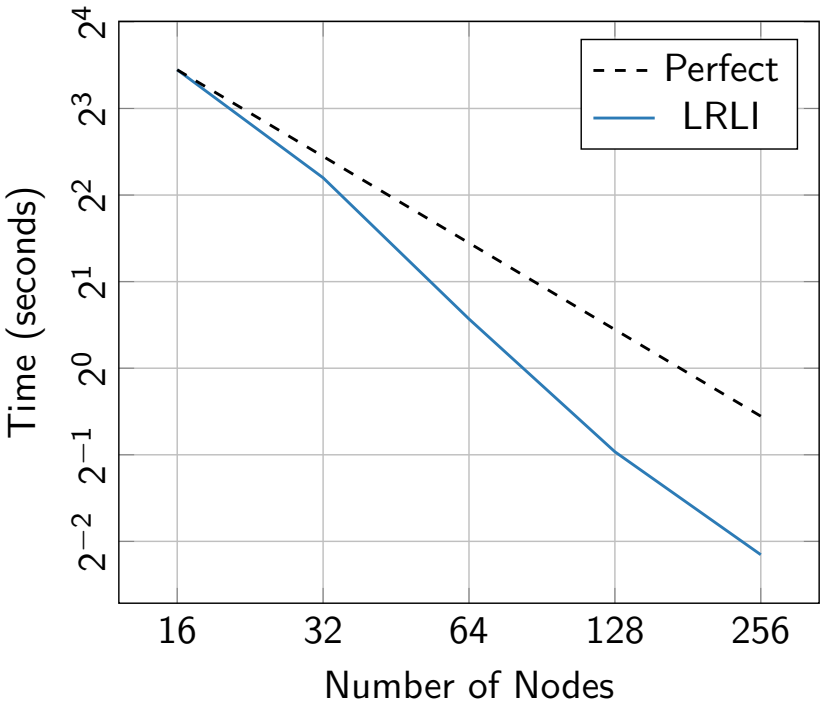
	1 core	20 cores	Par. Speedup	40 cores	Par. Speedup
TT-Toolbox	15.68	8.34	**1.9**×	8.752	**1.8**×
MPI-ATTAC	9.2	0.44	**20.9**×	0.27	**33.9**×
Speedup	**1.7**×	**18.95**×		**32.2**×	

Table: Single-node performance results on TT Model 1 and comparison with the MATLAB TT-Toolbox.

Numerical Experiments – **MPI-ATTAC**



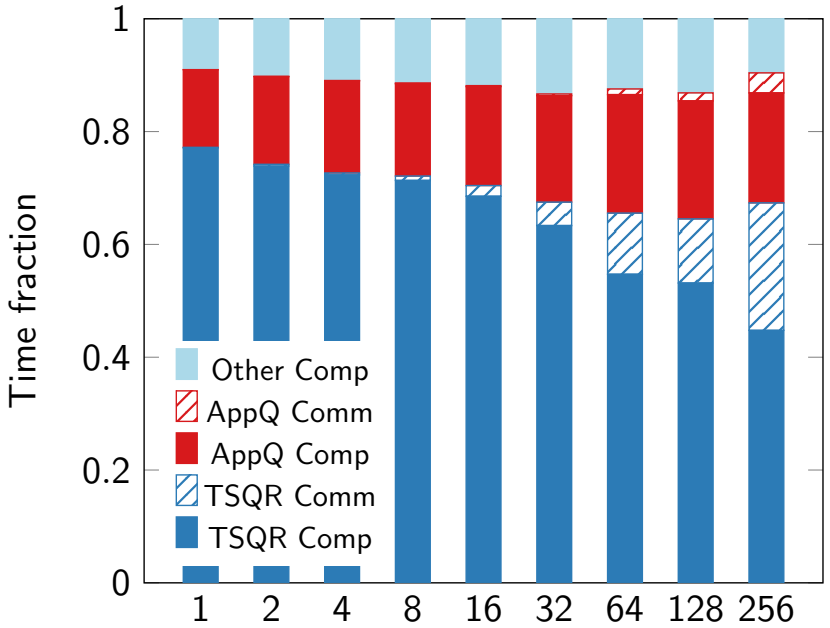
(a) Parallel scaling for Model 2



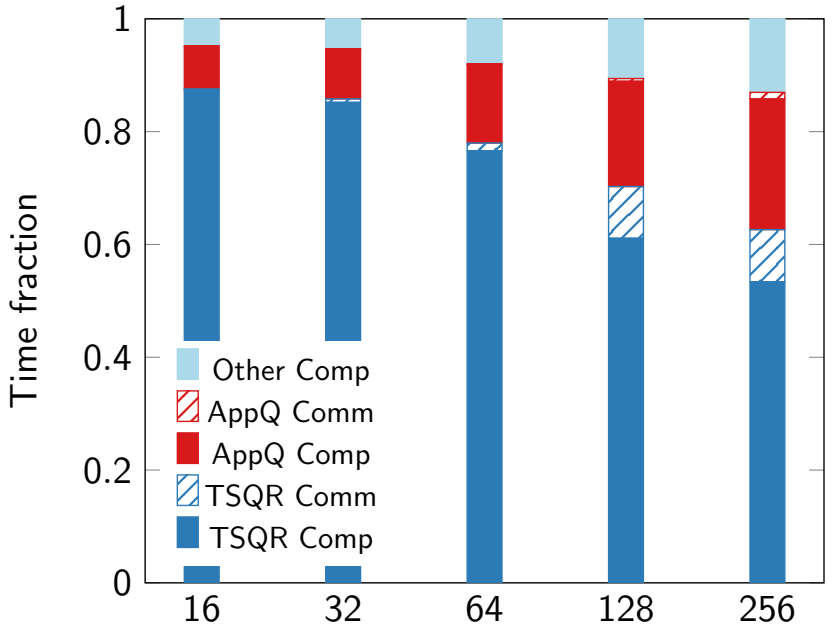
(b) Parallel scaling for Model 3

Figure: Time breakdown and and parallel scaling of LRLI variant of TT rounding.

Performance Breakdown – **MPI-ATTAC**



(a) Time breakdown for Model 2



(b) Time breakdown for Model 3

- communication cost increases with processors
- $O(NR^2 \log P)$ words and $O(N \log P)$ messages

Outline

Summary

- Algorithms for Tensor Train Arithmetic and Computation
- Scalable MPI implementation (BSD-2Clauses)

Summary

- Algorithms for Tensor Train Arithmetic and Computation
- Scalable MPI implementation (BSD-2Clauses)

Summary

- Algorithms for Tensor Train Arithmetic and Computation
- Scalable MPI implementation (BSD-2Clauses)

Perspective

- Linear solvers (TT-GMRES, AMEn)
- Randomized rounding

Gitlab Repo:
https://gitlab.com/aldaas/mpi_attac

Bibliographies


Demmel, J., Grigori, L., Hoemmen, M., and Langou, J. (2012).
Communication-optimal parallel and sequential QR and LU factorizations.
SIAM Journal on Scientific Computing, 34(1):A206–A239.


HA, G. Ballard, P. Benner (2020).
Parallel algorithms for tensor train arithmetic.


Oseledets, I. V. (2011).
Tensor-train decomposition.
SIAM J. Sci. Comput., 33(5):2295–2317.