

Tensor compression of volume integral operators in Discontinuous Galerkin

Anthony Torosyan¹, Bertrand Bannwarth¹, Olivier Chadebec¹, Jean-Michel Guichon¹,
G rard Meunier¹ and Jean-Ren  Poirier²

¹Univ. Grenoble Alpes, CNRS, Grenoble INP, G2Elab, 38000 Grenoble, France

²LAPLACE, Universit  de Toulouse, CNRS, INPT, UPS, Toulouse, France

In the paper, we show that volume integral operators discretized in Discontinuous Galerkin can be interpreted in tensor representations and we give the idea for a new assembly strategy. A numerical example illustrates the compression and the computational time obtained.

Index Terms—Volume integral equations, Tensor representations, Discontinuous Galerkin.

I. INTRODUCTION

THE Volume Integral Method (VIM) is a general framework for solving electromagnetic problems. It permits to discretize only the active parts of the geometry but leads to full matrix systems (i.e. with almost only non zeros coefficients). Thus, the compression of the volume integral operator is a fundamental part of the discretization in the VIM. It must also incorporate efficient algebraic properties in order to reduce the computational cost for solving the linear system.

Several methods were created in order to handle this difficulty (e.g. the Fast Multipole Method and Hierarchical matrices [1]). In this work, the paradigm adopted is the one of tensor representations (in particular the tensor train [2]). These representations allow to compress efficiently tensors defined on algebraic spaces and lead to powerful algebraic properties (e.g. solvers of linear systems [3]). In [4], it is shown that these notions can be extended on functional spaces.

In high frequency electromagnetism, some first attempts were realised (see [5]). Here, we have in mind electrostatic, magnetostatic and eddy current formulations as in [1] and [6]. We adopt a Discontinuous Galerkin point of view for the discretization because it is well adapted to the assembly procedure that we give in the paper. It allows also to consider more general meshes which is a limitation of tensor representations for the compression of volume integral operators (because of the number of degrees of freedom). The Discontinuous Galerkin discretization of the formulation in [6] is presented in [7]. The extension of this work would be to solve the linear system (with the algorithm in [3]).

II. TENSOR REPRESENTATION

A. Tensor train

Let $\mathcal{A} \in \mathbb{C}^{n_1 \times \dots \times n_d}$ be a d -dimensional tensor. The tensor train representation [2] of $\mathcal{A}$ is given by (with the multi-index $\mathbf{i} := (i_1, \dots, i_d)$)

$$\mathcal{A}_{\mathbf{i}} := \sum_{\mathbf{k}=1}^{\mathbf{r}} \mathcal{C}_1(k_0, i_1, k_1) \dots \mathcal{C}_d(k_{d-1}, i_d, k_d) \quad (1)$$

with $\mathcal{C}_l \in \mathbb{C}^{r_{l-1} \times n_l \times r_l}$ ($k_0 = k_d := 1$ to make uniform the notations) the l -th core and the integer r_l the l -th rank.

If $n := n_l$ and $r := r_l$, the number of coefficients in the tensor train representation is dnr^2 while in the canonical representation it is n^d .

To compute the tensor train representation, we can use the tt-svd and the tt-cross algorithms (see respectively [2], [8]). It is also possible to compress a given tensor in tensor train representation with the rounding procedure (see [2]).

B. Tensorisation

After the discretization of a volume integral operator, we obtain a matrix $\mathbf{A}_{\text{vim}} \in \mathbb{C}^{n \times n}$ where n is the number of unknowns in the mesh. This does not allow to use efficiently the tensor train representation. To fix this, we assume that $n = \prod_{k=1}^d p_k$ and that n is not a prime number.

C. Interpolation of functions

The tensor train representation presented in [2] is constructed to deal only with objects of algebraic spaces. Thanks to the framework developed in [4], it is possible to represent, in particular, functions in tensor train.

Let $f \in L^2(I)$ where $I := [0, 1]$ and $\bigcup_{q=0}^{m-1} I_q$ a subdivision of I (with $m := 2^d$). It is possible to define an operator $T_{2,d} : L^2(I) \rightarrow (\bigotimes_{k=1}^d \mathbb{R}^2) \otimes L^2(I)$ such that $T_{2,d}(f) =: F$ (see [4]). Note that $T_{2,d}$ is an isometry and a continuous version of the tensorisation principle (i.e. increase the number of dimensions of a tensor).

Let $V_S := \text{Span}(\phi_1, \dots, \phi_S) \subset L^2(I)$ be a finite dimensional approximation space (e.g. a space of polynomials). Following [4], we define the tensor train approximation $\tilde{F}_1(x) := \sum_{k_S=0}^S \mathcal{C}_{\mathbf{i}, k_S} \phi_{k_S}(x)$ of F where

$$\mathcal{C}_{\mathbf{i}, k_S} := \sum_{\mathbf{k}=1}^{\mathbf{r}} c_1(k_0, i_1, k_1) \dots c_{d+1}(k_d, k_S, k_{d+1}). \quad (2)$$

Note that everything in this part holds for compact domains Ω where a partition in m elements is available (and m is not a prime number).

III. VOLUME INTEGRAL OPERATORS

A. Representation in Discontinuous Galerkin

Let $\Omega \subset \mathbb{R}^3$ be a bounded Lipschitz domain discretized by $\mathcal{Q} := \bigcup_{q=0}^{n-1} Q_q$ with $n := 2^d$, $d > 1$ and where Q_q is a quadrangle. We define the mapping $f_q : Q_{\text{ref}} := [0, 1]^2 \rightarrow Q_q$ and assume that $|\det J_{f_q}(\mathbf{x})| = K$, $K > 0$ and $q \in \{0, \dots, n-1\}$, for the simplicity (i.e. $\mathcal{Q}$ is regular). Our aim is to compress the volume integral operators in [1] and [6]. This operator is defined as $P : L^2(\Omega) \rightarrow L^2(\Omega)$ such that $P(v) := g * v$ where $*$ denotes the convolution and g is the Green kernel of the Laplace equation in $\mathbb{R}^3$ (i.e. $g(\mathbf{x}, \mathbf{y}) := 1/(4\pi\|\mathbf{x} - \mathbf{y}\|_2)$ with $\|\cdot\|_2$ the euclidean norm). In the paper, we consider scalar-valued functions v although it can be vector-valued. The generalization is straightforward.

We denote by V_h the Discontinuous Galerkin approximation space. Note that $\dim(V_h) = np$ where p is the order of the Lagrange basis per element. After the discretization, the tensor to assembly is

$$\mathcal{A}_{\mathbf{i}, \mathbf{j}, i_S, j_S} := \int_{Q_j} \psi_{\mathbf{j}, j_S}(\mathbf{x}) \int_{Q_i} g(\mathbf{x}, \mathbf{y}) \psi_{\mathbf{i}, i_S}(\mathbf{y}) d\mathbf{y} d\mathbf{x} \quad (3)$$

where $\psi_{\mathbf{q}, q_S}$ denotes the q_S -th basis function of Q_q . The integration by parts formula gives

$$\mathcal{A}_{\mathbf{i}, \mathbf{j}, i_S, j_S} = K^2 \int_{Q_{\text{ref}}} \psi_{j_S}^{\text{ref}}(\mathbf{x}) \int_{Q_{\text{ref}}} G_{\mathbf{i}, \mathbf{j}}(\mathbf{x}, \mathbf{y}) \psi_{i_S}^{\text{ref}}(\mathbf{y}) d\mathbf{y} d\mathbf{x} \quad (4)$$

where $G_{\mathbf{i}, \mathbf{j}}(\mathbf{x}, \mathbf{y}) := g(f_{\mathbf{j}}(\mathbf{x}), f_{\mathbf{i}}(\mathbf{y}))$. We recognize the tensorisation of a function defined above with $T_{2,d}$ (i.e. $G = T_{2,d}(g)$). For the assembly, we consider firstly the part of $\mathcal{A}$ where $\mathbf{i} \neq \mathbf{j}$ (to avoid singularities on G). To achieve this we introduce G^s such that

$$G_{\mathbf{i}, \mathbf{j}}^s(\mathbf{x}, \mathbf{y}) := \begin{cases} G_{\mathbf{i}, \mathbf{j}}(\mathbf{x}, \mathbf{y}), & \text{if } \mathbf{i} \neq \mathbf{j} \\ s(\mathbf{i}, \mathbf{x}, \mathbf{y}), & \text{else} \end{cases} \quad (5)$$

where s is a function which permits to avoid singularities on G . The interpolation of G^s can be done and gives $\tilde{G}_{\mathbf{i}, \mathbf{j}}^s(\mathbf{x}, \mathbf{y}) = \sum_{k_S, l_S=0}^S C_{\mathbf{i}, \mathbf{j}, k_S, l_S} \phi_{k_S}(\mathbf{x}) \phi_{l_S}(\mathbf{y})$. In practice, the space of approximation can be a Lagrange polynomial space. If we define these basis functions at Gauss points, we avoid inaccuracies link to the interpolation of G^s . The tensor $\mathcal{C}$ is computed by a tt-cross on the values of G^s at Gauss points. Then it is possible to assembly the tensor $\mathcal{A}^s$ (by computing integrals in (4)) which is equal to $\mathcal{A}$ when $\mathbf{i} \neq \mathbf{j}$. Then by adding a correction on the $\mathbf{i} = \mathbf{j}$ part, we obtain $\mathcal{A}$ in tensor train representation. Note that the only approximation in the computation of $\mathcal{A}$ is in the tt-cross procedure for $\mathcal{C}$.

IV. NUMERICAL EXPERIMENTS

For the experience, $\Omega := [0, 1]^2$ and we discretize it with a regular grid. We take order one Lagrange basis for V_h and for the interpolation of the Green kernel.

For simplicity, we define s as $s := \theta$, $\theta \geq 0$ and we choose the parameter to optimize the computation of $\mathcal{A}$ (it changes the accuracy and the compression of the approximation). In our case, $\theta := 10^4$. It could be interesting to look for more general cutoff functions who remove singularities and lead to

Degrees of freedom (np)	Accuracy	Assembly time (s)
$2^{16} \approx 7.10^4$	$1,39.10^{-7}$	91,68
$2^{18} \approx 3.10^5$	7.10^{-7}	146,8
$2^{20} \approx 10^6$	$2,44.10^{-7}$	273,1
$2^{22} \approx 4.10^6$	$1,61.10^{-7}$	553

TABLE I

ASSEMBLY RESULTS IN ACCURACY AND COMPUTATIONAL TIME FOR DIFFERENT n .

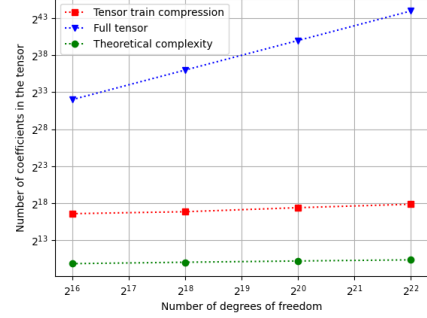


Fig. 1. Assembly results for the compression for different n . The accuracy of tt-cross and rounding procedures is 10^{-8} .

the best trade of between accuracy and compression.

In order to evaluate the accuracy of the assembly, which is linked to the approximation on $\mathcal{C}$, we randomly select n_{test} pairs of Gauss points (located in $Q_q \times Q_r$, $q \neq r$) and compare with the values of g . Here, $n_{\text{test}} := 50\sqrt{n}$.

In (Tab.I) and (Fig.1), some results are given for the accuracy, the computational time and the compression. It takes less than 10 min to assembly the operators for more than 10^6 degrees of freedom. For the compression, the tensor train representation has a complexity almost in $p \log_2(n)$ instead of the exponential complexity of the canonical representation.

ACKNOWLEDGMENT

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

REFERENCES

- [1] Siau, J., Chadebec, O., Perrussel, R., Poirier, J. R. (2015). Hybrid cross approximation for a magnetostatic formulation. IEEE Transactions on Magnetics, 51(3), 1-4.
- [2] Oseledets, I. V. (2011). Tensor-train decomposition. SIAM Journal on Scientific Computing, 33(5), 2295-2317.
- [3] Dolgov, S. V. (2013). TT-GMRES: solution to a linear system in the structured tensor format. Russian Journal of Numerical Analysis and Mathematical Modelling, 28(2), 149-172.
- [4] Ali, M., Nouy, A. (2023). Approximation theory of tree tensor networks: Tensorized univariate functions. Constructive Approximation, 58(2), 463-544.
- [5] Poirier, J. R., Coulaud, O., Kaya, O. (2020). Fast BEM solution for 2-D scattering problems using quantized tensor-train format. IEEE Transactions on Magnetics, 56(3), 1-4.
- [6] Nguyen, T. T., Meunier, G., Guichon, J. M., Chadebec, O., Nguyen, T. S. (2014). An integral formulation for the computation of 3-D eddy current using facet elements. IEEE transactions on magnetics, 50(2), 549-552.
- [7] Torosyan, A., Bannwarth, B., Chadebec, O., Guichon, J. M., Meunier, G., Poirier, J. R. (2025). Volume integral eddy current problem as a Friedrichs' system. Submitted to Compumag 2025.
- [8] Oseledets, I., Tyrtshnikov, E. (2010). TT-cross approximation for multi-dimensional arrays. Linear Algebra and its Applications, 432(1), 70-88.