Fast evaluation of trigonometric polynomials from hyperbolic crosses

Markus Fenn* Stefan Kunis† Daniel Potts‡

Abstract

The discrete Fourier transform in d dimensions with equispaced knots in space and frequency domain can be computed by the fast Fourier transform (FFT) in $\mathcal{O}(N^d \log N)$ arithmetic operations. In order to circumvent the 'curse of dimensionality' in multivariate approximation, interpolations on sparse grids were introduced. In particular, for frequencies chosen from an hyperbolic cross and spatial knots on a sparse grid fast Fourier transforms that need only $\mathcal{O}(N \log^d N)$ arithmetic operations were developed. Recently, the FFT was generalised to nonequispaced spatial knots by the so called NFFT.

In this paper, we propose an algorithm for the fast Fourier transform on hyperbolic cross points for nonequispaced spatial knots in two and three dimensions. We call this algorithm sparse NFFT (SNFFT). Our new algorithm is based on the NFFT and an appropriate partitioning of the hyperbolic cross. Numerical examples confirm our theoretical results.

2000 *Mathematics Subject Classification.* 65F10, 65F15, 65T40.
Key words and phrases. trigonometric approximation, hyperbolic cross, sparse grids, fast Fourier transform for nonequispaced knots, NFFT, FFT

1 Introduction

In multivariate approximation one has to deal with the so called 'curse of dimensionality', i.e., the number of degrees of freedom for representing an approximation of a function with a prescribed accuracy depends exponentially on the dimensionality of the considered problem. This obstacle can be circumvented to some extend by the interpolation on sparse grids and the related approximation on hyperbolic cross points in the Fourier domain, see, e.g., [17, 14, 3]. Let

$$g(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathbb{Z}^d} \hat{g}_{\boldsymbol{k}} \, \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}} \,,$$

be a function of dominating mixed smoothness or from a Korobov space. Instead of approximating g on the standard tensor product grid $\left\{\boldsymbol{k} = (k_1, \ldots, k_d)^\top \in \mathbb{Z}^d : \max\{|k_1|, \ldots, |k_d|\} < N\right\}$ with $\mathcal{O}(N^d)$ degrees of freedom, it can be approximated with only $\mathcal{O}(N \log^{d-1} N)$ degrees of freedom from the hyperbolic cross $\left\{\boldsymbol{k} = (k_1, \ldots, k_d)^\top \in \mathbb{Z}^d : (1 + |k_1|) \cdots (1 + |k_d|) < N\right\}$. The approximation error deteriorates only by a factor of $\log^{d-1} N$, cf. [14].

*Markus.Fenn@uni-mannheim.de, University of Mannheim, Institute of Mathematics, D–68159 Mannheim
†kunis@math.uni-luebeck.de, University of Lübeck, Institute of Mathematics, D–23560 Lübeck
‡potts@mathematik.tu-chemnitz.de, Chemnitz University of Technology, Department of Mathematics, D–09107 Chemnitz

The fast evaluation of trigonometric polynomials with equispaced knots in space and frequency domain can be computed by the fast Fourier transform (FFT) in only $\mathcal{O}(N^d \log N)$ arithmetic operations [4]. If the frequencies are chosen from a hyperbolic cross and the spatial knots lie on a sparse grid there exist fast algorithms of order $\mathcal{O}(N \log^d N)$, see [1, 11].
In [6, 2, 13], the FFT has been generalised by the fast Fourier transform at nonequispaced knots (NFFT) which requires $\mathcal{O}(N^d \log N + M)$ arithmetic operations for the evaluation of a trigonometric polynomial at M arbitrary knots.
In this paper, we present an algorithm for the fast evaluation of trigonometric polynomials from hyperbolic crosses, where in contrast to [1, 11], the spatial knots can be chosen arbitrarily. We will call this algorithm sparse NFFT (SNFFT).

The outline of this paper is as follows. In Section 2, we show how the NFFT can be coupled with hyperbolic crosses. The main idea consists in an appropriate partitioning of the index set and the application of the NFFT to the resulting blocks. In Section 3, we define the two dimensional hyperbolic cross and its block partition for the SNFFT. Fast algorithms for the sparse discrete cosine and sine transforms at nonequispaced spatial knots in two dimensions are given in Section 4. In Section 5, we introduce a modified three dimensional hyperbolic cross which easily can be partitioned into blocks again. Finally, Section 6 presents numerical examples and a discussion of the results.

2 Coupling NFFT with hyperbolic crosses

Our new fast algorithm for the evaluation of trigonometric polynomials from hyperbolic crosses at arbitrary knots is based on the application of the NFFT to an appropriate partitioning of the hyperbolic cross. First, we sketch the NFFT, then we explain how to couple it with block partitionings.
Let $\boldsymbol{N} = (N_1, \ldots, N_d)^\top \in 2\mathbb{N}^d$, $I_{\boldsymbol{N}}^d := \big\{-\frac{N_1}{2}, \ldots, \frac{N_1}{2} - 1\big\} \times \cdots \times \big\{-\frac{N_d}{2}, \ldots, \frac{N_d}{2} - 1\big\}$, and $\mathbb{T}^d := \big[-\frac{1}{2}, \frac{1}{2}\big)^d$. The d-variate NFFT$(N_1, \ldots, N_d)$ computes approximations $\tilde{f}$ of the trigonometric polynomial

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in I_{\boldsymbol{N}}^d} \hat{f}_{\boldsymbol{k}} \, \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}} \tag{2.1}$$

at arbitrary knots $\boldsymbol{x}_\ell \in \mathbb{T}^d$, $\ell = 0, \ldots, M-1$. The main idea for the fast computation of (2.1) consists in the approximation of f by linear combinations of translates of a window function. Original approaches used Gaussian bells [6, 5] or B-splines [2] as window functions. A general approach to NFFTs can be found in [15, 13]. An early review of several algorithms for the NFFT is given in [16]. A free NFFT software package is available on our web page [12] (see also [9]).
While the straightforward evaluation of (2.1) requires $\mathcal{O}(|I_{\boldsymbol{N}}^d| M)$ arithmetic operations, the fast computation by NFFT needs

$$\mathcal{O}(\alpha^d |I_{\boldsymbol{N}}^d| \log |I_{\boldsymbol{N}}^d| + m^d M)$$

arithmetic operations. Both parameters, the *oversampling factor* α and the *cut-off parameter* m have to be chosen in accordance with the desired accuracy of the fast algorithm. In general the approximation error introduced by the NFFT decays exponentially in m, where the basis of the exponent depends on α. In our numerical experiments, we will focus on the Gaussian window function. Then, by [7], the error can be estimated by

$$\frac{|f(\boldsymbol{x}_\ell) - \tilde{f}(\boldsymbol{x}_\ell)|}{\sum_{\boldsymbol{k}\in I_{\boldsymbol{N}}^d} |\hat{f}_{\boldsymbol{k}}|} \le d\, 2^{d+1}\, \mathrm{e}^{-m\pi(1-1/(2\alpha-1))} \,. \tag{2.2}$$

Next, we want to apply the NFFT in connection with hyperbolic crosses. Our aim is the fast approximate evaluation of the trigonometric polynomial

$$f(\boldsymbol{x}) = \sum_{\boldsymbol{k}\in H_J^d} \hat{f}_{\boldsymbol{k}}\, \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}} \tag{2.3}$$

at arbitrary knots $\boldsymbol{x}_\ell \in \mathbb{T}^d$, $\ell = 0, \ldots, M-1$ and for a hyperbolic cross H_J^d which can be partitioned into blocks of indices $H_J^d = \bigcup_r (I_{\boldsymbol{N}_r}^d + \boldsymbol{\rho}_r)$ with frequency shifts $\boldsymbol{\rho}_r \in \mathbb{Z}^d$. Then, the sum in (2.3) can be split up according to the blocks as

$$f(\boldsymbol{x}_\ell) = \sum_r \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{\rho}_r\boldsymbol{x}_\ell} \sum_{\boldsymbol{k}\in I_{\boldsymbol{N}_r}^d} \hat{f}_{\boldsymbol{k}+\boldsymbol{\rho}_r}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell} \tag{2.4}$$

with the 'nonuniform twiddle factors' $\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{\rho}_r\boldsymbol{x}_\ell}$. Now we apply the NFFT of size $|I_{\boldsymbol{N}_r}^d|$ on every block. Due to the triangle inequality, the overall error remains bounded by (2.2). The number of arithmetic operations on every block is $\mathcal{O}(\alpha^d |I_{\boldsymbol{N}_r}^d| \log |I_{\boldsymbol{N}_r}^d| + m^d M)$. So our main task consists in the construction of an adequate partition of the hyperbolic cross with only few blocks which leads to a fast overall algorithm.

Remark 2.1 We would like to emphasise a technical detail concerning NFFTs of short size. Obviously, NFFTs with small $N_1, \ldots, N_d$, should be computed directly. The case where only few N_i are small needs more care. We exemplify our solution for $d = 2$ and $N_1 \le m < N_2$. Splitting up the sum in equation (2.1) into both dimensions yields

$$f(\boldsymbol{x}_\ell) = f\big((\boldsymbol{x}_\ell)_1, (\boldsymbol{x}_\ell)_2\big) = \sum_{k_1\in I_{N_1}^1} \Big(\sum_{k_2\in I_{N_2}^1} \hat{f}_{k_1,k_2}\, \mathrm{e}^{-2\pi\mathrm{i}k_2(\boldsymbol{x}_\ell)_2} \Big)\, \mathrm{e}^{-2\pi\mathrm{i}k_1(\boldsymbol{x}_\ell)_1} \,. \tag{2.5}$$

Now the computation can be done by an one dimensional NFFT for the inner bracket, followed by a direct computation of the outer sum in a total of $\mathcal{O}\big(N_1(\alpha N_2 \log N_2 + mM)\big)$ arithmetic operations. □

3 NFFT on hyperbolic cross points – the bivariate case

Let $J \in \mathbb{N}_0$ and $N = 2^{J+2}$. We define the following index sets as building blocks for our partitioning of the hyperbolic crosses in two and three dimensions. For $r \in \mathbb{N}_0$, let

$$\begin{array}{lcll} H_r^- & := & \{ & -2^{r+1}, \ldots, -2^r - 1 \quad \}, \\ H_r^0 & := & \{ & -\lfloor 2^{r-1} \rfloor, \ldots, \lceil 2^{r-1} \rceil - 1 \quad \}, \\ H_r^+ & := & \{ & 2^r, \ldots, 2^{r+1} - 1 \quad \}. \end{array}$$

Obviously, the sets H_r^- and H_r^+ are just shifted versions of H_r^0, i.e.,

$$H_r^- = -\lceil \tfrac{3}{2}\, 2^r \rceil + H_r^0 \qquad \text{and} \qquad H_r^+ = \lfloor \tfrac{3}{2}\, 2^r \rfloor + H_r^0 \,. \tag{3.1}$$

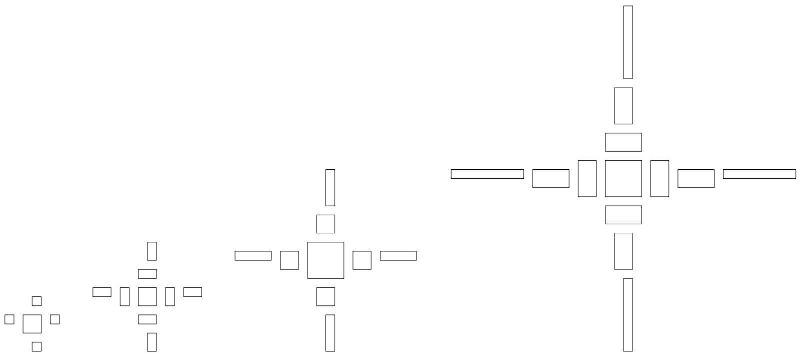

Figure 3.1: Hyperbolic cross points in 2D for $J = 0, \ldots, 3$.

Furthermore, we have that

$$\left|H_r^-\right| = \left|H_r^0\right| = \left|H_r^+\right| = 2^r\,, \qquad H_{r_1}^0 \times \ldots \times H_{r_d}^0 = I^d_{(N_1,\ldots,N_d)^\top} \tag{3.2}$$

for $N_1 = 2^{r_1}, \ldots, N_d = 2^{r_d}$ and $r_1, \ldots, r_d \geq 1$.
Let us now define the blocks of the hyperbolic cross in 2D. For the levels $r = 0, \ldots, \lceil \frac{J}{2} \rceil$, let

$$\begin{array}{lcl}
H_{J,r}^{\text{right}} & := & H_r^0 \times H_{J-r}^+,\\
H_{J,r}^{\text{top}} & := & H_{J-r}^+ \times H_r^0,\\
H_{J,r}^{\text{left}} & := & H_r^0 \times H_{J-r}^-,\\
H_{J,r}^{\text{bottom}} & := & H_{J-r}^- \times H_r^0,\\
H_{J,r} & := & H_{J,r}^{\text{right}} \cup H_{J,r}^{\text{top}} \cup H_{J,r}^{\text{left}} \cup H_{J,r}^{\text{bottom}},\\
H_J^{\text{centre}} & := & H^0_{\lfloor \frac{J}{2} \rfloor + 1} \times H^0_{\lfloor \frac{J}{2} \rfloor + 1}\,.
\end{array}$$

Now the *two dimensional hyperbolic cross* is the index set $H_J^2 \subset I^2_{(N,N)^\top}$ given by

$$H_J^2 := H_J^{\text{centre}} \cup \bigcup_{r=0}^{\lceil \frac{J}{2} \rceil} H_{J,r}\,,$$

cf. Figure 3.1. Since the cardinalities for these index sets are

$$\left|H_{J,r}^{\text{right}}\right| = \left|H_{J,r}^{\text{top}}\right| = \left|H_{J,r}^{\text{left}}\right| = \left|H_{J,r}^{\text{bottom}}\right| = 2^J\,, \qquad \left|H_{J,r}^{\text{centre}}\right| = 2^{2(\lfloor \frac{J}{2} \rfloor + 1)}\,,$$

the total number of hyperbolic cross points is $|H_J^2| = (J+4)2^{J+1}$, compared to $|I^2_{(N,N)^\top}| = N^2 = 4^{J+2}$ indices for the full tensor product grid.
Following equation (2.4), we are interested in the computation of

$$f(\boldsymbol{x}_\ell) = \sum_{\boldsymbol{k} \in H_J^{\text{centre}}} \hat{f}_{\boldsymbol{k}}\ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} + \sum_{r=0}^{\lceil \frac{J}{2} \rceil} \sum_{\boldsymbol{k} \in H_{J,r}} \hat{f}_{\boldsymbol{k}}\ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \tag{3.3}$$

at arbitrary knots $\boldsymbol{x}_\ell \in \mathbb{T}^2$, $\ell = 0, \ldots, M-1$. We start by computing the centre block, i.e., the first sum in (3.3) by a bivariate NFFT$(2^{\lfloor \frac{J}{2} \rfloor + 1}, 2^{\lfloor \frac{J}{2} \rfloor + 1})$ with arithmetic complexity

$\mathcal{O}(J2^J + M)$. Next we consider the following sums

$$f_{J,r}^{\text{label}}(\boldsymbol{x}_\ell) := \sum_{\boldsymbol{k}\in H_{J,r}^{\text{label}}} \hat{f}_{\boldsymbol{k}}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell}$$

where label $\in$ {right, top, left, bottom}. We explain the computation of a left block. By using equation (3.1), we obtain

$$\begin{aligned}\sum_{\boldsymbol{k}\in H_{J,r}^{\text{left}}} \hat{f}_{\boldsymbol{k}}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell} &= \sum_{k_1\in H_r^0}\sum_{k_2\in H_{J-r}^-} \hat{f}_{k_1,k_2}\, \mathrm{e}^{-2\pi\mathrm{i}(k_1(\boldsymbol{x}_\ell)_1+k_2(\boldsymbol{x}_\ell)_2)} \\ &= \mathrm{e}^{2\pi\mathrm{i}\lceil\frac{3}{2}2^{J-r}\rceil(\boldsymbol{x}_\ell)_2} \sum_{\boldsymbol{k}\in H_r^0\times H_{J-r}^0} \hat{f}_{k_1,k_2-\lceil\frac{3}{2}2^{J-r}\rceil}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell}\,.\end{aligned}$$

Due to (3.2) each of these blocks can be computed by a bivariate NFFT$(2^r, 2^{J-r})$ with arithmetic complexity $\mathcal{O}(J2^J + M)$, see also Remark 2.1.
Since the number of blocks is $\mathcal{O}(J)$, the overall complexity for computing $f(\boldsymbol{x}_\ell)$ for $\ell = 0,\ldots,M-1$ is $\mathcal{O}(J^2 2^J + JM)$. We refer to the following algorithm on hyperbolic cross points as sparse NFFT (SNFFT).

Algorithm 3.1 (SNFFT 2D)

Input: $J \in \mathbb{N}_0$, $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$ *for* $\boldsymbol{k} \in H_J^2$,
$M \in \mathbb{N}$, $\boldsymbol{x}_\ell \in \mathbb{T}^2$ *for* $\ell = 0,\ldots,M-1$.

1. *Compute the values*
$$\tilde{f}(\boldsymbol{x}_\ell) = \sum_{\boldsymbol{k}\in H_J^{\text{centre}}} \hat{f}_{k_1,k_2}\,\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell}\,.$$
by a bivariate NFFT$(2^{\lfloor\frac{J}{2}\rfloor+1}, 2^{\lfloor\frac{J}{2}\rfloor+1})$.

2. *For* $r = 0,\ldots,\lceil\frac{J}{2}\rceil$ *compute*
$$\begin{aligned}\tilde{f}(\boldsymbol{x}_\ell) &= \tilde{f}(\boldsymbol{x}_\ell) \\ &+ \mathrm{e}^{-2\pi\mathrm{i}\lfloor\frac{3}{2}2^{J-r}\rfloor(\boldsymbol{x}_\ell)_2} \sum_{\boldsymbol{k}\in H_r^0\times H_{J-r}^0} \hat{f}_{k_1,k_2+\lfloor\frac{3}{2}2^{J-r}\rfloor}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell} \\ &+ \mathrm{e}^{-2\pi\mathrm{i}\lfloor\frac{3}{2}2^{J-r}\rfloor(\boldsymbol{x}_\ell)_1} \sum_{\boldsymbol{k}\in H_{J-r}^0\times H_r^0} \hat{f}_{k_1+\lfloor\frac{3}{2}2^{J-r}\rfloor,k_2}\,\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell} \\ &+ \mathrm{e}^{2\pi\mathrm{i}\lceil\frac{3}{2}2^{J-r}\rceil(\boldsymbol{x}_\ell)_2} \sum_{\boldsymbol{k}\in H_r^0\times H_{J-r}^0} \hat{f}_{k_1,k_2-\lceil\frac{3}{2}2^{J-r}\rceil}\ \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell} \\ &+ \mathrm{e}^{2\pi\mathrm{i}\lceil\frac{3}{2}2^{J-r}\rceil(\boldsymbol{x}_\ell)_1} \sum_{\boldsymbol{k}\in H_{J-r}^0\times H_r^0} \hat{f}_{k_1-\lceil\frac{3}{2}2^{J-r}\rceil,k_2}\,\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell}\,.\end{aligned}$$
by four bivariate NFFT$(2^r, 2^{J-r})$.

Output: $\tilde{f}(\boldsymbol{x}_\ell)$ *approximate value of* $f(\boldsymbol{x}_\ell)$, $\ell = 0,\ldots,M-1$.

Algorithm 3.1 reads in matrix-vector notation as

$$\boldsymbol{A}_J\hat{\boldsymbol{f}} = \left[\boldsymbol{A}_{J,0} \mid \boldsymbol{A}_{J-1,1} \mid \ldots \mid \boldsymbol{A}_{\lfloor\frac{J}{2}\rfloor,\lceil\frac{J}{2}\rceil} \mid \boldsymbol{A}_J^{\text{centre}}\right] \left[\begin{array}{c} \hat{\boldsymbol{f}}_{J,0} \\ \hline \vdots \\ \hline \hat{\boldsymbol{f}}_J^{\text{centre}} \end{array}\right]$$

where the sub-matrices are given by

$$\boldsymbol{A}_{J,r} := \left[\boldsymbol{A}_{J,r}^{\text{right}} \mid \boldsymbol{A}_{J,r}^{\text{top}} \mid \boldsymbol{A}_{J,r}^{\text{left}} \mid \boldsymbol{A}_{J,r}^{\text{bottom}}\right], \qquad \boldsymbol{A}_{J,r}^{\text{label}} := \left(\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\boldsymbol{x}_\ell}\right)_{\ell=0,\ldots,M-1;\boldsymbol{k}\in H_{J,r}^{\text{label}}}$$

for label $\in \{\text{right}, \text{top}, \text{left}, \text{bottom}\}$.

4 NDCT and NDST on hyperbolic cross points in 2D

Similar algorithms can be constructed for the discrete cosine transform and sine transform, based on the fast discrete cosine and sine transforms for nonequispaced knots, developed in [8]. The bivariate fast cosine transform at nonequispaced knots NFCT(N_1, N_2) computes approximations of

$$f(\boldsymbol{x}_\ell) = \sum_{k_1=0}^{N_1-1}\sum_{k_2=0}^{N_2-1} \hat{f}_{k_1,k_2} \cos\left(2\pi k_1(\boldsymbol{x}_\ell)_1\right)\cos\left(2\pi k_2(\boldsymbol{x}_\ell)_2\right)$$

and the bivariate fast sine transform at nonequispaced knots NFST(N_1, N_2) computes approximations of

$$f(\boldsymbol{x}_\ell) = \sum_{k_1=1}^{N_1-1}\sum_{k_2=1}^{N_2-1} \hat{f}_{k_1,k_2} \sin\left(2\pi k_1(\boldsymbol{x}_\ell)_1\right)\sin\left(2\pi k_2(\boldsymbol{x}_\ell)_2\right)$$

at arbitrary knots $\boldsymbol{x}_\ell \in [0, \frac{1}{2}]^2$, $\ell = 0, \ldots, M-1$.
A coupling with hyperbolic crosses can be done as follows. Again let $N = 2^{J+2}$. Here we use the index sets depicted in Figure 4.1. For $r = 0, \ldots, J+2$ define

$$H'_{J,r} := \{0, \ldots, 2^{J+2-r}-1\} \times \{\lfloor 2^{r-1}\rfloor, \ldots, 2^r - 1\}, \qquad H'_J := \bigcup_{r=0}^{J+2} H'_{J,r}$$

The cardinalities for these index sets are

$$\left|H'_{J,0}\right| = 2^{J+2}, \quad \left|H'_{J,r}\right| = 2^{J+1}, \quad \left|H'_J\right| = (J+4)\,2^{J+1}.$$

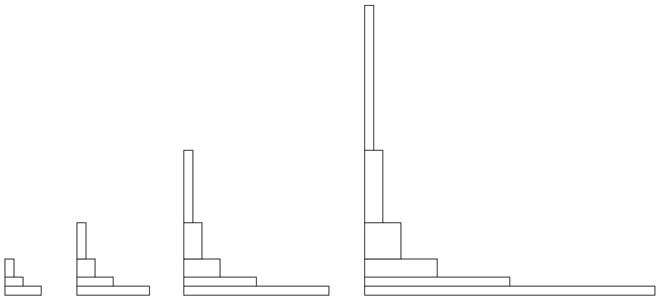

Figure 4.1: Hyperbolic cross points for the NFCT in 2D for $J = 0, \ldots, 3$.

Then the sparse NFCT can be computed by

$$\begin{aligned}
f(\boldsymbol{x}_\ell) &= \sum_{(k_1,k_2)^\top \in H'_J} \hat{f}_{k_1,k_2} \cos(2\pi k_1 (\boldsymbol{x}_\ell)_1) \cos(2\pi k_2 (\boldsymbol{x}_\ell)_2) \\
&= \sum_{r=0}^{J+2} \sum_{(k_1,k_2)^\top \in H'_{J,r}} \hat{f}_{k_1,k_2} \cos(2\pi k_1 (\boldsymbol{x}_\ell)_1) \cos(2\pi k_2 (\boldsymbol{x}_\ell)_2) \\
&= \sum_{r=0}^{J+2} \cos(\lfloor 2^{r-1} \rfloor 2\pi (\boldsymbol{x}_\ell)_2) \sum_{k_1=0}^{2^{J+2-r}-1} \sum_{k_2=0}^{2^{r-1}-1} \hat{f}_{k_1,k_2+2^{r-1}} \cos(2\pi k_1 (\boldsymbol{x}_\ell)_1) \cos(2\pi k_2 (\boldsymbol{x}_\ell)_2) \\
&+ \sum_{r=1}^{J+2} \sin(\lfloor 2^{r-1} \rfloor 2\pi (\boldsymbol{x}_\ell)_2) \sum_{k_1=0}^{2^{J+2-r}-1} \sum_{k_2=0}^{2^{r-1}-1} \hat{f}_{k_1,k_2+2^{r-1}} \cos(2\pi k_1 (\boldsymbol{x}_\ell)_1) \sin(2\pi k_2 (\boldsymbol{x}_\ell)_2) \,.
\end{aligned}$$

Using the fast algorithms from [8] we obtain an overall arithmetic complexity of $\mathcal{O}(J^2 2^J + JM)$.

5 NFFT on hyperbolic cross points – the trivariate case

Let us consider the hyperbolic cross in three dimensions. For $r = 1, \ldots, J+1$, we define the index sets

$$\begin{aligned}
H_{J,0}^{\text{front}} &:= H_J^2 \times \{0\}\,, & H_{J,r}^{\text{front}} &:= H_{J-r-1}^2 \times H_{r-1}^+\,, \\
H_{J,0}^{\text{rear}} &:= H_{J-1}^2 \times \{-1\}\,, & H_{J,r}^{\text{rear}} &:= H_{J-r-1}^2 \times H_{r-1}^-\,.
\end{aligned}$$

The *three dimensional hyperbolic cross* is given by

$$H_J^3 := \bigcup_{r=0}^{J+1} H_{J,r}^{\text{front}} \cup \bigcup_{r=0}^{J+1} H_{J,r}^{\text{rear}}\,,$$

cf. Figure 5.1. The total number of hyperbolic cross points is

$$\left|H_J^3\right| = \left|H_J^2\right| + \left|H_{J-1}^2\right| + \sum_{r=1}^{J+1} \left|H_{J-r-1}^2\right| \left(\left|H_{r-1}^+\right| + \left|H_{r-1}^-\right| \right) = 2^{J-1}\left(J^2 + 11J + 26\right).$$

The arithmetic complexity of the resulting algorithm is $\mathcal{O}(J^3 2^J + J^2 M)$, since for $r = 1, \ldots, J+1$ we have to compute $\mathcal{O}(J-r)$ trivariate NFFTs with complexity $\mathcal{O}(J 2^J + M)$

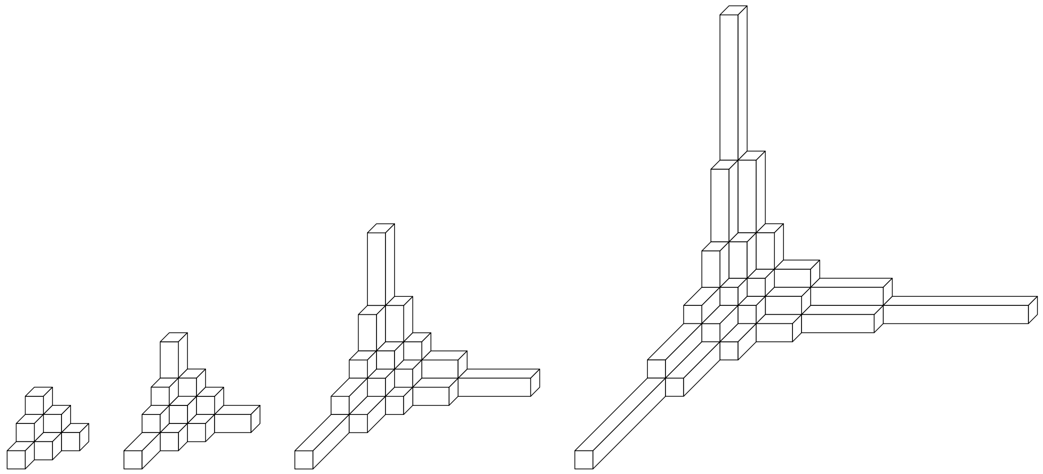

Figure 5.1: Hyperbolic cross points in 3D for $J = 0, \ldots, 3$, only the part $k_1,\ k_2,\ k_3 \le 0$ is shown.

each. Unfortunately, this algorithm has drawbacks: we have to compute NFFTs for $\mathcal{O}(J^2)$ blocks of our partition. Thus, the (asymptotic) arithmetic complexity for an equal number of knots and Fourier coefficients $M = |H_J^3|$ is $\mathcal{O}(J^4 2^J)$, i.e., not optimal. Furthermore, the second part of the NFFTs for the blocks is the most time consuming part for interesting problem sizes J.
Therefore, we use a simplification $\tilde{H}_J^3$ of the hyperbolic cross with $H_J^3 \subset \tilde{H}_J^3 \subset I^3_{(N,N,N)^\top}$, which can easily be partitioned into only $\mathcal{O}(J)$ blocks but has a total number of $\mathcal{O}(2^{\frac{3}{2}J})$ points. For $r = 0, \ldots, \lceil \frac{J}{2} \rceil$, we define the following index sets

$$
\begin{array}{lcllllll}
\tilde{H}_{J,r}^{\mathrm{top}} & := & H_{J-r}^{+} & \times & H_r^0 & \times & H_r^0\,, \\
\tilde{H}_{J,r}^{\mathrm{left}} & := & H_r^0 & \times & H_{J-r}^{+} & \times & H_r^0\,, \\
\tilde{H}_{J,r}^{\mathrm{front}} & := & H_r^0 & \times & H_r^0 & \times & H_{J-r}^{+}\,, \\
\tilde{H}_{J,r}^{\mathrm{bottom}} & := & H_{J-r}^{-} & \times & H_r^0 & \times & H_r^0\,, \\
\tilde{H}_{J,r}^{\mathrm{right}} & := & H_r^0 & \times & H_{J-r}^{-} & \times & H_r^0\,, \\
\tilde{H}_{J,r}^{\mathrm{rear}} & := & H_r^0 & \times & H_r^0 & \times & H_{J-r}^{-}\,, \\
\tilde{H}_{J,r} & := & \multicolumn{6}{l}{\tilde{H}_{J,r}^{\mathrm{left}} \cup \tilde{H}_{J,r}^{\mathrm{right}} \cup \tilde{H}_{J,r}^{\mathrm{top}} \cup \tilde{H}_{J,r}^{\mathrm{bottom}} \cup \tilde{H}_{J,r}^{\mathrm{front}} \cup \tilde{H}_{J,r}^{\mathrm{rear}}\,.}
\end{array}
$$

The centre block is given by

$$\tilde{H}_J^{\mathrm{centre}} := H^0_{\lfloor \frac{J}{2} \rfloor + 1} \times H^0_{\lfloor \frac{J}{2} \rfloor + 1} \times H^0_{\lfloor \frac{J}{2} \rfloor + 1}\,.$$

The cardinalities for these index sets are

$$\left|\tilde{H}_{J,r}^{\mathrm{left}}\right| = \left|\tilde{H}_{J,r}^{\mathrm{right}}\right| = \left|\tilde{H}_{J,r}^{\mathrm{top}}\right| = \left|\tilde{H}_{J,r}^{\mathrm{bottom}}\right| = \left|\tilde{H}_{J,r}^{\mathrm{front}}\right| = \left|\tilde{H}_{J,r}^{\mathrm{rear}}\right| = 2^{J+r}\,, \qquad \left|\tilde{H}_J^{\mathrm{centre}}\right| = 2^{3(\lfloor \frac{J}{2} \rfloor + 1)}\,.$$

The *modified three dimensional hyperbolic cross* $\tilde{H}_J^3$ is given by

$$\tilde{H}_J^3 := \tilde{H}_J^{\mathrm{centre}} \cup \bigcup_{r=0}^{\lceil \frac{J}{2} \rceil} \tilde{H}_{J,r}\,,$$

cf. Figure 5.2. Thus, the total number of hyperbolic cross points is $|\tilde{H}_J^3| = 2^J 6(2^{\lceil \frac{J}{2} \rceil+1} - 1) + 2^{3(\lfloor \frac{J}{2} \rfloor +1)}$, compared to $|I^3_{(N,N,N)^\top}| = N^3 = 8^{J+2}$ indices for the full tensor product grid. Similar to equation (3.3), we are now interested in the computation of

$$f(\boldsymbol{x}_\ell) = \sum_{\boldsymbol{k} \in \tilde{H}_J^{\text{centre}}} \hat{f}_{\boldsymbol{k}} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} + \sum_{r=0}^{\lceil \frac{J}{2} \rceil} \sum_{\boldsymbol{k} \in \tilde{H}_{J,r}} \hat{f}_{\boldsymbol{k}} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell}$$

at arbitrary knots $\boldsymbol{x}_\ell \in \mathbb{T}^3$, $\ell = 0, \ldots, M-1$. We compute each of the blocks as in the two dimensional case and end up with the following algorithm of arithmetic complexity $\mathcal{O}(J 2^{J+\lceil \frac{J}{2} \rceil} + JM)$.

Algorithm 5.1 (SNFFT 3D)

Input: $J \in \mathbb{N}_0$, $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$ *for* $\boldsymbol{k} \in \tilde{H}_J^3$,
$M \in \mathbb{N}$, $\boldsymbol{x}_\ell \in \mathbb{T}^3$ *for* $\ell = 0, \ldots, M-1$.

1. *Compute the values*

$$\tilde{f}(\boldsymbol{x}_\ell) = \sum_{\boldsymbol{k} \in H^0_{\lfloor \frac{J}{2} \rfloor+1} \times H^0_{\lfloor \frac{J}{2} \rfloor+1} \times H^0_{\lfloor \frac{J}{2} \rfloor+1}} \hat{f}_{k_1,k_2,k_3} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} .$$

by a trivariate NFFT$(2^{\lfloor \frac{J}{2} \rfloor+1}, 2^{\lfloor \frac{J}{2} \rfloor+1}, 2^{\lfloor \frac{J}{2} \rfloor+1})$.

2. *For* $r = 0, \ldots, \lceil \frac{J}{2} \rceil$ *compute*

$$\begin{aligned}
\tilde{f}(\boldsymbol{x}_\ell) &= \tilde{f}(\boldsymbol{x}_\ell) \\
&+ \mathrm{e}^{2\pi \mathrm{i} \lceil \frac{3}{2} 2^{J-r} \rceil (\boldsymbol{x}_\ell)_1} \sum_{\boldsymbol{k} \in H^0_{J-r} \times H^0_r \times H^0_r} \hat{f}_{k_1 - \lceil \frac{3}{2} 2^{J-r} \rceil, k_2, k_3} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \\
&+ \mathrm{e}^{2\pi \mathrm{i} \lceil \frac{3}{2} 2^{J-r} \rceil (\boldsymbol{x}_\ell)_2} \sum_{\boldsymbol{k} \in H^0_r \times H^0_{J-r} \times H^0_r} \hat{f}_{k_1, k_2 - \lceil \frac{3}{2} 2^{J-r} \rceil, k_3} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \\
&+ \mathrm{e}^{2\pi \mathrm{i} \lceil \frac{3}{2} 2^{J-r} \rceil (\boldsymbol{x}_\ell)_3} \sum_{\boldsymbol{k} \in H^0_r \times H^0_r \times H^0_{J-r}} \hat{f}_{k_1, k_2, k_3 - \lceil \frac{3}{2} 2^{J-r} \rceil} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \\
&+ \mathrm{e}^{-2\pi \mathrm{i} \lfloor \frac{3}{2} 2^{J-r} \rfloor (\boldsymbol{x}_\ell)_1} \sum_{\boldsymbol{k} \in H^0_{J-r} \times H^0_r \times H^0_r} \hat{f}_{k_1 + \lfloor \frac{3}{2} 2^{J-r} \rfloor, k_2, k_3} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \\
&+ \mathrm{e}^{-2\pi \mathrm{i} \lfloor \frac{3}{2} 2^{J-r} \rfloor (\boldsymbol{x}_\ell)_2} \sum_{\boldsymbol{k} \in H^0_r \times H^0_{J-r} \times H^0_r} \hat{f}_{k_1, k_2 + \lfloor \frac{3}{2} 2^{J-r} \rfloor, k_3} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} \\
&+ \mathrm{e}^{-2\pi \mathrm{i} \lfloor \frac{3}{2} 2^{J-r} \rfloor (\boldsymbol{x}_\ell)_3} \sum_{\boldsymbol{k} \in H^0_r \times H^0_r \times H^0_{J-r}} \hat{f}_{k_1, k_2, k_3 + \lfloor \frac{3}{2} 2^{J-r} \rfloor} \ \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{x}_\ell} .
\end{aligned}$$

by six trivariate NFFT$(2^r, 2^r, 2^{J-r})$.

Output: $\tilde{f}(\boldsymbol{x}_\ell)$ *approximate value of* $f(\boldsymbol{x}_\ell)$, $\ell = 0, \ldots, M-1$.

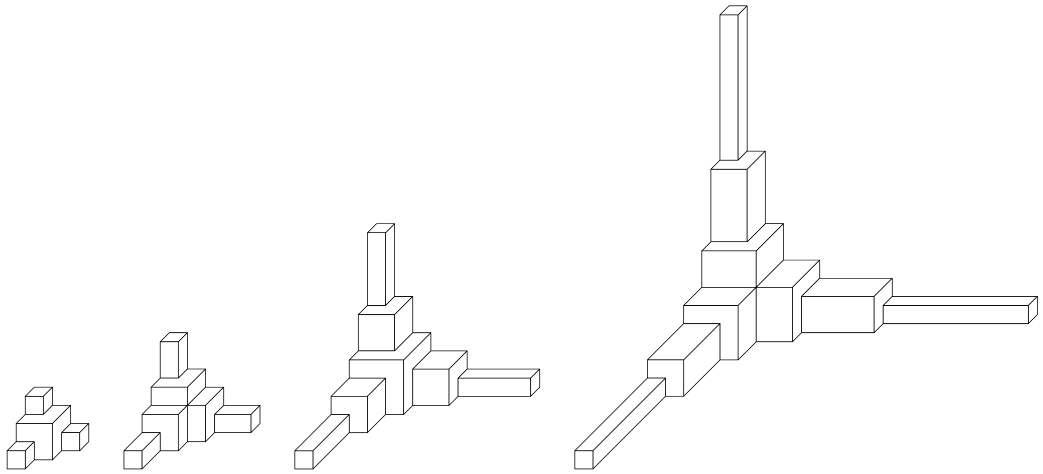

Figure 5.2: Modified hyperbolic cross in 3D for $J = 0, \dots, 3$, only the part k_1, k_2, $k_3 \le 0$ is shown.

6 Numerical results

Our algorithms were implemented in C and tested on an AMD Athlon$^{\mathrm{TM}}$XP 2700+ with 2GB main memory, SuSe-Linux (kernel 2.4.20-4GB-athlon, gcc 3.3) using double precision arithmetic. Further, we have used the libraries FFTW 3.0.1 [10] and NFFT 2.0.1 [12]. In the following, we compare Algorithm 3.1 and Algorithm 5.1 with the straightforward summation (2.3), denoted by SNDFT (sparse nonequispaced discrete Fourier transform) and with the 'ordinary' NFFT where $N = 2^{J+2}$ and all Fourier coefficients with an index not in the sets H_J^2 and $\tilde{H}_J^3$, respectively, are set to zero. We have chosen random knots $\boldsymbol{x}_\ell \in [-\frac{1}{2}, \frac{1}{2}]^d$ and random Fourier coefficients $\hat{f}_{\boldsymbol{k}} \in \{a + b\mathrm{i} : a, b \in [0, 1]\}$.
All tests use an oversampling factor of $\alpha = 2$ and the Gaussian window function. We precomputed the Gaussian at 10^5 equispaced evaluations points. Then, a linear interpolation scheme is used during the NFFT to compute an actual value of the window function at a certain knot $\boldsymbol{x}_\ell$.

First, we examine the error caused by the various approximations within the SNFFT in Algorithms 3.1 and 5.1. The relative error

$$E_\infty := \frac{|f(\boldsymbol{x}_\ell) - \tilde{f}(\boldsymbol{x}_\ell)|}{\sum_{\boldsymbol{k} \in H_{\boldsymbol{N}}^d} |\hat{f}_{\boldsymbol{k}}|} \tag{6.1}$$

is shown in Figure 6.1. According to the estimate in (2.2), the error decays exponentially as m increases. Due to our approximation of the Gaussian window function, it saturates at a level of 10^{-10}.
Next, we are interested in computation times and memory requirements. Here, we choose the number of evaluation knots equal to the number of Fourier coefficients on the hyperbolic cross, i.e., $M = (J + 4)2^{J+1}$ for $d = 2$ and $M = 2^J 6(2^{\lceil \frac{J}{2} \rceil + 1} - 1) + 2^{3(\lfloor \frac{J}{2} \rfloor + 1)}$ for $d = 3$. We compare the computation time and the memory requirements of the SNFFT, of the straightforward summation SNDFT, and of the 'ordinary' NFFT. Table 6.1 shows the theoretical CPU-time and the memory requirements. The actually required CPU times of all three algorithms are shown in Figure 6.2. As expected, the SNFFT outperforms the other algorithms. So we

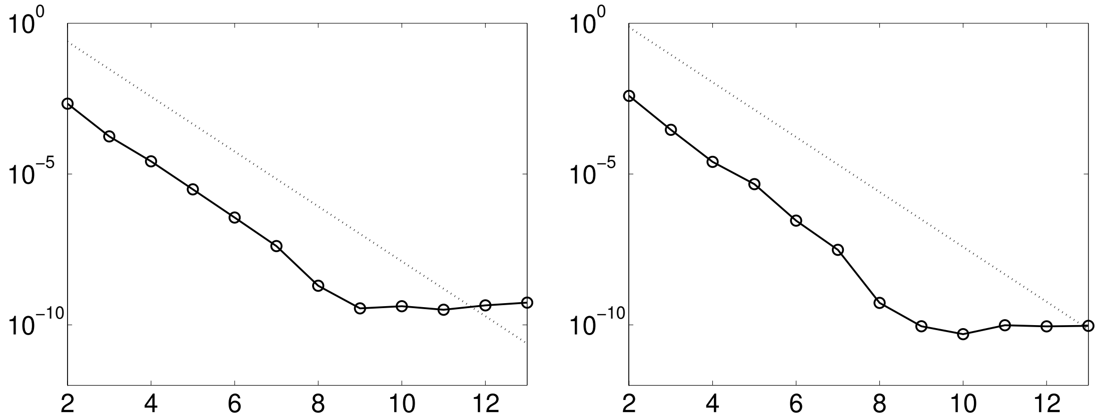

Figure 6.1: The error E_∞ (solid) and the error estimate given in (2.2) (dashed). Left: Algorithm 3.1 with $J = 7$ and $m = 2, \ldots, 13$. Right: Algorithm 5.1 with $J = 5$ and $m = 2, \ldots, 13$.

obtain, e.g., for $d = 2$, $J = 12$ and $M = |H_{12}^d| = 131072$, a CPU-time of 37 seconds for the SNFFT compared to 37 minutes for the SNDFT.

In the second test, we face the memory requirements of all three algorithms as shown in Figure 6.3. Here, the SNFFT needs only a constant amount of 2 MByte more for precomputations than the SNDFT.

	$d = 2$		$d = 3$	
algorithm	time	memory	time	memory
NFFT	$J2^{2J}$	2^{2J}	$J2^{3J}$	2^{3J}
SNDFT	$J^2 2^{2J}$	$J2^J$	2^{3J}	$2^{\frac{3}{2}J}$
SNFFT	$J^2 2^J$	$J2^J$	$J2^{\frac{3}{2}J}$	$2^{\frac{3}{2}J}$

Table 6.1: Theoretical order of magnitude for CPU-time and memory requirements.

Concluding remarks

In this paper, we proposed fast algorithms for the evaluation of trigonometric polynomials from hyperbolic cross points at arbitrary knots. Thus, we generalised the algorithms presented in [1, 11] for frequencies chosen from a hyperbolic cross and knots on a sparse grid. Our algorithms are based on a block partition of the hyperbolic cross, coupled with NFFTs on every block. The numerical results have shown the superiority of the proposed algorithms with respect to computing time, whereas the memory requirements and the approximation error remain bounded.

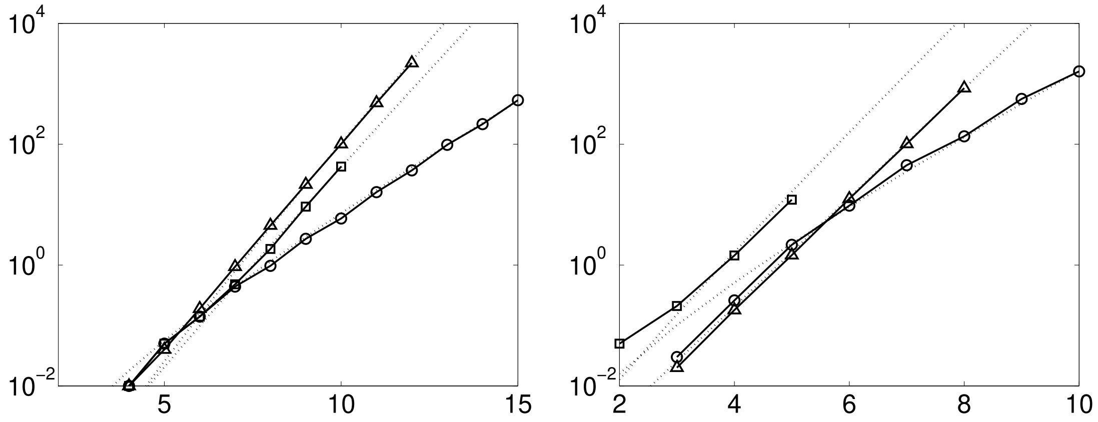

Figure 6.2: Elapsed CPU-time in seconds (solid) and theoretical orders of magnitude given in Table 6.1 (dashed) for the SNFFT (circle), SNDFT (triangle), and NFFT (square). Left: Algorithm 3.1 with $J = 2, \dots, 15$, $m = 4$. Right: Algorithm 5.1 with $J = 2, \dots, 10$, $m = 4$.

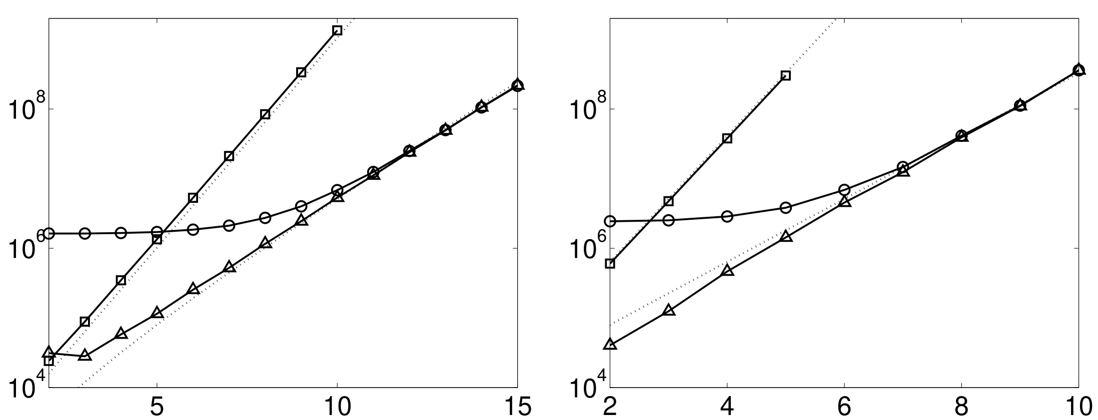

Figure 6.3: Memory requirements in bytes (solid) and theoretical orders of magnitude given in Table 6.1 (dashed) for the SNFFT (circle), SNDFT (triangle), and NFFT (square). Left: Algorithm 3.1 with $J = 2, \dots, 15$, $m = 4$. Right: Algorithm 5.1 with $J = 2, \dots, 10$, $m = 4$.

References

[1] G. Baszenski and F.-J. Delvos. A discrete Fourier transform scheme for Boolean sums of trigonometric operators. In C. K. Chui, W. Schempp, and K. Zeller, editors, *Multivariate Approximation Theory IV*, ISNM 90, pages 15–24. Birkhäuser, Basel, 1989.

[2] G. Beylkin. On the fast Fourier transform of functions with singularities. *Appl. Comput. Harmon. Anal.*, 2:363 – 381, 1995.

[3] H.-J. Bungartz and M. Griebel. Sparse grids. *Acta Numerica*, 13:147 – 269, 2004.

[4] J. W. Cooley and J. W. Tukey. An algorithm for machine calculation of complex Fourier series. *Math. Comput.*, 19:297 – 301, 1965.

[5] A. J. W. Duijndam and M. A. Schonewille. Nonuniform fast Fourier transform. *Geophysics*, 64:539 – 551, 1999.

[6] A. Dutt and V. Rokhlin. Fast fourier transforms for nonequispaced data. SIAM *J. Sci. Stat. Comput.*, 14:1368 – 1393, 1993.

[7] B. Elbel and G. Steidl. Fast Fourier transform for nonequispaced data. In C. K. Chui and L. L. Schumaker, editors, *Approximation Theory IX*, Nashville, 1998. Vanderbilt University Press.

[8] M. Fenn and D. Potts. Fast summation based on fast trigonometric transforms at nonequispaced nodes. *Numer. Linear Algebra Appl.*, 12:161 – 169, 2005.

[9] J. Fessler and B. Sutton. NUFFT - nonuniform FFT toolbox for Matlab. http://www.eecs.umich.edu/˜fessler/code/index.html, 2002.

[10] M. Frigo and S. G. Johnson. FFTW, a C subroutine library. http://www.fftw.org/.

[11] K. Hallatschek. Fouriertransformation auf dünnen Gittern mit hierarchischen Basen. *Numer. Math.*, 63:83–97, 1992.

[12] S. Kunis and D. Potts. NFFT, Softwarepackage, C subroutine library. http://www.math.uni-luebeck.de/potts/nfft, 2002 – 2005.

[13] D. Potts, G. Steidl, and M. Tasche. Fast Fourier transforms for nonequispaced data: A tutorial. In J. J. Benedetto and P. J. S. G. Ferreira, editors, *Modern Sampling Theory: Mathematics and Applications*, pages 247 – 270, Boston, 2001. Birkhäuser.

[14] F. Sprengel. A class of function spaces and interpolation on sparse grids. *Numer. Funct. Anal. Optim.*, 21:273 – 293, 2000.

[15] G. Steidl. A note on fast Fourier transforms for nonequispaced grids. *Adv. Comput. Math.*, 9:337 – 353, 1998.

[16] A. F. Ware. Fast approximate Fourier transforms for irregularly spaced data. SIAM *Rev.*, 40:838 – 856, 1998.

[17] C. Zenger. Sparse grids. In *Parallel algorithms for partial differential equations (Kiel, 1990)*, volume 31 of *Notes Numer. Fluid Mech.*, pages 241–251. Vieweg, Braunschweig, 1991.