Fast discrete algorithms for sparse Fourier expansions of high dimensional functions

Ying Jiang\textsuperscript{a}, Yuesheng Xu\textsuperscript{a,b,*}

\textsuperscript{a} Department of Scientific Computing and Computer Applications, Sun Yat-sen University, Guangzhou 510275, PR China
\textsuperscript{b} Department of Mathematics, Syracuse University, Syracuse, NY 13244, USA

\textbf{A R T I C L E I N F O}

\textbf{Article history:}
Received 17 September 2009
Accepted 7 October 2009
Available online 14 October 2009

\textbf{Keywords:}
Fast discrete algorithm
Hyperbolic cross Fourier expansion
Sparse grid

\textbf{A B S T R A C T}

We develop a fast discrete algorithm for computing the sparse Fourier expansion of a function of \(d\) dimension. For this purpose, we introduce a sparse multiscale Lagrange interpolation method for the function. Using this interpolation method, we then design a quadrature scheme for evaluating the Fourier coefficients of the sparse Fourier expansion. This leads to a fast discrete algorithm for computing the sparse Fourier expansion. We prove that this method gives the optimal approximation order \(O(n^{-s})\) for the sparse Fourier expansion, where \(s > 0\) is the order of the Sobolev regularity of the function to be approximated and where \(n\) is the order of the univariate trigonometric polynomial used to construct the sparse multivariate approximation, and requires only \(O(n \log 2^{d-1} n)\) number of multiplications to compute all of its Fourier coefficients. We present several numerical examples with \(d = 2, 3\) and \(4\) that confirm the theoretical estimates of approximation order and computational complexity and compare the numerical performance of the proposed method with that of a well-known existing algorithm. We also have a numerical example for \(d = 8\) to test the efficiency of the propose algorithm for functions of a higher dimension.

\textcopyright 2009 Elsevier Inc. All rights reserved.

1. Introduction

Sparse Fourier expansion of \(d\)-dimensional functions has been introduced for theoretical analysis [1–3] and practical applications [4–8]. However, computing the corresponding Fourier...
coefficient remains a challenging issue, since calculating these Fourier coefficients involves high

dimensional oscillation integrations. Efforts to address this issue have been made by number of

researchers. A discrete algorithm for computing these Fourier coefficients of bivariate functions was

proposed in [9]. This method was extended to d-dimensional functions in [10] and implemented in [11]. It

was proved in [10] that the algorithm uses $\Theta(n \log^d n)$ number of multiplications for computing all

necessary Fourier coefficients where $n$ is the order of the univariate trigonometric polynomial used to

construct the sparse multivariate approximation, and shown in [12] that the algorithm has the approximation

order $\Theta(n^{-s/(s+1)})$, where $s$ is the order of the Sobolev regularity of the approximated function. Although it does not seem that this non-optimal approximation order can be improved in general, higher approximation order was indeed observed in certain numerical experiments (see Examples in Section 4 of this papers). Moreover, the oscillation problem of the integrals remains untreated. The main purpose of this paper is to present a fully discrete algorithm for numerical computation of the Fourier coefficients, which preserves the optimal approximation order $\Theta(n^{-s})$ for a function that has the Sobolev regularity of order $s$ for each variable with $s > 0$ and at the same time requires $\Theta(n \log^{2d-1} n)$ number of multiplications to perform the entire computation.

In development of numerical integration methods to efficiently compute the $d$-dimensional oscillatory integrals which define the Fourier coefficients, we face two types of challenges. The first difficulty is to deal with the oscillation of the integrands and the second is to treat integration in high dimensions. Numerical integration of an oscillatory integral is a difficult task. To tackle the difficulty, we adopt the product integration method which was originated in [13] so that the integrals of the oscillatory factors are evaluated exactly. For recent development of numerical integration of oscillatory integrals, see [14–18]. The key idea to treat the second type of the challenges is to construct an approximation of the non-oscillatory factor of the integrand with no more than $\Theta(n \log^{d-1} n)$ number of functional evaluations for computing all integrals needed for the sparse Fourier expansion and then to write the resulting numerical quadrature formula as discrete Fourier transforms of certain vectors so that the fast Fourier transform can be applied.

To ensure that the proposed quadrature algorithms will not ruin the approximation order that the sparse Fourier expansion enjoys, we also demand that the approximation of the non-oscillatory factor of the integrand has the optimal order. To meet this demand, we introduce a multiscale high order Lagrange interpolation on sparse grids to approximate the non-oscillatory factor of the integrand. In this multiscale high order Lagrange interpolation scheme, we employ Lagrange interpolating wavelets introduced in [21,22]. The research progress of sparse grid methods can be found from a mastery review paper [23] and recent publications [2,24–31]. Note that multidimensional integrals were treated in [32] in the context of expanding a given function by the spherical harmonic functions. The multidimensional quadratures by using common zeros of orthogonal polynomials are discuss in [33]. In a more general context, lattice rules were proposed in [34,35] to efficiently evaluate multidimensional integrals.

To prepare the research presented in this paper, we now review the approximation of a $d$-dimensional function by sparse trigonometric polynomials. We begin with an introduction of the appropriate Sobolev space. Let $\mathbb{N} := \{1, 2, \ldots\}$, $\mathbb{Z} := \{\ldots, -1, 0, 1, \ldots\}$ and for $n \in \mathbb{N}$, set $\mathbb{Z}_n := \{0, 1, \ldots, n-1\}$. For $d \in \mathbb{N}$, we use $L^2(I^d)$ with $I := [0, 2\pi]$ for the standard Hilbert space of the square integrable functions on $I^d$, with the usual inner product $\langle \cdot, \cdot \rangle$ and the norm $\| \cdot \| := \langle \cdot, \cdot \rangle^{\frac{1}{2}}$. It is well known that the Fourier basis $e_k, I := [k \in \mathbb{Z}_d] \in \mathbb{Z}^d$, defined by

$$e_k(x) := \frac{1}{(2\pi)^{d/2}} e^{i\theta k \cdot x}, \quad x \in I^d,$$

constitutes an orthonormal basis for the space $L^2(I^d)$, where $i$ denotes the imaginary unit. For $s \geq 0$ and for $\phi, \psi \in L^2(I^d)$, we define

$$\langle \phi, \psi \rangle_s = \sum_{l \in \mathbb{Z}^d} \langle \phi, e_l \rangle \langle \psi, e_l \rangle \prod_{k \in \mathbb{Z}_d} (1 + \theta_k^2)^s. \quad (1.1)$$

All functions $\phi \in L^2(I^d)$ with property that $\| \phi \|_{H^s_{mix}(I^d)} := \langle \phi, \phi \rangle_s^{\frac{1}{2}} < \infty$ form a subspace $H^s_{mix}(I^d)$ of $L^2(I^d)$. It can be verified that $H^s_{mix}(I^d)$ is a Sobolev space endowed with the inner product $\langle \phi, \psi \rangle_s$
Fig. 1. (a) Values of $|\langle f, e_l \rangle| : l \in \mathbb{Z}_{32}^2$ and (b) index set $L_{32}^2$.

defined by (1.1). Note that $H_{\text{mix}}^s(I^d)$ is a proper subspace of the standard Sobolev space $W^{s,2}(I^d)$. It can be found in [2,3,36] that

$$H_{\text{mix}}^s(I^d) = W^{s,2}(I) \otimes \cdots \otimes W^{s,2}(I).$$

For an $n \in \mathbb{N}$, we introduce the index set of the full grids

$$\mathbb{Y}_n^d := \{ l \in \mathbb{Z}^d : |l_k| + 1 \leq n, k \in \mathbb{Z}_d \},$$

and define the subspace $W_n \subset L^2(I^d)$, by $W_n := \text{span} \{ e_l : l \in \mathbb{Y}_n^d \}$. For $f \in L^2(I^d)$, we let $f_n$ denote the orthogonal projection of $f$ onto subspace $W_n$. Hence, $f_n$ has the expression

$$f_n = \sum_{l \in \mathbb{Y}_n^d} \langle f, e_l \rangle e_l. \quad (1.2)$$

It is well known that there exists a positive constant $c$ such that for all $n \in \mathbb{N}$ and for all $f \in H_{\text{mix}}^s(I^d)$,

$$\|f - f_n\|_{L^2(I^d)} \leq cn^{-s} \|f\|_{H_{\text{mix}}^s(I^d)}.$$

It is clear from (1.2) that $f_n$ requires computing $\Theta(n^d)$ inner products $\langle f, e_l \rangle, l \in \mathbb{Y}_n^d$. Even when the fast Fourier transform is used, it still requires $\Theta(n^d \log n)$ number of multiplications to compute the approximation. When $n$ and $d$ are large, the computational costs for computing $f_n$ is huge. It is desirable to develop sparse approximations of function $f$ which has only $\Theta(n \log^{d-1} n)$ terms, with preserving the optimal approximation order $\Theta(n^{-s})$ when $f \in H_{\text{mix}}^s(I^d)$. This has been achieved by using the hyperbolic cross approximation [3,6,7]. Indeed, the optimal convergence property of hyperbolic cross approximation was shown in [3] (Theorem 2.1, Chapter II), while its number of Fourier bases needed is bounded by $\Theta(n \log^{d-1} n)$ (see [6]).

We now recall these results. To explain the insight leading to the construction of the sparse approximation, we first consider an example with $d = 2$. For the function

$$f(x, y) := xy, \quad (x, y) \in I^2,$$

the values $|\langle f, e_l \rangle| : l \in \mathbb{Z}_{32}^2$ are shown in image (a) of Fig. 1. It can be observed that the values $|\langle f, e_l \rangle| : l := [l_0, l_1]$, decreases rapidly, as $|l_0 l_1|$ increases. In general, for $d, n \in \mathbb{N}$, we introduce an index set by setting

$$\mathbb{L}_n^d := \left\{ l \in \mathbb{Z}^d : \prod_{k \in \mathbb{Z}_d} (|l_k| + 1) \leq n \right\}.$$
It can be proved that $L^d_n \subseteq Y^d_n$. However, unlike the full index set $Y^d_n$, $L^d_n$ is a sparse index set. The sparsity of index sets $L^d_{32}$ and $L^d_{33}$ is illustrated in image (b) of Figs. 1 and 2, respectively. We now use the index set $L^d_n$ to encode approximations of functions defined on $I^d$. Mainly, associated with the index set $L^d_n$, we define a subspace $T_n$ of $W_n$ by

$$T_n := \text{span} \{ e_l : l \in \mathbb{N}_n^d \}.$$  
(1.3)

For $f \in L^2(I^d)$, we let $\tilde{f}_n$ denote the orthogonal projection of $f$ onto subspace $T_n$ and we call $\tilde{f}_n$ the sparse approximation of $f$. The sparse approximation $\tilde{f}_n$ has the specific representation

$$\tilde{f}_n = \sum_{l \in L^d_n} \langle f, e_l \rangle e_l.$$  
(1.4)

By $N_n$ we denote the cardinality of index set $\mathbb{N}_n^d$. The next theorem which estimates the different between $\tilde{f}_n$ and $f$, and $N_n$ can be found in [3,6,36].

**Theorem 1.1.** If $s \geq 0$, then there exists a positive constant $c$ such that for all $n \in \mathbb{N}$ and for all $f \in H^s_{\text{mix}}(I^d)$,

$$\|f - \tilde{f}_n\| \leq cn^{-s} \|f\|_{H^s_{\text{mix}}(I^d)}.$$  
(1.5)

For a fixed $d \in \mathbb{N}$, there exists a positive constant $c$ such that for all $n \in \mathbb{N}$ with $n > 1$,

$$N_n \leq cn \log^{d-1} n.$$  
(1.6)

This paper will focus on the fast computation of using the sparse expansion (1.4). We organize this paper in five sections and an appendix. In Section 2, we develop an efficient multiscale Lagrange interpolation on sparse grids for $d$-dimensional functions. In particular, we design the multiscale Lagrange interpolation formula with $O(n \log^{d-1} n)$ number of basis functions used and establish its optimal approximation order. Using this formula, we introduce in Section 3 a quadrature formula for computing Fourier coefficients, in which the non-oscillatory factor of the integrand is replaced by its multiscale Lagrange interpolation, and then the fast Fourier transform is used to design a fast quadrature algorithm for the implementation of the quadrature formula. We prove that the discrete sparse Fourier expansion using the quadrature preserves the optimal approximation order and computing all the needed Fourier coefficients requires only $O(n \log^{2d-1} n)$ number of multiplications. Numerical examples are presented in Section 4 to confirm the theoretical estimates for approximation order and computational complexity, and to compare the numerical performance of the proposed algorithm with that of a well-known existing algorithm. As demonstrated by numerical examples in Section 4, when the function to be approximated has a low order smoothness, the proposed algorithm
outperforms significantly the existing algorithm. While the function has a higher order smoothness, we see that with a similar approximation accuracy, the proposed algorithm uses less terms but more computing time. A few conclusive remarks are presented in Section 5. We collect several technical lemmas in the Appendix.

2. Multiscale Lagrange interpolation on sparse grids

Aiming at developing a fast algorithm for computing the Fourier coefficients of a function defined on \( l^d \), we construct in this section an approximation of the function by a multiscale piecewise polynomial interpolation on sparse grids on \( l^d \). A remarkable work about approximating bivariate functions on sparse grids is introduced in [37]. In [37], a fast algorithm for approximating functions by bivariate spline wavelets on sparse grids is proposed, which uses bivariate periodic spline wavelets to approximate smooth periodic functions. It was shown there that the number of multiplications used in the fast algorithm is \( O(n \log n) \). However, when a function is not periodic, the algorithm in [37] cannot obtain the desired approximation order. To handle the non-periodic functions, we develop a fast algorithm for approximating the functions by piecewise Lagrange interpolation polynomials on sparse grids. We first review the univariate multiscale piecewise polynomial interpolation originally developed in [21]. We then describe the tensor product type multiscale piecewise polynomial interpolation formula on \( l^d \), upon which the multiscale piecewise polynomial interpolation formula on sparse grids is developed.

We begin with a description of a multiscale decomposition of \( C(l) \) by piecewise polynomials following [21]. Choose two contractive mappings \( \phi_\rho : l \rightarrow l, \rho \in \mathbb{Z}_2 \), defined by \( \phi_0(x) = \frac{x}{2} \) and \( \phi_1(x) = \frac{x + 2\pi}{2}, x \in l \), and let \( \Psi := \{ \phi_\rho : \rho \in \mathbb{Z}_2 \} \). It is clear that \( l \) is an invariant set relative to the mappings \( \Psi \) in the sense that \( l = \Psi(l) \). According to [21], a subset \( V \) of \( l \) is said refinable (relative to the mappings \( \Psi \)) if \( V \subset \Psi(V) \). Associated with a refinable set \( V := \{ v_\ell : 0 \leq v_0 < v_1 < \cdots < v_{m-1} < 2\pi \} \), the fundamental Lagrange polynomials \( \ell_\rho \) of degree \( m-1 \) are defined for \( x \in l \) by \( \ell_\rho(x) := \prod_{q=0,q\neq r}^{m-1} \frac{x-v_q}{v_r-v_q}, \) for \( r \in \mathbb{Z}_m \). We define linear operators \( \mathcal{T}_\rho : L^\infty(l) \rightarrow L^\infty(l), \rho \in \mathbb{Z}_2 \), for \( f \in L^\infty(l) \) by

\[
(\mathcal{T}_\rho f)(x) := \begin{cases} 
(f \circ \phi^{-1}_\rho)(x), & x \in \phi_\rho(l), \\
0, & x \notin \phi_\rho(l).
\end{cases}
\]

With the operators \( \mathcal{T}_\rho, \rho \in \mathbb{Z}_2 \), we define a sequence of subspaces \( F_N \) for all \( N \in \mathbb{N} \) recursively by

\[
F_N := \bigoplus_{\rho \in \mathbb{Z}_2} \mathcal{T}_\rho F_{N-1},
\]

where \( F_0 := \text{span}\{\ell_\rho : r \in \mathbb{Z}_m\} \). Note that for all \( N \in \mathbb{N}, F_{N-1} \subset F_N \). Thus, the space \( F_N, N \in \mathbb{N} \), can be decomposed as the direct sum of the space \( F_{N-1} \) and its complement space \( G_N \) in \( F_N \). Upon letting \( G_0 := F_0 \), we have the multiscale decomposition \( F_N = G_0 \oplus G_1 \oplus \cdots \oplus G_N \).

We next describe bases of spaces \( F_N \) and \( G_N, N \in \mathbb{N}_0 := \{ 0, 1, \ldots \} \). We set \( \ell_{0,r} := \ell_r, r \in \mathbb{Z}_m \), and clearly, \( F_0 = G_0 = \text{span}\{\ell_{0,r} : r \in \mathbb{Z}_m\} \). For all \( N \in \mathbb{N} \) and \( p_N := [\rho_\gamma : \gamma \in \mathbb{Z}_N] \in \mathbb{Z}_N^2 \), let \( \mathcal{T}_{p_N} := \mathcal{T}_{p_{N-1}} \circ \cdots \circ \mathcal{T}_{p_0} \) and \( \mu(p_N) := \sum_{\gamma \in \mathbb{Z}_N} \rho_\gamma 2^\gamma \). For each \( N \in \mathbb{N} \), the functions

\[
\ell_{N,r} := \mathcal{T}_{p_N} \ell_r, \quad \text{where} \quad r = m\mu(p_N) + r', \quad p_N \in \mathbb{Z}_N^2, \quad r' \in \mathbb{Z}_m
\]

form a basis of space \( F_N \). For all \( N \in \mathbb{N} \) and \( p_N := [\rho_\gamma : \gamma \in \mathbb{Z}_N] \in \mathbb{Z}_N^2 \), we let \( \phi_{p_N} := \phi_0 \circ \cdots \circ \phi_0 \).

Let \( V_N := \{ v_{N,r} : r \in \mathbb{Z}_m \} \), where \( v_{N,r} := \phi_{p_N}(v_r), \) with \( p_N \in \mathbb{Z}_N^2, r' \in \mathbb{Z}_m \). Let \( m = m\mu(p_N) + r' \). For all \( N \in \mathbb{N} \), we define index sets \( \mathcal{W}_N \) by \( \mathcal{W}_N := \{ r \in \mathbb{Z}_m : v_{N,r} \in \mathbb{N} \backslash V_{N-1} \} \). It was proved in Theorem 4.5 of [21] that for all \( N \in \mathbb{N}, G_N = \text{span}\{\ell_{N,r} : r \in \mathcal{W}_N\} \). Let \( \mathcal{W}_0 := \mathbb{Z}_m \).

The basis functions \( \ell_{N,r} \) and \( v_{N,r}, N \in \mathbb{N}_0 \), and \( r \in \mathcal{W}_N \), satisfy the property (see [21]) that \( \ell_{N,r}(v_{N,r'}) = \delta_{r,r'} \), where \( \delta_{r,r'} := 1 \) if \( r = r' \), and 0 otherwise. These basis functions also satisfy a refinement equation, which is presented in the next lemma. Let \( a_{\lambda,k} := \ell_{0,r}(v_{1,k}), r \in \mathbb{Z}_m \) and \( \kappa \in \mathbb{Z}_2 \).

For \( x \in \mathbb{R} \), we let \([x]\) denote the largest integer not greater than \( x \).
Lemma 2.1. For given $j \in \mathbb{N}_0$ and $r \in \mathbb{Z}_{2m}$, if $r = m\mu(p_j) + r'$ with $p_j \in \mathbb{Z}_2$ and $r' \in \mathbb{Z}_m$, then

$$\ell_{j,r} = \sum_{k \in \mathbb{Z}_{2m}} a_{r',\kappa} \ell_{j+1,2m\left\lfloor \frac{r}{m} \right\rfloor + \kappa}.$$ (2.2)

Proof. We first prove Eq. (2.2) for $j = 0$ and for all $r \in \mathbb{Z}_m$. Since $\ell_{0,r}$ is a polynomial of order $m$, using the definition of $a_{r,\kappa}$, $\kappa \in \mathbb{Z}_{2m}$, we obtain that

$$\ell_{0,r} = \sum_{k \in \mathbb{Z}_{2m}} a_{r,\kappa} \ell_{1,k}.$$ (2.3)

By noting that $\left\lfloor \frac{r}{m} \right\rfloor = 0$, from (2.3) we conclude (2.2) for $j = 0$ and $r \in \mathbb{Z}_m$.

We next prove Eq. (2.2) for $j \in \mathbb{N}$ and $r \in \mathbb{Z}_{2m}$. Substituting (2.3) into the definition of $\ell_{j,r}$ yields that

$$\ell_{j,r} = \sum_{k \in \mathbb{Z}_{2m}} a_{r',\kappa} \mathcal{T}_p \ell_{1,k}, \quad r = m\mu(p_j) + r', \quad p_j \in \mathbb{Z}_2, \quad r' \in \mathbb{Z}_m.$$ (2.4)

Let $p_{j+1,0} := [0, \rho_0, \ldots, \rho_{j-1}]$ and $p_{j+1,1} := [1, \rho_0, \ldots, \rho_{j-1}]$. From the definition of $\ell_{1,k}$, $\kappa \in \mathbb{Z}_{2m}$, we have that $\mathcal{T}_p \ell_{1,k} = \mathcal{T}_p \mathcal{T}_0 \ell_{0,k} = \mathcal{T}_p \ell_{1,k+1}$ and $\mathcal{T}_p \ell_{1,k+m} = \mathcal{T}_p \mathcal{T}_1 \ell_{0,k} = \mathcal{T}_p \ell_{1,k+1}$, for $\kappa \in \mathbb{Z}_m$. Using these relations in (2.4), we observe that

$$\ell_{j,r} = \sum_{k \in \mathbb{Z}_{2m}} a_{r',\kappa} \mathcal{T}_{p_{j+1,0}} \ell_{0,k} + \sum_{k \in \mathbb{Z}_{m}} a_{r',\kappa+m} \mathcal{T}_{p_{j+1,1}} \ell_{0,k}.$$ (2.5)

Note that $\mu(p_{j+1,0}) = 2\mu(p_j)$ and $\mu(p_{j+1,1}) = 2\mu(p_j) + 1$. From (2.1) and (2.5), we have that

$$\ell_{j,r} = \sum_{k \in \mathbb{Z}_{2m}} a_{r',\kappa} \ell_{j+1,2m\mu(p_j)+\kappa}.$$ (2.6)

Noting that $\mu(p_j) = \left\lfloor \frac{r}{m} \right\rfloor$, Eq. (2.2) follows from (2.6). \qed

The interpolation projection $\mathcal{P}_N : C(I) \to F_N$ is defined for all $f \in C(I)$ by

$$\mathcal{P}_N f := \sum_{r \in \mathbb{Z}_{2^N m}} f(v_{N,r}) \ell_{N,r}.$$ (2.7)

For $j \in \mathbb{N}$, we define $\mathcal{Q}_j : C(I) \to G_j$ by $\mathcal{Q}_0 := \mathcal{P}_0$ and $\mathcal{Q}_j := \mathcal{P}_j - \mathcal{P}_{j-1}$. Thus, for $f \in C(I)$,

$$\mathcal{P}_N f = \sum_{j \in \mathbb{N}_{N+1}} \mathcal{Q}_j f, \quad N \in \mathbb{N}_0$$ (2.8)

which is a multiscale decomposition of $\mathcal{P}_N f$.

We next derive a formula for $\mathcal{Q}_j f$. To do this, we let $\mathbb{W}_0 := \mathbb{Z}_m$ and define linear functionals $\eta_{j,r} : C(I) \to \mathbb{R}, j \in \mathbb{N}_0$ and $r \in \mathbb{Z}_{2^m}$ by

$$\eta_{0,r}(f) := f(v_{0,r}), \quad \text{and} \quad \eta_{j,r}(f) := f(v_{j,r}) - \sum_{q \in \mathbb{Z}_m} f(v_{j-1,r-1,2^m q + q}) a_{q,r} \text{ mod } 2m.$$ (2.9)

Lemma 2.2. If $f \in C(I)$, then for all $j \in \mathbb{N}_0$,

$$\mathcal{Q}_j f = \sum_{r \in \mathbb{W}_j} \eta_{j,r}(f) \ell_{j,r}.$$ (2.9)

Proof. Because $\mathcal{Q}_0 = \mathcal{P}_0$, Eq. (2.9) for $j = 0$ can be obtained directly from the definition of $\mathcal{P}_0$. Since $\mathcal{Q}_j f = \mathcal{P}_j f - \mathcal{P}_{j-1} f$, to establish (2.9) for $j > 0$ we need to represent $\mathcal{P}_{j-1} f$ in terms of $\ell_{j,r}, r \in \mathbb{Z}_{2^m}$. Since for all $r \in \mathbb{Z}_{2^m}$, there exist $\tau \in \mathbb{Z}_{2^m-1}$ and $q \in \mathbb{Z}_m$ such that $r = \tau m + q$, from the definition
of $\mathcal{P}_{j-1}$ we have that
\[ \mathcal{P}_{j-1}f = \sum_{r \in \mathbb{Z}_{2j-1}} \sum_{q \in \mathbb{Z}_m} f(v_{j-1,r,m+q})\ell_{j-1,mr+q}. \] (2.10)

Note that for each $r \in \mathbb{Z}_{2j-1}$ and $\kappa \in \mathbb{Z}_{2m}$,
\[ \sum_{q \in \mathbb{Z}_m} f(v_{j-1,r,m+q})a_{q,\kappa} = \sum_{q \in \mathbb{Z}_m} f(v_{j-1,m,\lfloor \frac{r}{m} \rfloor+q})a_{q,r \mod 2m} \]
where $r = 2mr + \kappa$.

Substituting (2.2) into (2.10), we obtain that
\[ \mathcal{P}_{j-1}f = \sum_{r \in \mathbb{Z}_{2j-1}} \left( \sum_{q \in \mathbb{Z}_m} f(v_{j-1,m,\lfloor \frac{r}{m} \rfloor+q})a_{q,r \mod 2m} \right)\ell_{j,r}. \] (2.11)

Combining (2.11) with the definition of $\mathcal{P}_j$, we obtain that
\[ \mathcal{Q}_jf = \sum_{r \in \mathbb{Z}_{2j-1}} \eta_{j,r}(f)\ell_{j,r}. \] (2.12)

Since $\mathcal{Q}_jf \in \mathcal{G}_j = \text{span}\{\ell_{j,r} : r \in \mathbb{W}_j\}$ (see Theorem 4.5 in [21]), we observe that $\mathcal{Q}_jf \in \text{span}\{\ell_{j,r} : r \in \mathbb{W}_j\}$. Thus, (2.9) follows from (2.12). \[\square\]

Substitution of (2.9) into (2.8) leads to a multiscale formula for the Lagrange piecewise polynomial interpolation on interval $I$, which we present in the following proposition.

**Proposition 2.3.** If $f \in C(I)$, then for all $N \in \mathbb{N}_0$,
\[ \mathcal{P}_Nf = \sum_{j \in \mathbb{N}_0+1} \sum_{r \in \mathbb{W}_j} \eta_{j,r}(f)\ell_{j,r}. \] (2.13)

We next extend the multiscale Lagrange interpolation formula to high dimensional cases by the tensor product. For this purpose, we review the tensor product $\otimes$ of linear functionals. We denote by $\mathcal{L}(C(I^d))$ the set of linear functionals on $C(I^d)$ which satisfy the condition: For each $\lambda \in \mathcal{L}(C(I^d))$, there exist $n \in \mathbb{N}$ and $\zeta_1, \ldots, \zeta_n \in \mathbb{R}$ such that for all $f \in C(I^d)$, $\lambda(f) := \sum_{j \in \mathbb{Z}_m} a_j f(x_j)$. Suppose that $d_1, d_2 \in \mathbb{N}$. For any $\lambda_r \in \mathcal{L}(C(I^{d_r})), r \in \{1, 2\}$ with $\lambda_r(f) := \sum_{j \in \mathbb{Z}_m} a^r_j f(x^r_j)$, we define $\lambda_1 \otimes \lambda_2$ for all $f \in C(I^{d_1+d_2})$ by
\[ (\lambda_1 \otimes \lambda_2)(f) := \sum_{j_1 \in \mathbb{Z}_m} \sum_{j_2 \in \mathbb{Z}_m} a^1_{j_1} a^2_{j_2} f(x^1_{j_1}, x^2_{j_2}). \]

It is clear that $\lambda_1 \otimes \lambda_2$ is a linear functional on $C(I^{d_1+d_2})$. The tensor product of functions in $L^\infty(I^{d_1})$ and $L^\infty(I^{d_2})$ is defined for all $g_r \in L^\infty(I^{d_r}), r \in \{1, 2\}$ by $(g_1 \otimes g_2)(x, y) := g_1(x)g_2(y)$, for $x \in I^{d_1}$ and $y \in I^{d_2}$. It can be shown that for all $\lambda_r \in \mathcal{L}(C(I^{d_r}))$ and $g_r \in C(I^{d_r}), r \in \{1, 2\}$, there holds that
\[ (\lambda_1 \otimes \lambda_2)(g_1 \otimes g_2) = (\lambda_1(g_1)) \otimes (\lambda_2(g_2)). \] (2.14)

Suppose that $\mathcal{A}_r : C(I^{d_r}) \rightarrow L^\infty(I^{d_r}), r \in \{1, 2\}$, are two linear operators satisfying that for each $r \in \{1, 2\}$, there exist a positive integer $n_r \in \mathbb{N}$, linear functionals $\lambda_{r,j} \in \mathcal{L}(C(I^{d_r}))$, and functions $g_{r,j} \in L^\infty(I^{d_r}), j \in \mathbb{Z}_{n_r}$ such that
\[ \mathcal{A}_rf := \sum_{j \in \mathbb{Z}_{n_r}} \lambda_{r,j}(f)g_{r,j}, \quad \text{for all } f \in C(I^{d_r}). \]

We define $\mathcal{A}_1 \otimes \mathcal{A}_2 : C(I^{d_1+d_2}) \rightarrow L^\infty(I^{d_1+d_2})$ for all $f \in C(I^{d_1+d_2})$ by
\[ (\mathcal{A}_1 \otimes \mathcal{A}_2)f := \sum_{j_1 \in \mathbb{Z}_{n_1}} \sum_{j_2 \in \mathbb{Z}_{n_2}} (\lambda_1(j_1) \otimes \lambda_2(j_2))(f)(g_{1,j_1} \otimes g_{2,j_2}). \]
From formula (2.14) and the definition of $A_1 \otimes A_2$, we see that for $g_r \in C(I^{d_r})$, $r \in \{1, 2\}$,

$$(A_1 \otimes A_2)(g_1 \otimes g_2) = (A_1 g_1) \otimes (A_2 g_2).$$

(2.15)

For all $N \in \mathbb{N}_0$ and $d \in \mathbb{N}$, we define the projection

$$\mathcal{P}_N^d \coloneqq \mathcal{P}_N \otimes \cdots \otimes \mathcal{P}_N
\quad d$$

(2.16)

and observe that $\mathcal{P}_N^d$ are the standard Lagrange interpolation projections on full grids. Indeed, if we define for all $N \in \mathbb{N}_0$ and $r := [r_k : k \in \mathbb{Z}_d] \in \mathbb{Z}_d^{N+1}$, $v_{N,r} := [v_{N,r_k} : k \in \mathbb{Z}_d]$ and $\ell_{N,r} := \ell_{N,r_0} \otimes \cdots \otimes \ell_{N,r_{d-1}}$, then for all $f \in C(I^d)$, operator $\mathcal{P}_N^d$ has the form

$$\mathcal{P}_N^d f = \sum_{r \in \mathbb{Z}_d^{N+1}} f(v_{N,r}) \ell_{N,r}.$$

(2.17)

Eq. (2.17) indicates that $\mathcal{P}_N^d f$ is an approximation of function $f$ on full grids. It is clear that the number of terms used in the approximation $\mathcal{P}_N^d f$ is $2^{hd} m^d$.

We next describe the multiscale Lagrange interpolation on sparse grids. For all $j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{Z}_d^d$, we let $Q_j \coloneqq Q_{j_0} \otimes \cdots \otimes Q_{j_{d-1}}$. For all $d \in \mathbb{N}$ and $j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d$, we define the index sets $W_j^d := W_{j_0} \otimes \cdots \otimes W_{j_{d-1}}$. For all $j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d$ and $r := [r_k : k \in \mathbb{Z}_d] \in \mathbb{W}_j^d$, we define $\eta_{j,r} := \eta_{j_0,r_0} \otimes \cdots \otimes \eta_{j_{d-1},r_{d-1}}$ and $\ell_{j,r} := \ell_{j_0,r_0} \otimes \cdots \otimes \ell_{j_{d-1},r_{d-1}}$.

**Lemma 2.4.** Let $d \in \mathbb{N}$. If $f \in C(I^d)$, then for all $j \in \mathbb{N}_0^d$,

$$Q_j f = \sum_{r \in \mathbb{W}_j^d} \eta_{j,r} f(\ell_{j,r}).$$

(2.18)

**Proof.** We prove this result by induction on $d$. Eq. (2.9) shows that (2.18) holds for $d = 1$. We assume that (2.18) holds for $d = d'$ and consider the case $d = d' + 1$. For $j := [j_k : k \in \mathbb{Z}_{d'+1}] \in \mathbb{N}_0^{d'}$ and $j' := [j_k : k \in \mathbb{Z}_d]$, from the definition of $Q_j$, we know that $Q_j = Q_{j'} \otimes Q_{j_f}$. Substituting (2.9) into this equation and using the induction hypothesis, for all $f \in C(I^{d'+1})$ we have that

$$Q_j f = \sum_{r \in \mathbb{W}_{j'}^d} \sum_{r' \in \mathbb{W}_{j_f}^d} (\eta_{j',r} \otimes \eta_{j_f,r'})(f)(\ell_{j',r} \otimes \ell_{j_f,r'}).$$

(2.19)

By the definitions of $\eta_{j,r}$ and $\ell_{j,r}, j \in \mathbb{Z}_{d'+1}^d, r \in \mathbb{W}_{j_f}^{d'+1}$, and (2.19) we obtain Eq. (2.18) for $d = d' + 1$. The induction principle ensures that Eq. (2.18) holds for any $d \in \mathbb{N}$. □

**Lemma 2.4** with the formula $\mathcal{P}_N^d f = \sum_{j \in \mathbb{Z}_N^{d+1}} Q_j f$ leads to the tensor product multiscale Lagrange interpolation formula.

**Theorem 2.5.** Let $d \in \mathbb{N}$. If $f \in C(I^d)$, then for all $N \in \mathbb{N}_0$,

$$\mathcal{P}_N^d f = \sum_{j \in \mathbb{Z}_N^{d+1}} \sum_{r \in \mathbb{W}_j^d} \eta_{j,r} f(\ell_{j,r}).$$

(2.20)

With the formula in **Theorem 2.5**, we now develop a multiscale Lagrange interpolation of $f$ on sparse grids. For each $N \in \mathbb{N}$, let

$$\mathbb{S}_N^d := \left\{ j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{Z}_N^{d+1} : \sum_{k \in \mathbb{Z}_d} j_k \leq N \right\}.$$
For \( f \in C(l^d) \) and \( N \in \mathbb{N} \), the multiscale Lagrange interpolation of \( f \) on sparse grids is defined by

\[
\delta_N f = \sum_{j \in \mathbb{N}_0} \sum_{r \in \mathbb{W}^d_j} \eta_{j,r}(f) \ell_{j,r}.
\]  

(2.21)

Out next task is to estimate the difference between \( \delta_N f \) and \( f \). This is done by a sequence of technical lemmas.

**Lemma 2.6.** For \( N \in \mathbb{N}_0 \), if \( f \in C(l^d) \), then

\[
(\mathcal{P}_N^d - \delta_N) f = \sum_{j \in \mathbb{N}_0} \sum_{r \in \mathbb{W}^d_j} \eta_{j,r}(f) \ell_{j,r}.
\]  

(2.22)

**Proof.** From the definitions of \( \mathcal{P}_N^d \) and \( \delta_N \), we have that

\[
(\mathcal{P}_N^d - \delta_N) f = \sum_{j \in \mathbb{N}_0} \sum_{r \in \mathbb{W}^d_j} \eta_{j,r}(f) \ell_{j,r}.
\]  

(2.23)

Substituting (2.21) into (2.23), we obtain equality (2.22). □

In the following lemmas, we show that the set of operators \( Q_j \), \( j \in \mathbb{N}_0^d \), is bounded. For \( r := [r_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d \), we let \( |r/m| := \lfloor |r_k/m| : k \in \mathbb{Z}_d \rfloor \) and for \( j \in \mathbb{N}_0^d \), we define \( U^j_f \) := \( \{ u := [u_k : k \in \mathbb{Z}_d] : u_k \in \mathbb{Z}_{2^{dk}} \} \) and \( \mathcal{V}^j \) := \( \{ u \in \mathbb{V}^j \} : \text{there exists } r \in \mathbb{W}^d_j \text{ such that } |r/m| = u \} \). We also define for each \( j \in \mathbb{N}_0^d \) and \( u \in \mathbb{U}^j_f \), \( \Omega_{j,u} := \{ x := [x_k : k \in \mathbb{Z}_d] : x_k \in \left[ 2\pi u_k \frac{2\pi (u_k + 1)}{2} \frac{k}{2k} \right], k \in \mathbb{Z}_d \} \).

**Lemma 2.7.** For \( j \in \mathbb{N}_0^d \), if \( f \in C(l^d) \), then

\[
\| Q_f \|_{\infty} = \max \{ \| Q_f \|_{\Omega_{j,u}} : u \in \mathcal{V}^j \}.
\]  

(2.24)

**Proof.** Since \( \bigcup_{u \in \mathcal{V}^j} \Omega_{j,u} = l^d \) and \( \mathcal{Q} \cap \Omega_{j,u} = 0 \) if \( u \neq u' \), we have for all \( f \in C(l^d) \) and \( j \in \mathbb{N}_0^d \) that \( \| Q_f \|_{\infty} = \max \{ \| Q_f \|_{\Omega_{j,u}} : u \in \mathbb{U}^j_f \} \). It follows from the definitions of \( \ell_{j,r} \) and \( \Omega_{j,u} \) that for all \( j \in \mathbb{N}_0^d \) and \( r \in \mathbb{W}^d_j \), \( \supp(\ell_{j,r}) = \Omega_{j,|r/m|} \). Hence, from Lemma 2.4, we have that if \( u \notin \mathcal{V}^j \), then \( Q_f \|_{\Omega_{j,u}} = 0 \). We obtain the desired result (2.24). □

In the next lemma, we establish the uniform boundedness of \( Q_j \).

**Lemma 2.8.** There exists a positive constant \( c \) such that for all \( f \in C(l^d) \) and \( j \in \mathbb{N}_0^d \),

\[
\| Q_f \|_{\infty} \leq c \| f \|_{\infty}.
\]  

(2.25)

**Proof.** Lemma 2.7 ensures that it suffices to estimate the norms \( \| Q_f \|_{\Omega_{j,u}} \). For each \( u \in \mathcal{V}^j \), there holds

\[
Q_f \|_{\Omega_{j,u}} = \sum_{r \in \mathbb{W}^d_j, |r/m| = u} \eta_{j,r}(f) \ell_{j,r} |\Omega_{j,u}|
\]  

(2.26)

According to the definition of \( \eta_{j,r} \), \( j \in \mathbb{N}_0^d \) and \( r \in \mathbb{W}^d_j \), there exists a positive constant \( c_0 \) such that for all \( f \in C(l^d) \), \( j \in \mathbb{N}_0^d \) and all \( r \in \mathbb{W}^d_j \), \( \eta_{j,r}(f) \leq c_0 \| f \|_{\infty} \). From the definition of \( \ell_{j,r} \), there exists a positive constants \( c_1 \) such that for all \( j \in \mathbb{N}_0^d \) and \( r \in \mathbb{W}^d_j \), \( \ell_{j,r} \|_{\infty} \leq c_1 \). Note that for fixed \( u \in \mathcal{V}^d_j \),
the number of vectors $r \in \mathbb{V}^d$ satisfying $|r/m| = u$ is not greater than $m^d$. Thus, from (2.26) and the two estimates above, we obtain that there exists a positive constants $c$ such that for all $f \in C(l^d)$, all $j \in \mathbb{N}_0^d$ and all $u \in \mathbb{V}^d$, $||Q_j f ||_{\Omega, u} \leq c ||f||_\infty$. By combining this estimate with (2.24), we obtain estimate (2.25). □

We next investigate the decay rate of $||Q_j f||_\infty$ when $f$ has additional order of smoothness. We define below the space that consists of all functions of bounded mixed derivatives up to order $m$. For an $\alpha := [\alpha_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d$, let $|\alpha|_\infty := \max \{|\alpha_k| : k \in \mathbb{Z}_d\}$ and $|\alpha| := \sum_{k \in \mathbb{Z}_d} \alpha_k$. Suppose that $\Omega \subset \mathbb{R}^d$ is compact. For a function $f \in C^m(\Omega)$ and for $\alpha := [\alpha_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d$ with $|\alpha| \leq m$, we let

$$f^{(\alpha)}(x) := \left( \frac{\partial |\alpha|}{\partial x_0^{\alpha_0} \cdots \partial x_{d-1}^{\alpha_{d-1}}} f \right)(x), \quad x := [x_k : k \in \mathbb{Z}_d] \in \Omega.$$ 

We define the space $X^m(l^d) := \{ f : l^d \to \mathbb{R} : f^{(\alpha)} \in C(l^d), |\alpha|_\infty \leq m \}$ with the norm $||f||_{X^m(l^d)} := \max \{|f^{(\alpha)}|_\infty : \alpha \in \mathbb{N}_0^d, |\alpha|_\infty \leq m\}$. It was shown in [38] that linear combinations of functions $f_0 \otimes f_1 \cdots \otimes f_{d-1}$ for $f_k \in X^m(\Omega)$, $k \in \mathbb{Z}_d$, are dense in $X^m(l^d)$ and $||f_0 \otimes f_1 \otimes \cdots \otimes f_{d-1}||_{X^m(l^d)} = \prod_{k \in \mathbb{Z}_d} ||f_k||_{X^m(\Omega)}$. We also define the polynomial space

$$P_m^d := \text{span} \left\{ \prod_{k \in \mathbb{Z}_d} x_k^{m_k} : x_k \in \mathbb{I}, m_k \in \mathbb{Z}_m, k \in \mathbb{Z}_d \right\}.$$ 

We next show that for all $j \in \mathbb{N}_0^d$ with $|j| := \sum_{k \in \mathbb{Z}_d} |j_k| > 0$, $Q_j$ vanish on $P_m^d$.

**Lemma 2.9.** For all $j \in \mathbb{N}_0^d$ with $|j| > 0$ and $p \in P_m^d$, there holds

$$\left( Q_j p \right)(x) = 0, \quad x \in l^d.$$ \hspace{1cm} (2.27)

**Proof.** For all $m := [m_k : k \in \mathbb{Z}_d] \in \mathbb{Z}_m^d$, we let $p_m(x) := \prod_{k \in \mathbb{Z}_d} x_k^{m_k}$ for $x := [x_k : k \in \mathbb{Z}_d] \in l^d$. Since any $p \in P_m^d$ is a linear combination of $p_m$, $m \in \mathbb{Z}_m^d$, it suffices to show the claim that for all $j \in \mathbb{N}_0^d$ with $|j| > 0$, $m \in \mathbb{Z}_m^d$ and $x \in l^d$, $(Q_j p_m)(x) = 0$.

We first prove the claim for $d = 1$. When $d = 1$, for each $m' \in \mathbb{Z}_m$, $p_m'$ is a polynomial of order $m'$. Because for each $j \in \mathbb{N}_0$, $P_j$ is a Lagrange piecewise polynomial interpolation on interval $l$, $Q_j p_m' = Q_{j-1} p_m'$, for $j \in \mathbb{N}$. Hence, the claim in this case follows immediately.

We now consider the cases when $d > 1$. Let $j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{N}_0^d$ with $|j| > 0$. From property (2.15) of tensor products and the definition of $Q_j$, we have for all $m := [m_k : k \in \mathbb{Z}_d] \in \mathbb{Z}_m^d$ and $x := [x_k : k \in \mathbb{Z}_d] \in l^d$ that

$$\left( Q_j p_m \right)(x) = \prod_{k \in \mathbb{Z}_d} \left( Q_{j_k} p_{m_k} \right)(x_k).$$ \hspace{1cm} (2.28)

Since $|j| > 0$, there exists a $k' \in \mathbb{Z}_d$ such that $j_{k'} > 1$. Thus, by using the claim with $d = 1$, we obtain for all $x \in l$ that $(Q_{j_{k'}} p_{m_{k'}})(x) = 0$. This with (2.28) implies that the claim holds for $d > 1$. □

The result in the last lemma is used to establish the decay rate of $||Q_j f||_\infty$.

**Lemma 2.10.** For a fixed $d \in \mathbb{N}$, there exists a positive constant $c$ such that for all $f \in X^m(l^d)$ and $j \in \mathbb{N}_0^d$ with $|j| > 0$,

$$||Q_j f||_\infty \leq c 2^{-m|j|} ||f||_{X^m(l^d)}.$$ \hspace{1cm} (2.29)
Lemma 2.9 (2.21)
Lemma 2.11
For a fixed $d$ according to the triangle inequality, we have that

\[ \|Q_j f|_{\Omega_j u}\|_\infty \leq c2^{-m|j|}\|f\|_{X^m(I^d)}. \] (2.30)

By the Taylor Theorem (see, Theorem 13.18 of [39]), there exists a positive constant $c$ such that for all $\mathbf{j} \in \mathbb{N}_0^d$ with $|\mathbf{j}| > 0$ and all $\mathbf{u} \in \mathbb{V}_j^d$, there is a polynomial $p_{\mathbf{j}, \mathbf{u}} \in P_m^d$ such that

\[ \|f - p_{\mathbf{j}, \mathbf{u}}|_{\Omega_j u}\|_\infty \leq c2^{-m|j|}\|f\|_{X^m(I^d)}. \] (2.31)

Note that

\[ \|Q_j f|_{\Omega_j u}\|_\infty \leq \|Q_j p_{\mathbf{j}, \mathbf{u}}|_{\Omega_j u}\|_\infty + \|Q_j (f - p_{\mathbf{j}, \mathbf{u}})|_{\Omega_j u}\|_\infty. \] (2.32)

From Lemma 2.8 and (2.31), there exists a positive constant $c$ such that for all $\mathbf{j} \in \mathbb{N}_0^d$ with $|\mathbf{j}| > 0$ and all $\mathbf{u} \in \mathbb{V}_j^d$,

\[ \|Q_j (f - p_{\mathbf{j}, \mathbf{u}})|_{\Omega_j u}\|_\infty \leq c2^{-m|j|}\|f\|_{X^m(I^d)}. \] (2.33)

Since $p_{\mathbf{j}, \mathbf{u}} \in P_m^d$, from Lemma 2.9, we conclude that $\|Q_j p_{\mathbf{j}, \mathbf{u}}|_{\Omega_j u}\|_\infty = 0$. Substituting this equation and inequality (2.33) into (2.32) yields the desired inequality (2.30).

Combining Lemmas 2.6 and 2.10 yields the following estimate on $\|(P_N^d - \delta_N f)\|_\infty$.

**Lemma 2.11.** For a fixed $d \in \mathbb{N}$, there exists a positive constant $c$ such that for all $N \in \mathbb{N}$ and $f \in X^m(I^d)$,

\[ \|(P_N^d - \delta_N f)\|_\infty \leq cN^{d-1}2^{-mN}\|f\|_{X^m(I^d)}. \] (2.34)

**Proof.** It follows from Lemmas 2.6 and 2.10 that there exists a positive constant $c$ such that for all $N \in \mathbb{N}$ and $f \in X^m(I^d)$,

\[ \|(P_N^d - \delta_N f)\|_\infty \leq c\sum_{j \in \mathbb{Z}_N^{d+1}\setminus \mathbb{Z}_N^d}2^{-m|j|}\|f\|_{X^m(I^d)}. \] (2.35)

The desired estimate (2.34) follows from (2.35) and Lemma 3.7 of [23].

We now estimate the difference between $f$ and $\delta_N f$ by using Lemma 2.11.

**Theorem 2.12.** For a fixed $d \in \mathbb{N}$, there exists a positive constant $c$ such that for all $N \in \mathbb{N}$ and $f \in X^m(I^d)$,

\[ \|f - \delta_N f\|_\infty \leq cN^{d-1}2^{-mN}\|f\|_{X^m(I^d)}. \] (2.36)

**Proof.** According to the triangle inequality, we have that

\[ \|f - \delta_N f\|_\infty \leq \|f - P_N^d f\|_\infty + \|(P_N^d - \delta_N f)\|_\infty. \] (2.37)

Since $P_N^d f$ is a piecewise polynomial interpolation of function $f$ on the grid $\underbrace{V_N \otimes \cdots \otimes V_N}_d$, it is well known that there exists a positive constant $c$ such that for all $N \in \mathbb{N}$ and $f \in X^m(I^d)$,

\[ \|P_N^d f - f\|_\infty \leq c2^{-mN}\|f\|_{X^m(I^d)}. \] (2.38)

Substituting (2.34) and (2.38) into (2.37), we obtain (2.36).

We next estimate the computational costs for computing $\delta_N f$. By $\mathcal{M}_N$ we denote the number of basis functions used in formula (2.21) for a given $N \in \mathbb{N}$, and by $\mathcal{C}(\mathcal{X})$ we denote the cardinality of a set $\mathcal{X}$. 

...
There exists a positive constant $c$ such that for all $N \in \mathbb{N}$,
\[ M_N \leq cN^{d-1/2}N. \tag{2.39} \]

**Proof.** From the definition of $\delta_N$, we have that $M_N = \sum_{j \in S_N^d} C(\mathcal{W}_N^d, j)$. According to the definition of $\mathcal{W}_N^d$, we obtain for all $j := \{k : k \in Z_d\} \in S_N^d$ that $C(\mathcal{W}_N^d) = \prod_{k \in Z_d} C(\mathcal{W}_k)$. Noting that $C(\mathcal{W}_0) = m$ and for all $j \in \mathbb{N}$, $C(\mathcal{W}_j) = 2^{-j}m$, we observe that for all $j \in S_N^d$, $C(\mathcal{W}_N^d) \leq m^d2^{j-1}$, where $1$ is the vector in $\mathbb{R}^d$ with all components equal to 1. Consequently,
\[ M_N \leq m^d \sum_{j \in S_N^d} 2^{j-1}. \tag{2.40} \]

By employing Lemma 3.6 of [23] in (2.40), we obtain the desired estimate (2.39). $\square$

### 3. A numerical quadrature strategy

For a given function $f \in L^2(I^d)$, the construction of the sparse approximation $\tilde{f}_n$, requires computing the coefficients $(f, e_l, 1 \in \mathbb{L}_n^d$. These are integrals over the $d$-dimensional cube $I^d$ and they are normally computed by numerical methods. Numerical evaluation of integrals of high dimensions is a difficult issue. When $\max |l_k : k \in Z_d | \gg 1$, evaluation of $(f, e_l)$ involves computing oscillatory integrals. Numerical computation of an oscillatory integral of a high dimension is a even more challenging task. In this section, we will propose a numerical quadrature strategy for computing the integrals
\[ (f, e_l) = \int_{I^d} f(x) \hat{e}_l(x) dx, \quad 1 \in \mathbb{L}_n^d. \tag{3.1} \]

We require that the discrete sparse Fourier expansion $\tilde{f}_n$ of a given function $f$, where its Fourier coefficients $(f, e_l, 1 \in \mathbb{L}_n^d$ are computed by the numerical quadrature strategy, preserves the same optimal approximation order as the sparse expansion $\tilde{f}_n$ has. Moreover, we demand that the total number of multiplications used in the numerical integration of all necessary Fourier coefficients cannot exceed $O(n \log^{2d-1} n)$.

To meet the first requirement, we will employ the multiscale Lagrange interpolation formula on sparse grids introduced in Section 2 to approximate the function $f$ by the sum of $Q_{j, l}, j \in S_N^d$. As a result, the integral (3.1) will be the sum of the integrals of $(Q_{j, l}) \hat{e}_l, j \in S_N^d, \text{on } I^d$. Moreover, each of these integrals can be rewritten as a discrete Fourier transform of a vector of the values of $\eta_{l, r}(f)$, $r \in \mathbb{W}_j^d$ so that the fast Fourier transform can be applied.

By employing the multiscale Lagrange interpolation of function $f$ on sparse grids, integral (3.1) can be computed by replacing $f$ by $\delta_N f$. Specifically, for all $j \in S_N^d$, $r \in \mathbb{W}_j^d$ and $1 \in \mathbb{L}_n^d$, we define $A_{j, r}(1) := \int_{I^d} \ell_{j, r}(x) \hat{e}_l(x) dx$. The quadrature formula for the integral (3.1) is given by
\[ Q_N(f, 1) := \sum_{j \in S_N^d} \sum_{r \in \mathbb{W}_j^d} \eta_{l, r}(f) A_{j, r}(1). \tag{3.2} \]

For a given continuous function $f$, the quadrature formulas $Q_N(f, 1), 1 \in \mathbb{L}_n^d$, lead to the discrete sparse Fourier expansion
\[ \tilde{f}_{n, N} := \sum_{1 \in \mathbb{L}_n^d} Q_N(f, 1) e_1. \]

In the next theorem, we show that the discrete sparse Fourier expansion preserves the optimal approximation order if the function $f$ has an appropriate order of regularity. To state the theorem, we let $|x|$ denote the smallest integer not less than $x \in \mathbb{R}$. 

\[ \]
Theorem 3.1. Let $s \geq 0$. Choose $N = \lceil \log_2 n \rceil$ and $m \geq s + \epsilon$ for any arbitrarily small $\epsilon > 0$. Then there exist a positive constant $c$ and a positive integer $n_0$ such that for all $f \in H_{\text{mix}}(I^d) \cap X^m(I^d)$ and for all $n \in \mathbb{N}$ with $n \geq n_0$,

$$\|f - f_{n,N}\| \leq cn^{-s}(\|f\|_{H_{\text{mix}}(I^d)} + \|f\|_{X^m(I^d)}).$$

(3.3)

Proof. By the triangle inequality, we obtain that

$$\|f - f_{n,N}\| \leq \|f - \tilde{f}_n\| + \|\tilde{f}_n - f_{n,N}\|.$$  

(3.4)

Since $f \in H_{\text{mix}}(I^d)$, the first term on the right hand side of (3.4) can be estimated by using Theorem 1.1. It remains to consider the second term.

For each $l \in \mathbb{Z}^d$, the error of the quadrature (3.2) is denoted by $\epsilon_N(f, l) := \langle f, e_l \rangle - Q^l(f, l)$. Appealing to the definition of $\tilde{f}_n$ and $f_{n,N}$, we have that

$$\tilde{f}_n - f_{n,N} = \sum_{l \in \mathbb{Z}^d} \epsilon_N(f, l)e_l.$$  

(3.5)

Because $e_l, l \in \mathbb{L}_n$, are an orthonormal basis for the space $T_n$, we conclude that

$$\|\tilde{f}_n - f_{n,N}\| = \left( \sum_{l \in \mathbb{L}_n} |\epsilon_N(f, l)|^2 \right)^{\frac{1}{2}} \leq \left( \sum_{l \in \mathbb{Z}^d} |\epsilon_N(f, l)|^2 \right)^{\frac{1}{2}}.$$  

(3.6)

From the definition of $Q_N(f, l)$, we know that $Q_N(f, l) = \langle \delta_N f, e_l \rangle$. Thus, by employing the definition of $\epsilon_N(f, l)$, we have that $\langle f - \delta_N f, e_l \rangle = \epsilon_N(f, l)$. Hence, noting that $e_l, l \in \mathbb{Z}^d$, are an orthonormal basis, we obtain that

$$\|f - \delta_N f\| = \left( \sum_{l \in \mathbb{Z}^d} |\langle f - \delta_N f, e_l \rangle|^2 \right)^{\frac{1}{2}} = \left( \sum_{l \in \mathbb{Z}^d} |\epsilon_N(f, l)|^2 \right)^{\frac{1}{2}}.$$  

(3.7)

Combining (3.6) and (3.7) leads to $\|\tilde{f}_n - f_{n,N}\| \leq \|f - \delta_N f\|$. Noting that $\|f - \delta_N f\| \leq (2\pi)^d \|f - \delta_N f\|_{X^m(I^d)}$, from Theorem 2.12, there exists a positive constant such that

$$\|\tilde{f}_n - f_{n,N}\| \leq cn^{d-1}2^{-mN} \|f\|_{X^m(I^d)}.$$  

(3.8)

Using the choice $N = \lceil \log_2 n \rceil$ and the hypothesis on $m$, we conclude that there exist a positive constant $c$ and a positive integer $n_0$ such that for all $n \in \mathbb{N}$ with $n \geq n_0$,

$$\|\tilde{f}_n - f_{n,N}\| \leq cn^{-s+\epsilon} (\log_2 n)^{d-1} \|f\|_{X^m(I^d)} \leq cn^{-s} \|f\|_{X^m(I^d)},$$  

(3.9)

completing the proof of this theorem. □

A comment on the regularity requirement of function $f$ as stated in Theorem 3.1 is in order. Theorem 1.1 shows that the hypothesis $f \in H_{\text{mix}}(I^d)$ is enough to ensure the sparse approximation $\tilde{f}_n$ has the optimal approximation order. However, it is not sufficient to guarantee that $f_{n,N}$ preserves the optimal approximation order due to the use of the quadrature formula (3.2) in approximating the Fourier coefficients. For this reason, we impose in Theorem 3.1 an extra smoothness requirement $f \in X^m(I^d)$ with $m \geq s + \epsilon, \epsilon > 0$. The smoothness requirement $f \in H_{\text{mix}}(I^d) \cap X^m(I^d)$ in Theorem 3.1 means that the functions $f$ are $\epsilon$ degree smoother at the interior points of $I^d$ than at the boundary points of $I^d$.

We next develop a fast algorithm for computing the values of $Q_N(f, l), l \in \mathbb{L}_n$. To clarify the ideals for developing this fast algorithm, we present only the important propositions and theorems in this section, with their technical details being presented in Appendix A. Throughout the rest of this paper, we always choose $N = \lceil \log_2 n \rceil$ and thus, to simplify our notation, we use $\tilde{f}_n$ for $f_{n,N}$.
According to Theorems 1.1 and 2.13, when $N = \lceil \log_2 n \rceil$, using formula (3.2) directly to compute $Q_n(f, I)$ for all $I \in \mathbb{L}_n^d$ leads $O(n^2 \log^{2d-2} n)$ number of multiplications. Thus, we need to find a better way to compute these quantities. To do this, for a given $f \in C(I^d)$ and $j \in \mathbb{S}_N^d$ we first describe a formula for computing

$$Q_j(f, I) := \sum_{r \in \mathbb{W}_j^d} \eta_{j,r}(f) A_{j,r}(I) \quad (3.10)$$

for all $I \in \mathbb{L}_n$, by rewriting the right hand side of (3.10) in terms of the discrete Fourier transform of vector $[\eta_{j,r}(f) : r \in \mathbb{W}_j^d]$. For this development, we define the notation as follows. Let $Z_{2^{-1}} := \{0\}$ and $X_0 := \mathbb{W}_0$ and $X_j := \mathbb{W}_1$ for all $j \in \mathbb{N}$. For all $j \in \mathbb{N}_d$ and $q \in X_j$, we define index set $\mathbb{W}_{j,q} := \{2mu + q : u \in Z_{2j-1}\}$. For all $j := [j_k : k \in Z_d] \in \mathbb{N}_d^d$ we also define $X_j^d := X_{j_0} \otimes \cdots \otimes X_{j_{d-1}}$. For all $j := [j_k : k \in Z_d] \in \mathbb{N}_d^d$ and $q := [q_k : k \in Z_d] \in X_j^d$, we let $\mathbb{W}_{j,q}^d := \mathbb{W}_{j_0,0} \otimes \cdots \otimes \mathbb{W}_{j_{d-1},q_{d-1}}$. For all $j := [j_k : k \in Z_d]$ with $j_k \geq -1$, we let $2^j := [2^k : k \in Z_d]$ and $Z_{2^j}^d := Z_{2^j_0} \otimes \cdots \otimes Z_{2^j_{d-1}}$. For all $j \in \mathbb{N}_d$ and $q \in X_j$ and $l \in \mathbb{Z}$, we let

$$t_{j,q}(l) := \begin{cases} \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \ell_{0,q}(x) e^{-ixl} \, dx, & j = 0 \\ \frac{1}{\sqrt{2\pi}} \frac{1}{2^{j-1}} \int_{\mathbb{R}} \ell_{1,q}(x) e^{-ix2^{j-1}} \, dx, & j \geq 1. \end{cases}$$

For all $j := [j_k : k \in Z_d] \in \mathbb{S}_N^d$, $q := [q_k : k \in Z_d] \in X_j^d$ and $l := [l_k : k \in Z_d] \in \mathbb{L}_n$, we define

$$t_{j,q}(l) := \prod_{k \in Z_d} t_{j_k,q_k}(l_k). \quad (3.11)$$

We also let $D_j := \text{diag}[2^{1-j_k} : k \in Z_d]$ for all $j := [j_k : k \in Z_d] \in \mathbb{N}_d^d$.

With these notations, we write formula (3.10) in the form of the discrete Fourier transform of $\eta_{j,r}(f)$. For all $j \in \mathbb{S}_N^d$ and $q \in X_j^d$, we define vector $\hat{f}_{j,q}$ by

$$\hat{f}_{j,q} := \sum_{u \in \mathbb{Z}_{2^{j-1}}} \eta_{j,2mu+q}(f) e^{-i2\pi f u}, \quad I \in \mathbb{L}_n.$$  

**Proposition 3.2.** If $n, N \in \mathbb{N}$ and $f \in C(I^d)$, then for all $I \in \mathbb{L}_n$ and $j \in \mathbb{S}_N^d$,

$$Q_j(f, I) = \sum_{q \in X_j^d} t_{j,q}(I) \left( \hat{f}_{j,q} \right)_1. \quad (3.12)$$

**Proof.** From the definition of $Q_j(f, I)$ and Lemma A.2, we have that

$$Q_j(f, I) = \sum_{q \in X_j^d} \sum_{r \in \mathbb{W}_j^d} \eta_{j,r}(f) A_{j,r}(I). \quad (3.13)$$

According to the definition of $\mathbb{W}_{j,q}$, we know that $\mathbb{W}_{j,q}^d = \left\{2mu + q : u \in Z_{2j-1}^{d-1}\right\}$. From this and (3.13), we obtain that

$$Q_j(f, I) = \sum_{q \in X_j^d} \sum_{u \in \mathbb{Z}_{2^{j-1}}} \eta_{j,2mu+q}(f) A_{j,2mu+q}(I). \quad (3.14)$$
Substituting (A.34) into (3.14) leads to
\[
Q_q(f, l) = \sum_{q \in X_j^d} t_{j, q}(l) \left( \sum_{u \in \mathbb{Z}_j^d} \eta_{j, 2\pi u + q}(f)e^{-2\pi iT u} \right).
\] (3.15)

From the definition of \( \left( \hat{f}_{j, q} \right)_1 \) and (3.15), we obtain equality (3.12). \( \Box \)

The next formula for \( Q_N(f, l) \) is an immediate consequence of Proposition 3.2.

**Proposition 3.3.** If \( n, N \in \mathbb{N} \) and \( f \in C(l^d) \), then for all \( l \in \mathbb{Z}_n^d \)
\[
Q_N(f, l) = \sum_{j \in \mathbb{N}} \sum_{q \in X_j^d} t_{j, q}(l) \left( \hat{f}_{j, q} \right)_1.
\] (3.16)

When using the formula (3.16) to compute \( Q_N(f, l) \), we must have the values of \( \left( \hat{f}_{j, q} \right)_1 \) and \( t_{j, q}(l) \) for all \( j \in \mathbb{N}_N \), \( q \in X_j^d \) and \( l \in \mathbb{Z}_n^d \). We first show how we compute the values of \( \left( \hat{f}_{j, q} \right)_1 \). Note that \( \hat{f}_{j, q} \) is the discrete Fourier transform of the vector \( f_{j, q} := \left[ \eta_{j, 2\pi u + q}(f) : u \in \mathbb{Z}_j^d \right] \). Hence, for each \( j \in \mathbb{N}_N \) and \( q \in X_j^d \), we compute \( \left( \hat{f}_{j, q} \right)_1 \). For \( l \in \mathbb{Z}_n^d \), we obtain the value \( \left( \hat{f}_{j, q} \right)_1 \) by employing the periodicity of the discrete Fourier transform. To explain this point, for each \( j \in \mathbb{N} \) and \( l \in \mathbb{Z} \), we define \( L_j : \mathbb{Z} \rightarrow \mathbb{Z}_j^d \) by
\[
L_j(l) := \begin{cases} l \mod 2^{j-1}, & \text{if } l \mod 2^{j-1} \geq 0, \\ 2^{j-1} + l \mod 2^{j-1}, & \text{if } l \mod 2^{j-1} < 0. \end{cases}
\]

For \( j = 0 \) and for all \( l \in \mathbb{Z} \), we let \( L_0(l) := l \). For \( j := \left[ j_k : k \in \mathbb{Z}_d \right] \in \mathbb{S}^d_N \) and for all \( l := \left[ l_k : k \in \mathbb{Z}_d \right] \in \mathbb{Z}^d \), we define \( L_j(l) := \left[ L_j(l_k) : k \in \mathbb{Z}_d \right] \). By using the periodicity of the discrete Fourier transform of \( f_{j, q} \), we have for all \( q \in X_j^d \) and all \( l \in \mathbb{Z}_n^d \) that
\[
\left( \hat{f}_{j, q} \right)_1 = \left( \hat{f}_{j, q} \right)_{L_j(l)}.
\] (3.17)

By combining Eqs. (3.16) and (3.17), we obtain the alternative formula
\[
Q_N(f, l) = \sum_{j \in \mathbb{N}} \sum_{q \in X_j^d} t_{j, q}(l) \left( \hat{f}_{j, q} \right)_{L_j(l)}.
\] (3.18)

To obtain the values of \( t_{j, q}(l) \), \( j \in \mathbb{S}^d_N \), \( q \in X_j^d \) and \( l \in \mathbb{Z}_n^d \), we next derive the discrete formula for computing \( t_{j, q}(l) \) for all \( j \in \mathbb{Z}_N \), \( q \in X_j \) and \( l \in \mathbb{Y}_n \). To this end, we consider the formula for computing
\[
t_q(\omega) := \int_I \ell_q(x)e^{-i\omega x}dx,
\] (3.19)
for all \( q \in \mathbb{Z}_m \) and \( \omega \in \mathbb{R} \). For each \( q \in \mathbb{Z}_m \) and \( \theta \in \mathbb{Z}_{m-1} \), we define the index set
\[
S_{q, \theta} = \left\{ s \in \mathbb{Z}_m^{m-\theta-1} : s_k < s_{k+1} \text{ for } k \in \mathbb{Z}_{m-\theta-2}, s_k \neq q \text{ for } k \in \mathbb{Z}_{m-\theta-1} \right\}.
\] (3.20)
For \( q \in \mathbb{Z}_m \), we first write
\[
\ell_q(x) = \sum_{\theta \in \mathbb{Z}_m} c_{q, \theta} x^\theta, \quad x \in I.
\] (3.21)
where
\[
c_{q,\theta} = \begin{cases}
(-1)^{m-\theta-1} \left( \prod_{s \in \mathbb{Z}_m, s \neq q} (v_q - v_s) \right)^{-1} \left( \sum_{s \in \mathbb{Z}_m, k \neq q, k \neq s} v_s \right), & \theta \in \mathbb{Z}_{m-1}, \\
\left( \prod_{s \in \mathbb{Z}_m, s \neq q} (v_q - v_s) \right)^{-1}, & \theta = m - 1.
\end{cases}
\] (3.22)

We then have for all \( q \in \mathbb{Z}_m \) and \( \omega \in \mathbb{R} \) that
\[
t_q(\omega) = \sum_{\theta \in \mathbb{Z}_m} c_{q,\theta} \tilde{l}_\theta(\omega),
\] (3.23)
where \( \tilde{l}_\theta(\omega) := \int_I x^\theta e^{-i\omega x} \, dx \) are computed exactly. In fact, we have the following result.

**Lemma 3.4.** For all \( \theta \in \mathbb{Z}_m \) and \( \omega \in \mathbb{R} \), there holds the formula
\[
\tilde{l}_\theta(\omega) = \begin{cases}
(2\pi)^{\theta+1} / \theta + 1, & \omega = 0, \\
- \sum_{i=0}^{\theta-1} \theta! (2\pi)^{\theta-i} e^{-i2\pi \omega} / (\theta-i)! (i\omega)^{\theta+1} + \theta! / (i\omega)^{\theta+1}, & \omega \neq 0.
\end{cases}
\] (3.24)

**Proof.** The proof for the case \( \omega = 0 \) is trivial. We next prove Eq. (3.24) for \( \omega \in \mathbb{R} \) with \( \omega \neq 0 \). To do this, an induction on \( \theta \) with integration by parts proves that if \( \omega \in \mathbb{C} \) with \( \omega \neq 0 \), then for all \( \theta \in \mathbb{Z}_m \),
\[
\int_I x^\theta e^{i\omega} \, dx = \sum_{i=0}^{\theta-1} (-1)^i \theta! (2\pi)^{\theta-i} e^{i2\pi \omega} / (\theta-i)! (i\omega)^{\theta+1} + (-1)^{\theta+1} \theta! / (i\omega)^{\theta+1}.
\] (3.25)
Replacing \( \omega \) by \( -i\omega \) in Eq. (3.25), we obtain for \( \omega \neq 0 \) that
\[
\int_I x^\theta e^{i\omega} \, dx = - \sum_{i=0}^{\theta-1} \theta! (2\pi)^{\theta-i} e^{i2\pi \omega} / (\theta-i)! (i\omega)^{\theta+1} + \theta! / (i\omega)^{\theta+1},
\] (3.26)
yielding the desired formula. \( \square \)

Note that when \( j = 0 \), for all \( q \in \mathbb{X}_0 \), there holds \( \ell_{0,q} = \ell_q \). Thus, from the definition of \( t_{0,q} \) and (3.23), we know for all \( q \in \mathbb{X}_0 \) and \( l \in \mathbb{Y}_n \) that
\[
t_{0,q}(l) = \frac{1}{\sqrt{2\pi}} \sum_{\theta \in \mathbb{Z}_m} c_{q,\theta} \tilde{l}_\theta(l).
\] (3.27)
We recall that when \( j \geq 1 \), for all \( q \in \mathbb{X}_j = \mathbb{W}_1 \),
\[
t_{j,q}(l) = \frac{1}{2^{j-1} \sqrt{2\pi}} \int_I \ell_{1,q}(x) e^{-ik \xi^{[2]}-1} \, dx.
\]
Note that for each \( q \in \mathbb{W}_1 \), \( \ell_{1,q} \) is generated by scaling and translating a function \( \ell_r, r \in \mathbb{Z}_m \). Specifically, for all \( q \in \mathbb{W}_1 \), \( \ell_{1,q} = \mathcal{T}_{[q/m]} \ell_q \mod m \). Thus, we have that
\[
t_{j,q}(l) = \frac{1}{2^{j-1} \sqrt{2\pi}} \int_{[q/m] \pi}^{(1+|q/m|)\pi} \ell_q \mod m (2(x - [q/m] \pi)) e^{-ik \xi^{[2]}-1} \, dx.
\] (3.28)
Replacing \( 2(x - [q/m] \pi) \) by \( x \) in the integral of (3.28), we obtain for all \( j \geq 1 \), \( q \in \mathbb{X}_j \) and \( l \in \mathbb{Y}_n \) that
\[
t_{j,q}(l) = \frac{e^{-i2\pi |q/m|/2^j}}{2^{j-1} \sqrt{2\pi}} \int_0^{2\pi} \ell_q \mod m (x) e^{-ik x/2} \, dx.
\] (3.29)
Combining (3.19), (3.23), (3.27) and (3.29), we see for all \( j \in \mathbb{Z}_N \), \( q \in X_j \) and \( l \in Y_n \) that

\[
    t_{j,q}(l) = \frac{e^{-2\pi i q/m}}{2l/\sqrt{2\pi}} \sum_{\theta \in \mathbb{Z}_m} c_{q \mod m, \theta} \tilde{t}_0(l/2^j). \tag{3.30}
\]

We now describe the fast algorithm for computing \( Q_N(f, I) \), \( I \in \mathbb{Z}_n^d \).

**Algorithm 3.5.** Given \( n \in \mathbb{N} \), \( N = \lceil \log_2 n \rceil \) and function \( f \in C(I^d) \), we choose a refinable set \( \{v_r : r \in \mathbb{Z}_m\} \).

**Step 1** Compute \( f_{j,q} := \left[ \eta_{j,2m+q}(f) : u \in \mathbb{Z}_{2^j-1}^d \right] \), \( j \in \mathbb{S}_N^d \) and \( q \in X_j^d \).

**Step 2** Compute \( \tilde{f}_{j,q} \), \( j \in \mathbb{S}_N^d \) and \( q \in X_j^d \), by applying the fast Fourier transform to \( f_{j,q} \).

**Step 3** Compute \( c_{q,0}, q, \theta \in \mathbb{Z}_m \), according to Eq. (3.22).

**Step 4** Compute \( \tilde{t}_0(l/2^j) \), \( l \in \mathbb{V}_n \), \( j \in \mathbb{Z}_N \) and \( q \in X_j \) according to formula (3.24).

**Step 5** Compute \( t_{j,q}(l) \), \( l \in \mathbb{V}_n \), \( j \in \mathbb{Z}_N \) and \( q \in X_j \) according to formula (3.30).

**Step 6** Compute \( t_{j,q}(l) \) for all \( j \in \mathbb{S}_N^d \), \( q \in X_j^d \) and \( l \in \mathbb{I}_n^d \) by (3.11).

**Step 7** Compute \( Q_N(f, I) \) for all \( I \in \mathbb{I}_n^d \) according to formula (3.18).

The number of multiplications \( M_n \) used in Algorithm 3.5 is estimated in the next theorem.

**Theorem 3.6.** Let \( m \in \mathbb{N} \) and \( d \in \mathbb{N} \) be fixed. If for each \( x \in I^d \), the number of multiplications used in computing \( f(x) \) is constant, then there exists a positive constant \( c \) such that for all \( n \in \mathbb{N} \),

\[
    M_n \leq cn \log^2 d n. \tag{3.31}
\]

**Proof.** For simplicity, we use \( M^{(v)} \) to denote the number of multiplications used in the \( v \)th step in Algorithm 3.5, \( v = 1, 2, \ldots, 7 \).

In Step 1, we compute \( f_{j,q} \), \( j \in \mathbb{S}_N^d \) and \( q \in X_j^d \). For each \( j \in \mathbb{S}_N^d \), \( q \in X_j^d \) and \( u \in \mathbb{Z}_{2^j-1}^d \), the number of the multiplications used in computing \( 2m \mu + q \) is \( \Theta(1) \). Lemma A.2 shows that the total number of elements of \( f_{j,q} \) for all \( j \in \mathbb{S}_N^d \) and \( q \in X_j^d \) equals to the number of \( \eta_{j,r}(f) \) for all \( j \in \mathbb{S}_N^d \) and \( r \in \mathbb{V}_j^d \). From the definition of \( \eta_{j,r} \), \( j \in \mathbb{S}_N^d \) and \( r \in \mathbb{V}_j^d \), we know that the number of functional evaluations used in computing \( \eta_{j,r}(f) \) is \( \Theta(1) \). Thus, by hypothesis, the number of the multiplications used in computing \( \eta_{j,r}(f) \) is a constant. By Theorem 2.13, the total number of \( \eta_{j,r}(f) \) for all \( j \in \mathbb{S}_N^d \) and \( r \in \mathbb{V}_j^d \) is \( \Theta(N^{d-1} 2^N) \). Note that \( N < \log_2 n + 1 \). Thus, \( M^{(1)} = \Theta(n \log^2 d n) \).

We use \( M_{j,q}^{(2)} \) to denote the number of multiplications used in computing the fast Fourier transform of \( f_{j,q} \). It is well known that there exists a positive constant \( c \) such that for all \( n \in \mathbb{N} \), \( j \in \mathbb{S}_N^d \), and \( q \in X_j^d \),

\[
    M_{j,q}^{(2)} \leq c |j| 2^{|j|}. \tag{3.32}
\]

Note that for all \( j \in \mathbb{S}_N^d \), \( |j| \leq N \). Applying Lemma 3.6 of [23] to the right hand side of (3.32) leads to \( M^{(2)} \leq c N d^2 2^N \). Again, since \( N < \log_2 n + 1 \), there exists a positive constant \( c \) such that for all \( n \in \mathbb{N} \),

\[
    M^{(2)} = \Theta(n \log^2 d n). \tag{3.22}
\]

The number of multiplications used in Step 3 to compute \( c_{q,0}, q, \theta \in \mathbb{Z}_m \), according to equations (3.22) is a constant with respect to \( n \), hence, \( M^{(3)} = \Theta(1) \).

In Step 4, from Eq. (3.24), for each \( l \in \mathbb{V}_n \), \( j \in \mathbb{Z}_N \) and \( q \in X_j \), the number of the multiplications used in computing \( \tilde{t}_0(l/2^j) \) is \( \Theta(1) \). Noting that there are \( n N (2n - 1) \) elements in \( \{ \tilde{t}_0(l/2^j) : l \in \mathbb{V}_n, j \in \mathbb{Z}_N, \theta \in \mathbb{Z}_m \} \) and \( N < \log_2 n + 1 \), we conclude that \( M^{(4)} = \Theta(n \log^2 n) \).

From Eq. (3.30), for each \( l \in \mathbb{V}_n \), \( j \in \mathbb{Z}_N \) and \( q \in X_j \), the number of the multiplications used in Step 5 for computing \( t_{j,q}(l) \) is \( \Theta(1) \). Noting that there are \( n N (2n - 1) \) elements in \( \{ t_{j,q}(l) : l \in \mathbb{V}_n, j \in \mathbb{Z}_N, q \in X_j \} \) and \( N < \log_2 n + 1 \), there holds \( M^{(5)} = \Theta(n \log^2 n) \).
Theorem 1.1 shows that there are $\Theta(N^d n \log^{d-1} n)$ elements in $\{ t_{j,q}(l) : l \in L_n^d, j \in S_n^d, q \in X_n^d \}$. For each $l \in L_n^d, j \in S_n^d$ and $q \in X_n^d$, from (3.11) the number of the multiplications used in computing $t_{j,q}(l)$ is a constant. Thus, by noting that $N < \log_2 n + 1$, $\mathcal{M}(6) = \Theta(n \log^{d-1} n)$.

In Step 7, to compute $Q_n(f, l)$ for all $l \in L_n^d$ according to Eq. (3.18) needs $\mathcal{O}(N^d n \log^{d-1} n)$ number of multiplications. That is, $\mathcal{M}(7) = \Theta(n \log^{d-1} n)$.

The total number of multiplications used in Algorithm 3.5 equals to the sum of the number of multiplications used in each step. Thus, we conclude for all $n \in \mathbb{N}$ that

$$\mathcal{M}_n = \sum_{v=1}^{7} \mathcal{M}^{(v)} = \Theta(n \log^{d-1} n),$$

proving the result of this theorem. □

To close this section we remark on the constants which appear in Theorems 3.1 and 3.6. These two constants depend on the spacial dimension $d$. It is useful to know how they depend on $d$. For this purpose, we use $c_e$ and $c_t$ to denote the constants appeared in Theorems 3.1 and 3.6, respectively. For all $d \in \mathbb{N}$, we also let $A(d) := 2 \left( \frac{1}{(d - 1)!} + c \right)$, where $c$ is a positive constant independent of $d$. It follows from [2,6] that the constant in (1.5) is independent of $d$ and the constant in (1.6) is bounded above by $A(d)$. It can be verified that the constant $c_e$ appeared in Theorem 3.1 and the constant $c_t$ appeared in Theorem 3.6 have the bound

$$c_e \leq c \left( \frac{(2\pi)^{m+1}}{m!} \right)^d A(d) \quad \text{and} \quad c_t \leq c \frac{m^d}{(d - 1)!} A(d),$$

respectively, where $c$ is independent of $m$ or $d$. It is clear from the above estimates that these two constants depend on $d$ but their dependency on $d$ is not severe. This is confirmed by Figs. 3 and 4 of Examples 1 and 2 in Section 4.

4. Numerical examples

We present in this section eight numerical examples to demonstrate the accuracy and computational efficiency of Algorithm 3.5. In particular, we compare the performance of Algorithm 3.5 with that of the well-known Fourier transform on sparse grids (FTSG) implemented in [11] in both approximation accuracy and computational efficiency. The computer programs for the examples are run on a workstation with a Intel(R) (TM) 3.06 GHz AT/AT compatible CPU and 3 GB memory. In the numerical examples, we use “CT” to denote computing time spent in calculating all coefficients $Q_n(f, l), l \in L_n^d$, measured in seconds. We define the compression rate by $CR := \mathcal{N}_n/n^d$. We also define the relative error $Err := \| \hat{f} - \tilde{f} \|_n$ and the approximation order $AO := \log_2 \| f - \hat{f} \|_n$. We use $\mathcal{M}_n$ to represent the number of multiplications actually used in computation.

In the first six examples, we will plot the constants which appear in the estimates of approximation accuracy and computational costs. For this purpose, we define $C_{error} := \| \hat{f} - \tilde{f} \|_n/n^d$ and $C_{time} := \mathcal{CT}/n \log^{d-1} n$. The quantities $C_{error}$ and $C_{time}$ are computed values of the constants $c_e$ and $c_t$ that appear in the estimates of approximation order and number of multiplications used in Algorithm 3.5, respectively. We will plot these two quantities for the first five examples presented below. In Examples 1 and 3–6, the parameter $m$ in the proposed numerical quadrature is chosen to be 2 and the refinable set is $V := \{ v_r := \frac{x+1}{2} : x \in \mathbb{Z}_2 \}$. In Example 2, $m$ is chosen to be 3 and the refinable set is $V := \{ v_r := \frac{x}{3} : x \in \mathbb{Z}_3 \}$. To computing “Err” and “$C_{error}$”, we replace $f$ in the definitions of “Err” and “$C_{error}$” by $\tilde{f}_n$ with $n = 65,536, 4096, 4096$ and 1024 in the cases of $d = 2, 3, 4$ and $d = 8$, respectively.

The first two examples are designed to confirm the theoretical estimates of Algorithm 3.5 and to compare its computational performance with that of Fourier transform on sparse grids (FTSG) for functions with the Sobolev regularity $s > 1$ with $d = 2, 3, 4$. The functions to be approximated in Example 1 are in spaces $H_{mix}^{1.5-p} (I^d)$, while the functions in Example 2 are in spaces $H_{mix}^{2.5-p} (I^d)$ where
\[ \varepsilon > 0 \text{ for } d = 2, 3, 4. \] Therefore, the theoretical approximation order are \( 1.5 - \varepsilon \) and \( 2.5 - \varepsilon \), respectively for these two types of functions. We choose \( s = 1.49 \) and \( s = 2.49 \), respectively, in computing “Err” for these cases. Since for fixed \( n \) and \( d \), the numbers \( \text{CR} \) with respect to different functions are the same, we present only in Example 1 these numbers.

Example 1. We consider in this example the functions

\[ g_d(x) = \prod_{k \in \mathbb{Z}_d} (x_k - \pi)^2, \quad x \in I^d, \]

for \( d = 2, 3 \), where \( x := [x_k : k \in \mathbb{Z}_d] \). The numerical results are listed in Tables 1, 2 and 9 for the case \( d = 2 \), the case \( d = 3 \) and the case \( d = 4 \), respectively.

We plot in Fig. 3 the values of \( \text{C}_{\text{error}} \) (image (a)) and \( \text{C}_{\text{time}} \) (image (b)), respectively. In image (a), we use the solid blue (resp. dash red) line with \( * \), \( \Delta \), and \( \Box \) to represent the values of \( \text{C}_{\text{error}} \) of Algorithm 3.5 (resp. the FTSG) for the cases \( d = 2, 3, 4 \), respectively. In image (b), we use the solid blue lines with \( * \), \( \Delta \) and \( \Box \) to present the values of \( \text{C}_{\text{time}} \) of Algorithm 3.5 for the cases \( d = 2, 3, 4 \), respectively (Table 3).
### Table 1
Numerical results for function $g_2$ of two variables in Example 1.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT CR Err AO</td>
<td>CT Err AO</td>
</tr>
<tr>
<td>64</td>
<td>0.01 8.52% 2.20e−2 1.95e+3</td>
<td>0.01 4.26e−2</td>
</tr>
<tr>
<td>128</td>
<td>0.06 5.28% 6.53e−3 1.7523 5.02e+3</td>
<td>0.01 1.63e−2 1.3969</td>
</tr>
<tr>
<td>256</td>
<td>0.19 3.16% 1.88e−3 1.7964 1.23e+4</td>
<td>0.02 6.19e−3 1.3969</td>
</tr>
<tr>
<td>512</td>
<td>0.53 1.85% 5.31e−4 1.8239 2.92e+4</td>
<td>0.03 2.33e−3 1.4096</td>
</tr>
<tr>
<td>1024</td>
<td>1.52 1.06% 1.48e−4 1.8431 6.76e+4</td>
<td>0.05 8.71e−4 1.4196</td>
</tr>
<tr>
<td>2048</td>
<td>4.11 0.59% 4.06e−5 1.8660 1.54e+5</td>
<td>0.06 3.26e−4 1.4178</td>
</tr>
</tbody>
</table>

### Table 2
Numerical results for function $g_3$ of three variables in Example 1.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT CR Err AO</td>
<td>CT Err AO</td>
</tr>
<tr>
<td>16</td>
<td>0.01 2.71% 1.58e−1 1.32e+3</td>
<td>0.01 9.01e−1</td>
</tr>
<tr>
<td>32</td>
<td>0.06 1.14% 1.13e−1 0.4836 5.42e+3</td>
<td>0.01 4.75e−1 0.9236</td>
</tr>
<tr>
<td>64</td>
<td>0.31 0.443% 5.48e−2 1.0441 1.88e+4</td>
<td>0.02 2.26e−1 1.0716</td>
</tr>
<tr>
<td>128</td>
<td>1.41 0.162% 2.00e−2 1.4542 5.86e+4</td>
<td>0.03 9.82e−2 1.3791</td>
</tr>
<tr>
<td>256</td>
<td>5.79 0.056% 6.67e−3 1.5842 1.69e+5</td>
<td>0.04 4.09e−2 1.3974</td>
</tr>
<tr>
<td>512</td>
<td>21.86 0.019% 2.03e−3 1.7162 4.63e+5</td>
<td>0.05 1.66e−2 1.4115</td>
</tr>
<tr>
<td>1024</td>
<td>74.43 0.006% 5.98e−4 1.7633 1.22e+6</td>
<td>0.09 6.60e−3 1.4174</td>
</tr>
<tr>
<td>2048</td>
<td>21.86 0.019% 2.03e−3 1.7162 4.63e+5</td>
<td>0.05 1.66e−2 1.4115</td>
</tr>
</tbody>
</table>

### Table 3
Numerical results for function $g_4$ of four variables in Example 1.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT CR Err AO</td>
<td>CT Err AO</td>
</tr>
<tr>
<td>16</td>
<td>0.06 1.14% 1.13e−1 5.82e+3</td>
<td>0.01 2.67e−0</td>
</tr>
<tr>
<td>32</td>
<td>0.55 0.0357% 1.52e−2 2.8942 2.96e+4</td>
<td>0.01 1.68e−0 0.6684</td>
</tr>
<tr>
<td>64</td>
<td>3.59 0.0069% 6.09e−3 1.3196 1.24e+5</td>
<td>0.02 9.29e−1 0.8547</td>
</tr>
<tr>
<td>128</td>
<td>21.72 0.0013% 2.11e−3 1.5292 4.54e+5</td>
<td>0.03 4.66e−1 0.9953</td>
</tr>
<tr>
<td>256</td>
<td>115.16 0.0003% 7.64e−4 1.4656 1.52e+6</td>
<td>0.08 2.19e−1 1.0894</td>
</tr>
<tr>
<td>512</td>
<td>562.16 0.00039% 2.76e−4 1.4689 4.72e+6</td>
<td>0.17 9.75e−2 1.1675</td>
</tr>
<tr>
<td>1024</td>
<td>0.44 4.19e−2</td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td>1.06 1.76e−2</td>
<td>1.2185</td>
</tr>
</tbody>
</table>

### Example 2
We consider in this example the tensor of quadratic B-Spline. We define quadratic B-Spline by

$$
B(x) = \begin{cases} 
\frac{1}{2}x^2, & 0 \leq x < \frac{1}{3}, \\
-x^2 + x - \frac{1}{6}, & \frac{1}{3} \leq x < \frac{2}{3}, \ x \in I. \\
\frac{1}{2}(1-x)^2, & \frac{2}{3} \leq x \leq 1.
\end{cases}
$$

We also define

$$
B_d(x) = \prod_{k \in \mathbb{Z}_d} B(x_k) \quad x \in I^d,
$$
Table 4
Numerical results for function $B_2$ of two variables in Example 2.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th></th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\text{CT}$</td>
<td>$\text{Err}$</td>
<td>$\text{AO}$</td>
</tr>
<tr>
<td>64</td>
<td>0.04</td>
<td>6.02e-4</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>0.05</td>
<td>1.16e-4</td>
<td>2.3756</td>
</tr>
<tr>
<td>256</td>
<td>0.18</td>
<td>2.20e-5</td>
<td>2.3985</td>
</tr>
<tr>
<td>512</td>
<td>0.55</td>
<td>4.13e-6</td>
<td>2.4133</td>
</tr>
<tr>
<td>1024</td>
<td>1.61</td>
<td>7.71e-7</td>
<td>2.4213</td>
</tr>
<tr>
<td>2048</td>
<td><strong>4.54</strong></td>
<td><strong>1.43e-7</strong></td>
<td>2.4307</td>
</tr>
<tr>
<td>4096</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8192</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5
Numerical results for function $B_3$ of three variables in Example 2.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th></th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\text{CT}$</td>
<td>$\text{Err}$</td>
<td>$\text{AO}$</td>
</tr>
<tr>
<td>16</td>
<td>0.25</td>
<td>3.10e-2</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>0.64</td>
<td>8.23e-3</td>
<td>1.9133</td>
</tr>
<tr>
<td>64</td>
<td>3.21</td>
<td>1.79e-3</td>
<td>2.2009</td>
</tr>
<tr>
<td>128</td>
<td>14.65</td>
<td>3.69e-4</td>
<td>2.2783</td>
</tr>
<tr>
<td>256</td>
<td>60.29</td>
<td>7.55e-5</td>
<td>2.2891</td>
</tr>
<tr>
<td>512</td>
<td><strong>233.39</strong></td>
<td><strong>1.53e-5</strong></td>
<td>2.3029</td>
</tr>
<tr>
<td>1024</td>
<td>886.95</td>
<td>3.03e-6</td>
<td>2.3361</td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 6
Numerical results for function $B_4$ of four variables in Example 2.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th></th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\text{CT}$</td>
<td>$\text{Err}$</td>
<td>$\text{AO}$</td>
</tr>
<tr>
<td>16</td>
<td>1.47</td>
<td>2.98e-2</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>10.85</td>
<td>1.29e-2</td>
<td>1.2079</td>
</tr>
<tr>
<td>64</td>
<td>76.53</td>
<td>4.28e-3</td>
<td>1.5917</td>
</tr>
<tr>
<td>128</td>
<td>723.62</td>
<td>1.01e-3</td>
<td>2.0833</td>
</tr>
<tr>
<td>256</td>
<td><strong>5316.86</strong></td>
<td><strong>2.25e-4</strong></td>
<td>2.1664</td>
</tr>
<tr>
<td>512</td>
<td>25425.60</td>
<td>4.89e-5</td>
<td>2.2020</td>
</tr>
<tr>
<td>1024</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

for $d = 2, 3, 4$, where $x := [x_k : k \in \mathbb{Z}_d]$. The numerical results are listed in Tables 4–6 for the cases $d = 2, 3, 4$, respectively. The values of $C_{\text{error}}$ (image (a)) and $C_{\text{time}}$ (image (b)), are also plotted in Fig. 4, respectively, with the same notations as in Fig. 3.

The numerical results listed for the above two examples confirm the theoretical estimates for approximation order and computational complexity, of the proposed method. These numerical results also show that when the Sobolev regularity of a function is 1.5 or 2.5, Algorithm 3.5 uses less terms in the Fourier expansion but more computing time in producing similar approximation results in comparison to the FTSG for $d = 2, 3, 4$.

In the next three examples, we compare the computational performance of Algorithm 3.5 with that of the Fourier transform on sparse grids (FTSG) for functions with the Sobolev regularity equal to 0.5. That is, these functions are in $H^{0.5-\varepsilon}(I^d)$ where $\varepsilon > 0$. Therefore, the theoretical approximation order is $0.5 - \varepsilon$. In these examples, we choose $s = 0.49$ in computing "Err".

Example 3. We consider the functions

$$f_d(x) = \prod_{k \in \mathbb{Z}_d} x_k, \quad x \in I^d$$

(4.1)
for $d = 2, 3, 4$ and $8$, where $x := [x_k : k \in \mathbb{Z}_d]$. The numerical results are listed in Tables 7–9 for the cases $d = 2$, $d = 3$ and $d = 4$, respectively. The values of $C_{\text{error}}$ (image (a)) and $C_{\text{time}}$ (image (b)), are plotted in Fig. 5, respectively, with the same notations as in Fig. 3.

**Example 4.** We consider the functions

$$a_d(x) = \prod_{k \in \mathbb{Z}_d} x_k^{0.1}, \quad x \in I^d$$

for $d = 2, 3, 4$, where $x := [x_k : k \in \mathbb{Z}_d]$. For each $d = 2, 3, 4$, $a_d \in H^0.5-\varepsilon_{\text{mix}}(I^d)$ where $\varepsilon > 0$ (Fig. 6). Therefore, the theoretical approximation order is $0.5 - \varepsilon$. To compute the “Err”, we choose $s = 0.49$. The numerical results are listed in Tables 10–12 for the cases $d = 2, 3, 4$, respectively. The values of $C_{\text{error}}$ (image (a)) and $C_{\text{time}}$ (image (b)), are plotted in Fig. 6, respectively, with the same notations as in Fig. 3.

| Table 7 | Numerical results for function $f_2$ of two variables in Example 3. |
|--------|-----------------|-----------------|
| $n$    | Algorithm 3.5   | FTSG            |
|        | CT | Err | AO | CT | Err | AO |
| 64     | 0.02 | 6.23e−2 | 0.01 | 2.59e−1 |
| 128    | 0.05 | 4.46e−2 | 0.4822 | 1.98e−1 | 0.3875 |
| 256    | 0.17 | 3.19e−2 | 0.4835 | 1.50e−1 | 0.4005 |
| 512    | 0.52 | 2.27e−2 | 0.4909 | 1.13e−1 | 0.4086 |
| 1024   | 1.47 | 1.61e−2 | 0.4956 | 8.42e−2 | 0.4244 |
| 2048   | 4.05 | 1.14e−2 | 0.4980 | 6.25e−2 | 0.4300 |
| 4096   | 0.52 | 2.27e−2 | 0.4909 | 1.13e−1 | 0.4086 |
| 8192   | 0.20 | 3.40e−2 | 0.4392 |
| 16384  | 0.45 | 2.52e−2 | 0.4321 |

| Table 8 | Numerical results for function $f_3$ of three variables in Example 3. |
|--------|-----------------|-----------------|
| $n$    | Algorithm 3.5   | FTSG            |
|        | CT | Err | AO | CT | Err | AO |
| 16     | 0.01 | 1.62e−1 | 0.01 | 5.17e−1 |
| 32     | 0.05 | 1.17e−1 | 0.4695 | 4.78e−1 | 0.1132 |
| 64     | 0.29 | 8.48e−2 | 0.4644 | 4.17e−1 | 0.1970 |
| 128    | 1.36 | 6.04e−2 | 0.4895 | 3.51e−1 | 0.2486 |
| 256    | 5.66 | 4.29e−2 | 0.4936 | 2.88e−1 | 0.2854 |
| 512    | 21.38 | 7.07e−2 | 0.4956 | 2.31e−1 | 0.3182 |
| 1024   | 69.57 | 2.15e−2 | 0.4980 | 1.83e−1 | 0.3360 |
| 2048   | 0.11 | 1.43e−1 | 0.1032 |

| Table 9 | Numerical results for function $f_4$ of four variables in Example 3. |
|--------|-----------------|-----------------|
| $n$    | Algorithm 3.5   | FTSG            |
|        | CT | Err | AO | CT | Err | AO |
| 16     | 0.06 | 1.77e−1 | 0.01 | 4.61e−1 |
| 32     | 0.48 | 1.32e−1 | 0.4232 | 4.56e−1 | 0.0157 |
| 64     | 3.52 | 9.76e−2 | 0.4356 | 4.49e−1 | 0.0223 |
| 128    | 21.38 | 7.07e−2 | 0.4652 | 4.18e−1 | 0.1032 |
| 256    | 114.69 | 4.97e−2 | 0.5085 | 3.74e−1 | 0.1605 |
| 512    | 560.78 | 3.52e−2 | 0.4977 | 3.26e−1 | 0.1982 |
| 1024   | 2048 | 0.41 | 2.76e−1 | 0.2402 |
| 2048   | 0.86 | 2.29e−1 | 0.2693 |
Fig. 5. (a) Values $C_{\text{error}}$ and (b) Values $C_{\text{time}}$ of functions $f_d$, $d = 2, 3$ and 4, in Example 3.

Fig. 6. (a) Values $C_{\text{error}}$ and (b) Values $C_{\text{time}}$ of functions $a_d$, $d = 2, 3$ and 4, in Example 4.

Table 10
Numerical results for function $a_2$ of two variables in Example 4.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT</td>
<td>Err</td>
</tr>
<tr>
<td>64</td>
<td>0.02</td>
<td>3.15e−2</td>
</tr>
<tr>
<td>128</td>
<td><strong>0.05</strong></td>
<td>2.30e−2</td>
</tr>
<tr>
<td>256</td>
<td>0.19</td>
<td>1.68e−2</td>
</tr>
<tr>
<td>512</td>
<td>0.53</td>
<td>1.22e−2</td>
</tr>
<tr>
<td>1024</td>
<td>1.49</td>
<td>8.73e−3</td>
</tr>
<tr>
<td>2048</td>
<td>4.11</td>
<td>6.13e−3</td>
</tr>
<tr>
<td>4096</td>
<td>0.05</td>
<td>5.64e−2</td>
</tr>
<tr>
<td>8192</td>
<td>0.13</td>
<td>4.08e−2</td>
</tr>
<tr>
<td><strong>16384</strong></td>
<td><strong>0.30</strong></td>
<td><strong>2.94e−2</strong></td>
</tr>
</tbody>
</table>

Example 5. We consider in this example the functions

$$b_d(x) = \prod_{k \in \mathbb{Z}^d} \left(\pi^2 - (x_k - \pi)^2\right)^{0.1}, \quad x \in \mathbb{I}^d,$$
for $d = 2, 3, 4$, where $x := [x_k : k \in \mathbb{Z}_d]$. For each $d = 2, 3, 4$, $b_d \in H_{\text{mix}}^{0.5 - \varepsilon}(I^d)$ where $\varepsilon > 0$. To compute the “Err”, we choose $s = 0.5$. The numerical results are listed in Tables 13–15 for the

| Table 11 | Numerical results for function $a_3$ of three variables in Example 4. |
|-----------|------------------|-----------|
| $n$       | Algorithm 3.5    | FTSG      |
|          | CT   | Err  | AO | CT   | Err  | AO |
| 16       | **0.02** | 7.56e−2 |     |     | 0.01 | 8.64e−1 |
| 32       | 0.06  | 5.34e−2 | 0.5014 |     | 0.01 | 7.36e−1 | 0.2313 |
| 64       | 0.31  | 3.83e−2 | 0.4788 |     | 0.02 | 6.16e−1 | 0.2551 |
| 128      | 1.41  | 2.75e−2 | 0.4807 |     | 0.02 | 5.01e−1 | 0.2986 |
| 256      | 5.77  | 1.93e−2 | 0.5049 |     | 0.03 | 3.99e−1 | 0.3305 |
| 512      | 21.86 | 1.31e−2 | 0.5670 |     | 0.03 | 3.12e−1 | 0.3548 |
| 1024     | 74.06 | 8.99e−3 | 0.5370 |     | 0.08 | 2.41e−1 | 0.3737 |
| **2048** |       | **0.19** | **1.84e−1** |     |     |     |

| Table 12 | Numerical results for function $a_4$ of four variables in Example 4. |
|-----------|------------------|-----------|
| $n$       | Algorithm 3.5    | FTSG      |
|          | CT   | Err  | AO | CT   | Err  | AO |
| 16       | **0.06** | 8.48e−2 |     |     | 0.01 | 8.31e−1 | 0.0512 |
| 32       | 0.52  | 5.93e−2 | 0.5150 |     | 0.01 | 8.31e−1 |
| 64       | 3.59  | 4.26e−2 | 0.4782 |     | 0.02 | 7.66e−1 | 0.1175 |
| 128      | 21.74 | 3.08e−2 | 0.4691 |     | 0.03 | 6.79e−1 | 0.1739 |
| 256      | 116.52| 2.21e−2 | 0.4789 |     | 0.06 | 5.83e−1 | 0.2199 |
| 512      | 568.19| 1.55e−2 | 0.5096 |     | 0.13 | 4.87e−1 | 0.2596 |
| 1024     | 0.34  | 3.99e−1 | 0.0512 |     |     |     |
| **2048** |       | **0.86** | **3.22e−1** |     |     |     |

| Table 13 | Numerical results for function $b_2$ of two variables in Example 5. |
|-----------|------------------|-----------|
| $n$       | Algorithm 3.5    | FTSG      |
|          | CT   | Err  | AO | CT   | Err  | AO |
| 64       | 0.02  | 3.09e−2 |     |     | 0.01 | 3.51e−1 |
| 128      | 0.06  | 2.01e−2 | 0.6204 |     | 0.01 | 2.51e−1 | 0.4838 |
| 256      | **0.17** | **1.31e−2** |     |     | 0.02 | 1.78e−1 | 0.4958 |
| 512      | 0.32  | 8.59e−3 | 0.6088 |     | 0.02 | 1.25e−1 | 0.5099 |
| 1024     | 1.51  | 5.63e−3 | 0.6095 |     | 0.02 | 8.72e−2 | 0.5195 |
| 2048     | 4.09  | 3.68e−3 | 0.6134 |     | 0.06 | 6.05e−2 | 0.5274 |
| 4096     | 0.11  | 4.18e−2 | 0.5334 |     |     |     |
| 8192     | 0.25  | 2.86e−2 | 0.5475 |     |     |     |
| **16384**|       | **0.53** | **1.97e−2** |     |     |     |

| Table 14 | Numerical results for function $b_3$ of three variables in Example 5. |
|-----------|------------------|-----------|
| $n$       | Algorithm 3.5    | FTSG      |
|          | CT   | Err  | AO | CT   | Err  | AO |
| 16       | **0.03** | **1.04e−1** |     |     | 0.01 | 9.04e−1 |
| 32       | 0.12  | 6.02e−2 | 0.7887 |     | 0.02 | 7.72e−1 | 0.2277 |
| 64       | 0.36  | 3.70e−2 | 0.7022 |     | 0.02 | 6.28e−1 | 0.2978 |
| 128      | 1.32  | 2.34e−2 | 0.6610 |     | 0.03 | 4.93e−1 | 0.3492 |
| 256      | 5.46  | 1.49e−2 | 0.6512 |     | 0.03 | 3.78e−1 | 0.3832 |
| 512      | 21.68 | 9.39e−3 | 0.6661 |     | 0.06 | 2.84e−1 | 0.4125 |
| 1024     | 79.42 | 5.72e−3 | 0.7151 |     | 0.16 | 2.11e−1 | 0.4286 |
| **2048** |       | **0.36** | **1.54e−1** |     |     |     |
Table 15
Numerical results for function \( b_4 \) of four variables in Example 5.

<table>
<thead>
<tr>
<th>( n )</th>
<th>Algorithm 3.5</th>
<th>FTSG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT</td>
<td>Err</td>
</tr>
<tr>
<td>16</td>
<td>0.06</td>
<td>1.26e−1</td>
</tr>
<tr>
<td>32</td>
<td>0.49</td>
<td>7.03e−2</td>
</tr>
<tr>
<td>64</td>
<td>3.55</td>
<td>4.21e−2</td>
</tr>
<tr>
<td>128</td>
<td>21.61</td>
<td>2.63e−2</td>
</tr>
<tr>
<td>256</td>
<td>115.42</td>
<td>1.67e−2</td>
</tr>
<tr>
<td>512</td>
<td>567.06</td>
<td>1.05e−2</td>
</tr>
<tr>
<td>1024</td>
<td>21.61</td>
<td>3.92e−1</td>
</tr>
<tr>
<td>2048</td>
<td>1.77</td>
<td>3.04e−1</td>
</tr>
</tbody>
</table>

Fig. 7. (a) Values \( C_{\text{error}} \) and (b) Values \( C_{\text{time}} \) of functions \( b_d \), \( d = 2, 3 \) and 4, in Example 5.

Examples 3–5 show that Algorithm 3.5 outperforms the exiting algorithm FTSG. The items in the tables of these examples highlighted by underlines indicate that Algorithm 3.5 using significantly less terms of the Fourier expansion and less computational time produces better approximation results in comparison to the FTSG for all dimensions \( d = 2, 3, 4 \). In passing, we point it out that within a given tolerance error the proposed algorithm uses significantly less terms of the Fourier expansion to represent the given function in comparison to the FTSG. This is highly desirable since it will further save computational time when the sparse Fourier expansion with less terms is used in an application context.

In the last example, we test the computational performance of Algorithm 3.5 for functions with \( d = 8 \).

Example 6. We consider function \( f_d \) defined in (4.1) with \( d = 8 \). It is easy to see that for each \( d = 8 \), \( f_d \in H_{\text{mix}}^{0.5-\varepsilon}(I^d) \) where \( \varepsilon > 0 \). Therefore, the theoretical approximation order is \( 0.5 - \varepsilon \). The numerical results are listed in Table 16.

5. Conclusive remarks

The fully discrete algorithm for computing the sparse Fourier approximation is introduced in this paper. The algorithm generates a sparse Fourier expansion for a function in \( H_{\text{mix}}^{0.5-\varepsilon}(I^d) \) having the optimal approximation order \( \Theta(n^{-s}) \) and requiring \( \Theta(n \log^{2d-1} n) \) number of multiplications to compute
Table 16
Numerical results for function $f_8$ of eight variables in Example 6.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Algorithm 3.5</th>
<th></th>
<th>FTSG</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CT</td>
<td>CR</td>
<td>Err</td>
<td>AO</td>
</tr>
<tr>
<td>16</td>
<td>22.70</td>
<td>2.61e−5%</td>
<td>2.58e−1</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>491.91</td>
<td>6.03e−7%</td>
<td>2.08e−1</td>
<td></td>
</tr>
<tr>
<td>64</td>
<td>6813.34</td>
<td>1.27e−8%</td>
<td>1.61e−1</td>
<td>0.3108</td>
</tr>
<tr>
<td>128</td>
<td></td>
<td>0.51</td>
<td>4.18e−1</td>
<td>0</td>
</tr>
<tr>
<td>256</td>
<td></td>
<td>1.45</td>
<td>4.12e−1</td>
<td>0.0209</td>
</tr>
<tr>
<td>512</td>
<td></td>
<td>3.59</td>
<td>4.05e−1</td>
<td>0.0247</td>
</tr>
</tbody>
</table>

all the needed Fourier coefficients. As demonstrated by numerical examples in Section 4, when the function to be approximated has a low order smoothness, the proposed algorithm outperforms significantly the existing algorithm. While with a similar approximation accuracy, the proposed algorithm uses less terms in the Fourier expansion but requires more computing time, when the function to be approximated has a higher order smoothness.

Finally, we remark on a theoretical comparison of the numerical performance of the proposed algorithm that has the approximation order $E_1 := cn^{-s}$ and computational complexity $C_1 := cn \log^{2d-1} n$ with that of an algorithm (compared algorithm) that has an approximation order $E_2 := cn^{-s+1}$ and computational complexity $C_2 := cn \log^d n$. Here, for simplicity, we assume that all the coefficients are the same. When $0 < s \leq 1$, the compared algorithm has no approximation order. In this case, the proposed algorithm is better. This has been confirmed by Examples 3–6 presented in the last section. Now, suppose that $s > 1$ and that we are required to compute the Fourier approximation up to the tolerance error $\epsilon > 0$. That is, $E_1 = E_2 = \epsilon$. Then, in terms of $\epsilon$, we have that

$$C_1 = \frac{c^{1+1/s}}{s} \epsilon^{-1/s} \log^{2d-1} (c/\epsilon)$$
and

$$C_2 = \frac{c^{1+1/(s-1)}}{s-1} \epsilon^{-1/s} \epsilon^{1/(s-1)} \log^d (c/\epsilon).$$

Clearly, if $d$ is fixed and $\epsilon$ is small enough, $C_2$ is larger than $C_1$.

Acknowledgments

The second author was supported in part by the US National Aeronautics and Space Administration under Cooperative Agreement NNX07AC37A, by the US National Science Foundation under grants CCR-0407476 and DMS-0712827, by the Natural Science Foundation of China under grants 10371122 and 10631080.

Appendix

In this appendix, we include several technical lemmas used in Section 3 for developing Algorithm 3.5. We first present properties of index set $W_j^d$. We use the notation $A \cup^\perp B$ for $A \cup B$ when $A \cap B = \emptyset$.

Lemma A.1. For each $j \in \mathbb{N}_0$, there holds

$$W_j = \bigcup_{q \in \mathcal{X}_j} W_{j,q}. \quad \text{(A.1)}$$
We prove this lemma by induction on $j$. From the definition of $\mathbb{W}_j$ and $\mathbb{W}_{j,q}$, $q \in X_j$, we can see that (A.1) holds when $j \in \{0, 1\}$. We assume that (A.1) holds when $j = j'$ with $j' > 1$. Note that $X_{j'} = \mathbb{W}_1$ when $j' \geq 1$. This means that when $j = j'$,

$$\mathbb{W}_{j'} = \bigcup_{q \in \mathbb{W}_1} \mathbb{W}_{j',q}. \quad (A.2)$$

We now consider the case when $j = j' + 1$. Let $r \in \mathbb{W}_{j'+1}$. From the definition of $\mathbb{W}_{j'+1}$, we know that

$$v_{j'+1,r} \in V_{j'+1} \setminus V_j. \quad (A.3)$$

According to Theorem 3.4 of [21], there holds

$$V_{j'+1} \setminus V_j = \Psi (V_j \setminus V_{j'-1}). \quad (A.4)$$

Combining (A.3) and (A.4), we obtain that there exists $\rho \in \mathbb{Z}_2$ such that $v_{j'+1,r} \in \phi_\rho (V_j \setminus V_{j'-1})$. This means that there exists $\tilde{r} \in \mathbb{W}_j$ such that $v_{j',r} \in V_j \setminus V_{j'-1}$ and

$$v_{j'+1,r} = \phi_\rho (v_{j',\tilde{r}}). \quad (A.5)$$

From the definition of $v_{j',\tilde{r}}$, we know that

$$v_{j',\tilde{r}} = \phi_{p_{j'}}(v_{r'}), \quad (A.6)$$

where $p_{j'} := [\rho_{j'} : \gamma \in \mathbb{Z}_{j'}] \in \mathbb{Z}_{2}^{\mathbb{Z}_{j'}}$, $r' \in \mathbb{Z}_m$ and

$$\tilde{r} = m\mu(p_{j'}) + r'. \quad (A.7)$$

Let $p_{j'+1} := [\rho_0, \ldots, \rho_{j'-1}, \rho]$. Substituting (A.6) into (A.5) leads to

$$v_{j'+1,r} = \phi_{p_{j'+1}}(v_{r'}). \quad (A.8)$$

Note that $\phi_{p_{j'+1}}(v_{r'}) = v_{j'+1,m\mu(p_{j'+1})+r'}$ and for all $j \in \mathbb{N}_0$ and $r_1, r_2 \in \mathbb{Z}_{2^j}$, $v_{j,r_1} = v_{j,r_2}$ if and only if $r_1 = r_2$. Thus, from (A.8), we obtain that

$$r = m\mu(p_{j'+1}) + r'. \quad (A.9)$$

According to the definition of $p_{j'+1}$, we know that

$$\mu(p_{j'+1}) = 2^{j'} \rho + \mu(p_{j'}). \quad (A.10)$$

Substituting (A.10) into (A.9), from (A.7) we obtain that

$$r = 2^{j'} m\rho + \tilde{r}. \quad (A.11)$$

Since (A.2) holds and $\tilde{r} \in \mathbb{W}_j$, there exists $q \in \mathbb{W}_1$ and $u \in \mathbb{Z}_{2^{j'-1}}$ such that

$$\tilde{r} = 2mu + q. \quad (A.12)$$

Combining (A.11) and (A.12) leads to equality

$$r = 2m(2^{j'-1} \rho + u) + q. \quad (A.13)$$

Noting that $2^{j'-1} \rho + u \in \mathbb{Z}_{2^{j'}}$, from (A.13) we have that

$$r \in \mathbb{W}_{j'+1,q}. \quad (A.14)$$
Thus, we know that
\[ W_{j+1} \subseteq \bigcup_{q \in \mathbb{W}_1} W_{j+1,q}. \]  
(A.15)

Noting that the cardinality of index sets \( V_{j+1} \) and \( V_j \) are \( 2^{j+1} m \) and \( 2^j m \), respectively, from the definition of \( W_{j+1} \) we know that
\[ C(W_{j+1}) \geq 2^j m. \]  
(A.16)

By the definition of \( W_{j+1,q} \), we find that
\[ C\left( \bigcup_{q \in \mathbb{W}_1} W_{j+1,q} \right) = 2^j m. \]  
(A.17)

Combining (A.15)–(A.17), we conclude that (A.1) holds when \( j = j' + 1 \). The induction principle ensures that equality (A.1) holds. □

We now extend the result of Lemma A.1 to the \( d \)-dimensional case.

**Lemma A.2.** For all \( j \in \mathbb{N}_0^d \), there holds
\[ W_j^d = \bigcup_{q \in \mathbb{X}_j^d} W_{1,q}. \]  
(A.18)

**Proof.** We prove this result by induction on \( d \). Lemma A.1 shows that (A.18) holds when \( d = 1 \). Assume that (A.18) holds when \( d = d' \). That is, for all \( j \in \mathbb{N}_0^{d'} \),
\[ W_j^{d'} = \bigcup_{q \in \mathbb{X}_j^{d'}} W_{1,q}. \]  
(A.19)

We now consider the case when \( d = d' + 1 \). Let \( j := [j_k : k \in \mathbb{Z}_{d' + 1}] \in \mathbb{N}_0^{d'+1} \) and \( j' := [j_k : k \in \mathbb{Z}_{d' + 1} \setminus \{0\}] \). From the definition of \( W_j^{d'+1} \), we know that
\[ W_{j+1}^{d'+1} = W_{j_0} \otimes W_{j'}^d. \]  
(A.20)

Substituting (A.1) and (A.19) into (A.20), we have that
\[ W_j^{d'+1} = \left( \bigcup_{q \in \mathbb{X}_j} W_{1,q} \right) \otimes \left( \bigcup_{q \in \mathbb{X}_j^{d'}} W_{1,q} \right). \]  
(A.21)

Note that in general for any finite sets \( A, B, A' \) and \( B' \),
\[ (A \cup \perp B) \otimes (A' \cup \perp B') = (A \otimes A') \cup \perp (B \otimes B'). \]

Thus, from (A.21) we obtain Eq. (A.18) for \( d = d' + 1 \). Thus, for all \( d \in \mathbb{N} \), Eq. (A.18) holds by the induction principle. □

We now turn to investigating \( A_{j,r}(I) \). We first show that for all \( j \in \mathbb{N} \) and \( r \in \mathbb{W}_j \), there exists an \( r' \in \mathbb{W}_1 \) such that \( \ell_{j,r} \) is generated by scaling and translating \( \ell_{1,r'} \). For all \( r \in \mathbb{W}_1 \), we assume that \( \ell_{1,r}(x) = 0 \) if \( x \not\in I \).
Lemma A.3. Let \( j \in \mathbb{N} \). For all \( r \in \mathbb{N}_{j} \) if \( r = 2mu + q \) where \( u \in \mathbb{Z}_{2^j} \) and \( q \in \mathbb{W}_{1} \), then
\[
\ell_{j,r}(x) = \ell_{1,q}(2^{j-1}x - 2\pi u), \quad x \in I.
\] (A.22)

**Proof.** We prove this lemma by induction on \( j \). The lemma clearly holds when \( j = 1 \). We assume that the lemma holds when \( j = j' \). That is, for all \( r \in \mathbb{N}_{j'} \), if \( r = 2mu + q \) where \( u \in \mathbb{Z}_{2^{j'}} \) and \( q \in \mathbb{W}_{1} \), then for all \( x \in I \),
\[
\ell_{j',r}(x) = \ell_{1,q}(2^{j'-1}x - 2\pi u).
\] (A.23)

We now consider the case when \( j = j' + 1 \). Let \( r \in \mathbb{N}_{j'+1} \) and
\[
\rho = m\mu(p_{j'+1}) + r_0.
\] (A.24)

where \( u \in \mathbb{Z}_{2^{j'}} \) and \( q \in \mathbb{W}_{1} \). Note that \( \mathbb{W}_{j'+1} \subset \mathbb{Z}_{2^{j'+1}} \). Thus, there exist \( p_{j'+1} := [\rho_{j'} : \gamma \in \mathbb{Z}_{j'+1}] \in \mathbb{Z}_{2^{j'+1}} \) and \( r_0 \in \mathbb{Z}_{m} \) such that
\[
r = m\mu(p_{j'+1}) + r_0.
\] (A.25)

Let \( p_{j'} := [\rho_{j'} : \gamma \in \mathbb{Z}_{j'}] \) and \( r_1 := m\mu(p_{j'}) + r_0 \). From the definition of \( \ell_{j'+1,r} \) and \( \ell_{j',r_1} \), we know that
\[
\ell_{j'+1,r} = \tau_{\rho_{j'}} \ell_{j',r_1}.
\] (A.26)

Set \( p := [\rho_{j'} : \gamma \in \mathbb{Z}_{j'+1} \setminus \{0,j'\}] \). From (A.24) and (A.25), we see that
\[
q = m\rho_0 + r_0
\] (A.27)

and
\[
u = \mu(p) + 2^{j'-1}\rho_{j'}.
\] (A.28)

From (A.27) and the definition of \( p \), we have that
\[
r_1 = 2m\mu(p) + q.
\] (A.29)

Since \( \mu(p) \in \mathbb{Z}_{2^{j'}} \) and \( q \in \mathbb{W}_{1} \), according to **Lemma A.1** and (A.29), we know that \( r_1 \in \mathbb{W}_{j'} \). Thus, from (A.23) and (A.29) we obtain for all \( x \in I \) that
\[
\ell_{j',r_1}(x) = \ell_{1,q}(2^{j'-1}x - 2\pi \mu(p)).
\] (A.30)

We now consider the case when \( \rho_{j'} = 0 \) and the case when \( \rho_{j'} = 1 \), separately. In the case when \( \rho_{j'} = 0 \), by combining (A.26) and (A.30), we obtain for all \( x \in I \) that
\[
\ell_{j'+1,r}(x) = \ell_{1,q}(2^jx - 2\pi \mu(p)).
\] (A.31)

Noting that \( u = \mu(p) \) when \( \rho_{j'} = 0 \), from (A.31) we have for all \( x \in I \) that
\[
\ell_{j'+1,r}(x) = \ell_{1,q}(2^jx - 2\pi u).
\] (A.32)

In the case when \( \rho_{j'} = 1 \), replacing \( x \) in (A.30) by \( 2x - \pi \) and substituting it into (A.26), we obtain for all \( x \in I \) that
\[
\ell_{j'+1,r}(x) = \ell_{1,q}(2^jx - 2\pi (\mu(p) + 2^{j'-1})).
\] (A.33)

Eq. (A.28) ensures that \( u = \mu(p) + 2^{j'-1} \) when \( \rho_{j'} = 1 \). Thus, from (A.33) we have for all \( x \in I \) that
\[
\ell_{j'+1,r}(x) = \ell_{1,q}(2^jx - 2\pi u).\]

By combining this equation with (A.32), we have that (A.22) holds when \( j = j' + 1 \). By the induction principle, we conclude that this lemma holds for all \( j \in \mathbb{N} \). \( \square \)

**Lemma A.3** is applied to re-express \( A_{j,r}(I) \) in the next lemma.
Lemma A.4. For all $r \in \mathbb{V}^d$, there holds

$$A_{j, r} (l) = t_{j, q} (l) e^{-i 2 \pi l^T d j u}, \quad l \in \mathbb{L}^d_n, \quad j \in \mathbb{S}^d_N,$$

(A.34)

where $u \in \mathbb{Z}^d_{2j-1}$ and $q \in \mathbb{X}^d$ such that $r = 2m u + q$.

Proof. Let $l := [l_k : k \in \mathbb{Z}_d] \in \mathbb{L}^d_n, \quad j := [j_k : k \in \mathbb{Z}_d] \in \mathbb{S}^d$, and $r := [r_k : k \in \mathbb{Z}_d] \in \mathbb{V}^d$. From the definition of $A_{j, r} (l)$, we have that

$$A_{j, r} (l) = \prod_{k \in \mathbb{Z}_d} A_{j, r_k} (l_k) \quad (A.35)$$

Write $r = 2m u + q$, with $u := [u_k : k \in \mathbb{Z}_d] \in \mathbb{Z}^d_{2j-1}$ and $q := [q_k : k \in \mathbb{Z}_d] \in \mathbb{X}^d$. Thus, for all $k \in \mathbb{Z}_d$, there holds $r_k = 2m u_k + q_k$. Fix $k \in \mathbb{Z}_d$. If $j_k = 0$, then $u_k = 0$ and $r_k = q_k$. It follows that when $j_k = 0$,

$$A_{j, r_k} (l_k) = e^{-i 2 \pi l_k u_k / 2^{k-1}} t_{j_k, q_k} (l_k).$$

(A.36)

Suppose that $j_k \geq 1$. From Lemma A.3, we know for all $x \in l$ that

$$\ell_{j, r_k} (x) = \ell_{1, q_k} (2^{k-1} x - 2 \pi u_k).$$

Thus, from the definition of $A_{j, r_k} (l_k)$, we obtain that

$$A_{j, r_k} (l_k) = \frac{1}{\sqrt{2 \pi}} \int_l \ell_{1, q_k} (2^{k-1} x - 2 \pi u_k) e^{-i l_k x} dx.$$  (A.37)

We change the variable by replacing $2^{k-1} x - 2 \pi u_k$ with $x$ on the right hand side of (A.37) and obtain that

$$A_{j, r_k} (l_k) = \frac{1}{\sqrt{2 \pi}} \frac{e^{-i 2 \pi l_k u_k / 2^{k-1}}}{2^{k-1}} \int_{2^{k-1} x - 2 \pi u_k} \ell_{1, q_k} (x) e^{-i l_k x / 2^{k-1}} dx.$$  (A.38)

Noting that $\ell_{1, q_k} (x) = 0$ if $x \not\in l$, from the definition of $t_{j_k, q_k}$ and (A.38) we have that

$$A_{j, r_k} (l_k) = e^{-i 2 \pi l_k u_k / 2^{k-1}} t_{j_k, q_k} (l_k).$$  (A.39)

Substituting (A.36) and (A.39) into (A.35), from the definition of $t_{j, q} (l)$ we obtain (A.34). □

References


