Max-Planck-Institut
für Mathematik
in den Naturwissenschaften
Leipzig

On the efficient convolution with the Newton potential

by

Wolfgang Hackbusch, Kishore Kumar Naraparaju, and Jan Schneider

Preprint no.: 53 2009

On the efficient convolution with the Newton potential

W. Hackbusch*, K.K. Naraparaju* and J. Schneider*
*Max-Planck-Institute for Mathematics in the Sciences
Inselstrasse 22-26, D-04103 Leipzig, Germany.

Abstract

The convolution $\int_{\mathbb{R}^d} \frac{1}{\|x-y\|} f(y)dy$, where f is smooth, except for some local singularities, arises for example in electronic structure calculations. An efficient convolution with the Newton potential in d dimensions has been proposed in [3]. The convolution is approximated on a refined grid and additional approximations are introduced for efficient evaluation. This paper studies the performance of the method and a precise error analysis of the method is discussed.

Key Words: Convolution, Refined grid, Tensor product representation, Pointwise smoothness, 2-microlocal spaces

1 Introduction

Consider $f \in L^2(R^d)$. The integral

$$u(x) = (\mathcal{K}f)(x) = \int_{\mathbb{R}^d} \frac{1}{\|x-y\|} f(y)dy \text{ for } x \in \mathbb{R}^d \tag{1}$$

represents the convolution of the Newton potential $1/\|.\|$ with f. For $d = 3$, the kernel $\frac{1}{4\pi\|x-y\|}$ is the fundamental solution of the Laplace operator $\triangle$ in $\mathbb{R}^3$. Hence, up to a factor, u from (1) is the solution of the Poisson equation $-\triangle u = f$ in $\mathbb{R}^3$.

Assume that the given function f is smooth except for some local singularities and f is assumed to have a bounded support. This models the electron density of a many-particle (which is rather smooth except in the neighbourhood of the atom centres) in electron structure calculations. The most efficient representation of such a function f is an *hp* approach using local refinements as well as high polynomial degrees.

An efficient convolution with the Newton potential has been proposed by W. Hackbusch in [3]. The function is approximated in S, the space generated by piecewise orthonormal basis functions $\Phi^l_{\mathbf{i},\alpha}$ (have the smallest possible support) defined on a locally refined grid combining the regular grids of size $h_l = 2^{-l}$ ($l < 0$) in the refinement zones

$$B_l = [-2h_l\mathbf{1}, 2h_l\mathbf{1}) = \{x \in \mathbb{R}^d : -2h_l \le x_j \le 2h_l \text{ for } 1 \le j \le d\} \subseteq B_{l-1}.$$

Instead of computing the integral (1) exactly, u is approximated on the same kind of mesh (project u orthogonally onto S). In the projection, the coefficients

$$\int_{\mathbb{R}^d}\int_{\mathbb{R}^d} \frac{\Phi^{l'}_{\mathbf{i},\alpha}\Phi^{l}_{\mathbf{j},\beta}}{\|x-y\|}dxdy$$

need to be computed (the basis functions $\Phi^l_{\mathbf{i},\alpha}$ are from different levels l and l'). Further discretization errors are introduced in order to get a fast evaluation of the above integrals. This

approximation is justified under restrictive assumptions on the regularity of the given function, see Example 3.3 in [3].

The precise analysis of the approximation is the subject of this article, where our work can be considered as a continuation of [3]. Since the given function f has singular behaviour in the neighbourhood of a point x_0 (only one singular point is considered for simplicity), the main difficulty is the formulation of the pointwise smoothness of the function. 2-microlocal spaces $C^{s,s'}_{x_0}$, where $s, s' > 0$ are the natural choice to describe pointwise smoothness of such functions [7, 8]. The advantage of the 2-microlocal setting is that it provides a characterization in terms of wavelet coefficients. The additional approximation in the method is justified in the framework of 2-microlocal spaces and also a pointwise error estimate for the approximation of the convolution is obtained.

The contents of this paper are organized as follows. The representation of the given function f is described in Section 2. In Section 3, the approximation of u is described. Error analysis of the added approximation is discussed in Section 4. In Section 5, computational results are provided.

2 Representation of f

The given function f is smooth except for some local singularities. We approximate the given function f efficiently by piecewise polynomials defined on the locally refined grid (fine mesh near the singularities and the coarser mesh in the regular part of the support of f). The generation of the refined grid and the representation of the function f is described below. For convenience of the reader we follow the same notation as in [3].

Uniform grids in $\mathbb{R}^d$

Let $h_l = 2^{-l}$ for $l \in \mathbb{Z}$ be the step sizes corresponding to the l th level ($l < 0$ associated with a large step size). Define multidimensional intervals (cubes) by

$$\mathbf{I}^l_v = [vh_l, (v+\mathbf{1})h_l) = I^l_{v_1} \times I^l_{v_2} \times ... \times I^l_{v_d} \quad \text{for} \quad v \in \mathbb{Z}^d, l \in \mathbb{Z},$$

where the position $v = (v_1, v_2, .., v_d)$ is used as a multi-index. This defines the mesh $M_l = \left\{\mathbf{I}^l_v \,:\, v \in \mathbb{Z}^d\right\}$ for $l \in \mathbb{Z}$.

Let $P_{\alpha_\delta}(x)$ be the Legendre polynomial of degree α_δ defined in $[-1, 1]$. Then the basis functions $\Phi^l_{v,\alpha}$ of degree $\alpha = (\alpha_1, \alpha_2, .., \alpha_d)$ are defined as tensor products of univariate basis functions:

$$\Phi^l_{v,\alpha}(x_1, x_2, ..x_d) = \prod_{\delta=1}^{d} \Phi^l_{v_\delta,\alpha_\delta}(x_\delta) \qquad (v \in \mathbb{Z}^d, l \in \mathbb{Z}, \mathbf{0} \le \alpha \le p\mathbf{1}),$$

where the univariate basis function $\Phi^l_{v_\delta,\alpha_\delta}(x)$ is defined as

$$\Phi^l_{v_\delta,\alpha_\delta}(x_\delta) = \begin{cases} \sqrt{\frac{2v_\delta+1}{h_l}} P_{\alpha_\delta}(-1 - 2v_\delta + \frac{2x_\delta}{h_l}) & \text{if } x_\delta \in [v_\delta h_l, (v_\delta+1)h_l), \\ 0 & \text{otherwise.} \end{cases}$$

The basis functions $\Phi^l_{v,\alpha}$ have support in $\mathbf{I}^l_v \in M_l$ and satisfy $\left\|\Phi^l_{v,\alpha}\right\|_{L^2} = 1$. Then the space

$$S_l = \operatorname{span}\left\{\Phi^l_{v,\alpha} \,:\, v \in \mathbb{Z}^d, \mathbf{0} \le \alpha \le p\mathbf{1}\right\}$$

forms an orthonormal system.

The basis functions $\Phi^l_{v,\alpha}$ of subsequent levels satisfy the relation

$$\Phi^l_{v,\eta} = \sum_{\mathbf{0}\le\alpha\le\eta} \sum_{\mathbf{0}\le\mathbf{i}\le\mathbf{1}} \xi_{\eta,\alpha,\mathbf{i}} \Phi^{l+1}_{2v+\mathbf{i},\alpha} \text{ with } \xi_{\eta,\alpha,\mathbf{i}} = (-1)^{<\eta+\alpha,\mathbf{1}-\mathbf{i}>} \xi_{\eta,\alpha},\ \xi_{\eta,\alpha} = \prod_{\delta=1}^{d} \xi_{\eta_\delta,\alpha_\delta}, \tag{2}$$

where the coefficients $\xi_{\eta_\delta,\alpha_\delta}$ are independent of l and v_δ and can easily be computed, see [3].

Locally refined mesh in $\mathbb{R}^d$

As usual we refine our mesh in the neighbourhood of each singular point. Now we explain its construction: Restrict the infinitely many levels $l \in \mathbb{Z}$ to $\underline{L} \le l \le \overline{L}$ and associate a sequence of nested boxes (considered as centered in a singular point)

$$\emptyset \neq B_{\overline{L}} \subset .. \subset B_{l+1} \subset B_l ... \subset B_{\underline{L}} \subset R^d \text{ with } B_l = [\mathbf{a}_l h_l, \mathbf{b}_l h_l),$$

where $\mathbf{a}_l, \mathbf{b}_l \in 2\mathbb{Z}^d$ and $[\mathbf{a}_l h_l, \mathbf{b}_l h_l) = [a_{l,1} h_l, b_{l,1} h_l) \times [a_{l,2} h_l, b_{l,2} h_l) \times .... \times [a_{l,d} h_l, b_{l,d} h_l)$.
In order to avoid an immediate jump from a fine step to a neighbouring coarse step size, we require that the boxes B_l are properly nested. The precise condition is: There is some $m \in \mathbb{N}$ with $\text{dist}_\infty(\partial B_{l-1}, \partial B_l) \ge m h_{l-1}$.

Approximate f in the refined mesh defined by

$$M = \{\mathbf{I}_v^l \in M_l : \mathbf{I}_v^l \subset B_l \setminus B_{l+1}\}.$$

The refined mesh in two dimension is shown in Fig. 1. The corresponding function space is given by

$$S = \text{span}\{\Phi_{v,\alpha}^l : \mathbf{I}_v^l \in M, \mathbf{0} \le \alpha \le p\mathbf{1}\}.$$

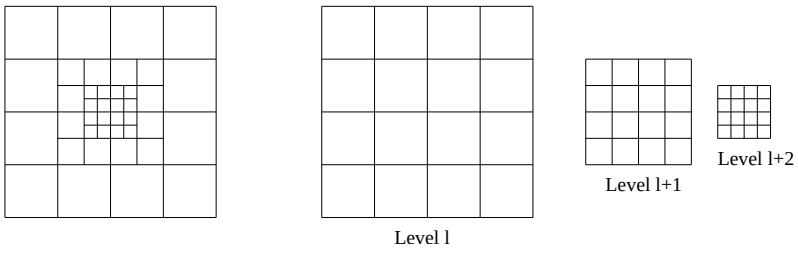

Figure 1: Refined mesh in two dimension for $m = 1$

The basis functions in S are orthonormal. Any function in S has a support contained in $B_{\underline{L}}$.

Approximation of f

From the very beginning we consider f as already approximated in S but keep the symbol f for the approximated function. By the definition of S there are coefficients $f_{\mathbf{j},\beta}^l$ of $f \in S$ so that

$$f = \sum_{l=\underline{L}}^{\overline{L}} f^l \text{ with } f^l = \sum_{\mathbf{j},\beta} f_{\mathbf{j},\beta}^l \Phi_{\mathbf{j},\beta}^l \in S_l,$$

where $\text{supp}(f^l) \subset \overline{B_l \setminus B_{l+1}}$. The representation of f is unique.

Remark 1: *We can allow a representation*

$$f = \sum_{l=\underline{L}}^{\overline{L}} f^l \text{ with } f^l = \sum_{\mathbf{j},\beta} f_{\mathbf{j},\beta}^l \Phi_{\mathbf{j},\beta}^l \in S_l \text{ where } supp(f^l) \subset \overline{B_l}.$$

In this case the restriction $f\,|_{B_l \setminus B_{l+1}}$ may involve contributions from all levels $l, l-1, .., \underline{L}$. This representation is not unique.

3 Convolution with $\frac{1}{\|x\|}$

The convolution with the Newton potential defines the operator

$$u(x) = (\mathcal{K}f)(x) = \int_{\mathbb{R}^d} \frac{f(y)}{\|x-y\|} dy.$$

As described above we use for the function f its representation in the space S. Instead of computing the above integral exactly, we approximate u by projecting orthogonally on to S.

Let P, P_l denote the L^2 orthogonal projection of the function onto S and S_l respectively. Then the Galerkin approximation of $\mathcal{K}$ is given by $P\mathcal{K}P$. Therefore, for $f \in S$, the function $u = P\mathcal{K}f \in S$ is to be determined.

The coefficients of $u \in S$ are given by

$$u = \sum_{l'=\underline{L}}^{\overline{L}} \sum_{\substack{\mathbf{i} \in \mathbb{Z}^d \text{ with} \\ \mathbf{I}_{\mathbf{i}}^{l'} \subset B_l \setminus B_{l+1}}} \sum_{\mathbf{0} \le \alpha \le p\mathbf{1}} u_{\mathbf{i},\alpha}^{l'} \Phi_{\mathbf{i},\alpha}^{l'} \quad \text{with} \quad u_{\mathbf{i},\alpha}^{l'} = < \mathcal{K}f, \Phi_{\mathbf{i},\alpha}^{l'} > .$$

Since $f \in S$ is a linear combination of certain $\Phi_{\mathbf{j},\beta}^{l}$, the quantities

$$< \mathcal{K}\Phi_{\mathbf{j},\beta}^{l}, \Phi_{\mathbf{i},\alpha}^{l'} > = \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \frac{\Phi_{\mathbf{i},\alpha}^{l'} \Phi_{\mathbf{j},\beta}^{l}}{\|x-y\|} dxdy \tag{3}$$

need to be computed.

Remark 2:

(a). If $l' = l$, the basis functions in the above integral are at the same level.

(b). If $l' \neq l$, it can be reduced to the case $l' = l$. Let $l' > l$. Replace $\Phi_{\mathbf{j},\beta}^{l}$ by basis functions of level $l+1$, see formula (2). By repeating this process $(l'-l)$ times, we get the case $l' = l$. But this procedure increases the data size by $2^{l'-l}$.

To overcome the above disadvantage, an approximation of $P\mathcal{K}P$ which requires (3) for only equal levels $l = l'$ has been introduced in [3]. A brief description of the approximation is given below.

Modified approximation

The approximation to be defined is based on the smoothness property of the convolution. Roughly speaking, the key idea is that the image $\mathcal{K}\phi$ is smoother than the function ϕ itself. A precise analysis of this point is described in Section 4.

Consider $f^l \in S_l$ with support B_l. The image $u = P\mathcal{K}f^l \in S$ is the exact Galerkin result. Fix a level λ and consider the restriction $u^\lambda = u|_{B_\lambda \setminus B_{\lambda+1}}$. This function extended by zero outside belongs to S_λ. Hence $u^\lambda = P_\lambda \mathcal{K} P_l f^l$ holds. As mentioned before, the computation of $P_\lambda \mathcal{K} P_l f^l$ for $\lambda \neq l$ is possible, but expensive. Instead we replace now $P_\lambda \mathcal{K} P_l$ by $P_{min\{\lambda,l\}} \mathcal{K} P_{min\{\lambda,l\}}$. Then the construction looks as follows:

Represent $f \in S$ in the form $f = \sum_l f^l$ with $f^l \in S_l$, $\text{supp}(f^l) \subset \overline{B}_l$. Then $u = \sum_l P\mathcal{K}f^l$ and we approximate now each term $P\mathcal{K}f^l \in S$ by $u_l = \sum_{\lambda=\underline{L}}^{l} u^{\lambda,l}$ with

$$u^{l,l} = (P_l \mathcal{K} P_l f^l) \mid_{B_l},$$
$$u^{\lambda,l} = (P_\lambda \mathcal{K} P_\lambda f^l) \mid_{B_\lambda \setminus B_{\lambda+1}}, \text{ for } \underline{L} \le \lambda < l.$$

Altogether the approximation of $\mathcal{K}f$ is

$$u = \sum_{l=\underline{L}}^{\overline{L}} \sum_{\lambda=\underline{L}}^{l} u^{\lambda,l}.$$

The most accurate result is to be expected for the representation of f with $\mathrm{supp}(f^l) \subset \overline{B_l \setminus B_{l+1}}$. Due to the approximation, the coefficients (3) are now only needed for the case $l' = l$. The evaluation of the convolution at level l is described below.

Remark 3: *Note that the replacement is only done for* $x \in B_\lambda$ *with* $\mathrm{dist}(x, B_l) \geq h_\lambda$, *see the construction of* B_l *in section 2 (or section 3.3 in [3]).*

Convolution at level l

Since $\Phi^l_{\mathbf{i},\alpha}(x) = \Phi^l_{\mathbf{0},\alpha}(x - \mathbf{i}h_l)$,

$$N^l_{\mathbf{i}-\mathbf{j},\alpha,\beta} = \int\int \frac{\Phi^l_{\mathbf{i},\alpha}(x)\Phi^l_{\mathbf{j},\beta}(y)}{\|x-y\|}dxdy = \int\int \frac{\Phi^l_{\mathbf{i}-\mathbf{j},\alpha}(x)\Phi^l_{\mathbf{0},\beta}(y)}{\|x-y\|}dxdy$$

depends on the difference $\mathbf{i} - \mathbf{j}$.

The L^2 orthogonal projection of $F(x) = \int \frac{f^l(y)}{\|x-y\|}dy$ onto S_l yields

$$P_l F = P_l \mathcal{K} f^l = \sum_{\mathbf{i}\in\mathbb{Z}^d} \sum_{\mathbf{0}\leq\alpha,\beta\leq p\mathbf{1}} \left(\sum_{\mathbf{j}\in\mathbb{Z}^d} N^l_{\mathbf{i}-\mathbf{j},\alpha,\beta} f^l_{\mathbf{j},\beta} \right) \Phi^l_{\mathbf{i},\alpha}.$$

Set

$$F^l_{\mathbf{i},\alpha} = \sum_{\mathbf{0}\leq\beta\leq p\mathbf{1}} \sum_{\mathbf{j}\in\mathbb{Z}^d} N^l_{\mathbf{i}-\mathbf{j},\alpha,\beta} f^l_{\mathbf{j},\beta}.$$

Besides β summation it is a discrete convolution and can be obtained by fast Fourier Transform (FFT), provided the quantities $N^l_{\mathbf{k},\alpha,\beta}$ are known. $F^l_{\mathbf{i},\alpha}$ are the coefficients of

$$P_l F = \sum_{\mathbf{i}\in\mathbb{Z}^d} \sum_{\mathbf{0}\leq\alpha\leq p\mathbf{1}} F^l_{\mathbf{i},\alpha} \Phi^l_{\mathbf{i},\alpha}.$$

Let $N = \#\{\mathbf{j} : \mathbf{j}h_l \in B_l\}$ be the data size. Applying fast Fourier transform, the above discrete convolution requires $O(Np^d logN)$ arithmetical operations.

Computation of $N^l_{\mathbf{i},\alpha,\beta}$

The coefficients $N^l_{\mathbf{i},\alpha,\beta} = \int\int \frac{\Phi^l_{\mathbf{i},\alpha}(x)\Phi^l_{\mathbf{0},\beta}(y)}{\|x-y\|}dxdy$ can be determined exactly as described in [5] at least for $d \leq 3$. An efficient approximation has been proposed in [3]. First, we obtain the coefficients $N^l_{\mathbf{i},\alpha,\beta}$ for $\|\mathbf{i}\|_\infty \geq 2$ (far field where the integrands are smooth) recursively. Once $N^l_{\mathbf{i},\alpha,\beta}$ are known for $2 \leq \|\mathbf{i}\|_\infty \leq 3$, the coefficients $N^l_{\mathbf{i},\alpha,\beta}$ for $\|\mathbf{i}\|_\infty \leq 1$ (near field where the integrands are singular) can be computed by simple algebraic operations without any quadrature error. In [3], it has been shown that the coefficients $N^l_{\mathbf{i},\alpha,\beta}$ can be determined efficiently and inexpensively for a fixed range of indices $\mathbf{i}$ with $0 \leq i_1 \leq i_2 \leq .. \leq i_d$ and $\alpha \leq \beta$ once for all.

Tensor product approximation

So far the approximation of the convolution in $\mathbb{R}^d$ is discussed in the original form. In this case, for $d \geq 3$ the data size may become large because of the exponent d. The tensor product

approximation reduces the data size and the computational complexity. A brief description on the tensor product approximation is given below.

If the function has the product form

$$f(x) = f_1(x_1).f_2(x_2)....f_d(x_d),$$

and the kernel function had the form

$$k(x) = k_1(x_1).k_2(x_2)....k_d(x_d),$$

then the convolution

$$\begin{aligned}(k * f)(x) &= \int_{\mathbb{R}^d} k(x-y)f(y)dy \\ &= \left(\int_{\mathbb{R}} k(x_1-y_1)f_1(y)dy_1\right) ... \left(\int_{\mathbb{R}} k(x_d-y_d)f_1(y)dy_d\right) \\ &= (k_1 * f_1)(x_1)....(k_d * f_d)(x_d)\end{aligned}$$

could be reduced to d one-dimensional convolutions. For simplicity, here only one term is considered in the tensor product representations of f and k. In general they are represented by the sum of r such terms (r is called the rank). The data size and the computational work increase only proportionally to d if the spatial dimension increases. The complete details of the tensor product representation of the Newton potential $\frac{1}{\|x-y\|}$ and the evaluation of the convolution is given in [3]. The details of tensor approximation for our numerical experiments are described in section 5.

4 Justification of the approximation

In this Section, we justify the approximation defined in Section 2 and study the smoothness of the convolution u.

Error analysis.

Remark 4: *Although our numerical results (section 5) are based on the polynomial approach, described in the previous sections, we will justify our approximation scheme in the wavelet setting, because this somehow covers the polynomial case. In fact, we can write f in (multi-) wavelet setting by splitting f^l orthogonally into $f^l = P_{l-1}f^l + (I - P_{l-1})f^l$ for $\underline{L}+1 \leq l \leq \bar{L}$, then we have*

$$f = \sum_{l=\underline{L}}^{\bar{L}} f^l \quad \textit{with } f^l \in S_l \quad \textit{and } f^l \perp S_\lambda \quad \textit{for } \lambda < l.$$

In [3] (section 3.3) the same argumentation is given in more details.

As described in Section 3, we have replaced $P_\lambda \mathcal{K} P_l$ by $P_{min\{\lambda,l\}}\mathcal{K}P_{min\{\lambda,l\}}$ (for $\lambda < l$ and the condition from Remark 3). This replacement will now be justified below. We assume that the source function f and the convolution u are represented in the same grid system.
For simplicity, we assume that f exhibits singular behaviour at only one point x_0 in its bounded support. Away from x_0 the function f is considered to be more regular. First, let us consider the one dimensional case.

To formulate the variable smoothness of the function, we use the setting of the 2-microlocal spaces $C_{x_0}^{s,s'}$, which were designed to model local smoothness behaviour, see [7, 8] for details. In those spaces $C_{x_0}^{s,s'}$ the parameter s reflects the smoothness at the point x_0 where its difference

to the smoothness in the neighbourhood is given by s'. We are interested in the case $s, s' > 0$, where the functions are at least continuous and x_0 is a point of "bad" smoothness s compared to its "better" surrounding with $s + s'$. For those situations the spaces $C_{x_0}^{s,s'}$ are the natural choice. Even more, because they provide a wavelet characterization (see Proposition 1.4. in [7]): Let us represent the function f as

$$f(x) = \sum_{j,k \in \mathbb{Z}} d_{j,k}(f)\psi_{j,k}(x)$$

in the wavelet system $\{\psi_{j,k}\}$ with coefficients $d_{j,k}(f)$, where we assume that the system is orthonormal with sufficient regularity and $|\operatorname{supp} \psi_{j,k}| \leq 2^{-j} |\operatorname{supp} \psi_0|$ (ψ_0 is the mother wavelet, see [1, 2]). Then f belongs to $C_{x_0}^{s,s'}$ if and only if

$$|d_{j,k}(f)| \leq C 2^{-(\frac{1}{2}+s)j} (1 + 2^j |x_0 - 2^{-j}k|)^{-s'}. \tag{4}$$

For $s' > 0$ this decay condition is weaker at x_0 than away from it, hence, it fits well to what we said above.

Now, consider a point y in the support of the function $f \in C_{x_0}^{s,s'}$. The absolute error $\delta f_l = f - P_l f$ at the level l in the wavelet approximation of f at the point y is given by

$$|\delta f_l(y)| = \left| \sum_{j=l+1}^{\infty} \sum_{i=1}^{N} d_{j,k_{j,i}} \psi_{j,k_{j,i}}(y) \right|,$$

where the summation over i is due to the contribution of the wavelet functions at the point y, which are defined on different intervals whose index is denoted by $k_{j,i}$, $i = 1, \ldots, N$, at each level (due to the overlapping of the wavelets). Using (4), we have

$$\begin{aligned} |\delta f_l(y)| &\leq C \sum_{j=l+1}^{\infty} \sum_{i=1}^{N} h_j^{(\frac{1}{2}+s)} (1 + 2^j \left|x_0 - 2^{-j} k_{j,i}(y)\right|)^{-s'} \left|\psi_{j,k_{j,i}}(y)\right| \\ &\leq C' \sum_{j=l+1}^{\infty} \sum_{i=1}^{N} h_j^{(\frac{1}{2}+s)} h_j^{-\frac{1}{2}} (1 + 2^j \left|x_0 - 2^{-j} k_{j,i}(y)\right|)^{-s'} \\ &= C' \sum_{j=l+1}^{\infty} \sum_{i=1}^{N} h_j^s (1 + 2^j |x_0 - 2^{-j} k_{j,i}(y)|)^{-s'}. \end{aligned} \tag{5}$$

Now we want to get rid of the sum over i. Because N does not depend on j we can estimate

$$\sum_{i=1}^{N} h_j^s (1 + 2^j \left|x_0 - 2^{-j} k_{j,i}(y)\right|)^{-s'} \leq N h_j^s (1 + 2^j \left|x_0 - 2^{-j} k_j^*(y)\right|)^{-s'}, \tag{6}$$

where $2^{-j} k_j^*(y)$ is the nearest point to x_0 among $2^{-j} k_{j,i}(y)$, $i = 1, \ldots, N$. Notice that in case $s' > 0$ we have

$$\left(1 + 2^j |a - b|\right)^{-s'} \leq \left(1 + 2^j |a - c|\right)^{-s'} \left(1 + 2^j |b - c|\right)^{s'}$$

for all $a, b, c \in \mathbb{R}$. Combining this with (5) and (6) we can write

$$\begin{aligned} |\delta f_l(y)| &\leq C'' \sum_{j=l+1}^{\infty} h_j^s (1 + 2^j |x_0 - y|)^{-s'} (1 + 2^j |y - 2^{-j} k_j^*(y)|)^{s'} \\ &\leq C''' \sum_{j=l+1}^{\infty} h_j^s (1 + 2^j |x_0 - y|)^{-s'}, \end{aligned}$$

since $|y - 2^{-j} k_j^*(y)| \leq \tilde{c} 2^{-j}$ by the support property of the ψ_j's.

As $j \to \infty$, the value $(1+2^j\,|x_0-y|)^{-s'}$ is bounded by $(1+2^{l+1}\,|x_0-y|)^{-s'}$ and we estimate further

$$\begin{aligned} |\delta f_l(y)| &\lesssim (1+2^{l+1}\,|x_0-y|)^{-s'} \sum_{j=l+1}^{\infty} h_j^s \\ &\lesssim h_{l+1}^s (1+2^{l+1}\,|x_0-y|)^{-s'} \\ &\lesssim h_l^s (1+2^l|x_0-y|)^{-s'}. \end{aligned}$$

Now we apply our kernel $k(x,y) = \|x-y\|^{-1}$, which is smooth off the diagonal and certainly satisfies there also

$$0 < \hat{c} \le \left| \frac{\partial}{\partial y} k(x,y) \right| \le \widehat{C},$$

for all $y \in I_v^l$. Therefore, we have

$$\begin{aligned} \left| \int_{I_v^l} k(x,y)\delta f_l(y)dy \right| &= \left| \int_{I_v^l} [k(x,y)-k(x,y')]\delta f_l(y)dy \right| \quad (\text{since } \int_{I_v^l} \delta f_l(y)dy = 0) \\ &\lesssim h_l^s \int_{I_v^l} \left| k(x,y)-k(x,y') \right| (1+2^l|x_0-y|)^{-s'} dy \\ &\lesssim h_l^{s+1} \int_{I_v^l} \left| \frac{\partial}{\partial y} k(x,\tilde{y}) \right| (1+2^l|x_0-y|)^{-s'} dy \quad (\text{by meanvalue theorem}) \\ &\lesssim h_l^{s+1} \int_{I_v^l} \left| \frac{\partial}{\partial y} k(x,y) \right| (1+2^l|x_0-y|)^{-s'} dy \end{aligned} \tag{7}$$

Now consider the three dimensional case. Let y be a point in $\mathbf{I}_v^l$, then similar arguments as above show

$$|\delta f_l(y)| \lesssim h_l^s (1+2^l|x_0-y|)^{-s'}$$

and (7) reads as

$$\left| \int_{\mathbf{I}_v^l} \frac{\delta f_l(y)}{\|x-y\|} dy \right| \lesssim h_l^{s+1} \int_{\mathbf{I}_v^l} \frac{(1+2^l|x_0-y|)^{-s'}}{\|x-y\|^2} dy. \tag{8}$$

Finally, let's study the effect of the replacement of $P_\lambda \mathcal{K} P_l$ by $P_\lambda \mathcal{K} P_\lambda$ on the approximation of the convolution function. The refinement is done such that $h_l^s(1+2^l|x_0-x|)^{-s'} \le \varepsilon$ at each point x in the support of the function f. Therefore, an error of $u = \mathcal{K}f$ of the size $\int_{B_l} \frac{\epsilon}{\|x-y\|} dy$ is acceptable.

Let $x \in B_\lambda \setminus B_{\lambda+1}$ and $y \in B_l$, where $l > \lambda$. Replacing $P_\lambda \mathcal{K} P_l$ by $P_\lambda \mathcal{K} P_\lambda$ means to replace δf_l in B_l by δf_λ. Using (8), δf_λ leads to the error contribution

$$h_\lambda^{s+1} \int_{B_l} \frac{(1+2^\lambda|x_0-y|)^{-s'}}{\|x-y\|^2} dy \tag{9}$$

from B_l.

Since the replacement of $P_\lambda \mathcal{K} P_l$ by $P_\lambda \mathcal{K} P_\lambda$ is only made if $\|x-y\| \ge h_\lambda$, $\frac{h_\lambda^{1+s}}{\|x-y\|^2} \le \frac{h_\lambda^s}{\|x-y\|}$ holds. Furthermore, there exists a constant $\tilde{C}$ such that

$$(1+2^\lambda|x_0-y|)^{-s'} \le \tilde{C}(1+2^\lambda|x_0-x|)^{-s'},$$

Using the last two inequalities to estimate the error in (9), it follows by our refinement strategy, that the error contribution from B_l is of acceptable size:

$$h_\lambda^{s+1} \int_{B_l} \frac{(1+2^\lambda|x_0-x|)^{-s'}}{\|x-y\|^2} dy \lesssim \int_{B_l} \frac{\varepsilon dy}{\|x-y\|}.$$

Smoothness of the convolution function u

Now, we test the behaviour of the convolution function $u(x)$ in the integral

$$u(x) = (\mathcal{K}f)(x) = \int_{\mathbb{R}^3} \frac{1}{\|x-y\|} f(y)dy \text{ for } x \in \mathbb{R}^3.$$

The convolution function is the solution of the Poisson equation $-\triangle u = f$ up to a factor. As described above, let us assume that $f \in C_{x_0}^{s,s'}$. Here $s, s' > 0$.

Since f belongs to the two-microlocal space $C_{x_0}^{s,s'}$, by Theorem 4 in [8], $u(x)$ belongs to $C_{x_0}^{s+2,s'}$. The convolution u is smoother than f. The coefficients $d_{j,k}$ in the wavelet expansion of u have the following decay property [7]

$$|d_{j,k}| \leq C2^{-(\frac{3}{2}+s+2)j}(1+2^j\left|x_0-2^{-j}k\right|)^{-s'}.$$

One can easily show that the absolute error δu_l at the level l in the wavelet approximation of u at the point y is given by

$$|\delta u_l(y)| \leq C' h_l^{s+2}(1+2^l|x_0-y|)^{-s'}.$$

The proof of the above estimate easily follows from the estimates obtained before. This is the error in the wavelet expansion of the convolution of the function f with the Newton potential at level l.

The above estimates shows that u does not need a more refined grid. In general, u may also need another grid which might be very different from the f-grid and it may require a more extended grid in the case of a fast decaying function f because u will decay less strongly than f.

5 Numerical Results

In this Section we provide the numerical results to show the efficiency of the method. As explained in Section 3, to evaluate the discrete convolution at a level l, we need to compute the coefficients $N^l_{\mathbf{i},\alpha,\beta} = \int\int \frac{\Phi^l_{\mathbf{i},\alpha}(x)\Phi^l_{\mathbf{0},\beta}(y)}{\|x-y\|}dxdy$ for a fixed range of the indices $\mathbf{i}, \alpha, \beta$. Instead of exact computation an easy and good approximation has been proposed in [3].

Since $N^l_{\mathbf{i},\alpha,\beta} = 2^{l(1-d)}N^0_{\mathbf{i},\alpha,\beta}$ (from the properties of $N^l_{\mathbf{i},\alpha,\beta}$), it is sufficient to determine $N^0_{\mathbf{i},\alpha,\beta}$ only for the level $l = 0$.

We evaluated the coefficients $N^0_{\mathbf{i},\alpha,\beta}$ (dimension $d = 3$) efficiently for a fixed range of index $\mathbf{i}$ and for the polynomial degrees $1 \leq \alpha_i, \beta_i \leq 6,\ i = 1,2,3$ using the algorithm proposed in [3]. The coefficients $N^0_{\mathbf{i},\alpha,\beta}$ for some values of the indices $\mathbf{i}$,α,β are shown in Table 1. The complete data of the coefficients $N^0_{\mathbf{i},\alpha,\beta}$ for a range of indices $\|\mathbf{i}\|_\infty \leq 3,\ 0 \leq \alpha, \beta \leq p\mathbf{1}(p = 6)$ and $\|\mathbf{i}\|_\infty \leq 7,\ 0 \leq \alpha, \beta \leq p\mathbf{1}(p = 5)$ can be found in www.mis.mpg.de/scicomp/Gacoeff. Now we consider a few numerical examples to check the accuracy and efficiency of the method in computing the convolution. Although we justified the method in a much more general framework (2-microlocal spaces modelling functions with

$\mathbf{i}, \alpha, \beta$	$N^0_{\mathbf{i},\alpha,\beta}$
(1,1,1)(0,0,0)(0,0,0)	0.5787968780
(1,1,1)(0,1,0)(0,0,1)	-0.0171748095
(1,1,1)(1,1,1)(1,1,1)	-2.4206346E-3
(0,1,1)(0,0,0)(0,0,0)	0.7084949688
(0,1,1)(0,0,1)(0,0,1)	-0.0192376213
(0,1,1)(0,1,1)(0,0,0)	0.0437745422
(0,0,1)(0,0,0)(0,0,0)	0.9808849307
(0,0,1)(0,0,0)(0,0,1)	0.2506269809
(0,0,1)(1,1,1)(1,1,1)	-0.0280493477
(0,0,0)(0,0,0)(0,0,0)	1.8823119346
(0,0,0)(0,0,1)(0,0,1)	0.4388291645
(0,0,0)(1,1,1)(1,1,1)	0.1369316507
(0,0,0)(2,2,2)(2,2,2)	0.0486651243
(0,0,0)(6,6,6)(6,6,6)	0.0071713666

Table 1: $N^0_{\mathbf{i},\alpha,\beta}$ for some values of $\mathbf{i}, \alpha, \beta$

pointwise singularities), we basically restrict ourselves here to sums of Gaussians, which in quantum chemistry usually model the electron density.

Example 1: Consider

$$f(x_1, x_2, x_3) = e^{-(x_1^2+x_2^2+x_3^2)}$$

in the cube $[-4, 4]^3$. Since the function is decaying for $\|x\| \to \infty$ with less regular behaviour at $x = \mathbf{0} = (0, 0, 0)$, the simplest refinement structure uses

$$B_l = 2h_l[-\mathbf{1}, \mathbf{1}) = [-2h_l, 2h_l)^3.$$

Consider 3 levels $l = -1, 0, 1$. Here, $h_{-1} = 2; h_0 = 1; h_1 = 0.5$. As mentioned in section 3 the data size N in each level is equal to $4^3 = 64$ in this refinement strategy.

For simplicity, we have fixed the polynomial degree $p = 6$ uniformly on the refined grid. The construction of intervals at various levels in univariate direction is shown in Fig. 2.

As described in Section 3, the tensor product approximation of the given function f and kernel $k(x, y) = \frac{1}{\|x-y\|}$ leads (1) to 3 one dimensional convolutions. It reduces the computational complexity of the method. The Newton potential $\frac{1}{\|x-y\|}$ is approximated by exponential sums [3]

$$\frac{1}{\|x-y\|} \approx \sum_{\nu=1}^{k} \omega_\nu \prod_{\delta=1}^{3} \exp\left(-\vartheta_\nu (x_\delta - y_\delta)^2\right).$$

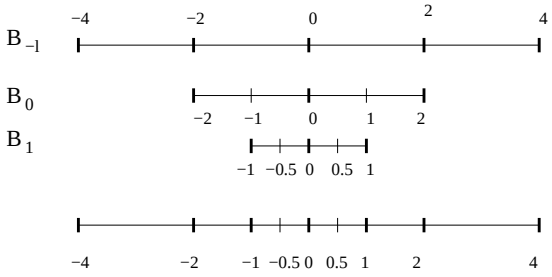

Figure 2: Mesh in univariate direction

In all of our numerical examples in this section, we considered the rank $k = 15$. The coefficients ω_ν and ϑ_ν corresponding to $k = 15$ are given in [4]. For $\|\mathbf{i}\|_\infty \geq 2$, the coefficients $N^l_{\mathbf{i},\alpha,\beta}$ are approximated by $E^l_{\mathbf{i},\alpha,\beta}$ (tensor product form) with sufficiently small error. For $\|\mathbf{i}\|_\infty \leq 1$, an additional correction $\delta N^l_{\mathbf{i},\alpha,\beta} = N^l_{\mathbf{i},\alpha,\beta} - E^l_{\mathbf{i},\alpha,\beta}$ is necessary. A tensor representation of the error $\delta N^l_{\mathbf{i},\alpha,\beta}$ has been described in [3]. In all of our numerical examples, we used rank 3 tensor representation of $\delta N^l_{\mathbf{i},\alpha,\beta}$.

An approximation solution is obtained on the refined grid and compared with the exact solution at some points of the cube. The exact value of the convolution of the function $f = e^{-(x_1^2+x_2^2+x_3^2)}$ with the Newton potential at a point ($\neq \mathbf{0}$) is given by

$$\frac{\pi^{3/2}}{r}\mathrm{erf}(r),$$

where r is the Euclidean distance of the point from the origin $\mathbf{0}$ and erf is the error function [9]. The exact value at the origin $\mathbf{0}$ is 2π. The exact values of u, approximated values and the error at some points along the diagonal in $B_{-1} \setminus B_0$ are shown in Table 2.

Point	Exact	Appr	Error
(4.000,4.000,4.000)	0.803718917001020	0.80371907694237	1.59940E-7
(3.875,3.875,3.875)	0.829645333653398	0.82964505243810	2.81215E-7
(3.750,3.750,3.750)	0.857300178134421	0.85729979718413	3.80950E-7
(3.625,3.625,3.625)	0.886862253242505	0.88686193313274	3.20109E-7
(3.500,3.500,3.500)	0.918535905116262	0.91853568199939	2.23116E-7
(3.375,3.375,3.375)	0.952555753453901	0.95255560280310	1.50650E-7
(3.250,3.250,2.250)	0.989192513202126	0.98919240826608	1.04493E-7
(3.125,3.125,3.125)	1.028760213730193	1.02876015985222	5.38779E-8
(3.000,3.000,3.000)	1.071625222635424	1.07162525002545	2.7390E-8
(2.875,2.875,2.875)	1.118217623617682	1.11821773548813	1.11870E-7
(2.750,2.750,2.750)	1.169045697401677	1.16904580232783	1.04926E-7
(2.625,2.625,2.625)	1.224714539998645	1.22471443865771	1.01340E-7
(2.500,2.500,2.500)	1.285950265987240	1.28594977802592	4.87961E-7
(2.375,2.375,2.375)	1.353631852087357	1.35363108474215	7.67345E-7
(2.250,2.250,2.250)	1.428833579301809	1.42883307013285	5.09168E-7
(2.125,2.125,2.125)	1.512882374017987	1.51288246098072	8.69627E-8
(2.000,2.000,2.000)	1.607436285416955	1.60743641481465	1.29397E-7

Table 2: The exact, approximated values of u and the error at some points in $B_{-1} \setminus B_0$.

The exact values, approximated values of u and the error at some points along the diagonal in $B_0 \setminus B_1$ are shown in Table 3.

Point	Exact	Appr	Error
(1.875,1.875,1.875)	1.714592857373497	1.71459233230972	5.25063E-7
(1.750,1.750,1.750)	1.837038481562164	1.83703820250376	2.79058E-7
(1.625,1.625,1.625)	1.978248926624814	1.9782485384616	3.88163E-7
(1.500,1.500,1.500)	2.142739144042205	2.14273892170467	2.22337E-7
(1.375,1.375,1.375)	2.336321405735543	2.33632152929874	1.23563E-7
(1.250,1.250,1.250)	2.566243260873728	2.56624236681007	8.94063E-7
(1.125,1.125,1.125)	2.840929620064991	2.84092885695697	7.63108E-7
(1.000,1.000,1.000)	3.168884047416819	3.16887980044927	4.24696E-6

Table 3: The exact, approximated values of u and the error at some points in $B_0 \setminus B_1$.

The exact, approximated values of u at some points along the diagonal in B_1 are shown in Table 4.

Point	Exact	Appr	Error
(0.875,0.875,0.875)	3.556245003230128	3.55624148246019	3.52076E-6
(0.750,0.750,0.750)	4.002766338616063	4.00276342410935	2.91450E-6
(0.625,0.625,0.625)	4.496780723171488	4.49677732355138	3.39962E-6
(0.500,0.500,0.500)	5.010889351733840	5.01088837478121	9.76952E-7
(0.375,0.375,0.375)	5.501068343437025	5.50106765407012	6.89366E-7
(0.250,0.250,0.250)	5.911624296406164	5.91162280001335	1.49639E-6
(0.125,0.125,0.125)	6.186375850544813	6.18637228005012	3.57049E-6
(0.000,0.000,0.000)	6.283185307179586	6.28318664385373	1.33667E-6

Table 4: The exact, approximated values of u and the error at some points in B_1.

The error obtained at arbitrary points in the given cube is shown in the Table 5.

Point	Exact	Appr	Error
(4.000,3.000,3.000)	0.95496037306	0.95496056455	2.38418E-7
(4.000,2.000,2.000)	1.13663019271	1.13663030122	1.19209E-7
(4.000,3.000,2.000)	1.03401255045	1.03401285644	3.57627E-7
(2.000,1.000,1.000)	2.27205099840	2.27205278819	1.66893E-6
(1.850,2.000,1.250)	1.85761633774	1.85761613394	1.19209E-7
(2.000,1.500,1.000)	2.06773524745	2.06773591752	7.15255E-7
(0.000,0.100,0.100)	6.24154754038	6.24154852153	9.53674E-7
(0.000,0.250,0.500)	5.68574447872	5.68574426967	2.09049E-7
(0.000,0.750,1.000)	3.90063626630	3.90063630380	3.75071E-8
(0.000,0.750,0.750)	4.28685532160	4.28685425270	1.06890E-6

Table 5: The exact, approximated values of u and the error at some arbitrary points in the cube $[-4,4]^3$.

Run time denotes the sum of the user time and the cpu time measured in seconds. It is obtained by the command "dtime" in the program (Fortran 90). In this example, the run time is 235 seconds.

Now, consider two different mesh refinement strategies in the cube $[-4,4]^3$ (different values of l). We look at the performance of these strategies in terms of local accuracy and computation time. Since the function has less regular behaviour in the neighbourhood of the origin $(0,0,0)$, it is enough to check their performance at the origin. We provide the error of the approximation at the origin in Table 6 ($p=6$ is considered uniformly).

Levels	Error	Run time
$l=-1$	6.03023E-4	100
$l=-1,0$	4.91370E-5	142
$l=-1,0,1$	1.33667E-6	235

Table 6: Error at the origin $(0,0,0)$ for different kinds of mesh.

The method gives better accuracy with the refined grid with three levels $l=-1,0,1$. The accuracy is considerably good also with two levels. A much better accuracy can be expected with higher degree of approximation $p \geq 7$.

Now, we consider the convolution in $[-8,8]^3$. We use the simplest mesh refinement structure

$$B_l = 2h_l[-\mathbf{1},\mathbf{1}) = [-2h_l, 2h_l)^3.$$

with four levels $l=-2,-1,0,1$. Therefore $h_{-2}=4$. Let us consider the polynomial degree $p=6$ uniformly on the refined grid. The approximate solution is obtained and compared with the exact values. The exact, approximated values of u and the error at some points along the diagonal are given in the Table 7. The accuracy does not change so much at the points in the cube $[-4,4]^3$ compared to the values shown in the above Tables. Same rate of convergence is observed even when the convolution is approximated on much larger supports with little increase in the computational time.

Point	Exact	Appr	Error
(8.0,8.0,8.0)	0.401859458488365	0.401859635473206	1.76984E-7
(7.5,7.5,7.5)	0.428650089054255	0.428649781241130	3.07813E-7
(7.0,7.0,7.0)	0.459267952558131	0.459267800885742	1.51672E-7
(6.5,6.5,6.5)	0.494596256601064	0.494596298106986	4.15059E-8
(6.0,6.0,6.0)	0.535812611317819	0.535812776761550	1.65443E-7
(5.5,5.5,5.5)	0.584522848710348	0.584522987448119	1.38737E-7
(5.0,5.0,5.0)	0.642975133581383	0.642974826590719	3.06990E-7
(4.5,4.5,4.5)	0.714416815090426	0.714416410612729	4.04477E-7
(4.0,4.0,4.0)	0.803718917001020	0.803718574662575	3.42338E-7
(3.5,3.5,3.5)	0.918535905116262	0.918535722193669	1.82922E-7
(3.0,3.0,3.0)	1.071625222635424	1.071625295155810	7.25203E-8
(2.5,2.5,2.5)	1.285950265987240	1.285949828443110	4.37544E-7
(2.0,2.0,2.0)	1.607436285416955	1.607436470086180	1.84669E-7
(1.5,1.5,1.5)	2.142739144042205	2.142738980425410	1.63616E-7
(1.0,1.0,1.0)	3.168884047416819	3.168879860829170	4.18658E-6
(0.5,0.5,0.5)	5.010889351733840	5.010890915129430	1.56339E-6
(0.0,0.0,0.0)	6.283185307179586	6.28318670497909	1.39779E-6

Table 7: The exact, approximated values of u and the error at some points in the cube $[-8,8]^3$.

In this case, the run time is 288.25 seconds. Now, consider the refinement structure

$$B_l = 4h_l[-\mathbf{1},\mathbf{1}) = [-4h_l, 4h_l)^3.$$

The same rate of convergence is observed when three levels l ranging from -1 to 1 are considered with the polynomial degree $p = 5$. In this refinement structure N is equal to $8^3 = 512$ in each level. In this case, the coefficients $N^0_{\mathbf{i},\alpha,\beta}$ are needed for a wide range of indices $\mathbf{i}$ ($\|\mathbf{i}\|_\infty \leq 7$) for all ($\mathbf{0} \leq \alpha, \beta \leq p\mathbf{1}$). The run time is 1140 seconds, because of the increased data size in this refinement structure compared to the previous one.

Example 2: Consider the function

$$f(x_1,x_2,x_3) = \frac{10}{\sqrt{\pi}} e^{-100(x_1^2+x_2^2+x_3^2)} + \frac{20}{\sqrt{\pi}} e^{-400(x_1^2+x_2^2+x_3^2)}$$

in $[-2,2]^3$. This function decays faster as $\|x\| \to \infty$ and has a peak at the origin $(0,0,0)$. Consider the simplest refinement structure

$$B_l = 2h_l[-\mathbf{1},\mathbf{1}) = [-2h_l, 2h_l)^3.$$

Since the function f has a large peak at $(0,0,0)$ and decays faster, a very fine mesh is needed in the neighbourhood of the origin. Consider the levels l ranging from 0 to 5 with uniform degree of approximation $p = 5$. Here $h_5 = 2^{-5}$. The exact value of the convolution is given by

$$\frac{\pi}{100}\frac{1}{r}\mathrm{erf}(10r) + \frac{\pi}{400}\frac{1}{r}\mathrm{erf}(20r).$$

The comparison of the approximate values with the exact values of the convolution and the error at some points along the diagonal in $B_0 \setminus B_1$ is given in Table 8.

Point	Exact	Appr	Error
(2.000,2.000,2.000)	0.011336246026464	0.01133614692086207	9.91056E-8
(1.875,1.875,1.875)	0.012091995761561	0.01209189646097236	9.93005E-8
(1.750,1.750,1.750)	0.012955709744530	0.01295559581911199	1.13925E-7
(1.625,1.625,1.625)	0.013952302801801	0.01395219203020205	1.10771E-7
(1.500,1.500,1.500)	0.015114994701951	0.01511488845364178	1.06248E-7
(1.375,1.375,1.375)	0.016489085129402	0.01648895286927748	1.32260E-7
(1.250,1.250,1.250)	0.018137993642342	0.01813782864846653	1.64993E-7
(1.125,1.125,1.125)	0.020153326269269	0.02015314107994888	1.85189E-7
(1.000,1.000,1.000)	0.022672492052927	0.02267247585593861	1.61969E-8

Table 8: The exact, approximated values of u and the error at some points in $B_0 \setminus B_1$.

The error in the approximation of the convolution at some points in $B_1 \setminus B_2$ is shown in Table 9.

Point	Exact	Appr	Error
(0.950,0.950,0.950)	0.023865781108345	0.02386577710502395	4.00332E-9
(0.900,0.900,0.900)	0.025191657836585	0.02519163811448026	1.97221E-8
(0.850,0.850,0.850)	0.026673520062267	0.02667350415009453	1.59121E-8
(0.800,0.800,0.800)	0.02834061506616	0.02834062992489371	1.48587E-8
(0.750,0.750,0.750)	0.030229989403903	0.03023002947593631	4.00720E-8
(0.700,0.700,0.700)	0.032389274361325	0.03238929853974601	2.41784E-8
(0.650,0.650,0.650)	0.034880757004503	0.03488074234103890	1.46634E-8
(0.600,0.600,0.600)	0.037787486754879	0.03778747025090653	1.65039E-8
(0.550,0.550,0.550)	0.041222712823504	0.04122269003321277	2.27902E-8

Table 9: The error in the approximation at some points in $B_1 \setminus B_2$.

In Tables 10 and 11 the exact, approximated values and error at some points in $B_2 \setminus B_3$ and $B_3 \setminus B_4$ are tabulated respectively.

Point	Exact	Appr	Error
(0.500,0.500,0.500)	0.045344984105855	0.0453449473771581	3.67286E-8
(0.450,0.450,0.450)	0.050383315673171	0.0503832820387457	3.36344E-8
(0.400.0.400,0.400)	0.056681230132319	0.0566812476774826	1.75451E-8
(0.350,0.350,0.350)	0.06477854872265	0.0647785824092972	3.36866E-8
(0.300,0.300,0.300)	0.075574973509746	0.0755749270255115	4.64842E-8
(0.250,0.250,0.250)	0.090689968145388	0.090689862675230	1.05470E-7

Table 10: The exact, approximate values at some points in $B_2 \setminus B_3$.

Point	Exact	Appr	Error
(0.225,0.225,0.225)	0.100766628475802	0.100766592404997	3.60708E-8
(0.200,0.200,0.200)	0.113362372897823	0.113362382908686	1.00108E-8
(0.175,0.175,0.175)	0.129555217076436	0.129555253984642	3.69082E-8
(0.150,0.150,0.150)	0.151121099936769	0.151120992970801	1.06965E-7
(0.125,0.125,0.125)	0.18106075891498	0.181059651708292	1.10720E-6

Table 11: The exact, approximate values and error at some points in $B_3 \setminus B_4$.

Point	Exact	Appr	Error
(0.100,0.100,0.100)	0.224130077524767	0.224129683833875	3.93690E-7
(0.075,0.075,0.075)	0.286277462628707	0.286277228187033	2.34441E-7
(0.050,0.050,0.050)	0.372101726207017	0.372098593336344	3.13287E-6

Table 12: The exact, approximate values and error at some points in $B_4 \setminus B_5$.

In Tables 12 and 13 the error at some points in $B_4 \setminus B_5$ and B_5 are tabulated respectively.

Point	Exact	Appr	Error
(0.025,0.025,0.025)	0.474882263815941	0.474882040710773	2.23105E-7
(0.000,0.000,0.000)	0.531736155271655	0.531730081079106	6.07419E-6

Table 13: The exact, approximate values and error at some points in B_5.

The run time in this example is 182 seconds. More accuracy can be achieved using the higher degree of approximation $p = 6$. The same rate of convergence is observed even when the convolution is approximated on much larger supports. We also studied the performance of various refinement structures. But the above described refinement structure gives better results in terms of both the accuracy and computational time.

Now we consider a function which exhibits singular behaviour at more than one point in its bounded support.

Example 3:

Consider the function

$$\begin{aligned} f(x_1,x_2,x_3) &= f_1(x_1,x_2,x_3) + f_2(x_1,x_2,x_3) + f_3(x_1,x_2,x_3) \\ &= \frac{10}{\sqrt{\pi}} e^{-100((x_1-1)^2+(x_2-1)^2+(x_3-1)^2)} + \frac{20}{\sqrt{\pi}} e^{-400(x_1^2+x_2^2+x_3^2)} \\ &+ \frac{10}{\sqrt{\pi}} e^{-100((x_1+1)^2+(x_2+1)^2+(x_3+1)^2)} \end{aligned}$$

in $[-2,2]^3$. Fig. 3 shows the behaviour of the function in two dimension.

The function $f(x_1,x_2,x_3)$ has less regular behaviour in the neighbourhood of the points $(-1,-1,-1)$, $(0,0,0)$ and $(1,1,1)$. We approximate the convolution of the Newton potential with the function f_2 in the cube $[-2,2]^3$ with refined mesh in the neighbourhood of the origin $(0,0,0)$ (level l ranging from 0 to 5), the function f_1 in the cube $[-5,3]^3$ with the refined mesh in the neighbourhood of the point $(-1,-1,-1)$ (level l ranging from -1 to 5) and the function f_3 in the cube $[-3,5]^3$ with refined mesh in the neighbourhood of the point $(1,1,1)$ (level l ranging from -1 to 5). An uniform degree of approximation $p = 5$ is considered. We used the simplest mesh refinement structure

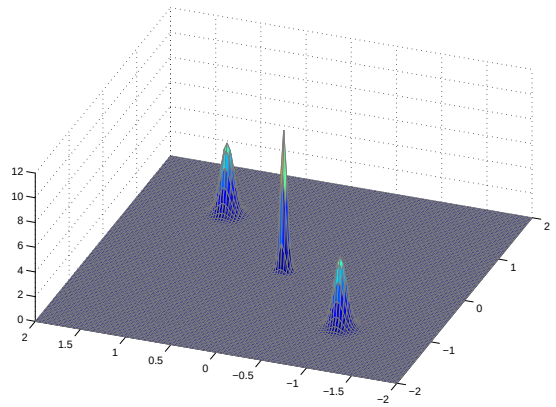

Figure 3: Two dimensional representation of f

$$B_l = 2h_l[-\mathbf{1},\mathbf{1}) = [-2h_l, 2h_l)^3.$$

The approximate values of the convolution at some points in the cube $[-2,2]^3$ are obtained by adding the approximate values of the convolution with the functions f_1, f_2 and f_3 respectively. The exact values of the convolution, approximate values and the error at some points in the neighbourhood of the point $(-1,-1,-1)$ cube $[-2,2]^3$ are shown in the Table 14.

Point	Exact	Appro	Error
(-2.000,-2.000,-2.000)	0.026451240728416	0.026450893945291	3.46783E-7
(-1.875,-1.875,-1.875)	0.029456402097419	0.029456032467360	3.69630E-7
(-1.750,-1.750,-1.750)	0.033370767523790	0.03337042231798	3.45205E-7
(-1.625,-1.625,-1.625)	0.038720962251857	0.038720540909295	4.21342E-7
(-1.500,-1.500,-1.500)	0.046554183682012	0.046553722730530	4.60951E-7
(-1.375,-1.375,-1.375)	0.059302850026797	0.059302433083371	4.16943E-7
(-1.250,-1.250,-1.250)	0.084240903739223	0.084240303549805	6.00189E-7
(-1.125,-1.125,-1.125)	0.157350963337238	0.157350431646816	5.31690E-7
(-1.000,-1.000,-1.000)	0.368094265412860	0.368091560558460	2.70485E-6
(-0.875,-0.875,-0.875)	0.159640652170520	0.159640698696725	4.65262E-8
(-0.750,-0.750,-0.750)	0.088962540179452	0.088962448908805	9.12706E-8
(-0.625,-0.625,-0.625)	0.066785022744624	0.066785067274290	4.45296E-8

Table 14: The exact, approximate values of the convolution and the error at some points in the neighbourhood of $(-1.0,-1.0,-1.0)$.

Table 15 shows the error at some points in the neighbourhood of $(0,0,0)$. Due to the symmetry we can observe the same rates of convergence as shown in Table 14 at the points in the neighbourhood of the point $(1.0,1.0,1.0)$. The run time in this example is 300 seconds. This is the sum of the run times of the approximation of the convolution with the functions f_1 and f_2 respectively. Since the functions f_1 and f_3 are equal, but located at different centres $(-1.0,-1.0,-1.0)$ and $(1.0,1.0,1.0)$, it is enough to find the approximated values of the convolution with one of the functions f_1 and f_3. The approximated values of the convolution with the other function can be easily obtained by shifting the centre. In general, the run time will be the sum of run times of the individual problems.

Point	Exact	Appro	Error
(-0.500,-0.500,-0.500)	0.057436979867417	0.05743695743465003	2.24327E-8
(-0.375,-0.375,-0.375)	0.054304053692831	0.05430405659021483	2.89738E-9
(-0.250,-0.250,-0.250)	0.056832380079339	0.05683238782970265	7.75036E-9
(-0.125,-0.125,-0.125)	0.073127783858187	0.07312771330296847	7.05552E-8
(0.000,0.000,0.000)	0.213521372375236	0.21352000244554086	1.36992E-6
(0.125,0.125,0.125)	0.073127783858187	0.07312771330296847	7.05552E-8
(0.250,0.250,0.250)	0.056832380079339	0.05683238782970265	7.75036E-9
(0.375,0.375,0.375)	0.054304053692831	0.05430405659021482	2.89738E-9
(0.500,0.500,0.500)	0.057436979867417	0.05743695743465003	2.24327E-8

Table 15: The error at some points in the neighbourhood of $(0,0,0)$.

Conclusions

The given function f (with singular behaviour in the neighbourhood of some points) is efficiently approximated on the refined grid. The basis polynomials are of higher order and have the support (smallest possible support) only on one cube. The convolution is approximated on the same grid and further discretizations are considered for an efficient evaluation. The added approximation is justified and pointwise smoothness of the convolution is studied in the framework of 2-microlocal spaces. The smoothness of the convolution is also studied. The numerical

examples show the efficiency of the method. Various refinement structures are considered. The simple refinement structure

$$B_l = 2h_l[-\mathbf{1}, \mathbf{1}) = [-2h_l, 2h_l)^3$$

gives the better results in terms of both accuracy and computation time. A uniform polynomial approximation (for simplicity) is used in all our numerical examples. As a further improvement one can use variable degree of approximation, i.e., a lower order approximation in very fine mesh regions and the higher order approximation in coarser ones on the refined grid.

References

[1] A. Cohen (2003): Numerical analysis of wavelet methods, Studies in Mathematics and its Applications, Elsevier.

[2] I. Daubechies (1988): Orthonormal bases of compactly supported wavelets, Comm. on pure and Appl. Math. 41.

[3] W. Hackbusch (2008): Efficient convolution with the Newton potential in d dimensions. Numer. Math. 110, 449-489.

[4] W. Hackbusch (2005): Entwicklungen nach Exponential summen, Technischer Bericht 4, Max-Planck-Institute fur Mathematik, Leipzig.

[5] W. Hackbusch (2002): Direct integration of the Newton potential over cubes, Computing, 68, 193-216.

[6] W. Hackbusch (2006): Approximation of $\frac{1}{\|x-y\|}$ by exponentials for wavelet applications, Computing, 76:359-366.

[7] S. Jaffard and Y. Meyer (1996): Wavelet methods for pointwise regularity and local oscillations of functions, Memoirs of AMS, Number 587.

[8] S.Jaffard (1991): Pointwise smoothness, two-microlocalization and wavelet coefficients, Publications Mathematiques, Vol 35, 155-168.

[9] T. Helgaker, P.R. Taylor (1995): Gaussian basis sets and molecular integrals: Modern electronic structure theory, part-II, D.R. Yarkony, Ed., World Scientific, Singapore, p. 725.