

Tensor Train-Based Approach for the Magnetic Moment Method

A. Zekri^{1,2}, O. Coulaud², L. Giraud² and J.-R. Poirier¹

¹LAPLACE, Université de Toulouse, CNRS, INPT, UPS, Toulouse, France

²Inria centre at university of Bordeaux

In this work, we focus on applying the Magnetic Moment Method (MMM) and enhancing its computational efficiency using a low-rank Tensor Train (TT) approach. By transforming the dense linear system generated by MMM into TT format, we achieve significant reductions in both computational cost and memory usage. This method is tested on several problems involving a ferromagnetic part with a regular mesh, demonstrating superior performance compared to traditional techniques.

Index Terms—Tensors , Linear systems , Integral equations , Computational electromagnetics

I. INTRODUCTION

VOLUME integral methods, such as the Magnetic Moment Method (MMM)[1], are powerful tools for modeling magnetic fields. These methods reduce the problem size by restricting computations to ferromagnetic regions, discretized into elementary elements with uniform magnetization. The MMM is particularly suited for problems involving ferromagnetic materials, where it offers accurate solutions while avoiding the need to mesh the surrounding air regions.

However, the system matrices resulting from these methods are dense, leading to high memory and computational demands. To address this, we propose using low-rank tensor formats, particularly the Tensor Train (TT) decomposition [2], to compress the system matrices and accelerate the solution process.

This approach has been successfully applied in methods like the Boundary Element Method (BEM) [3]. Here, we extend its application to the Magnetic Moment Method, leveraging TT decomposition to achieve significant improvements in efficiency.

We compare our method with H-Matrices, demonstrating that TT methods effectively balance accuracy, memory usage, and computational efficiency.

II. MAGNETIC MOMENT METHOD

The Magnetic Moment Method (MMM) efficiently models electrotechnical devices and computes electromagnetic fields, providing accurate magnetization distributions to derive the magnetic field, forces, and fluxes.

The equation to be solved is given for all $P \in \Omega_V$ by:

$$\mathbf{M}(P) - (\mu_r - 1) \times \frac{1}{4\pi} \int_{\Omega_V} \left(\frac{3 \mathbf{M}(Q) \cdot (\mathbf{r}_P - \mathbf{r}_Q)}{\|\mathbf{r}_P - \mathbf{r}_Q\|^5} (\mathbf{r}_P - \mathbf{r}_Q) - \frac{\mathbf{M}(Q)}{\|\mathbf{r}_P - \mathbf{r}_Q\|^3} \right) d\Omega_V = (\mu_r - 1) \times \mathbf{H}_s(P) \quad (1)$$

Where:

- Ω_V : The meshed region representing the active material.
- $\mathbf{M}(P)$: The magnetic moment at point P .
- μ_r : The relative permeability of the material.
- $\mathbf{H}_s(P)$: The source magnetic field at point P .

- $\mathbf{r}_P$ and $\mathbf{r}_Q$: Position vectors of points P and Q relative to the origin O .

Using a collocation method, we obtain a dense linear system to solve:

$$Ax = b \quad (2)$$

where $A \in \mathbb{R}^{N \times N}$ and $b \in \mathbb{R}^N$ and N is the number of degrees of freedom equal to three times the number of mesh elements.

Efficient numerical techniques, such as low-rank tensor train compression, are essential to make the approach scalable for large-scale problems.

III. TENSOR TRAIN FORMAT

Given a vector $X \in \mathbb{R}^N$, where N can be factorized as $N = \prod_{i=1}^d n_i$, it can be reshaped into a high-dimensional vector $\mathcal{X} \in \mathbb{R}^{n_1 \times \dots \times n_d}$.

A **Tensor Train (TT) vector** is a low-rank representation of this high-dimensional vector. Each element $\mathcal{X}(i_1, \dots, i_d)$ is approximated in the TT format as:

$$\mathcal{X}(i_1, \dots, i_d) \approx \mathcal{X}^{(1)}(i_1) \mathcal{X}^{(2)}(i_2) \dots \mathcal{X}^{(d)}(i_d),$$

where $\mathcal{X}^{(k)} \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$ is the k -th **TT-core** of the TT-vector, and r_k represents the **TT-rank** for the k -th mode.

To ensure the consistency of the representation, "boundary conditions" are imposed as $r_0 = r_d = 1$.

The Tensor Train format significantly reduces storage requirements and enables efficient computation of operations such as vector addition, dot products, and scalar multiplication, provided that the TT-ranks r_k remain small.

The TT format is able to reduce the storage complexity of the vector X from $O(N)$ to $O(d \cdot r^2 \cdot N^{\frac{1}{d}})$ where r is the maximum TT-rank, d the number of modes, and N the number of degrees of freedom.

IV. COMPRESSION IN TT FORMAT

To approximate a tensor in low TT-rank for a given accuracy ϵ , algorithms like **TT-cross** [4] have been developed. TT-cross avoids storing the full tensor by computing specific elements “on the fly” and iteratively constructing the TT representation.

After this initial approximation, **TT-rounding** refines the TT format, reducing ranks while preserving accuracy, thereby improving storage and computation. Further compression can be achieved using **QTT** [5], which decomposes each dimension into smaller sub-dimensions.

V. SOLVING LINEAR SYSTEMS IN TT FORMAT

While there are many methods to solve linear systems using TT representations, we focus on two approaches in this work:

- Using the GMRES solver with an operator based on the TT-matrix.
- Using a TT-based solver called **AMEN** (Alternating Minimal Energy Method) [6], available in the **ttpy** library [7].

VI. NUMERICAL RESULTS

To evaluate the Tensor Train (TT) approach, we model a ferromagnetic part Ω_V as a cube with a relative permeability of $\mu_r = 5000$. The domain Ω_V is discretized into cubic mesh elements. Below the cube, a current-carrying coil Ω_S generates a magnetic field, separated from it by an air gap Ω_T with a permeability of μ_0 .

The linear system arising from this discretization is solved using the TT format and is compared against the H-matrix.

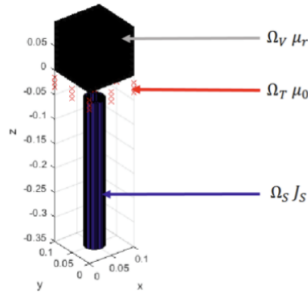


Fig. 1. Illustration of the Ferromagnetic Cube and Coil Setup for Numerical Testing

The threshold used for the approximation is $\epsilon = 10^{-6}$. The evaluation focuses on measuring the computation time required by the solvers to reach a scaled residual equal to ϵ and the compression rate ($\frac{\text{Compressed size}}{\text{Original size}} \times 100\%$) achieved for the matrix in each method. These metrics provide insight into the efficiency and scalability of the TT approach compared to other techniques.

Regarding the H-matrix approach, we approximate the matrix of the linear system using the Adaptive Cross Approximation (ACA) algorithm with the same threshold ϵ , and then solve it using GMRES as the solver.

The libraries used to implement and test both methods are written in Fortran. The computation time was measured on a system equipped with an Intel Core i7-13850HX CPU (20 cores, 5.3 GHz), 31 GiB of RAM, and a Linux operating system. The timings were recorded using Python’s time module.

TABLE I

THE LINEAR SYSTEM’S MATRIX ASSEMBLY TIME AND THE COMPRESSION RATE

N	TT-Cross	ACA
12,288	12.86 s (0.06 %)	19.34 s (59.78 %)
98,304	13.06 s (2.2 10^{-3} %)	460.78 s (19.71 %)
786,432	19.56 s (7.1 10^{-5} %)	-

The TT-Cross approach significantly outperforms ACA in both compression rate and assembly time, making it a superior choice for large-scale problems where storage and computational efficiency are critical.

TABLE II

COMPUTATION TIME OF THE SOLVE

N	AMEN	GMRES (TT Operator)	GMRES (H-matrix Operator)
12,288	1.21 s	2.63 s	6.09 s
98,304	5.63 s	58.09 s	238.10 s
786,432	19.92 s	829.35 s	-

The table shows AMEN’s superior efficiency and scalability compared to GMRES, with significantly shorter computation times. GMRES with a TT operator outperforms the one with H-matrix operator, but both are slower than AMEN, highlighting its advantage for large-scale systems.

VII. CONCLUSION

This study highlights the effectiveness of the Tensor Train (TT) approach for large-scale linear systems. Using a simple ferromagnetic cube model, the TT format achieved higher compression rates and significantly reduced storage compared to traditional methods, enabling the solution of problems that are infeasible with dense matrices on the same hardware.

It is important to note that the TT approach involves reshaping the matrix and vector into high-dimensional tensors to transition into the TT format, which can be challenging depending on the problem’s geometry. The way we reshape the linear system significantly impacts the method’s performance. The example presented illustrates a best-case scenario; however, we have also tested other cases where the TT approach remains efficient. In future work, we plan to explore more complex geometries.

VIII. ACKNOWLEDGMENTS

This work is supported by the ANR TensorVim project 22-CE50-0028.

REFERENCES

- [1] CHADEBEC, Olivier, COULOMB, J.-L., et JANET, Fleur. *A review of magnetostatic moment method*. IEEE Transactions on magnetics, 2006, vol. 42, no 4, p. 515-520.
- [2] OSELEDETS, Ivan V. *Tensor-train decomposition*. SIAM Journal on Scientific Computing, 2011, vol. 33, no 5, p. 2295-2317.
- [3] POIRIER, J.-R., COULAUD, Olivier, et KAYA, Oguz. Fast BEM solution for 2-D scattering problems using quantized tensor-train format. IEEE Transactions on Magnetism, 2020, vol. 56, no 3, p. 1-4.
- [4] OSELEDETS, Ivan et TYRTYSHNIKOV, Eugene. *TT-cross approximation for multidimensional arrays*. Linear Algebra and its Applications, 2010, vol. 432, no 1, p. 70-88.
- [5] KHOROMSKIY, Boris N. *O(d log N)-quantics approximation of N-d tensors in high-dimensional numerical modeling*. Constructive Approximation, 2011, vol.
- [6] DOLGOV, Sergey V. et SAVOSTYANOV, Dmitry V. *Alternating minimal energy methods for linear systems in higher dimensions*. SIAM Journal on Scientific Computing, 2014, vol. 36, no 5, p. A2248-A2271.
- [7] OSELEDETS, Ivan V. and contributors. *TPPY: Tensor Train Toolbox for Python*. Available at: <https://github.com/oseledets/ttpy>. Accessed: 10 January 2025.