SCATE: SCAlable Tensor dEcompositions for data analytics

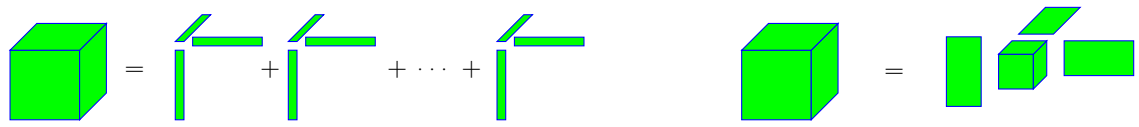
CES46 > Axe E.05 > Calcul haute performance, Modèles numériques, simulation, applications

Coordinated by: Suraj Kumar AAPG 2026 / JCJC 48 months

1 Context, positioning and objectives

More than 403 million terabytes of data is produced every day [1]. Analyzing such massive data is essential to extract valuable insights and make informed decisions. To accomplish this, it is crucial to design parallel and scalable approaches that efficiently utilize modern computing systems. Tensors are multi-dimensional arrays and used to store data in several domains [6], e.g., data mining, neuroscience and computer vision. Tensor decompositions help to identify inherent structure of data, achieve data compression and enable various ways of data analysis. Tensors are also used as operators to solve problems in applied mathematics, chemistry, and machine learning [6, 7]. Working with tensors is challenging as the computational effort and memory requirements grow exponentially with the number of dimensions. It is therefore necessary to exploit patterns in the tensor data. Leveraging the low-dimensional structure of high-dimensional data is a powerful approach in this context. Most tensor decompositions represent data in low-dimensional structures for efficient analysis and manipulation [6].

CP (also known as Canonical Polyadic or CANDECOMP or PARAFAC) and Tucker are the widely used tensor decompositions in the literature for data analytics. Both decompositions can be viewed as high order generalization of Singular Value Decomposition (SVD). CP decomposition is used to understand the latent components of the data, while Tucker decomposition is considered to be more appropriate for compression and multi-modal data analysis [6].



(a) CP decomposition. (b) Tucker decomposition.

Figure 1: CP and Tucker decompositions of a 3-dimensional tensor.

Figure 1 shows visual representations of both tensor decompositions for a 3-dimensional tensor. CP decomposition represents a tensor with the sum of rank one tensors. Whereas, Tucker decomposition represents a tensor with multiple factor matrices and a much smaller core tensor with the same number of dimensions as the original. As mentioned earlier, tensors are used to store data in several domains. For example, in neuroscience, a single trial to observe behaviors of neurons over time can be stored in a matrix and data of multiple trials in a 3-dimensional tensor. CP decomposition is a popular method to analyze such neuroscience data [9]. Similarly, a single experiment in combustion simulations or electron microscopy produces terabytes of data. For instance, tracking 64 variables on a 512×512×512 three-dimensional spatial grid for 128 time steps in a simulation requires to store 2^{40} entries. It is very difficult to transfer and analyze data of such experiments. Tucker decomposition is an appealing practice to compress such data before different types of analysis can be performed [3].

The data generated by many experiments in areas such as neuroscience, combustion simulation and hyperspectal imaging is so large that performing a CP or Tucker decomposition becomes infeasible without parallel computation. Therefore, it is important to focus on parallel algorithms for both decompositions. Recent advances in computing architectures have led to the development of parallel high performance computing (HPC) systems equipped with a large number of computational units – El Capitan at LLNL and Adastra at CINES, for example. Although computational performance has improved rapidly over the past few decades, progress in data movement rates has been comparatively limited. The current HPC systems face bottleneck due to the large volume of data transfers [5]. Thus

it is crucial to take communication costs into account while designing parallel algorithms. In doing so, one also reduces energy consumption – a must in today's world facing growing environmental and sustainability challenges. GPUs provide increased processing capabilities and superior energy efficiency compared to CPUs, making them a crucial component of HPC systems over the past decade. Developing CPU-only approaches may overlook a significant portion of the computational power in such systems, leading to poor efficiency. Therefore, it is important to focus on both types of processing units while developing algorithms for these systems. The SCATE project addresses parallelization, communication costs and scalability of tensor decompositions on the modern HPC systems. More precisely, the high level aim of the project is to devise parallel and communication optimal tensor decomposition algorithms which scale well on both homogeneous and heterogeneous systems. Establishing communication lower bounds is also an important part of the project as it helps one to devise efficient parallel algorithms.

Most existing tensor decomposition algorithms work with matrix (2-dimensional) representations of tensors at each step and rely on the parallelization of matrix operations. This approach neglects multi-dimensional properties of tensors and may not perform the computation efficiently. We focus on improving the performance of Tucker and CP decomposition algorithms by taking multi-dimensional properties of tensors into account.

Algorithms to compute Tucker decompositions: Truncated higher-order SVD (HOSVD) is one of the popular algorithms for computing a Tucker decomposition, thanks to its quasi-optimal numerical approximation [8]. In this algorithm, the core tensor is obtained using the full-format tensor and multiple tall-and-skinny factor matrices. For a 3-dimensional tensor $\mathcal{X}$, computation of the core tensor $\mathcal{Y}$ can be expressed in tensor notation as $\mathcal{Y} = \mathcal{X} \times_1 \mathsf{A}^{(1)^\mathsf{T}} \times_2 \mathsf{A}^{(2)^\mathsf{T}} \times_3 \mathsf{A}^{(3)^\mathsf{T}}$, where $\mathsf{A}^{(k)}$ is a tall-skinny factor matrix corresponding to mode k, and $\times_k$ denotes the tensor-times-matrix (TTM) operation in the kth mode [6]. This collective operation is known as the Multi-TTM computation [2] and is one of the main bottlenecks of the HOSVD algorithm. One approach to perform this computation is *in sequence*, i.e., $\mathcal{Y} = ((\mathcal{X} \times_1 \mathsf{A}^{(1)^\mathsf{T}}) \times_2 \mathsf{A}^{(2)^\mathsf{T}}) \times_3 \mathsf{A}^{(3)^\mathsf{T}}$. Another approach is to work with the inputs *all at once*, i.e., $\mathcal{Y}_{ijk} = \sum_{lmp} \mathcal{X}_{lmp} \cdot \mathsf{A}^{(1)}_{li} \cdot \mathsf{A}^{(2)}_{mj} \cdot \mathsf{A}^{(3)}_{pk}$. Daas et al. [2] show that the latter approach, which takes multi-dimensional properties of tensors into account, reduces communication significantly compared to the sequence approach for small and moderate number of processors. However there is not any clear winner for all settings. One can achieve better gains with a hybrid method that combines the strengths of both approaches. This perspective is not explored in [2]. Furthermore, the amount of operations for the Multi-TTM computation depends on how factor matrices are processed with the input tensor. It is also important to model the computation-communication tradeoff in the combined hybrid method such that the overall completion time of the Multi-TTM is minimized.

In addition, HOSVD algorithm also computes SVD for each mode to obtain the corresponding factor matrix. Optimal data distributions to perform SVDs and the Multi-TTM computation may not be compatible and may require to perform extra communication. As SVD is expensive, it is also important to consider alternatives, such as randomized SVD, for computing factor matrices. These issues must be addressed to achieve optimal completion time for the entire HOSVD algorithm.

Algorithms to compute CP decompositions: Computing a CP decomposition involves solving a non-linear optimization problem to minimize the approximation error. The workhorse algorithm to compute this decomposition uses an alternating least squares approach. This works in multiple iterations. For a d-dimensional decomposition, in each iteration, d matricized-tensor times Khatri-Rao product (MTTKRP) computations are performed. In this computation, a matrix representation of the original tensor is multiplied with the Khatri-Rao product of $d-1$ factor matrices. This is the bottleneck computation of the algorithm. Ballard et al. [4] show that the working with all the inputs at once without forming intermediates reduces communication costs significantly compared to the procedure where the Khatri-Rao product is computed first and the intermediate output is multiplied with a matricized rep-

resentation of the original tensor. Certain operations can be reused among all MTTKRPs of a single iteration. But, their approach restricts the reuse of operations as intermediates are not formed. The amount of operations in the MTTKRP computation also depends on how factor matrices are processed with the original tensor. It is important to study tradeoff among communications, computations and reuse of intermediate results, and design a method that achieves optimal completion time.

Objectives: The goal of the SCATE project is to develop Tucker and CP decomposition algorithms that scale efficiently on modern computing systems. The project first studies tradeoff among computations, communications and data reuse for the existing algorithms and proposes new methods whose completion times are optimal for homogeneous systems. After that, the proposed methods will be implemented and tested with real-world data-sets from neuroscience and combustion simulations on more than 5,000 cores and/or 512 nodes of GENCI computers. This implementation will also be used to accelerate hyperspectral data analysis in astrophysics. The project also extends our framework for heterogeneous systems composed of CPUs and GPUs.

With the help of other researchers from tensor community, the project also plans to setup a public repository to store dense tensors of real applications. We will add all the used real-world tensors to that repository. We will also add code snippets of real applications that generate dense tensors, such as correlation functions.

Our approach: As mentioned earlier, the computations and communication costs of Multi-TTM and MTTKRP depend on how we perform them, i.e., how matrices are processed with the input tensor. If we do not select it carefully, their completion times may be very far from the optimal ones. Even when we know how we want to perform the computations, combining *all at once* and *in sequence* approaches, using one after the other, in a straightforward way will only work if data distributions of inputs and temporaries between both approaches are compatible. However we can not expect any guarantees with respect to the optimal communication/computation costs. Considering optimal data distributions for only one approach may induce significant data movements for the other approach. Using optimal data distributions for both approaches may not be compatible or require significant data movement during the transition of the approach. Therefore, in order to take benefits of both *all at once* and *in sequence* approaches, it is important to study all possible ways to combine them. More precisely, we will i) analyze communication costs for all possible ways to perform both computations, ii) investigate tradeoff among amount of operations, communications and data reuse, and iii) design communication optimal parallel algorithms.

We will study 3– and 4-dimensional computations in detail as the number of ways is limited. Subsequently, we will generalize our findings and develop a dynamic programming based approach for d-dimensional computations. After performing a detailed study of Multi-TTM and MTTKRP, we will design parallel algorithms to minimize the overall completion times of both decompositions.

We will implement our parallel algorithms for homogeneous systems using MPI . All the communications would be performed with collectives. This is a simple and popular strategy to minimize communication for many linear algebra computations [3]. We plan to determine processor grid dimensions analytically based on the components of communication lower bounds. We will equate the communication cost expressions of all matrices and tensors to their respective lower bounds, and determine solutions that satisfy all constraints, especially those involving dominant terms. Since MPI collectives rely on global synchronization, they may not efficiently utilize all resources of heterogeneous systems. Therefore, we will develop task-based parallel algorithms for such systems, enabling efficient execution with runtime systems such as StarPU.

Tensor Train is another popular tensor representation. This is very well suited to work with high dimensional tensors and used often in molecular and quantum simulations. However, this has limited use in data analytics as it is hard to interpret its different components. Therefore, we focus on CP and Tucker decompositions in this project and leave Tensor Train decomposition for future work.

Dissemination and new collaborations: We expect to publish the obtained results to the major venues (conferences/journals) of the field (SC, IPDPS, SIMAX, SISC). Furthermore we plan to organize a minisymposium on tensor computations at SIAM Conference on Computational Science and Engineering 2027. This will facilitate fruitful discussions with other tensor experts and domain specialists who work with tensors, and open avenues for potential collaborations. The algorithms developed in this project will be freely available, and the code will be published under a free and open-source license.

2 Consortium

Scientific coordinator: Suraj Kumar (PI) holds an Inria Starting Faculty Position since October 2022 in the ROMA Inria team, which is hosted at École Normale Supérieure de Lyon (ENS Lyon). He is an expert in tensor computations, communication avoiding algorithms and efficient utilization of resources on heterogeneous systems. He worked at Inria Paris from November 2019 to September 2022 as a postdoctoral researcher on the design of parallel and communication efficient algorithms for Multi-TTM computations and tensor-train decompositions. He was also a postdoctoral researcher at Pacific Northwest National Laboratory, USA from May 2018 to October 2019. He worked there on NWChemEx project, whose main goal was to run molecular simulations efficiently on exascale computers. He defended his thesis in April 2017 on Scheduling of Dense Linear Algebra Kernels on Heterogeneous Resources. The PI will dedicate 75% of his time on this project.

Team: The SCATE project will take place within the Inria ROMA team, in collaboration with other members of the team. In particular, L. Marchal (DR2, CNRS) will bring his expertise in memory-aware computations and B. Uçar (DR2, CNRS) is an expert in the design and implementation of efficient sparse tensor algorithms on large-scale systems. We ask for a PhD student, 24 months of a postdoctoral researcher and two master interns. The PhD student will work with the PI and B. Uçar (PhD co-advisor) on designing optimal algorithms for Tucker and CP decompositions on homogeneous systems. The master interns will study the numerical robustness of the proposed algorithms with the PI. The postdoctoral researcher will work with the PI and L. Marchal on extending the framework for heterogeneous systems, and will also help the PI to advise the PhD student. The PI has an active collaboration with G. Ballard of Wake Forest University, USA, who is an expert in the design of communication-efficient algorithms, that will be beneficial to the implementation of the project.

References

[1] "How much data is generated every day?" https://soax.com/research/data-generated-per-day, accessed: 2025-09-28.

[2] H. Al Daas, G. Ballard, L. Grigori, S. Kumar, and K. Rouse, "Communication lower bounds and optimal algorithms for multiple tensor-times-matrix computation," *SIAM Journal on Matrix Analysis and Applications*, vol. 45, no. 1, pp. 450–477, 2024. [Online]. Available: https://arxiv.org/abs/2207.10437

[3] W. Austin, G. Ballard, and T. G. Kolda, "Parallel tensor compression for large-scale scientific data," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2016, pp. 912–922.

[4] G. Ballard, N. Knight, and K. Rouse, "Communication lower bounds for matricized tensor times khatri-rao product," in *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2018, pp. 557–567.

[5] DOE ASCAC Subcommittee Report, "Top ten exascale research challenges," 2014. [Online]. Available: https://www.osti.gov/biblio/1222713

[6] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[7] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[8] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen, "A new truncation strategy for the higher-order singular value decomposition," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A1027–A1052, 2012.

[9] A. H. Williams, T. H. Kim, F. Wang, S. Vyas, S. I. Ryu, K. V. Shenoy, M. Schnitzer, T. G. Kolda, and S. Ganguli, "Unsupervised discovery of demixed, low-dimensional neural dynamics across multiple timescales through tensor component analysis," *Neuron*, vol. 98, no. 6, pp. 1099–1115.e8, 2018.