Fast BEM Solution for 2-D Scattering Problems Using Quantized Tensor-Train Format

Jean-René Poirier, Olivier Coulaud, Oguz Kaya

▶ **To cite this version:**

Jean-René Poirier, Olivier Coulaud, Oguz Kaya. Fast BEM Solution for 2-D Scattering Problems Using Quantized Tensor-Train Format. IEEE Transactions on Magnetics, 2020, 56 (3), pp.1-4. 10.1109/TMAG.2019.2954584 . hal-03150956

HAL Id: hal-03150956
https://inria.hal.science/hal-03150956

Submitted on 4 Mar 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Abstract

It is common to accelerate the boundary element method by compression techniques (FMM, H-matrix / ACA) that enable a more accurate solution or a solution in higher frequency. In this work, we present a compression method based on a transformation of the linear system into Tensor-Train format by the quantization technique. The method is applied to a scattering problem on a canonical object with a regular mesh and improves the performance obtained from existing methods.

keywords: Boundary integral equations, electromagnetic diffraction, Quantized Tensor-Train

Fast BEM solution for 2D scattering problems using Quantized Tensor-Train format

J.-R. Poirier[*], O. Coulaud[†], and O. Kaya[‡]

March 4, 2021

Abstract

It is common to accelerate the boundary element method by compression techniques (FMM, H-matrix/ACA) that enable a more accurate solution or a solution in higher frequency. In this work, we present a compression method based on a transformation of the linear system into Tensor-Train format by the quantization technique. The method is applied to a scattering problem on a canonical object with a regular mesh and improves the performance obtained from existing methods.

keywords: Boundary integral equations, electromagnetic diffraction, Quantized Tensor-Train

1 Introduction

Many applications in science and engineering are formulated in terms of boundary integral equations. This is namely the case in electromagnetic scattering to avoid the use of an artificial condition to truncate the domain of study.

From a numerical point of view, the main difficulty is the solution of a full complex linear system

$$Ax = b.$$

Many compression for accelerating the solution of this linear system have been developed, which increase the amenable problem size from a few thousands to several millions. Continuing to improve performance and accelerate solution remains a topical issue. In this work, we reformulate the linear system in a tensor format and use tensor compression techniques for its storage and solution.

[*]LAPLACE, Université de Toulouse, CNRS, INPT, UPS, Toulouse, France and INRIA Bordeaux Sud-Ouest-HIEPACS

[†]INRIA Bordeaux Sud-Ouest-HIEPACS

[‡]Université Paris Saclay

As a numerical example, we will consider the scattering by a perfectly conducting cylinder Γ in the E-polarization ($u = E_z$). The involved boundary value problem to solve is then the Helmholtz equation with a Dirichlet boundary condition and a radiation condition at infinity.

The BEM solution can be written with a single layer potential [1]

$$\int_\Gamma G(x, x_s)\, j(x_s)\, \mathrm{d}\gamma(x_s) = -u^{\mathrm{inc}}(x),\ \forall x \text{ on } \Gamma, \tag{1}$$

where Γ is the boundary, u^{inc} the incident electric field, j the sought density current and G the Green's function.

In free space, this Green function is usually given by $G(x, x_s) = \frac{1}{4i} H_0^{(2)}(k|x - x_s|)$ where i is the imaginary unit, k the wave number and $H_0^{(2)}$ the Hankel function of second kind.

2 Solution with tensor techniques

For many years, various types of problems have been formulated using tensors instead of the classical matrix algebra [3]. More recently, tools have been developed to treat high order tensors (dimension higher than 3) based on the implementation of low-rank techniques to effectively reproduce the algebraic structure of the system. In the literature, among the most prominent tensor formats for an efficient and stable "hollow" representation of a very large system are Tensor-Train (TT) and Hierarchical Tucker decomposition. In this work, we have chosen to apply the TT format to formulate the integral equation problem for a fast solution.

2.1 Tensor-Train format

The TT format [2] consists of writing a d-dimensional tensor $\mathcal{T} \in \mathbb{R}^{n \times \ldots \times n}$ as a chain of 3-dimensional tensors according to the formula

$$\mathcal{T}(i_1, \cdots, i_d) = \sum_{\alpha_0, \cdots, \alpha_d} G_1(\alpha_0, i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \cdots G_1(\alpha_{d-1}, i_d, \alpha_d)$$

which could be rewritten in a matrix multiplication form

$$\mathcal{T}(i_1, \cdots, i_d) = \underbrace{G_1[i_1]}_{r_0 \times r_1} \underbrace{G_2[i_2]}_{r_1 \times r_2} \cdots \underbrace{G_3[i_3]}_{r_{d-1} \times r_d}$$

where

- G_i : TT-cores (matrices of size $r_{k-1} \times r_k$ with $r_0 = r_d = 1$),
- r_i : TT-ranks,
- $r = \max(r_i)$: The maximum rank among all cores.

The complexity of storage is $O(dnr^2)$ for a tensor with $O(n^d)$ elements. If r is small, the tensor is of low-rank requiring significantly less number of elements in its TT representation. As a result, operations involving the compressed low-rank tensor will be very fast.

The same format is given for a linear operator in $\mathbb{R}^{n^d \times n^d}$ which is represented by a $2d$-dimensional tensor $\mathcal{A}$ that couple elements as (i_n, j_n) for $n = 1, \ldots, d$ resulting from the couple (i, j). The TT representation of this tensor has the form

$$\mathcal{A}(i_1, \cdots, i_d; j_1, \cdots, j_d) = \underbrace{M_1[i_1, j_1]}\ \underbrace{M_2[i_2, j_2]} \cdots \underbrace{M_3[i_3, j_3]}$$

where $M_k[i_k, j_k]$ is a matrix of size $r_{k-1} \times r_k$.

The TT format provides all corresponding algebraic operations (addition, matrix-vector product, dot product, etc.) that are performed very efficiently for low-rank tensors owing to the reduced storage complexity.

2.2 Quantization and QTT format

Although some physical problems can naturally be formulated in terms of tensors and thereby solved by adapted techniques, a key point for the application of a tensor compression technique on integral equations is the quantization procedure [6], which enables a transition from a vector or a matrix to a tensor representation. For a vector $x \in \mathbb{R}^I$ and $I, d \in \mathbb{N}$ such that

$$I = I_1 I_2 \ldots I_d \text{ where } \mathrm{I}_k \in \mathbb{N} \text{ and for } k = 1, \ldots, d,$$

we can transform x to a d-dimensional tensor $\mathcal{X}$ as

$$x_i = \mathcal{X}(i_1, ..., i_d) \text{ with } i_k = 0, 1, .., I_k - 1 \text{ for } k = 1, \ldots, d)$$

using the following *flattening* of the multi-index $(i_1, \ldots, i_d)$:

$$i = i_1 + i_2 I_1 + .. + i_d I_1 I_2 ... I_{d-1}.$$

The low-rank representation of this vector in TT format is called Quantized Tensor-Train (QTT). An equivalent procedure can be applied to obtain a QTT representation for a matrix $A \in \mathbb{R}^{I \times I}$ with quantization in both dimensions.

This procedure has been successfully applied to approximate some functions or solutions of PDEs sampled on an equispaced mesh or grid, and is shown to lose effectiveness with mesh irregularities [6].

3 Solving Boundary Element Method with Tensor-Train Format

3.1 Compression with the TT-cross algorithm

A tensor $\mathcal{X}$ in full format can be approximated by $\tilde{\mathcal{X}}$ in the low-rank Tensor-Train (TT) format with the TT-SVD algorithm [5] such that

$$\left\| \tilde{\mathcal{X}} - \mathcal{X} \right\|_F \leq \varepsilon \left\| \mathcal{X} \right\|_F$$

where $\|.\|_F$ is the Frobenius norm and $\epsilon > 0$ the desired accuracy. The main limitation of this algorithm is the need for a full tensor, which is impractical to construct in most cases. To avoid this, cross-approximation techniques are developed that rely on the black-box evaluations of the function f defining $\mathcal{X}$, i.e.,

$$\mathcal{X} = f(i_1, \ldots, i_d)$$

and construct the approximate tensor $\tilde{\mathcal{X}}$ directly in the low-rank TT form [7]. TT-cross is an iterative framework that starts with an initial guess on the ranks of $\mathcal{X}$, finds the best approximation with these ranks, and finally increases the ranks to repeat the procedure until a convergence is achieved in the approximations $\tilde{\mathcal{X}}$ obtained in two successive iterations, which is determined according to a relative error parameter $\epsilon > 0$. The procedure performs $O(Ndr^2)$ evaluations of f per iteration where r is the rank used for the iteration. It is also possible to further reduce the ranks of this output tensor with an operation called TT-rounding [7] as TT-cross typically overestimates the tensor rank.

A vector/matrix can first be quantized, then constructed directly in TT form (without constructing full vector/matrix) with the TT-cross algorithm. In our experiments, we used the TT-cross implementation in the ttpy library, the Python version of TT-Toolbox [9].

3.2 Solution using AMEN solver

Once the matrix and vectors are expressed in the low-rank TT form, one needs to solve the linear system efficiently in this compressed framework. To do this, many algorithms are developed in the literature. Among the most prominent of these algorithms is the AMEN solver [7], which is a variant of the alternating linearization scheme (ALS). It starts with an initial guess on the solution vector x as well as its ranks, attempting to find the best solution for the core G_k in dimension k by fixing the cores in other dimensions, and sweeping over all dimensions until convergence. In addition, it calculates the residual vector $b - Ax$ in the TT form, and dynamically augments the rank of the solution vector in this direction to reveal optimal ranks, thereby combining rank search with a gradient descent step. Due to operating in low-rank tensor manifolds under very high compression, the solver works very efficiently in practice while keeping the desired accuracy.

4 Numerical Results

4.1 Validation with an analytic reference solution

We first consider a numerical example involving a metallic cylinder of radius 1m illuminated by an electromagnetic wave at frequency 0.6GHz (Fig. 1). This problem is well known and has an analytic solution for comparison.

The cylinder is discretized using a regular mesh with $N = 2^d$ elements. Fig. 2 presents the error with respect to the reference solution when a matrix of size $N \times N$ is built using TT-cross algorithm using three different values of ϵ.

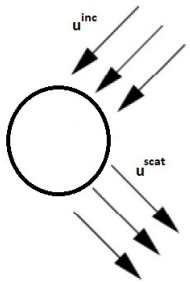

Figure 1: The scattering problem

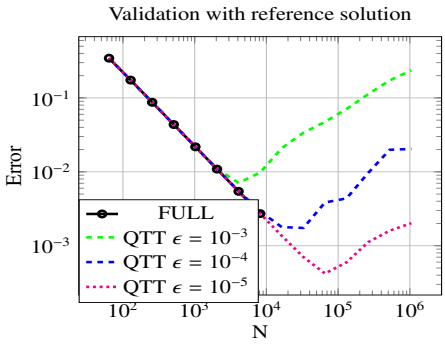

Figure 2: Error $\frac{\|u-u_{ref}\|}{\|u\|}$ on a cylinder with r=1m and f=0.6Ghz

As expected, the QTT improves the accuracy when increasing the number of unknowns thanks to the reduced storage. The accuracy is fixed by ϵ (compression accuract) because the AMEN solver has very fine one $\epsilon_{amen} << \epsilon$.

The numerical study shows that the computed error grows with the number of unknowns. Hence, to control it and keep the same accuracy, it is necessary to refine ϵ when increasing the size of the problem.

In this section, we consider the same physical problem with two other boundaries (rough surface) described as follows:

- Sinus profile

$$s(x) = h \sin\left(\frac{2\pi x}{L}\right), \text{ and} \tag{2}$$

- Weierstrass surface [8] given by

$$W(x) = h \sum_{n=1}^{n_2} \left(\frac{1}{b^{(2-D)n}} \cos\left(\frac{2\pi b^n x}{L}\right)\right), \tag{3}$$

with parameters D and b characterizing the fractal dimension and lacunarity of the surface and n_2 the number of scale in the surface.

These two examples are regularly discretized on the x-axis, though when $h \neq 0$ the mesh becomes irregular.

4.2 Memory storage

4.2.1 Cylinder with a regular mesh

We first consider the memory storage results for the cylinder with a regular mesh. Table 1

d	N	Full Matrix	H-matrix	TT-cross	TT-rounding
6	64	0.0625 MB	0.0568 MB	0.067 MB	0.007 MB
10	1024	16 MB	1.81 MB	0.33 MB	0.048 MB
14	16384	4 GB	39.32 MB	0.54 MB	0.063 MB
16	65536	64 GB	175.74 MB	0.7 MB	0.073 MB
18	262144	1 TB	770 MB	0.75 MB	0.079 MB
20	1048576	16 TB		1 MB	0.084 MB

Table 1: Memory storage for the matrix A

shows very promising compression rates obtained by the QTT technique in comparison with the H-matrix technique [10]. We point out the effectiveness of the TT-rounding algorithm, which further compresses the tensor in TT form obtained from the TT-cross algorithm. As mentioned, this compression is possible due to TT-cross needing to overestimate the ranks in attaining a good approximation, which is accounted for in a subsequent TT-rounding step providing the optimal compression.

We now study the influence of the frequency on the memory storage, which is presented in Fig. 3 using three discretization sizes for a metallic cylinder. We observe that the frequency has a stronger influence on the memory storage than the size of discretization.

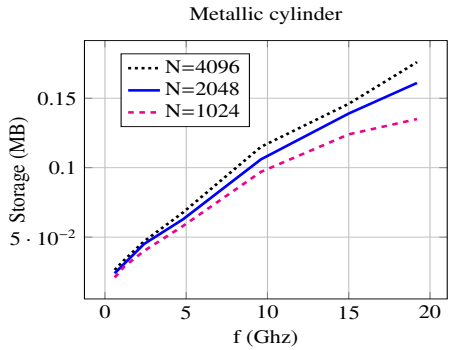

Figure 3: Memory storage relative to the frequency

Taking the previous observation into account, Fig. 4 draws the memory storage complexity for three cases:

- (a): for a given frequency f=0.6Ghz,
- (b): for a given frequency but with $\epsilon = 10^{-4} f(N)$ (with $f(N) < 1$ adapting to the increase in N to keep accuracy),
- (c): frequency increase with N relatively to the mesh (to keep the usual criterion of 10 points per wavelength)

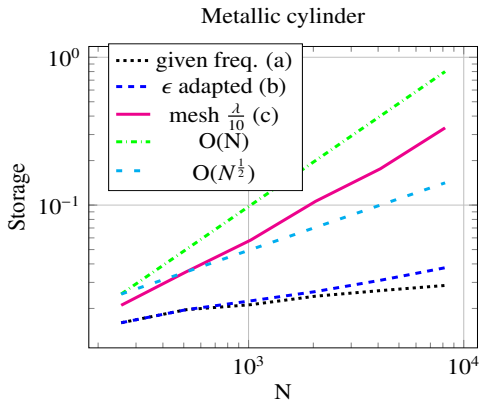

Figure 4: Memory storage using different frequency adaptation strategies

Comparing (a) with (b), we observe that refining ϵ to keep accuracy doesn't impact the compression significantly. We also note that despite the increase in frequency affecting the complexity in (c), the complexity manages to stay under $O(N)$.

4.2.2 Rough surface

Let us now study the QTT method when the object (and the mesh) lose some regularity. All cases consider a mesh of 16384 unknows. Table 2 shows the storage for two frequencies (low and high) for a sinus of length 10m and a variable heigh h.

Table 3 presents the storage for two frequencies (low and high) for a Weierstrass surface ($D = 1.5, b = 2$) of length 10m and height 0.1m. The complexity of the surface and irregularity of the mesh are governed by the number of scale n_2.

As expected the QTT method loses its effectiveness in terms of compression in these cases. On the same examples, the H-matrix is relatively stable, passing from 43 to 62 MB and from 44 to 85 MB for the sinus and from 44 to 51 MB and from 50 to 79 MB on the Weierstrass surface for the same frequencies. The memory storage still remains better in the QTT case but the computation time limits the use of the method when the complexity of the mesh/object grows.

h	f=0.6Ghz	f=30Ghz
0	0.05	0.07
0.01	0.07	0.14
0.1	0.14	0.43
0.2	0.18	0.86
0.5	0.30	3.65
1	0.47	5.81
2	0.90	7.09
5	2.59	

Table 2: Storage (MB) for a sine surface

n_2	f=0.6Ghz	f=30Ghz
1	0.16	0.51
2	0.34	1.66
3	0.53	3.69
5	1.91	16.97
7	12.68	

Table 3: Storage (MB) for a Weierstrass surface

4.3 Computation time

4.3.1 Cylinder

We now study the assembly and solver times, which are given in Table 4. The storage in terms of number of elements and the number of function evaluations for the TT-cross algorithm are also provided.

N	Storage (#elem)	f eval.	Ass. time (s)	Sol. time (s)
1024	19 421	71 248	0.16	0.02
4096	30 141	90 720	1.61	0.09
16384	53 433	188 256	1.06	0.44
65536	39 957	116 168	0.30	2.48
262144	51 569	150 368	0.32	12.18
1048576	55 813	160 712	0.39	135.47

Table 4: Computation times for cylinder

We note that the computations are very fast and the solver is the main cost. We also point out that the number of function evaluations is very high relative to the number of elements in the tensor obtained from TT-cross (which shrinks significantly after rounding). This observation explains that the method stays efficient in memory usage while computation time becoming its main limitation.

4.3.2 Rough surface

We conduct the same study on a Weierstrass surface with $n_2 = 3$ whose results are provided in Table 5. In this case, the compression remains very effective after rounding, yet the higher number of function evaluations done by TT-cross notably increases the assembly time. Nevertheless, the solver time remains to be the dominant cost.

N	Storage (#elem)	f eval.	Ass. time (s)	Sol. time (s)
1024	93 313	599 932	2.67	0.05
4096	153 673	1 079 188	3.62	0.12
16384	202 617	1 411 484	4.69	0.26
65536	256 681	1 775 448	7.03	5.60
262144	313 569	2 177 000	9.42	34.21
1048576	354 185	2 425 948	32.32	125.33

Table 5: Computation times for Weierstrass surface

4.3.3 Comparison with H-matrix

In this last example shown in Fig. 5, we provide a comparison of the computation times (assembly+solver) between the QTT method and H-matrix solver with respect to the frequency for two mesh sizes. The considered object is the Weierstrass surface.

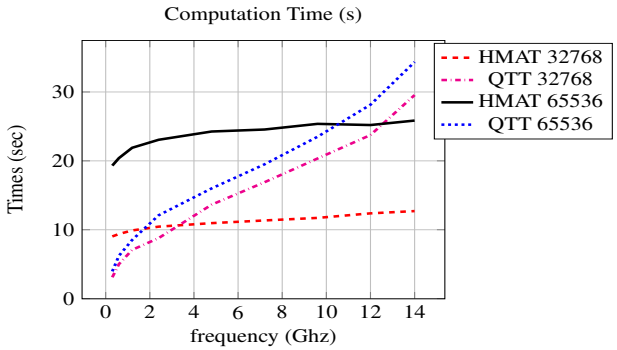

Figure 5: Influence of mesh size and frequency using QTT and H-matrix

As we observe in Fig. 5, the choice of the fastest method depends on the size and frequency, with QTT being more sensitive to the frequency increase and H-matrix to the mesh. In terms of memory usage, QTT still remains the most efficient.

5 Conclusion

In this work, we applied the quantized Tensor-Train format to the 2D scattering problem. We observed much higher compression and performance on canonical object and regular mesh than those obtained by H-matrix techniques. The performance on more complex problems or geometries has been numericaly studied. The method becomes less efficient but still remains useful in some cases. When the regularity of the mesh is insufficient, it could still be convenient to apply it as a preconditioner [4].

References

[1] R. F. Harrington, ”Field Compilation by Moment Methods” *Macmillan*, New York, 1968.

[2] I. V. Oseledets. 2011. ”Tensor-Train Decomposition” *SIAM J. Sci. Comput.*, vol 33, no 5 September 2011, pp 2295-2317.

[3] T. G. Kolda and B. W. Bader,”Tensor Decompositions and Applications,” *SIAM Review,* vol. 51, no. 3, pp. 455–500, 2009.

[4] Eduardo Corona, Abtin Rahimian, Denis Zorin,"A Tensor-Train accelerated solver for integral equations in complex geometries" *Journal of Computational Physics* Volume 334, April 2017, Pages 145–169

[5] Ivan Oseledets, Eugene Tyrtyshnikov,"TT-cross approximation for multidimensional arrays" *Linear Algebra and its Applications*,Volume 432, Issue 1,2010, Pages 70-88.

[6] Khoromskij, B. "O (d log N)-Quantics Approximation of N - d Tensors in High-DimensionalNumerical Modeling" *Constructive Approximation*,vol 34, 2009.

[7] Sergey V. Dolgov and Dmitry V. Savostyanov. "Alternating Minimal Energy Methods for Linear Systems in Higher Dimensions". *SIAM Journal on Scientific Computing* vol 36, no 5, pp. A2248-A2271, 2014.

[8] Falconer, K.,Fractal Geometry "Mathematical Foundations and Applications",John Wiley & Sons,2013.

[9] I. V. Oseledets, S. Dolgov, V. Kazeev, D. Savostyanov, O. Lebedeva, P. Zhlobich, T. Mach, and L. Song. *TTToolbox. https://github.com/oseledets/TT-Toolbox*.

[10] Daquin, Priscillia, Perrussel, Ronan, and Poirier, Jean-René. Hybrid Cross Approximation for the Electric Field Integral Equation. *Progress In Electromagnetics Research*, 2018, vol. 75, p. 79-90.