Design and Analysis of Tensor Decomposition Algorithms

Advisor: Pascal Koiran*

October 6, 2023

In tensor decomposition, the goal is to express an input tensor as the sum of a small number of rank one tensors. Assume for instance that T is a tensor of order 3, i.e., a 3-dimensional array. Then we want to write down T as

$$T = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i \tag{1}$$

where the u_i, v_i, w_i are (unknown) vectors. The smallest number of terms that can be achieved in such a decomposition is known as the tensor rank of T. For 2-dimensional arrays (i.e., matrices), this is just the matrix rank and in matrix notation we have $u \otimes v = uv^T$.

The study of tensor rank and tensor decomposition is well motivated from several points of view. In algebraic complexity, it is know that the complexity of bilinear problems such as matrix multiplication is characterized by the rank of an associated tensor. We can therefore prove lower bounds on the complexity of matrix multiplication and other bilinear problems by proving lower bounds on the rank of the corresponding tensor [6, 14]. Conversely, we can design faster algorithms by finding short decompositions of this tensor [1]. Tensor decomposition also occurs naturally in more applied areas such as machine learning [11], statistics or signal processing [3, 2]. In these areas the input tensor is usually not known a priori, but inferred from the data; and the components u_i, v_i, w_i have a special meaning in the application domain. For instance they can be used to reconstruct phylogenetic trees, or identify the topics occurring in a collection of documents [11]. Computing the tensor rank is NP-hard in the worst case [4, 12, 13]. In order to obtain efficient algorithms it is therefore necessary to make some assumptions on the input tensor. For instance, linear independence assumptions on the vectors occurring in the decomposition (1) have proved useful [11].

Goal of the internship. The student will work on tensor decomposition by methods of linear algebra such as (simultaneous) matrix diagonalisation. One prototypical example is the so-called "Jennrich algorithm" described in e.g. [11]. Such algorithms can be analyzed at varying levels of detail. In the literature, it is often assumed that objects such as eigenvalues and eigenvectors of matrices can be computed exactly. This is the case in the book [11]. Similar problems have been studied in the literature on arithmetic circuit reconstruction. As an introduction to this subject we recommend reading the reconstruction algorithm for sums of powers of linear forms in Section 5 of [5], which is particularly simple and elegant. In the language of tensors, the problem tackled in [5] is known as "symmetric tensor decompoosition" (meaning that $u_i = v_i = w_i$ in (1)). In that paper and in most of the literature on arithmetic circuit reconstruction, it is assumed that that polynomial roots can be computed exactly. Doing a very "high level" analysis as in [5, 11] is

*Project MC2, LIP laboratory, Ecole Normale Supérieure de Lyon. Email: firstname.lastname@ens-lyon.fr.

certainly fine, at least as a first step; but it doesn't yield a precise analysis of the complexity and accuracy of the algorithm. For a more precise analysis, at least two options are available.

(i) One can assume that the input tensor T is known exactly, and aim for a decision algorithm (does T admit an exact a decomposition of the required form?). This is the approach followed in [10, 7, 8].

(ii) One can try to find an approximate decomposition of T. This is the approach followed in [9] and it only requires the approximate computation of eigenvalues and eigenvectors.

The main goal of the internship is to design and analyze tensor decomposition algorithms with provable performance guarantees. More precisely, the student could work on the two following topics (or possibly just one of them):

1. *Undercomplete tensor decomposition.* Suppose for simplicity that T is a cubic tensor, of format $n \times n \times n$. In [9] we have only studied symmetric tensor decomposition, and only the case $r = n$ has been published so far. Therefore, one goal would be to get rid of the symmetry assumption and to allow for $r \leq n$ as well (the so-called *undercomplete* setting). We note that [9] contains a rather heavy analysis of errors due to finite precision arithmetic. For this internship it would make perfect sense to assume infinite precision arithmetic, which is easier to analyze. One would then analyze the complexity of the algorithm as a function of the desired precision on the output and of the other parameters of the problem (input size, condition number of the tensor).

2. *Overcomplete tensor decomposition.* This is the setting where $r > n$, and it is much less understood than the undercomplete setting. In fact, obtaining an effficient algorithm for the decomposition of "generic" tensors in that case is suggested as an open problem in [11]. The advisor has some concrete ideas to make progress on this problem, and the student could contribute to this work. Since this setting is relatively unexplored, it would make sense to stick at least initially to a rather "high level" style of analysis along the same lines as done in [5, 11] for the undercomplete setting.

Finally, depending on the student's interests we note that the algorithms could be implemented and tested.

Student's background. The student should of course be interested in the design and analysis of efficient algorithms. A good knowledge of linear algebra as taught for instance in *classes préparatoires* will also be very useful. For some parts of the above work plan, prior exposure to numerical linear algebra could be useful but it is certainly not a requirement. The relevant notions (numerical stability, condition numbers...) can be learnt during the internship.

References

[1] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic Complexity Theory*. Springer, 1997.

[2] Annie Cuyt, Ferre Knaepkens, and Wen-shin Lee. From exponential analysis to Padé approximation and tensor decomposition, in one and more dimensions. In *Computer Algebra in Scientific Computing: 20th International Workshop, CASC 2018.*, pages 116–130. Springer, 2018.

[3] Lieven De Lathauwer, Joséphine Castaing, and Jean-Franois Cardoso. Fourth-order cumulant-based blind identification of underdetermined mixtures. *IEEE Transactions on Signal Processing*, 55(6):2965–2973, 2007.

[4] Johan Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11(4):644–654, 1990.

[5] Neeraj Kayal. Efficient algorithms for some special cases of the polynomial equivalence problem. In *Symposium on Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, January 2011.

[6] Pascal Koiran. On tensor rank and commuting matrices. *arXiv preprint arXiv:2006.02374*, 2020.

[7] Pascal Koiran and Subhayan Saha. Black Box Absolute Reconstruction for Sums of Powers of Linear Forms. In *Proceedings FSTTCS 2022. arXiv:2110.05305.*

[8] Pascal Koiran and Subhayan Saha. Absolute reconstruction for sums of powers of linear forms: degree 3 and beyond. *Computational Complexity*, 32(2):1–66, 2023.

[9] Pascal Koiran and Subhayan Saha. Complete decomposition of symmetric tensors in linear time and polylogarithmic precision. In *CIAC (International Conference on Algorithms and Complexity)*, 2023. https://arxiv.org/abs/2211.07407.

[10] Pascal Koiran and Mateusz Skomra. Derandomization and absolute reconstruction for sums of powers of linear forms. *Theoretical Computer Science*, 887, 2021. arXiv preprint arXiv:1912.02021.

[11] Ankur Moitra. *Algorithmic aspects of machine learning*. Cambridge University Press, 2018.

[12] Marcus Schaefer and Daniel Stefankovic. The complexity of tensor rank. *Theory of Computing Systems*, 62(5):1161–1174, 2018. arXiv:1612.04338.

[13] Yaroslav Shitov. How hard is the tensor rank? *arXiv preprint arXiv:1611.01559*, 2016.

[14] Volker Strassen. Rank and optimal computation of generic tensors. *Linear Algebra and its Applications*, 52:645–685, 1983.