Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

Low-rank quadrature-based tensor approximation of the Galerkin projected Newton/Yukawa kernels

(revised version: January 2010)

by

Cristobal Bertoglio, and Boris N. Khoromskij

Preprint no.: 79 2008

Low-rank quadrature-based tensor approximation of the Galerkin projected Newton/Yukawa kernels

Cristóbal Bertoglio*
INRIA, CRI Rocquencourt, B.P. 105, 78153 Le Chesnay Cedex, France

Boris N. Khoromskij†
Max-Planck-Institute for Mathematics in the Sciences, Inselstr. 22-26, 04103 Leipzig, Germany

January 18, 2010

Abstract

Tensor-product approximation provides a convenient tool for efficient numerical treatment of high dimensional problems that arise, in particular, in electronic structure calculations in $\mathbb{R}^d$. In this work we apply tensor approximation to the Galerkin representation of the Newton and Yukawa potentials for a set of tensor-product, piecewise polynomial basis functions. To construct tensor-structured representations, we make use of the well-known Gaussian transform of the potentials, and then approximate the resulting univariate integral in $\mathbb{R}$ by special sinc quadratures. The novelty of the approach lies on the heuristic optimisation of the quadrature parameters that allows to reduce dramatically the initial tensor-rank obtained by the standard sinc-quadratures. The numerical experiments show that this approach gives tensor-ranks close to the optimal in 3D computations on large spatial grids and with linear complexity in the univariate grid size. Particularly, this scheme becomes attractive for the multiple calculation of the Yukawa potential when the exponents in gaussian functions vary during the computational process.

Key words: tensor-product approximation, Newton/Yukawa potentials, Gaussian integral transform, sinc-quadrature, electronic structure calculations.

1 Introduction

In the recent years, the idea of tensor approximation of operators and functions has lead to powerful numerical algorithms in large-scale problems of computational physics, in particular, in electronic structure calculations based on the Hartree-Fock or DFT models [6, 10, 8]. In these applications one deals with numerical computations of integral transforms which

*E-mail: cristobal.bertoglio@inria.fr
†E-mail: bokh@mis.mpg.de

include Green's kernels in $\mathbb{R}^d$ [5, 7, 9]. Efficient computation of such integral transforms often appears to be the bottleneck of the traditional numerical schemes.

For example, the Hartree-Fock equation for determination of the ground state of a molecular system consisting of M nuclei and N electrons is given by the following self-consistent eigenvalue problem in $L^2(\mathbb{R}^3)$,

$$(\mathcal{F}_\Phi \phi_i)(\mathbf{x}) = \lambda_i \, \phi_i(\mathbf{x}), \quad \int_{\mathbb{R}^3} \phi_i(\mathbf{x})\phi_j(\mathbf{x}) \, \mathrm{d}\mathbf{x} = \delta_{ij}, \; i,j = 1, ..., N \; , \tag{1.1}$$

with $\mathcal{F}_\Phi$ being the non-linear Fock operator

$$\mathcal{F}_\Phi(\cdot) := -\frac{1}{2}\Delta(\cdot) - \sum_{\nu=1}^{M} \frac{Z_\nu}{\|\mathbf{x} - a_\nu\|}(\cdot) + V_H(\mathbf{x})(\cdot) + V_E(\cdot) \; ,$$

where the Hartree potential is defined by

$$V_H(\mathbf{x}) := 2 \int_{\mathbb{R}^3} \frac{\rho(\mathbf{y}, \mathbf{y})}{\|\mathbf{x} - \mathbf{y}\|} \, \mathrm{d}\mathbf{y} \; ,$$

and the nonlocal exchange operator is given as

$$V_E \phi := - \int_{\mathbb{R}^3} \frac{\rho(\mathbf{x}, \mathbf{y})}{\|\mathbf{x} - \mathbf{y}\|} \phi(\mathbf{y}) \, \mathrm{d}\mathbf{y} \; .$$

Here, $1/\|\cdot\| : \mathbb{R}^3 \to \mathbb{R}$ corresponds to the Newton potential, and $Z_\nu \in \mathbb{R}_+$, $a_\nu \in \mathbb{R}^3$ $(\nu = 1, ..., M)$ specify charges and positions of M nuclei. The electron density matrix $\rho : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$, is given by $\rho(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{N} \phi_i(\mathbf{x})^* \phi_i(\mathbf{y})$.

Note that both exchange and the Hartree potentials contain the 3D convolution transform with the Newton convolving kernel, that has to be computed at each step of the iterations on nonlinearity. Hence, these terms represent the most complicated part in the numerical treatment of the Hartree-Fock equation, see, e.g., [7, 9, 10, 12]. Another popular modification of the Hartree-Fock and Schrödinger equations is based on the so-called Lippmann-Schwinger integral formulation, which contains also the convolution transform with the Yukawa potential $\frac{e^{-\lambda\|\mathbf{x}\|}}{\|\mathbf{x}\|}$ $(\lambda \in \mathbb{R}_+)$ [6, 8].

Traditionally, the Hartree-Fock equation is solved by meshless methods based on the usage of the so-called Gaussian type orbitals which allow analytical evaluation of the basic convolution transforms. Application of the finite element and wavelet methods [13, 6] might be of the particular interest if the volume integral transforms can be evaluated efficiently [13, 6, 12]. In particular, due to the recent development of tensor numerical methods, the combination of the finite element with nonlocal problem dependent basis sets leads to the linear scaling methods $O(n)$ [12].

In this case, the critical step corresponds to the efficient approximation in tensor-product format of the Galerkin representation to the Newton and Yukawa potentials on a set of piecewise polynomial basis functions associated with the large $n \times n \times n$ spatial grids. Then, this allows for instance the accurate computation of the convolution between Newton and density kernels with log-linear scaling in the univariate grid size n, i.e., $O(n \log n)$.

The goal of this paper is the description of an efficient black-box low tensor-rank approximation algorithm applied to the FEM-Galerkin matrices of the 3D Newton and Yukawa

potentials over the set of piecewise polynomial tensor-product basis functions. To construct tensor-structured representations, we make use of the well-known Gaussian transform of the potentials, and then approximate the resulting univariate integral in $\mathbb{R}$ by special *sinc*-quadratures [1, 2, 3, 15]. The novelty of the approach lies on the heuristic optimisation of the quadrature parameters that allow to reduce substantially the initial tensor-rank, obtained by the standard quadratures approximation. We describe and implement the algorithm that computes fast the optimised *sinc*-quadratures adapted to the required accuracy, the grid-size, discretisation interval, and to the type of finite elements.

The numerical experiments show that this approach gives near optimal tensor-ranks in 3D computations on large spatial grids and with linear complexity in the univariate grid size. This scheme becomes attractive for the multiple calculation of the Yukawa potential when the exponents $\lambda \geq 0$ vary during the computational process. It is also important to mention that the adaptive black-box scheme presented in this paper was successfully applied in electronic structure calculations [8, 12, 11].

The rest of this paper is organized as follows. Section 2 introduces tensor-product formats inspired from the finite element representation of the potentials of interest, and discusses the analytic tensor-product decomposition applied to the Galerkin integrals of the Newton kernel (over the computational hypercube $\Omega \subset \mathbb{R}^3$) in terms of a parametric integral over a family of rank-1 tensors. Section 3 explains how *sinc*-schemes are used to discretise the aforementioned integral representations. As the main contribution of this work, special attention is deserved to the appropriate choice of the free parameter in the quadrature to minimize the resultant tensor-rank with a computational cost asymptotically neglectable with respect to $O(n)$. Furthermore, we discuss in Sections 4.1 the extension to other classes of basis functions, and Section 4.2 summarizes the results approximation of the Yukawa potential based on the similar methodology. Finally, Section 5 includes a comparative analysis of the optimality of the low rank approximations obtained by our black-box algorithm, and includes some concluding remarks.

Throughout the paper we present several numerical illustrations on the efficiency of the proposed numerical algorithms of low tensor-rank approximation.

2 Tensor-product representations involving $\frac{1}{\|x\|}$

As said above, we are interested in the separable representation of the Galerkin and collocation matrices associated with the singular kernel $\frac{1}{\|x\|}$ in $\mathbb{R}^d$, and specifically, for the Newton kernel corresponding to $d = 3$.

Assume that the computational domain Ω corresponds to a hypercube in $\mathbb{R}^d$,

$$\Omega = \Omega_1 \times \cdots \times \Omega_d \subset \mathbb{R}^d \quad \text{with} \quad \Omega_\ell = [a^\ell, b^\ell] \subset [0, \infty) \ , \ \ell = 1, \ldots, d \ . \tag{2.1}$$

One can apply the Galerkin method of approximation to equation (1.1) in $\mathbb{R}^d$ for $d = 3$, with respect to certain basis functions set $\{\psi_{\mathbf{i}}\}$, where $\psi_{\mathbf{i}}$ are the tensor-product piecewise polynomials,

$$\psi_{\mathbf{i}}(\mathbf{x}) = \prod_{\ell=1}^{d} \psi_{i_\ell}^{(\ell)}(x_\ell) \quad \text{with} \quad x_\ell \in \Omega_\ell, \tag{2.2}$$

for $\mathbf{i} = (i_1, \ldots, i_d) \in \mathcal{I} := I_1 \times \cdots \times I_d$, $i_\ell \in I_\ell = \{1, ..., n_\ell\}$.

For example, the Galerkin matrix representation of the core potential is given by a tensor of order 6 with the entries,

$$\mathcal{N}_{\mathbf{ij}} = \sum_{\nu=1}^{M} Z_\nu \int_\Omega \frac{\psi_{\mathbf{i}}\psi_{\mathbf{j}}}{\|\mathbf{x} - a_\nu\|}\, \mathrm{d}\mathbf{x}, \quad \mathbf{i}, \mathbf{j} \in \mathcal{I}.$$

Assume that the function ρ is already presented in the basis set $\{\psi_{\mathbf{i}}\}$. Then, the projection-collocation scheme to approximate the Hartree potential described in [7], requires the convolution product with the following 3-tensor,

$$\mathcal{G} := [\mathcal{G}_{\mathbf{i}}]_{\mathbf{i}\in\mathcal{I}}, \quad \mathcal{G}_{\mathbf{i}} = \int_{\Omega_{\mathbf{i}}} \frac{\psi_{\mathbf{i}}(\mathbf{x})}{\|\mathbf{x}\|}\, \mathrm{d}\mathbf{x} \quad \text{where} \quad \Omega_{\mathbf{i}} = \mathrm{supp}(\psi_{\mathbf{i}}). \tag{2.3}$$

We consider the low rank tensor approximation of the real-valued arrays like $\mathcal{N} = [\mathcal{N}_{\mathbf{ij}}]$, $\mathcal{G} = [\mathcal{G}_{\mathbf{i}}]$, $\mathbf{i}, \mathbf{j} \in \mathcal{I}$.

Definition 2.1 *The canonical form of a tensor $\mathcal{N}$ (resp. $\mathcal{G}$) is given by*

$$\mathcal{N} = \sum_{k=1}^{r_{\mathcal{N}}} n_k \bigotimes_{\ell=1}^{d} \mathbf{V}_k^{(\ell)}, \ n_k \in \mathbb{R}, \ \mathbf{V}_k^{(\ell)} \in \mathbb{R}^{n\times n},$$

respectively,

$$\mathcal{G} = \sum_{k=1}^{r_{\mathcal{G}}} g_k \bigotimes_{\ell=1}^{d} \mathbf{B}_k^{(\ell)}, \ g_k \in \mathbb{R}, \ \mathbf{B}_k^{(\ell)} \in \mathbb{R}^{n},$$

with $r_{\mathcal{N}}$ (resp. $r_{\mathcal{G}}$) called a tensor-rank (Trank) and with the normalized canonical factors $\mathbf{V}_k^{(\ell)}$ (resp. $\mathbf{B}_k^{(\ell)}$), where $\bigotimes$ represents the Kronecker product of matrices or tensor-product of vectors.

Notice that the first expression represents the compressed form of a matrix $\mathcal{N}$ since the required storage size is only $\mathcal{O}(r\, d\, n^2)$, versus $\mathcal{O}(n^{2d})$ in the full format. Moreover, linear operations between elements in this tensor-product spaces can be performed very efficiently. For instance, a "matrix-vector multiplication" of $\mathcal{N}$ with a tensor $\mathcal{G} \in \mathbb{R}^{\mathcal{I}}$ requires only $\mathcal{O}(d\, r_{\mathcal{N}}\, r_{\mathcal{G}}\, n^2)$ operations, instead of $\mathcal{O}(n^{2d})$ operations in the full format. For more details about tensor-product formats we refer to [**?**, 10].

To simplify the presentation, we will start computing the tensor-product approximation of the coefficients tensor $\mathcal{G} := [\mathcal{G}_{\mathbf{i}}]_{\mathbf{i}\in\mathcal{I}}$. To that end, the Newton potential has to be decomposed in the separable tensor-product format, that can be performed by using its Laplace transform representation, i.e.,

$$\frac{1}{\|\mathbf{x}\|} = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} e^{-t^2\|\mathbf{x}\|^2}\, \mathrm{d}t = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} \prod_{\ell=1}^{d} e^{-t^2 (x_\ell)^2}\, \mathrm{d}t, \ \|\mathbf{x}\| > 0\ . \tag{2.4}$$

Moreover, inserting (2.4) into (2.3) and applying Fubini's theorem, the entries of the tensor $\mathcal{G}$ can be written in the form,

$$\mathcal{G}_{\mathbf{i}} = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} \int_\Omega \psi_{\mathbf{i}}(\mathbf{x}) e^{-\|\mathbf{x}\|^2 t^2}\, \mathrm{d}\mathbf{x}\, \mathrm{d}t = \int_{\mathbb{R}} \prod_{\ell=1}^{d} \mathbf{B}_{i_\ell}^{(\ell)}(t)\, \mathrm{d}t, \tag{2.5}$$

with

$$\mathbf{B}_{i_\ell}^{(\ell)}(t) = \pi^{-1/2d} \int_{\Omega_\ell} \psi_{i_\ell}^{(\ell)}(x_\ell) e^{-x_\ell^2 t^2} \mathrm{d}x_\ell, \tag{2.6}$$

that remains valid for $\|\mathbf{x}\| \geq 0$. Furthermore, since in the integral (2.5) we separate the spatial directions, the tensor $\mathcal{G}$ can be expressed using the integral representation via a family of rank-1 tensors $\mathcal{B}(t)$,

$$\mathcal{G} = \int_{\mathbb{R}} \mathcal{B}(t)\,\mathrm{d}t \quad \text{with} \quad \mathcal{B}(t) := \bigotimes_{\ell=1}^{d} \mathbf{B}^{(\ell)}(t) \quad \text{and} \quad \mathbf{B}^{(\ell)}(t) \in \mathbb{R}^{n_\ell}. \tag{2.7}$$

Now the task is to find an appropriate quadrature to approximate (2.7) for all elements $\mathcal{G}_{\mathbf{i}}$ simultaneously, and with possibly small numbers of terms.

Remark 2.2 *Since computation of the Galerkin matrix includes integration of the Newton kernel over the singularity, direct best-approximation of this kernel by exponential sums [4] cannot be applied straightforwardly, since the latter is defined only for $\|\mathbf{x}\| \geq h > 0$. Moreover, the coefficients used in these sums are generally not optimal for the integrated version, hence, the error control becomes troublesome.*

In the next section, we describe the black-box optimised *sinc*-quadratute scheme to approximate the integral in (2.7).

3 Quadrature-based Kronecker-sum decomposition

3.1 Improved sinc-quadrature

For a given precision $\varepsilon > 0$, we are looking for an accurate quadrature formula on $\mathbb{R}$ for the integral of a tensor-valued function (2.7), that involves as less quadrature points as possible, i.e.,

$$\mathcal{G} \approx \mathcal{G}^{(M)} = \sum_{k=-M}^{M} g_k \bigotimes_{\ell=1}^{d} \mathbf{B}^{(\ell)}(t_k), \quad g_k,\, t_k \in \mathbb{R}, \tag{3.1}$$

such that in the Frobenius-type norm

$$\|\mathcal{G} - \mathcal{G}^{(M)}\| \leq \varepsilon \|\mathcal{G}\|, \tag{3.2}$$

with the tensor-rank estimated by $r = 2M + 1$. We intentionally do not specify the norm at this point since the proper choice will be discussed later on.

A good possibility to construct such a quadrature can be based on the *sinc methods*, which are commonly used to interpolate and integrate analytic functions on $\mathbb{R}$ [15, 14]. For instance, a typical choice for the quadrature parameters

$$t_k = k\mathfrak{h}_M,\ g_k = \mathfrak{h}_M,\ \mathfrak{h}_M = C_0 \log(M)/M,\ C_0 \in \mathbb{R}_+, \tag{3.3}$$

leads to the exponential convergence in M, see [5],

$$\|\mathcal{G} - \mathcal{G}^{(M)}\| \leq C\, e^{-\beta\sqrt{M}} \|\mathcal{G}\| \quad \text{with} \quad C, \beta \in \mathbb{R}_+.$$

An improved convergence rate can be achieved by using a variable transformation $t = \sinh(u)$, and by taking advantage of the symmetry in t of the transformed integrand in (2.5), to obtain

$$\mathcal{G} = \int_{\mathbb{R}} \cosh(u) \bigotimes_{\ell=1}^{d} \mathbf{B}^{(\ell)}(\sinh(u))\, \mathrm{d}u \approx \sum_{k=0}^{M} g_k \bigotimes_{\ell=1}^{d} \mathbf{B}^{(\ell)}(t_k) := \mathcal{G}^{(M)}. \tag{3.4}$$

If both quadrature points and weights in (3.4) are choosen as

$$t_k = \sinh(k\mathfrak{h}_M), \tag{3.5}$$

and

$$g_k = \begin{cases} \mathfrak{h}_M & \text{for } k = 0 \\ 2\,\mathfrak{h}_M \cosh(k\mathfrak{h}_M) & \text{for } 0 < k < M, \end{cases} \tag{3.6}$$

with $\mathfrak{h}_M$ as above, then the quadrature (3.4)-(3.6) converges in M asymptotically as (cf. [5]),

$$\|\mathcal{G} - \mathcal{G}^{(M)}\| \leq C\, e^{-\beta M/\log M} \|\mathcal{G}\| \quad \text{with} \quad C, \beta \in \mathbb{R}_+.$$

Moreover, due to the symmetry of quadrature points, the tensor-rank is now estimated by $r = M + 1$.

Remark 3.1 *The quadrature formula proposed above allows a free choice of the parameter C_0 in Equation* (3.3)*. This choice has to be done carefully in order to control the relative approximation accuracy for the complete tensor (in the sense of equation* (3.2)*, and for some choice of the norm), while keeping the tensor-rank r as small as possible. The problem is that on one hand, all components of the target $\mathcal{G}$ are practically not computable for realistic grids (e.g., if $n = O(10^3)$), and on the other hand, the norm evaluation with the complete tensor may be expensive.*

3.2 Rank minimisation via calibration of C_0

In this section we will explain in detail how to choose C_0 and r in an optimal way, such that the rank-r approximating tensor fulfills the prescribed accuracy criteria $\varepsilon > 0$ as in (3.2).

First, let us define the proper norm-type functional that allows the efficient control of the approximation accuracy. Typically in multilinear algebra calculations, the Frobenius norm, $\|\cdot\|_F$, does a job. Practical application of this norm has, however, the limitation that the approximation criteria (3.2) requires all n^d entries of the exact tensor to be evaluated, that is exactly what we are trying to avoid.

We introduce the following error-functional which allows to estimate the relative Frobenius norm,

$$\|\mathcal{G}/|\mathcal{G}| - \mathcal{G}^{(M)}/|\mathcal{G}|\|_\infty := \max_{\mathbf{i}\in\mathcal{I}} \left\{ |\mathcal{G}_\mathbf{i} - \mathcal{G}^{(M)}_\mathbf{i}|/|\mathcal{G}_\mathbf{i}| \right\}. \tag{3.7}$$

Lemma 3.2 *For given $\varepsilon > 0$, the condition*

$$\max_{\mathbf{i}\in\mathcal{I}} \left\{ |\mathcal{G}_\mathbf{i} - \mathcal{G}^{(M)}_\mathbf{i}|/|\mathcal{G}_\mathbf{i}| \right\} \leq \varepsilon$$

implies

$$\|\mathcal{G} - \mathcal{G}^{(M)}\|_F \leq \varepsilon \|\mathcal{G}\|_F.$$

Proof. It is easy to show that

$$\frac{\|\mathcal{G}-\mathcal{G}^{(M)}\|_F^2}{\|\mathcal{G}\|_F^2}=\frac{\sum_{\mathbf{i}\in\mathcal{I}}|\mathcal{G}_{\mathbf{i}}-\mathcal{G}_{\mathbf{i}}^{(M)}|^2}{\|\mathcal{G}\|_F^2}\leq\varepsilon^2\frac{\sum_{\mathbf{i}\in\mathcal{I}}|\mathcal{G}_{\mathbf{i}}|^2}{\|\mathcal{G}\|_F^2}=\varepsilon^2,$$

what proves the assertion. ∎

The error-functional in (3.7), in the general case, still needs the knowledge of the exact tensor $\mathcal{G}$, and, moreover, it is computationally even more expensive than the cost for the exact Frobenius norm. However, taking advantage of the specific data structure in our particular application, we are able to calculate the good estimate of that functional efficiently by majorizing it over a very small subset of tensor entries.

In the following, we fix $\Omega=[0,1]^d$, $d=3$, and $n\times n\times n$ tensor grid with $n=2^p-1$, and use the piecewise constant basis functions

$$\psi_{i_\ell}(x_\ell):=\begin{cases}1 \text{ if } h\,(i_\ell-1)<x_\ell<h\,i_\ell\\ 0 \text{ otherwise,}\end{cases}\tag{3.8}$$

with the uniform grid spacing $h=1/n$ in all spatial directions. All computations are performed in MATLAB 7.4 on a MAC-OS machine (2.93 GHz, 4 GB). In particular, for the evaluation of $\mathbf{B}_{i_\ell}^{(\ell)}(t)$ in Equation (2.6), we approximate the integral over Ω_ℓ for a given set of values of t by means of the function `quadv` (adaptive Simpson's rule). In fact, even though we still cannot find an estimate for the accuracy of this approximation (as function of the global target precision ε), numerical tests show the the overall result is not too sensitive to this parameter, and that fixing the accuracy equal to ε (but probably too strong), we found out that always that the error curves are controlled by the accuracy of the sinc approximation itself or the computation of the reference elements (see Section 3.4).

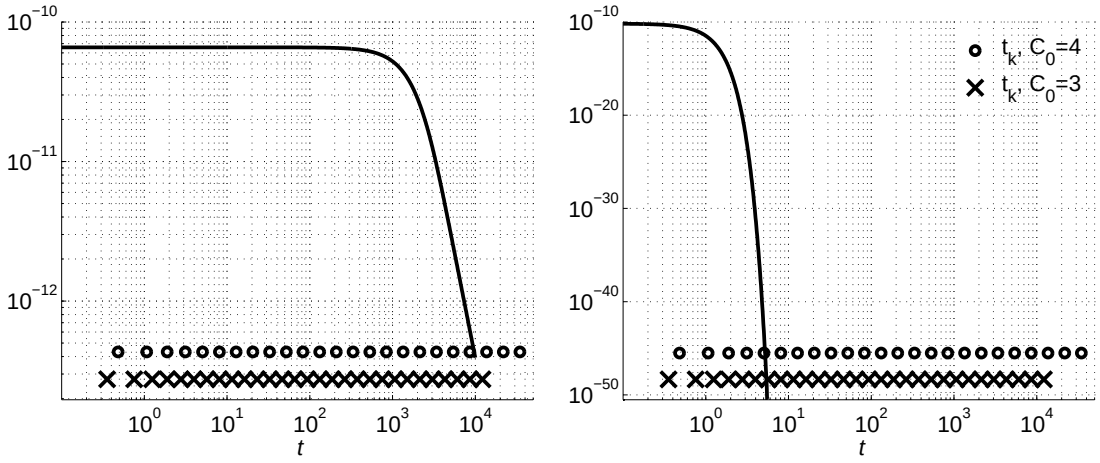

Figure 3.1: $\mathcal{B}_{\mathbf{i}_{min}}(t)$ (left), $\mathcal{B}_{\mathbf{i}_{max}}(t)$ (right) and location of the quadrature points for $r=30$ and different values of C_0.

First, we present some numerical examples illustrating the effect of optimisation by varying the quadrature parameter C_0. For fixed $n=2^{11}-1=2047$, Figure 3.1 presents

the integrand shapes for $\mathcal{B}_{\mathbf{i}_{min}}(t)$ and $\mathcal{B}_{\mathbf{i}_{max}}(t)$, with $\mathbf{i}_{min} := (1,1,1)$, $\mathbf{i}_{max} := (n,n,n)$ and $t \in [0, 10^4]$, and positions of the quadrature points, which are influenced by the value of C_0. From (2.5) and (2.6) it is clearly seen that $\mathcal{B}_{\mathbf{i}_{min}}(t) \leq \mathcal{B}_{\mathbf{i}_{max}}(t)\ \forall\ t$, what can be observed in Figure 3.1. Hence, due to the different decays of the integrands for different indicies $\mathbf{i}$, varying $C_0 > 0$ provides the way to improve accuracies of the respective quadratures. Then, we find some $C_0 > 0$ that simultaneously minimises the relative error of the sinc-quadrature for both integrands corresponding to $\mathbf{i}_{min}$ and $\mathbf{i}_{max}$. The error control for the rest of tensor entries relyes on the heuristic majorizing property explained as follows.

Figure 3.2 presents relative error curves for $\mathcal{G}_{\mathbf{i}_{min}}$, $\mathcal{G}_{\mathbf{i}_{max}}$ and for other selected tensor entries, corresponding two different values of C_0 (to compute reference values we use 3D quadrature rules, see Section 3.4). We emphasize here that the choice of C_0 is critical in the optimisation of the tensor-rank, since for all elements the same quadrature parameter, C_0 and M, are simultaneously used.

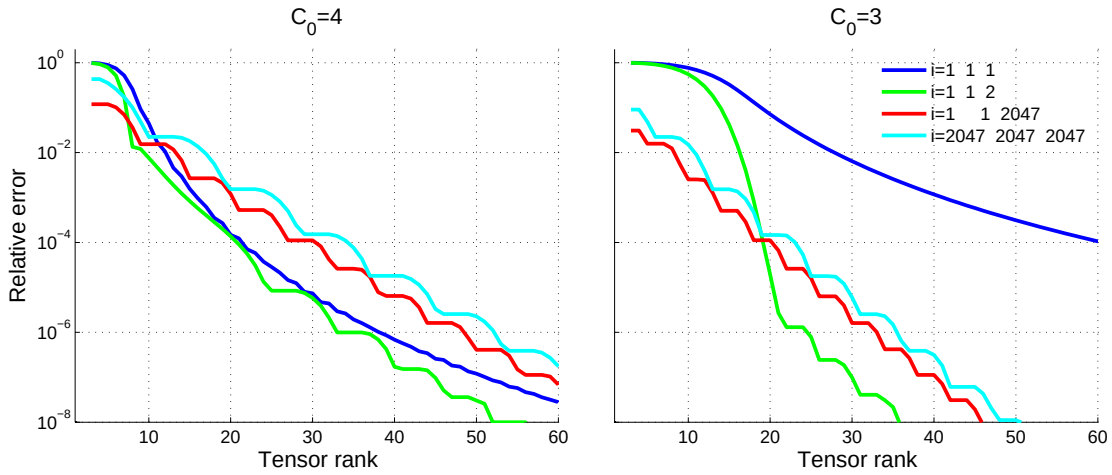

Figure 3.2: Relative error curves for two values of C_0 and different tensor entries.

We now observe that the upper error bound is simply given by either $\mathcal{G}_{\mathbf{i}_{min}}$ or $\mathcal{G}_{\mathbf{i}_{max}}$, depending on the value of C_0. In fact, a perturbation of this parameter will improve the relative accuracy in some elements and reduce it in others since all integrands are contained between $\mathcal{B}_{\mathbf{i}_{min}}(t)$ and $\mathcal{B}_{\mathbf{i}_{max}}(t)$.

This *majorizing property* can be used for choosing the quadrature parameters appropriately, avoiding the computation of all tensor components: the minimal Kronecker rank $r(C_0) = M + 1$ can be obtained by intersecting both error curves, related to $\mathbf{i}_{min}$ and $\mathbf{i}_{max}$, at the given accuracy level ε. In other words, the task is to find C_0 and the smallest M so that

$$|\mathcal{G}_{\mathbf{i}_{min}} - \mathcal{G}^{(M)}_{\mathbf{i}_{min}}| \leq \varepsilon |\mathcal{G}_{\mathbf{i}_{min}}| \quad \text{and} \quad |\mathcal{G}_{\mathbf{i}_{max}} - \mathcal{G}^{(M)}_{\mathbf{i}_{max}}| \leq \varepsilon |\mathcal{G}_{\mathbf{i}_{max}}|\ .$$

Figure 3.3 shows the effects of applying this criteria at two accuracy levels. Notice that the optimal values of C_0 obtained for the two different accuracies are quite close. This observation motivates us to optimize C_0 on a sequence of accuracy thresholds ε_j, $j = 1, \ldots, m$, with the grading $\varepsilon_j / \varepsilon_{j+1} = O(10)$ and $\varepsilon_1 = 10^{-1}$. Therefore, we minimize the Kronecker rank for each ε_j and provide the resulting optimal $C_0 = C_0(\varepsilon_j)$ as an initial guess for ε_{j+1}, until the

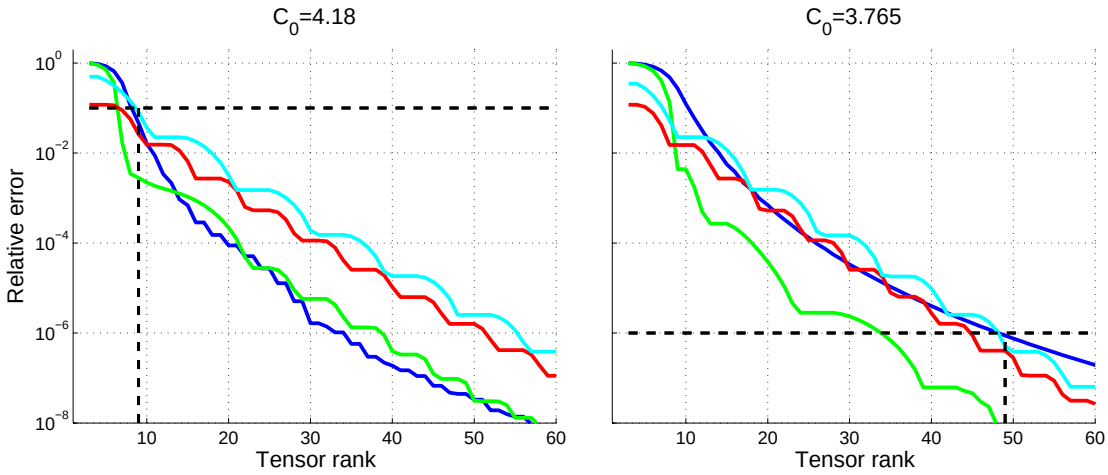

Figure 3.3: Optimal configurations of the error curves for accuracies $\varepsilon = 10^{-1}$ (left) and $\varepsilon = 10^{-6}$ (right) (dashed lines).

desired accuracy ε_m is achieved. This strategy avoids to deal with unnecessary higher ranks if the initial value of C_0 is far from the optimal one.

Finally, Table 3.1 summarizes the results for various values of n and ε.

ε	$p = 11$	$p = 13$	$p = 15$
10^{-4}	3.61 , 30	3.9 , 34	4.19 , 37
10^{-6}	3.765 , 49	4.04 , 53	4.28 , 58
10^{-8}	3.947 , 73	4.189 , 79	4.43 , 85
10^{-10}	4.161 , 97	4.409 , 105	4.65 , 112

Table 3.1: Optimal values of (C_0, r) for different grid sizes $n = 2^p - 1$.

3.3 Near-far fields decomposition (NFFD)

We can further reduce the tensor-rank using, instead of $\mathcal{G}_{\mathbf{i}_{min}}$ in the optimisation of C_0, the tensor element $\mathcal{G}_{\tilde{\mathbf{i}}_{min}}$, with $\tilde{\mathbf{i}}_{min} = (1, 1, 2)$ (the one with the second lowest decay of $\mathcal{B}$ in t). Thus, the task is now to find a new pair $(\tilde{C}_0, \tilde{M})$, so that for a given accuracy ε we have

$$|\mathcal{G}_{\mathbf{i}}^{(\tilde{M})} - \mathcal{G}_{\mathbf{i}}| \leq \varepsilon \, |\mathcal{G}_{\mathbf{i}}| \quad \text{for } \mathbf{i} \neq \mathbf{i}_{min} \ . \tag{3.9}$$

Respectively, the value $\mathcal{G}_{\mathbf{i}_{min}}^{(\tilde{M})}$ has to be corrected as

$$|(\mathcal{G}_{\mathbf{i}_{min}}^{(\tilde{M})} + \delta\mathcal{G}_{\mathbf{i}_{min}}) - \mathcal{G}_{\mathbf{i}_{min}}| \leq \varepsilon \, |\mathcal{G}_{\mathbf{i}_{min}}| \quad \text{with } \delta\mathcal{G}_{\mathbf{i}_{min}} := \mathcal{G}_{\mathbf{i}_{min}}^{(M)} - \mathcal{G}_{\mathbf{i}_{min}}^{(\tilde{M})} \ . \tag{3.10}$$

Now we express the new compressed tensor $\tilde{\mathcal{G}}^{(\tilde{M})}$ with the Kronecker rank $\tilde{r} = \tilde{M} + 2$ as

$$\mathcal{G} \approx \tilde{\mathcal{G}}^{(\tilde{M})} = \sum_{k=0}^{\tilde{M}} \tilde{g}_k \bigotimes_{\ell=1}^{d} \mathbf{B}^{(\ell)}(\tilde{t}_k) + \delta\mathcal{G}_{\mathbf{i}_{min}} \bigotimes_{\ell=1}^{d} \mathbf{e}_1^{(\ell)} = \sum_{k=0}^{\tilde{M}+1} \bar{g}_k \bigotimes_{\ell=1}^{d} \bar{\mathbf{B}}_k^{(\ell)} \tag{3.11}$$

with

$$\bar{g}_k = \begin{cases} \tilde{g}_k & \text{for } 0 \le k \le \tilde{M} \\ \delta\mathcal{G}_{\mathbf{i}_{min}} & \text{for } k = \tilde{M}+1 \end{cases} \quad \text{and} \quad \bar{\mathbf{B}}_k^{(\ell)} = \begin{cases} \mathbf{B}^{(\ell)}(\tilde{t}_k) & \text{for } 0 \le k \le \tilde{M} \\ \mathbf{e}_1^{(\ell)} & \text{for } k = \tilde{M}+1 \ , \end{cases} \tag{3.12}$$

where $\tilde{g}_k$ and $\tilde{t}_k$ are computed by formula (3.6) and (3.5) respectively with parameters $\tilde{C}_0$ and $\tilde{M}$ and

$$\left\{\mathbf{e}_1^{(\ell)}\right\}_{i_\ell} = \begin{cases} 1 & i_\ell = 1 \\ 0 & \text{otherwise} \ . \end{cases} \tag{3.13}$$

In Figure 3.4 we compare the error curves with optimal Kronecker rank for the two situations, with and without NFFD in the optimisation procedure. Note that the $\tilde{C}_0$-quadrature applied to $\mathcal{G}_{\mathbf{i}_{min}}$ converges much slower than for the other entries, indicating the need of a correction. Table 3.2 shows some optimal values for $\tilde{C}_0$ and $\tilde{r}$ obtained with the algorithm based on the NFFD strategy.

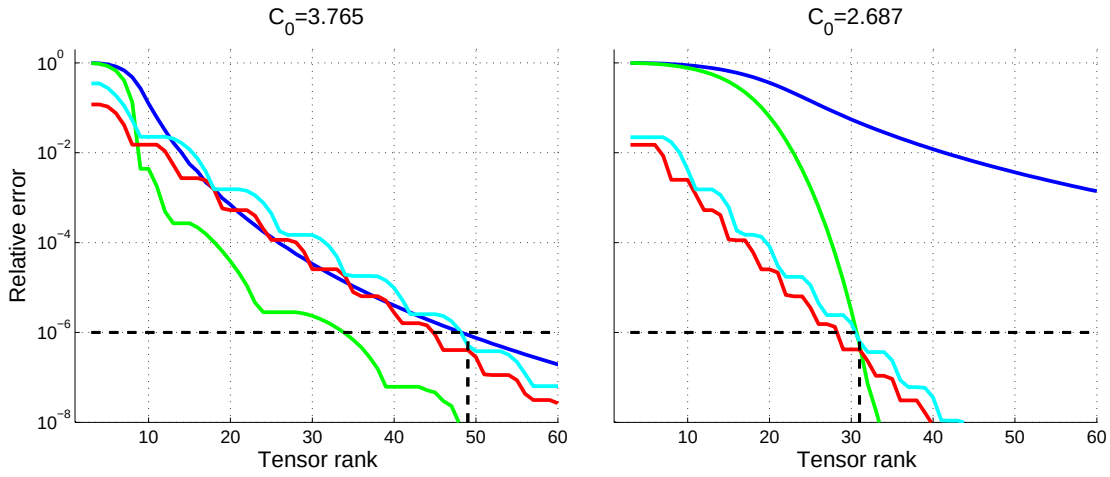

Figure 3.4: Optimal configurations without (left) and with (right) NFFD for $\varepsilon = 10^{-6}$.

ε	$p = 11$	$p = 13$	$p = 15$
10^{-4}	2.88 , 23	3.195 , 27	3.46 , 30
10^{-6}	2.687 , 32	2.97 , 37	3.26 , 41
10^{-8}	2.547 , 42	2.816 , 48	3.083 , 54
10^{-10}	2.461 , 51	2.724 , 58	2.983 , 65

Table 3.2: Optimal values of $(\tilde{C}_0, \tilde{r})$ for different grid sizes $n = 2^p - 1$, and $\varepsilon > 0$.

3.4 Black-box algorithm for tensor-rank optimisation

In this section, we will summarize the particular algorithm realizing the optimized Tensor decomposition of the Galerkin projected Newton kernel (TGN).

Algorithm TGN. Given the computational domain $\Omega = [0, b]^3$, the one-dimensional grid size n, the target accuracy $\varepsilon_* > 0$, and the level number $m = \lceil \log_{10} 1/\varepsilon_* \rceil$:

1. For $\tilde{\mathbf{i}}_{min} = (1, 1, 2)$ and $\mathbf{i}_{max} = (n, n, n)$, compute the reference values for $\mathcal{G}_{\tilde{\mathbf{i}}_{min}}$ and $\mathcal{G}_{\mathbf{i}_{max}}$ by approximating the integral (2.3) through a quadrature over the domain $\Omega_{\mathbf{i}}$, $\mathbf{i} = \tilde{\mathbf{i}}_{min}, \mathbf{i}_{max}$ (e.g., by a trapezoidal rule), with an accuracy higher than ε_*.

2. Initialize $C_0 = 3$, $\delta C_0 = 0.5$ and $\tilde{M}^0 = 3$ (recommended). For $j = 1, ..., m$:

   (a) Set $\varepsilon_j = 10^{-j}$, $p = 1$.

   (b) Find $M_1^j, M_2^j > \tilde{M}^{j-1}$ such that

   $$|\mathcal{G}_{\tilde{\mathbf{i}}_{min}} - \mathcal{G}_{\tilde{\mathbf{i}}_{min}}^{(M_1^j)}| \le \varepsilon_j |\mathcal{G}_{\tilde{\mathbf{i}}_{min}}| \quad \text{and} \quad |\mathcal{G}_{\mathbf{i}_{max}} - \mathcal{G}_{\mathbf{i}_{max}}^{(M_2^j)}| \le \varepsilon_j |\mathcal{G}_{\mathbf{i}_{max}}|,$$

   with

   $$\mathcal{G}_{\mathbf{i}}^{(M)} = \sum_{k=0}^{M} g_k \prod_{\ell=1}^{d} \mathbf{B}_{i_\ell}^{(\ell)}(t_k),$$

   g_k, t_k computed using quadrature formula (3.5)-(3.6) with $\mathfrak{h}_M = C_0 \log(M)/M$, and $\mathbf{B}_{i_\ell}^{(\ell)}(t_k)$ evaluated by approximating integral in (2.6).

   (c) Compute $s_p = \text{sign}\{M_1^j - M_2^j\}$ and if $p = 1$, set $s_0 = s_1$. Then:

   i. If $s_p = 0$, set $\tilde{M}^j = M_1^j$, $j = j + 1$, and $p = 1$ and go to Step 2b, or continue with Step 3 if $j = m$.

   ii. If $s_p = s_{p-1} \neq 0$, set $C_0 = C_0 + s_p \delta C_0$, $p = p + 1$ and repeat Step 2b.

   iii. If $s_p \neq s_{p-1}$, set $\delta C_0 = \alpha \delta C_0$, $C_0 = C_0 + s_p \delta C_0$, $p = p + 1$ and repeat Step 2b (with, e.g., $\alpha = 0.2$).

3. Let $\tilde{M} := \tilde{M}^m$ and $\tilde{C}_0 = C_0$ be the optimal quadrature parameters computed at Step 2. Set $\mathbf{i}_{min} = (1, 1, 1)$, and calculate the correction $\delta\mathcal{G}_{\mathbf{i}_{min}} = \mathcal{G}_{\mathbf{i}_{min}}^{(M_\infty)} - \mathcal{G}_{\mathbf{i}_{min}}^{(\tilde{M})}$, with $\mathcal{G}_{\mathbf{i}_{min}}^{(M_\infty)}$ computed with the quadrature parameters $\tilde{C}_0$, and choosing M_∞ large enough in order to satisfy

   $$|\mathcal{G}_{\mathbf{i}_{min}}^{(M_\infty)} - \mathcal{G}_{\mathbf{i}_{min}}^{(M_\infty - 1)}| \le \varepsilon_* |\mathcal{G}_{\mathbf{i}_{min}}^{(M_\infty - 1)}|.$$

4. Finally, compute the canonical vectors of the tensor $\tilde{\mathcal{G}}^{(\tilde{M})}$ from Equations (3.11)-(3.13), with the quadrature parameters $\tilde{C}_0$, $\tilde{M}$, $\delta\mathcal{G}_{\mathbf{i}_{min}}$ and the resulting tensor-rank $\tilde{r} = \tilde{M} + 2$.

We would like to estimate now the computational complexity of Algorithm TGN. For this purpose, we assume that each evaluation of the function $\mathbf{B}_{i_\ell}^{(\ell)}(t)$ costs $O(1)$. The result is summarized in the following Lemma.

Lemma 3.3 *For given $\varepsilon_* > 0$, and with fixed $\tilde{r}$, the total computational cost of Algorithm TGN can be estimated by*

$$O\left(d\,\tilde{r}\,n + d\tilde{r}^2\right) = O\left(d\,n\,\log 1/\varepsilon_* + d\,\log^2 1/\varepsilon_*\right).$$

Proof. It is clear that Step 1 has a cost $O(1)$, and that Step 4 is of $O(d\,\tilde{r}\,n) = O\left(d\,n\,\log 1/\varepsilon_*\right)$ complexity in view of the estimation $r = O\left(\log 1/\varepsilon_*\right)$ stipulated by the sinc-quadratures. However, for both Steps 2 and 3 we have to proceed more carefully.

Observing that for going from $j \to j+1$ in Step 2, M_1^j and M_2^j are searched starting from $\tilde{M}^{j-1}$, the complexity of Step 2 can be estimated by the expression

$$O\left(\sum_{j=1}^{m} N_j 2\,d \sum_{i=M_{1,2}^{j-1}}^{M_{1,2}^j} i\right), \tag{3.14}$$

with N_j the number of repetitions of the Step 2b (typically $N_j \leq 10$, equivalent to p in the algorithm explanation) and $M_{1,2}^j = \max\{M_1^j, M_2^j\}$.

Based on the exponential convergence of the quadrature presented in Section 3.1 and observed in the numerical experiments, we can assume that there is a bound for the rank in terms of the accuracy level j of the type

$$M_{1,2}^j \leq \bar{M}^j := \bar{C}\left(1 + \frac{\tilde{M}-1}{m} j\right)\ ,\ \bar{C} = O(1). \tag{3.15}$$

Then, replacing this expression in equation (3.14) and computing the double sum we obtain

$$N\,d\,\left(\tilde{M}^2 + m\,\tilde{M} + m - 1\right) < N\,d\,\log^2 1/\varepsilon_*, \tag{3.16}$$

since $m = O(\tilde{M}) = \log 1/\varepsilon_*$.

Analogously (but simpler), for Step 3 the complexity can be (roughly) estimated as

$$d/2\,\left(M_\infty^2 + M_\infty - \tilde{M}^2 - \tilde{M}\right) < d\,\tilde{r}^2 = O\left(d\,\log^2 1/\varepsilon_*\right),$$

assuming that $M_\infty = O(\tilde{M})$, which completes the proof. ∎

We summarize some numerical results in Table 3.3. There, t_{ref}, t_{opt} and t_{ass} correspond to the computing time for all reference values, rank optimisation and assembly of the canonical vectors, respectively.

We first note that t_{ref} is neglectable with respect to the others, except for very high accuracies due to the quadrature method used (MATLAB's `triplequad` with an adaptive Lobatto scheme in our case). However, these high accuracies are not used in practice, since the computation of the Frobenius norm can only be peformed until machine precision, i.e., $\varepsilon_*^2 \approx 10^{-16}$.

Note that the behavior $O(n)$ is also verified for t_{ass}. However, it remains practically invariable by increasing the rank $\tilde{r}$, that happens due to the vectorized format of the MATLAB's function `quadv` whereby the evaluation of $\mathbf{B}_{i_\ell}^{(\ell)}(t_k)$, for $k = 1, \ldots, M$, is performed. This has also an impact on t_{opt}, which grows in practice as $O(\tilde{r})$.

n	ε_*	$\tilde{r}$	t_{ref} [s]	t_{opt} [s]	$t_{ass}[s]$
500	10^{-2}	12	0.06	0.41	0.27
	10^{-5}	24	0.06	0.85	0.28
	10^{-11}	49	21	1.7	0.29
2000	10^{-2}	14	0.06	0.5	1
	10^{-5}	28	0.06	1.2	1
	10^{-11}	57	21	2.3	1.1
8000	10^{-2}	16	0.06	0.7	4.1
	10^{-5}	32	0.06	1.5	4.3
	10^{-11}	64	21.2	2.7	4.5
32000	10^{-2}	18	0.06	0.64	16.9
	10^{-5}	36	0.06	1.7	17.2
	10^{-11}	69	20.8	3.1	18.38

Table 3.3: Complexity results for the Algorithm TGN.

4 Generalisation of the method

4.1 Extension to other types of basis functions

When solving equation (1.1) by the finite element method the set of Galerkin continuous piecewise polynomial basis functions is usually required. Thus, we apply our scheme in a straightforward way to $\mathcal{G}_{\mathbf{i}}$ but now using tensor-product piecewise linear polynomials in (2.2),

$$\psi_{i_\ell}(x_\ell) = \begin{cases} \dfrac{x_\ell - (i_\ell - 1)h}{h} & \text{if } (i_\ell - 1)\, h \le x_\ell < i_\ell\, h \\ \dfrac{(i_\ell + 1)\, h - x_\ell}{h} & \text{if } i_\ell\, h \le x_\ell \le (i_\ell + 1)\, h \\ 0 & \text{otherwise,} \end{cases} \tag{4.1}$$

for the index set $i_\ell \in I_\ell = \{-n, \ldots, n\}$, corresponding to the computational domain $(-1, 1)^3$.

As said in the first sections, in electronic structure calculations it is of special interest to compute the Galerkin matrix $\mathcal{N} \in \mathbb{R}^{\mathcal{I}\times\mathcal{I}}$

$$\mathcal{N} = [\mathcal{N}_{\mathbf{ij}}]_{\mathbf{i},\mathbf{j}\in\mathcal{I}} \ , \ \mathcal{N}_{\mathbf{ij}} \ = \ \int_\Omega \frac{\boldsymbol{\psi}_{\mathbf{i}}(\mathbf{x})\boldsymbol{\psi}_{\mathbf{j}}(\mathbf{x})}{\|\mathbf{x}\|}\, \mathrm{d}\mathbf{x} \ , \tag{4.2}$$

using the basis functions in (4.1). Given the accuracy $\varepsilon > 0$, the optimised quadrature parameters for both tensors $\mathcal{G}$ and $\mathcal{N}$ are presented in Table 4.1.

Notice that the extension of our scheme to the case of high order polynomials does not change the basic concept of the method.

ε	$\mathcal{G}$	$\mathcal{N}$
10^{-4}	3.365 , 29	3.53 , 30
10^{-6}	3.39 , 43	3.52 , 46
10^{-8}	3.453, 63	3.582 , 66

Table 4.1: Optimal values of $(\tilde{C}_0, \tilde{r})$ for the case of Galerkin tensors $\mathcal{G}$ and $\mathcal{N}$ via piecewise linear basis functions with $n = 2047$ and $\Omega = [0, 1]^3$.

4.2 Extension to the Yukawa potential

As it was mentioned in the introduction, we would like also to approximate in tensor product format the Yukawa kernel integrated over a tensor-product set of basis functions as,

$$\mathcal{Y} = \int_{\Omega} \frac{e^{-\lambda\|\mathbf{x}\|}}{\|\mathbf{x}\|} \boldsymbol{\psi}_{\mathbf{i}}(\mathbf{x})\, \mathrm{d}\mathbf{x}\ ,\ \lambda > 0, \quad \mathbf{i} \in \mathcal{I}. \tag{4.3}$$

Analogous to the case of Newton kernel, we apply the gaussian integral to the Yukawa kernel in the form [8],

$$\frac{e^{-\lambda\|\mathbf{x}\|}}{\|\mathbf{x}\|} = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} e^{-\|\mathbf{x}\|^2 t^2 - (\lambda/2t)^2}\, \mathrm{d}t = \frac{1}{\sqrt{\pi}} \int_{\mathbb{R}} e^{-(\lambda/2t)^2} \prod_{\ell=1}^{d} e^{-t^2 (x_\ell)^2}\, \mathrm{d}t\ ,\ \mathbf{x} \in \Omega\ . \tag{4.4}$$

Inserting (4.4) into (4.3), and applying the Fubiny theorem, we obtain

$$\int_{\Omega} \frac{e^{-\lambda\|\mathbf{x}\|}}{\|\mathbf{x}\|} \boldsymbol{\psi}_{\mathbf{i}}(\mathbf{x})\, \mathrm{d}\mathbf{x} = \int_{\mathbb{R}} e^{-\frac{\lambda^2}{4t^2}}\, \mathcal{B}_{\mathbf{i}}(t)\, \mathrm{d}t\ , \tag{4.5}$$

with $\mathcal{B} = [\mathcal{B}_{\mathbf{i}}]_{\mathbf{i}\in\mathcal{I}}$ being the same tensor as in (2.7).

Although (4.5) differs from (2.7) only by the factor $e^{-\frac{\lambda^2}{4t^2}}$, the respective quadrature may converge much slower because the integrand has a sharp decay as $t \to 0$. The numerical experiments show that in this case it is more preferable to use the exponential-type nonsymmetric *sinc*-quadrature with the following parameters [8],

$$t_k^{(M)} = \exp(k\mathfrak{h}_M) \quad \text{and} \quad g_k^{(M)} = \mathfrak{h}_M \exp(k\mathfrak{h}_M), \quad \mathfrak{h}_M = C_0 \log(M)/M, \tag{4.6}$$

for $-M \le k \le M$ and $r = 2M + 1$.

Table 4.2 shows the results obtained for the Yukawa potential by repeating the Algorithm TGN presented in Section 3.4, but now with quadrature formula (4.6) and $\tilde{r} = 2\tilde{M} + 1$. It is important to note that quadrature (4.6) generates many small components for large values of λ, and hence some of them can be simply eliminated keeping the same accuracy in terms of the Frobenius norm with respect to the target tensor. This filtering of the canonical vectors implies a further tensor-rank reduction $\tilde{r} \to \hat{r}$, with $\tilde{r} > \hat{r}$.

5 On the rank optimality and conclusions

It is interesting to analyze if the rank obtained through the aforementioned heuristic method is close to the quasioptimal one obtained by the algebraic Tucker/canonical decompositions.

ε	$\lambda = 0.1$	$\lambda = 1$
10^{-4}	2.935 , 32→24	2.725 , 42→26
10^{-6}	2.637 , 52→38	2.515 , 60→37
10^{-8}	2.478 , 68→51	2.395 , 78→48

Table 4.2: Optimal values of $(\tilde{C}_0, \tilde{r} \to \hat{r})$ for $n = 2047$ and $\lambda = 0.1, 1$.

For this purpose, we generate (with Algorithm TGN) the low rank tensors with higher accuracy $\varepsilon = 10^{-9}$, and apply algebraic rank optimisation.

Figure 5.1 illustrates the tensor rank vs. relative error for best algebraic recompressions via the multigrid orthogonal Tucker decomposition [10] and for those obtained with NFFD applied to our sample tensor. In fact, it is known that the rank-r orthogonal Tucker model provides the lower bound for the canonical rank R, i.e., $r \leq R$.

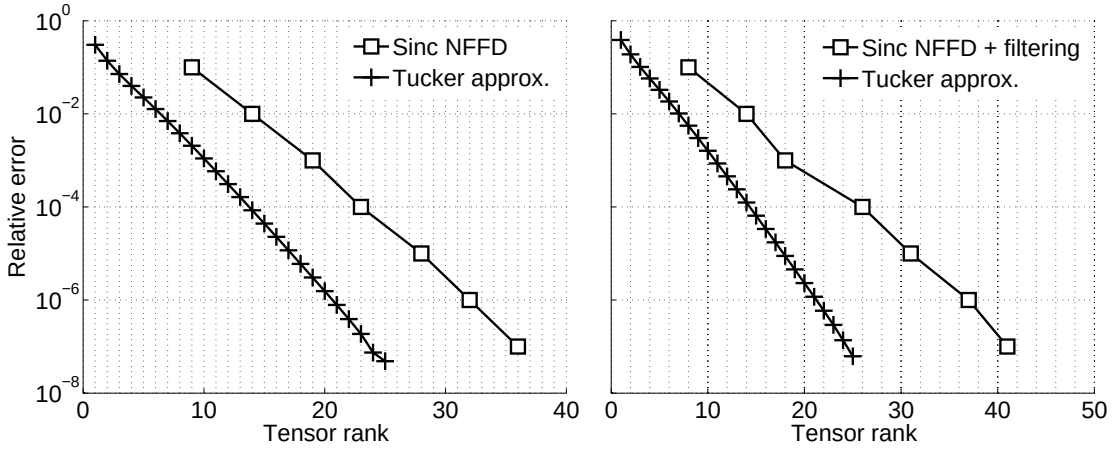

Figure 5.1: Rank recompression results for the sample tensor with $n = 2047$ and $\Omega = [0, 1]^3$ for both the Newton (left) and Yukawa ($\lambda = 1$) potentials.

Finally, we conclude that this work presents a methodology to approximate L^2-Galerkin projections of Green kernels in $\mathbb{R}^3$ onto the set of tensor-product basis functions, through canonical tensor-product sums. We apply analytic quadrature-based numerical decompositions followed by the algebraic optimizations, leading to almost optimal tensor ranks. The computation procedure is represented as the black-box scheme whose complexity scales linearly in the univariate grid size, $O(n)$.

The analytic step allows to decompose the volume integral representation of our Green's kernels over $\Omega \in \mathbb{R}^d$ into another integral of a family of rank-1 tensors (discrete separable functions) over $\mathbb{R}$. Then, we use improved *sinc*-quadratures to approximate this integral numerically. Here, the choice of the free parameter C_0 in the quadrature is crucial to reduce the number of evaluation points to reach the target accuracy, uniformly for all tensor entries.

After the description of the rank minimization scheme, we present optimal values for C_0 and tensor rank r for different grid sizes, and different sets of basis functions. The numerical

results encourage to further applications of this strategy to other type of Green kernels, like the Yukawa potential. Concerning the computational cost, the optimization involves only a small number of evaluations of some tensor elements. Therefore, the main cost of our scheme is (asymptotically) dominated by the calculation of the canonical vectors constituting the complete tensor from the optimized quadrature, that is of order $\mathcal{O}(n)$. The latter step is well paralellisable since the tensor entries can be computed independently. In practice, only few seconds are required for the complete rank decomposition on large spacial grids up to $n = 32000$.

Algorithm TGN was already successfully applied in numerical computations of various 3D convolution integrals [7] included in the Fock operator of the nonlinear Hartree-Fock equation in 3D, see [9, 10, 11, 12]. In particular, this includes fast multiple computations of the Coulomb and exchange convolution integrals in the tensor-structured numerical methods for solving the *ab initio* Hartree-Fock equation on large $n \times n \times n$ Cartesian grids, in the range $n \leq 10^4$, see [12].

Acknowledgements

The authors are thankful to Prof. Wolfgang Hackbusch (MPI MiS, Leipzig) for valuable discussions on the topic. We appreciate Venera Khoromskaia (MPI MiS, Leipzig) for providing the compression to Tucker format, and Marco Müller (University of Leipzig) for the assistance in testing Algorithm TGN.

References

[1] I.P. Gavrilyuk, W. Hackbusch and B.N. Khoromskij. *Data-Sparse Approximation to Operator-Valued Functions of Elliptic Operator.* Math. Comp. **73** (2003), 1297-1324.

[2] I.P. Gavrilyuk, W. Hackbusch and B.N. Khoromskij. *Data-Sparse Approximation to a Class of Operator-Valued Functions.* Math. Comp. **74** (2005), 681-708.

[3] I. P. Gavrilyuk, W. Hackbusch, and B. N. Khoromskij. *Tensor-product approximation to elliptic and parabolic solution operators in higher dimensions.* Computing **74** (2005), 131-157.

[4] W. Hackbusch *Entwicklungen nach Exponentialsummen.* Preprint 4/2005, MPI-MiS Leipzig.

[5] W. Hackbusch and B.N. Khoromskij. *Low-rank Kronecker product approximation to multi-dimensional nonlocal operators. Part I. Separable approximation of multi-variate functions.* Computing **76** (2006), 177-202.

[6] R.J. Harrison, G.I. Fann, T. Yanai, Z. Gan, and G. Beylkin. *Multiresolution quantum chemistry: Basic theory and initial applications.* J. of Chemical Physics, 121 (23): 11587-11598, 2004.

[7] B. N. Khoromskij. *Fast and Accurate Tensor Approximation of Multivariate Convolution with Linear Scaling in Dimension.* Preprint 36/2008, MPI MiS Leipzig (J. Comp. Appl. Math., accepted).

[8] B.N. Khoromskij. *On Tensor Approximation of Green Iterations for Kohn-Sham Equations.* Comput. and Visualization in Sci., **11** (2008) 259-271.

[9] B.N. Khoromskij and V. Khoromskaia. *Low Rank Tucker-Type Tensor Approximation to Classical Potentials.* Central European J. of Math. **5**(3) 2007, 1-28.

[10] B.N. Khoromskij and V. Khoromskaia. *Multigrid tensor approximation of function related multi-dimensional arrays.* SIAM J. on Sci. Comp., **31**(4), 3002-3026 (2009).

[11] B. N. Khoromskij, V. Khoromskaia, S. R. Chinnamsetty, H.-J. Flad. *Tensor Decomposition in Electronic Structure Calculations on 3D Cartesian Grids.* J. of Comput. Phys. 228 (2009), 5749-5762.

[12] B.N. Khoromskij, V. Khoromskaia, and H.-J. Flad. *Numerical Solution of the Hartree-Fock Equation in Multilevel Tensor-structured Format.* Preprint 44/2009, MPI MiS Leipzig (submitted).

[13] C. Le Bris. *Computational chemistry from the perspective of numerical analysis.* Acta Numerica (2005), 363 - 444.

[14] J. Lund and K.L. Bowers. *Sinc Methods for Quadrature and Differential Equations.* SIAM, Philadelphia, 1992.

[15] F. Stenger. *Numerical Methods Based on Sinc and Analytic Functions.* Springer Verlag, 1993.