TECHNISCHE UNIVERSITÄT
CHEMNITZ

Faculty of Mathematics
Professorship of Applied Functional Analysis

# Master's Thesis

**A frame-theoretical approach to the inversion of the NFFT**

Melanie Kircheis, B.Sc.

Chemnitz, June 28, 2018

| | |
|---|---|
| **Advisor:** | Prof. Dr. Daniel Potts |
| **Co-Advisor:** | Dr. Franziska Nestler |

**Kircheis, Melanie**
A frame-theoretical approach to the inversion of the NFFT
Master's Thesis, Faculty of Mathematics
Technische Universität Chemnitz, June 2018

# Contents

# 1 Introduction

The NFFT, short hand for nonequispaced fast Fourier transform, is a fast algorithm to evaluate a trigonometric polynomial

$$f(x) = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k \,\mathrm{e}^{2\pi \mathrm{i} k x} \tag{1.1}$$

at nonequispaced points $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$. In case we are given equispaced points and $M = N$, this evaluation can be realized by means of the FFT, a fast algorithm that is invertible. Hence, we are interested in an inversion also for non-equispaced data, i. e., the Fourier coefficients $\hat{f}_k$ shall be computed for given function values $f(x_j)$ of the trigonometric polynomial (1.1). Moreover, the number $N$ of nodes $x_j$ is independent from the number $M$ of Fourier coefficients $\hat{f}_k$ and hence the nonequispaced Fourier matrix

$$\boldsymbol{A} := \left(\mathrm{e}^{2\pi \mathrm{i} k x_j}\right)_{j=1,\, k=-\frac{M}{2}}^{N,\, \frac{M}{2}-1} \in \mathbb{C}^{N \times M}$$

which we would have to invert is rectangular in most cases.

Nevertheless, several approaches have been developed to compute an inverse NFFT (iNFFT). Already in [7] a method was explained which uses Lagrange interpolation as well as fast multipole methods. An approach for the overdetermined case can be found in [8] where the solution is computed iteratively by dint of the CG algorithm using $\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{A}$ with a diagonal matrix $\boldsymbol{W}$ with the voronoi weights. Furthermore, in [18] the CG method in connection with the NFFT was used to formulate an iterative algorithm for the underdetermined setting which deploys $\boldsymbol{A}\hat{\boldsymbol{W}}\boldsymbol{A}^*$ with weights $\hat{\boldsymbol{W}}$ based on kernel approximation. Recently, a direct method for the quadratic setting, i. e., $N = M$, was deduced in [17] which is also based on Lagrange interpolation but utilizes fast summation to evaluate the occuring sums.

In this thesis we develop another direct method for inverting the NFFT in general. For this purpose, we take as a motivation that for equispaced points an inversion can be realized by $\boldsymbol{A}\boldsymbol{A}^* \approx M\boldsymbol{I}_N$ and $\boldsymbol{A}^*\boldsymbol{A} \approx N\boldsymbol{I}_M$, respectively. We aim to generalize this result to find a good approximation of the inversion for nonequispaced nodes. To this end, we make use of the decomposition $\boldsymbol{A} \approx \boldsymbol{B}\boldsymbol{F}\boldsymbol{D}$ known from the NFFT approach and compute the sparse matrix $\boldsymbol{B}$ such that we receive approximations of the form $\boldsymbol{A}\boldsymbol{F}^*\boldsymbol{D}^*\boldsymbol{B}^* \approx M\boldsymbol{I}_N$ and $\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{B} \approx \boldsymbol{I}_{M_\sigma}$, respectively, where $M_\sigma \geq M$.

In other words, we are able to compute an inverse NFFT by dint of a modified adjoint NFFT. Analogously, an inverse adjoint NFFT can be obtained by modifying the NFFT. Hence, the inversions can be computed in $\mathcal{O}(M \log M + N)$ arithmetic operations. The corresponding precomputations for the entries of the matrix $\boldsymbol{B}$ are of complexity $\mathcal{O}(N^2)$ and $\mathcal{O}(M^2)$, respectively. Therefore, this method is especially beneficial in case we are given fixed nodes for several problems. By means of iterative solvers the approximation could be improved additionally. Finally, we show that these approaches can also be explained by means of frame approximation.

The present thesis is organized as follows. In Chapter 2 we introduce some important concepts from the field of Fourier analysis. There we mainly focus on the already mentioned algorithm, the NFFT. Afterwards, in Chapter 3 we deal with the inversion of this algorithm. A motivation for our approach is given and we will find out that there are four different cases to be considered which shall be treated separately in Sections 3.2 to 3.5. However, for all of these problems we use the minimization of a certain Frobenius norm, similar to [19]. In each section we start by deducing the main algorithm. Since we will see that those algorithms are of high complexity, two more algorithms will be explained, each reducing the computational costs of the previous. Given these algorithms, also numerical examples will be provided. Finally, in Chapter 4 another approach for inverting the NFFT shall be deduced. This happens based on ideas from the field of frame theory. Therefore, first of all, the main ideas of frames and approximation via frames will be introduced in Sections 4.1 and 4.2. Subsequently, in Section 4.3 we will use these ideas to develop an approach for the iNFFT, adapted from [14]. In the end, we will see that both derived frame-theoretical approaches can be traced back to the methods for the inversion of the NFFT as introduced in Chapter 3.

# 2 Fundamentals

In this chapter we will have a look at some important ideas from the field of Fourier analysis. Besides the term of Fourier series, two fast algorithms for the computation of a discrete Fourier transform for differently spaced nodes will be introduced.

## 2.1 Fourier series

We consider the Hilbert space $L_2(\mathbb{T})$ of all 1-periodic, complex-valued functions, where the so-called **torus** is given by $\mathbb{T} := \mathbb{R}/\mathbb{Z} \simeq \left[-\frac{1}{2}, \frac{1}{2}\right)$.

Very important for the theory of Fourier series are functions of the form

$$\left\{ \mathrm{e}^{2\pi \mathrm{i} kx} : k \in \mathbb{Z} \right\}.$$

These constitute an orthonormal basis of $L_2(\mathbb{T})$, for a proof see e. g. [10, 24]. Thereby, every function $f \in L_2(\mathbb{T})$ is uniquely representable in the form

$$f(x) = \sum_{k\in\mathbb{Z}} c_k(f)\, \mathrm{e}^{2\pi \mathrm{i} kx}, \tag{2.1}$$

where the sum converges to $f$ in the $L_2(\mathbb{T})$-norm, cf. [12, 26].

**Definition 2.1** A series of the form (2.1) is named **Fourier series**. The coefficients

$$c_k(f) := \int_{-\frac{1}{2}}^{\frac{1}{2}} f(x)\, \mathrm{e}^{-2\pi \mathrm{i} kx}\, \mathrm{d}x, \quad k \in \mathbb{Z}, \tag{2.2}$$

are termed **Fourier coefficients** of the function $f$. ◇

## 2.2 Fast Fourier transform

In applications the Fourier coefficients (2.2) are mostly approximated by a quadrature formula. For this purpose, we consider for $M \in 2\mathbb{N}$ the 1-periodic function $f$ as well as the equispaced points

$$x_j = \tfrac{j}{M} \in \left[-\tfrac{1}{2}, \tfrac{1}{2}\right), \quad j = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1.$$

The corresponding function values shall be defined as $f_j \coloneqq f(x_j)$. By using the rectangular rule, which is equivalent to the trapezoidal rule in the periodic setting, the Fourier coefficients can be approximated as follows.

$$c_k(f) \approx \hat{f}_k \coloneqq \frac{1}{M} \sum_{j=-\frac{M}{2}}^{\frac{M}{2}-1} f_j \, \mathrm{e}^{-2\pi \mathrm{i} jk/M}, \quad k \in \mathbb{Z}.$$

One can show that

$$\hat{f}_k \approx c_k(f) \quad \forall\, k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1,$$

is an acceptable approximation, see e. g. [12].

**Definition 2.2** The mapping

$$\mathbb{C}^M \to \mathbb{C}^M, \; \boldsymbol{f} \coloneqq \left(f_{-\frac{M}{2}}, \dots, f_{\frac{M}{2}-1}\right) \mapsto \hat{\boldsymbol{f}} \coloneqq \left(\hat{f}_{-\frac{M}{2}}, \dots, \hat{f}_{\frac{M}{2}-1}\right)$$

is called **discrete Fourier transform (DFT)** of order $M$. In matrix-vector notation the DFT can also be noted as $\hat{\boldsymbol{f}} = \boldsymbol{F}_M \boldsymbol{f}$ with the $M$-th **Fourier matrix**

$$\boldsymbol{F}_M = \left(\mathrm{e}^{-2\pi \mathrm{i} jk/M}\right)_{j,\,k=-\frac{M}{2}}^{\frac{M}{2}-1},$$

cf. [24, 26]. It can be shown that this transformation is invertible. The **inverse discrete Fourier transform (iDFT)** is given by

$$f_j = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k \, \mathrm{e}^{2\pi \mathrm{i} jk/M}, \quad j = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1.$$

$\diamond$

If we want to calculate a DFT in practice, it becomes apparent that this becomes costly very quickly. Thus, a DFT of length $M$ requires overall $M^2$ multiplications as well as $M(M-1)$ additions and therefore $\mathcal{O}(M^2)$ arithmetic operations. However, this complexity remains out of the question for practical usage.

Another method for computing a DFT is the **fast Fourier transform (FFT)**. For $M \in 2^{\mathbb{N}}$ it can be shown that the calculation of a DFT of length $M$ can be done by a computation of two DFT of length $\frac{M}{2}$. These correspond again to the computation of four DFT of length $\frac{M}{4}$, and so on. By means of this strategy we end up with costs of $\mathcal{O}(M \log M)$ arithmetic operations, see also [25, 24].

## 2.3 Nonequispaced fast Fourier transform

Given nonequispaced data instead, we need another fast algorithm to calculate a DFT at arbitrarily distributed nodes. This algorithm is referred to as **nonequispaced fast Fourier transform (NFFT)** and shall briefly be explained below, cf. [6, 2, 23, 22, 21, 16].

For given nodes $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$, $M \in 2\mathbb{N}$, as well as arbitrary coefficients $\hat{f}_k \in \mathbb{C}$, $k = -\frac{M}{2}, \dots, \frac{M}{2}$, we consider the computation of sums

$$f_j = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k \, \mathrm{e}^{2\pi \mathrm{i} k x_j}, \quad j = 1, \dots, N, \tag{2.3}$$

and the adjoint problem of the computation of sums

$$h_k = \sum_{j=1}^{N} f_j \, \mathrm{e}^{-2\pi \mathrm{i} k x_j}, \quad k = -\frac{M}{2}, \dots, \frac{M}{2} - 1, \tag{2.4}$$

for given values $f_j \coloneqq f(x_j) \in \mathbb{C}$, respectively.

### 2.3.1 The NFFT

We firstly restrict our attention to the problem (2.3). One can notice that this is equivalent to the evaluation of a trigonometric polynomial

$$f(x) = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k \, \mathrm{e}^{2\pi \mathrm{i} k x} \tag{2.5}$$

at given points $x_j$, $j = 1, \dots, N$. This function $f$ can now be approximated by a linear combination of translates of a 1-periodic function $\tilde{w}$, i. e.,

$$f(x) \approx s_1(x) \coloneqq \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} g_l \, \tilde{w}\left(x - \frac{l}{M_\sigma}\right),$$

where $M_\sigma = \sigma M$ with the so-called **oversampling factor** $\sigma \geq 1$. This corresponds to a discrete convolution. In the easiest case $\tilde{w}$ originates from periodization of a function $w\colon [-\frac{1}{2}, \frac{1}{2}) \to \mathbb{R}$. Let this so-called **window function** be chosen such that its 1-periodic version

$$\tilde{w}(x) = \sum_{r \in \mathbb{Z}} w(x + r)$$

has an absolute convergent Fourier series. By means of the convolution theorem and the definition

$$\hat{g}_k := \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} g_l \, \mathrm{e}^{-2\pi \mathrm{i} kl/M_\sigma}, \quad k \in \mathbb{Z},$$

$s_1$ can be represented as

$$\begin{aligned} s_1(x) &= \sum_{k=-\infty}^{\infty} c_k(s_1) \, \mathrm{e}^{2\pi \mathrm{i} kx} \\ &= \sum_{k=-\infty}^{\infty} \hat{g}_k \, c_k(\tilde{w}) \, \mathrm{e}^{2\pi \mathrm{i} kx} \\ &= \sum_{k=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \hat{g}_k \, c_k(\tilde{w}) \, \mathrm{e}^{2\pi \mathrm{i} kx} + \sum_{\substack{r=-\infty \\ r\neq 0}}^{\infty} \sum_{k=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \hat{g}_k \, c_{k+M_\sigma r}(\tilde{w}) \, \mathrm{e}^{2\pi \mathrm{i}(k+M_\sigma r)x}. \end{aligned} \tag{2.6}$$

Comparing (2.5) and (2.6) gives the intention for the following definition. We set

$$\hat{g}_k := \begin{cases} \dfrac{\hat{f}_k}{\hat{w}(k)} & : k \in \{-\frac{M}{2}, \dots, \frac{M}{2}-1\}, \\ 0 & : k \in \{-\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2}-1\} \setminus \{-\frac{M}{2}, \dots, \frac{M}{2}-1\}. \end{cases}$$

where

$$\hat{w}(k) = \int_{-\infty}^{\infty} w(x) \, \mathrm{e}^{-2\pi \mathrm{i} kx} \, \mathrm{d}x = \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{w}(x) \, \mathrm{e}^{-2\pi \mathrm{i} kx} \, \mathrm{d}x = c_k(\tilde{w}) \tag{2.7}$$

is the Fourier transform of $w$.

Further, we suppose that $w$ is small outside the interval $[-{}^{m}/_{M_\sigma}, {}^{m}/_{M_\sigma}]$, $m \ll M_\sigma$. Then $w$ can be approximated by $w_m(x) = \chi_{[-m/M_\sigma, m/M_\sigma]} \cdot w(x)$ which is compactly supported since $\chi_{[-m/M_\sigma, m/M_\sigma]}$ denotes the characteristic function of the interval $[-{}^{m}/_{M_\sigma}, {}^{m}/_{M_\sigma}]$. Thus, $\tilde{w}$ can be approximated by the 1-periodic function $\tilde{w}_m$ with

$$\sum_{k\in\mathbb{Z}} \hat{w}(k) \, \mathrm{e}^{2\pi \mathrm{i} kx} = \tilde{w}(x) \approx \tilde{w}_m(x) = \sum_{r\in\mathbb{Z}} w_m(x+r).$$

Hence, we obtain the following approximation

$$f(x_j) \approx s_1(x_j) \approx s(x_j) := \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} g_l \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) = \sum_{l=\lceil M_\sigma x_j \rceil - m}^{\lfloor M_\sigma x_j \rfloor + m} g_l \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big),$$

where simplification arises because many summands vanish.

*Remark* 2.3 Suitable window functions are for instance

- cardinal B-splines of order $2m$

$$w(x) := B_{2m}(M_\sigma x), \qquad \hat{w}(k) = \frac{1}{M_\sigma}\operatorname{sinc}^{2m}\left(\tfrac{\pi k}{M_\sigma}\right), \tag{2.8}$$

- the Gaussian

$$w(x) := \frac{1}{\sqrt{\pi b}}\, \mathrm{e}^{-\frac{(M_\sigma x)^2}{b}}, \qquad \hat{w}(k) = \frac{1}{M_\sigma}\, \mathrm{e}^{-b\left(\frac{\pi k}{M_\sigma}\right)^2}, \tag{2.9}$$

  with $b := \frac{2\sigma}{2\sigma-1}\frac{m}{\pi}$,

- powers of the sinc function

$$\begin{aligned} w(x) &:= \frac{M(2\sigma-1)}{2m}\operatorname{sinc}^{2m}\left(\frac{\pi M x(2\sigma-1)}{2M_\sigma}\right), \\ \hat{w}(k) &= B_{2m}\left(\frac{2mk}{(2\sigma-1)M}\right), \end{aligned} \tag{2.10}$$

  with $\sigma > 1$

- and Kaiser-Bessel functions

$$w(x) := \frac{1}{\pi}\begin{cases} \dfrac{\sinh(b\sqrt{m^2 - M_\sigma^2 x^2})}{\sqrt{m^2 - M_\sigma^2 x^2}} & : |x| \le \frac{m}{M_\sigma}, \\ \\ \dfrac{\sin(b\sqrt{M_\sigma^2 x^2 - m^2})}{\sqrt{M_\sigma^2 x^2 - m^2}} & : \text{otherwise}, \end{cases} \tag{2.11}$$

$$\hat{w}(k) = \begin{cases} \frac{1}{M_\sigma} I_0\left(m\sqrt{b^2 - \left(\frac{2\pi k}{M_\sigma}\right)^2}\right) & : k = -M_\sigma(1-\frac{1}{2\sigma}), \ldots, M_\sigma(1-\frac{1}{2\sigma}), \\ 0 & : \text{otherwise}, \end{cases}$$

  where $b := \pi(2 - \frac{1}{\sigma})$ and $I_0$ denotes the modified zero-order Bessel function.

For more details see [6, 2, 23, 5, 11, 9, 20, 16]. ◇

Thus, the algorithm can be summarized as follows.

**Algorithm 2.4** (NFFT)

For $N \in \mathbb{N}$, $M \in 2\mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$, be given nodes as well as Fourier coefficients $\hat{f}_k \in \mathbb{C}$ with $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$. Furthermore, we are given the oversampling factor $\sigma \geq 1$ and $M_\sigma := \sigma M$ as well as the window function $w$, the truncated function $w_m$ with $m \ll M_\sigma$ and their 1-periodic versions $\tilde{w}$ and $\tilde{w}_m$.

1. Set
$$\hat{g}_k := \begin{cases} \frac{\hat{f}_k}{\hat{w}(k)} & : k \in \{-\frac{M}{2}, \dots, \frac{M}{2} - 1\}, \\ 0 & : k \in \{-\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1\} \setminus \{-\frac{M}{2}, \dots, \frac{M}{2} - 1\}. \end{cases}$$
$\mathcal{O}(M)$

2. Compute
$$g_l := \frac{1}{M_\sigma} \sum_{k=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \hat{g}_k \, \mathrm{e}^{2\pi \mathrm{i} kl/M_\sigma}$$
for all $l = -\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1$, by an inverse FFT. $\mathcal{O}(M \log M)$

3. Compute
$$\tilde{f}_j := \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} g_l \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big), \quad j = 1, \dots, N.$$
$\mathcal{O}(N)$

Output: $\tilde{f}_j \approx f_j$ from (2.3), $j = 1, \dots, N$.

Complexity: $\mathcal{O}(M \log M + N)$

### 2.3.2 The adjoint NFFT

Now we consider the problem (2.4). Therefore, we define analogously to [20] the function
$$\tilde{g}(x) := \sum_{j=1}^{N} f_j \, \tilde{w}(x_j - x) \tag{2.12}$$

and calculate the Fourier coefficients of $\tilde{g}$.

$$\begin{aligned} c_k(\tilde{g}) &= \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{g}(x)\, \mathrm{e}^{-2\pi \mathrm{i} k x}\, \mathrm{d}x \\ &= \int_{-\frac{1}{2}}^{\frac{1}{2}} \sum_{j=1}^{N} f_j\, \tilde{w}(x_j - x)\, \mathrm{e}^{-2\pi \mathrm{i} k x}\, \mathrm{d}x \\ &= \sum_{j=1}^{N} f_j\, \mathrm{e}^{-2\pi \mathrm{i} k x_j} \int_{-\frac{1}{2}}^{\frac{1}{2}} \tilde{w}(y)\, \mathrm{e}^{2\pi \mathrm{i} k y}\, \mathrm{d}y \\ &= h_k\, c_{-k}(\tilde{w}). \end{aligned}$$

In other words, the values $h_k$ sought-after can be computed if $c_{-k}(\tilde{w})$ and $c_k(\tilde{g})$ are known. The Fourier coefficients of $\tilde{g}$ will be determined approximately by dint of the trapezoidal rule

$$c_k(\tilde{g}) \approx \frac{1}{M_\sigma} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \sum_{j=1}^{N} f_j\, \tilde{w}\Big(x_j - \tfrac{l}{M_\sigma}\Big)\, \mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma}.$$

Let the function $w$ moreover be well localized in time so that $\tilde{w}$ can be replaced by $\tilde{w}_m$ again. Then we obtain the approximation

$$\frac{c_k(\tilde{g})}{c_{-k}(\tilde{w})} \approx \frac{1}{M_\sigma \hat{w}(-k)} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \sum_{j=1}^{N} f_j\, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\, \mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma} =: \tilde{h}_k. \tag{2.13}$$

Hence, a fast algorithm can be formulated as follows.

**Algorithm 2.5** (adjoint NFFT)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$ be given nodes as well as $f_j \in \mathbb{C}$, $j = 1, \dots, N$. Furthermore, we are given the oversampling factor $\sigma \geq 1$, $M \in 2\mathbb{N}$ and $M_\sigma := \sigma M$ as well as the window function $w$, the truncated function $w_m$ with $m \ll M_\sigma$ and their 1-periodic versions $\tilde{w}$ and $\tilde{w}_m$.

1. Compute

$$g_l := \sum_{j=1}^{N} f_j\, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big), \quad l = -\tfrac{M_\sigma}{2}, \dots, \tfrac{M_\sigma}{2} - 1.$$

$\mathcal{O}(N)$

2. Compute

$$\hat{g}_k \coloneqq \frac{1}{M_\sigma} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} g_l \, \mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma}$$

for all $k = -\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1$, by an FFT. $\mathcal{O}(M \log M)$

3. Set

$$\tilde{h}_k \coloneqq \frac{\hat{g}_k}{\hat{w}(-k)}, \quad k = -\frac{M}{2}, \dots, \frac{M}{2} - 1.$$

$\mathcal{O}(M)$

Output: $\tilde{h}_k \approx h_k$ from (2.4), $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$.

Complexity: $\mathcal{O}(M \log M + N)$

### 2.3.3 Matrix-vector notation

By defining the nonequispaced Fourier matrix

$$\boldsymbol{A} \coloneqq \left(\mathrm{e}^{2\pi \mathrm{i} k x_j}\right)_{j=1,\, k=-\frac{M}{2}}^{N,\, \frac{M}{2}-1} \in \mathbb{C}^{N \times M} \tag{2.14}$$

as well as the vectors

$$\boldsymbol{f} \coloneqq (f_j)_{j=1}^{N}, \; \hat{\boldsymbol{f}} \coloneqq \left(\hat{f}_k\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \quad \text{and} \quad \boldsymbol{h} \coloneqq (h_k)_{k=-\frac{M}{2}}^{\frac{M}{2}-1},$$

the computation of the sums (2.3) can be rewritten as

$$\boldsymbol{f} = \boldsymbol{A}\hat{\boldsymbol{f}}.$$

Likewise the computation of the sums (2.4) can be rewritten as

$$\boldsymbol{h} = \boldsymbol{A}^* \boldsymbol{f},$$

where $\boldsymbol{A}^* = \overline{\boldsymbol{A}}^{\mathrm{T}}$ denotes the adjoint matrix of $\boldsymbol{A}$. In addition, we define

- the diagonal matrix

$$\boldsymbol{D} \coloneqq \operatorname{diag}\left(\frac{1}{M_\sigma \hat{w}(k)}\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \in \mathbb{C}^{M \times M},$$

- the truncated Fourier matrix

$$\boldsymbol{F} := \left(\mathrm{e}^{2\pi\mathrm{i}k\frac{l}{M_\sigma}}\right)_{l=-\frac{M_\sigma}{2},\, k=-\frac{M}{2}}^{\frac{M_\sigma}{2}-1,\ \frac{M}{2}-1} \in \mathbb{C}^{M_\sigma \times M},$$

- and the sparse matrix

$$\boldsymbol{B} := \left(\tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\right)_{j=1,\, l=-\frac{M_\sigma}{2}}^{N,\ \frac{M_\sigma}{2}-1} \in \mathbb{R}^{N \times M_\sigma}. \tag{2.15}$$

Therefore, also the steps of Algorithm 2.4 can be perceived as matrix-vector products. Now having a look at Algorithm 2.5 and keeping in mind that for real-valued $w$ we have

$$\hat{w}(-k) = \int_{-\infty}^{\infty} w(x)\,\mathrm{e}^{2\pi\mathrm{i}kx}\,\mathrm{d}x = \overline{\int_{-\infty}^{\infty} w(x)\,\mathrm{e}^{-2\pi\mathrm{i}kx}\,\mathrm{d}x} = \overline{\hat{w}(k)},$$

we recognize that this is exactly the adjoint of Algorithm 2.4. Hence, we receive the approximations

$$\boldsymbol{A} \approx \boldsymbol{B}\boldsymbol{F}\boldsymbol{D} \quad \text{and} \quad \boldsymbol{A}^* \approx \boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*.$$

*Remark* 2.6 It must be pointed out that because of consistency the factor $\frac{1}{M_\sigma}$ is here not located in the matrix $\boldsymbol{F}$ as usual but in the matrix $\boldsymbol{D}$. $\diamond$

# 3 Inversion of the NFFT

Now having introduced the fast algorithms for nonequispaced data, our aim is to find an inversion for these algorithms. This idea is encouraged by the fact that for equispaced data the inversion is well-known. Hence, we are now looking for an inversion of the NFFT as well as an inversion of the adjoint NFFT. To this end, we face the following two problems.

(1) Solve

$$\begin{gathered}\boldsymbol{A}\hat{\boldsymbol{f}} = \boldsymbol{f}, \\ \text{given: } \boldsymbol{f} \in \mathbb{C}^N, \text{ find: } \hat{\boldsymbol{f}} \in \mathbb{C}^M,\end{gathered} \tag{3.1}$$

i. e., reconstruct the Fourier coefficients $\hat{\boldsymbol{f}} = (\hat{f}_k)_{k=-\frac{M}{2},\dots,\frac{M}{2}-1}$ from function values $\boldsymbol{f} = (f_j)_{j=1,\dots,N}$. This is called inverse NFFT.

(2) Solve

$$\begin{gathered}\boldsymbol{A}^*\boldsymbol{f} = \boldsymbol{h}, \\ \text{given: } \boldsymbol{h} \in \mathbb{C}^M, \text{ find: } \boldsymbol{f} \in \mathbb{C}^N,\end{gathered} \tag{3.2}$$

i. e., reconstruct the coefficients $\boldsymbol{f} = (f_j)_{j=1,\dots,N}$ from the given data $\boldsymbol{h} = (h_k)_{k=-\frac{M}{2},\dots,\frac{M}{2}-1}$. This is named inverse adjoint NFFT.

In both problems the numbers $M$ and $N$ are independent. Therefore, different relations between them are possible so that underdetermined and overdetermined cases occur.

Firstly we consider the inverse NFFT in (3.1). It is obvious that except for the quadratic setting $M = N$ there are two different ways to choose $M$ and $N$. The first possibility is $M < N$, i. e., we are given more function values than Fourier coefficients, which we are supposed to find. That means, we are given more data than we have to compute and thus we are in an overdetermined setting. The second variation is the converse setting $M > N$. There we have to find more Fourier coefficients than we are given initial data. Hence, this is the underdetermined case. Analogously, the same relations can be considered for the inverse adjoint NFFT in (3.2). There $M$ belongs to the given data while $N$ goes with the wanted solution. In other words, the overdetermined case in now $M > N$ whereas the problem is underdetermined for $M < N$.

Therefore, we end up with four different cases which shall be considered separately below. In Section 3.2 we start with the underdetermined case of the inverse NFFT. Secondly, we survey the overdetermined case of the inverse adjoint NFFT in Section 3.3. Afterwards, in Section 3.4 the overdetermined case of the inverse NFFT will be explained and finally, in Section 3.5, we examine the underdetermined case of the inverse adjoint NFFT. Though, first of all, we motivate our approach.

## 3.1 Motivation

To clarify the idea of our approach we consider the case of equispaced points $x_j = \frac{j}{N} \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = -\frac{N}{2}, \dots, \frac{N}{2} - 1$. Thereby, we obtain the Fourier matrices in (2.14) as

$$\boldsymbol{A} = \left(\mathrm{e}^{2\pi\mathrm{i}k\frac{j}{N}}\right)_{j=-\frac{N}{2},\, k=-\frac{M}{2}}^{\frac{N}{2}-1,\ \frac{M}{2}-1} \qquad \text{and} \qquad \boldsymbol{A}^* = \left(\mathrm{e}^{-2\pi\mathrm{i}k\frac{j}{N}}\right)_{k=-\frac{M}{2},\, j=-\frac{N}{2}}^{\frac{M}{2}-1,\ \frac{N}{2}-1},$$

respectively. Now we study the composition of these two matrices, i. e., we have a look at products of both, starting with $\boldsymbol{A}^*\boldsymbol{A}$. Thus, we have

$$\boldsymbol{A}^*\boldsymbol{A} = \left[\sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \mathrm{e}^{-2\pi\mathrm{i}k\frac{j}{N}}\, \mathrm{e}^{2\pi\mathrm{i}l\frac{j}{N}}\right]_{k,\, l=-\frac{M}{2}}^{\frac{M}{2}-1} = \left[\sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \mathrm{e}^{2\pi\mathrm{i}j(l-k)/N}\right]_{k,\, l=-\frac{M}{2}}^{\frac{M}{2}-1}.$$

This matrix has obviously main diagonal $N$ because there we have $l = k$ and therefore $(\boldsymbol{A}^*\boldsymbol{A})_{k,k} = \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \mathrm{e}^0 = N$. For the remaining entries with $l \neq k$ it results from the geometric sum formula that

$$\begin{aligned}(\boldsymbol{A}^*\boldsymbol{A})_{k,l} &= \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} \left(\mathrm{e}^{2\pi\mathrm{i}(l-k)/N}\right)^j = \sum_{j=0}^{N-1} \left(\mathrm{e}^{2\pi\mathrm{i}(l-k)/N}\right)^j \cdot \left(\mathrm{e}^{2\pi\mathrm{i}(l-k)/N}\right)^{-N/2} \\ &= \frac{\left(\mathrm{e}^{2\pi\mathrm{i}(l-k)/N}\right)^N - 1}{\mathrm{e}^{2\pi\mathrm{i}(l-k)/N} - 1} \cdot \mathrm{e}^{-\pi\mathrm{i}(l-k)} = \frac{\mathrm{e}^{2\pi\mathrm{i}(l-k)} - 1}{\mathrm{e}^{2\pi\mathrm{i}(l-k)/N} - 1} \cdot \mathrm{e}^{-\pi\mathrm{i}(l-k)},\end{aligned}$$

if $N \nmid (l-k)$. Hence, it follows that $(\boldsymbol{A}^*\boldsymbol{A})_{k,l} = 0$ because the enumerator vanishes because of $\mathrm{e}^{2\pi\mathrm{i}r} = 1$ for all $r \in \mathbb{Z}$. Thus, it ensues $\boldsymbol{A}^*\boldsymbol{A} = N\boldsymbol{I}_M$ for $N \nmid (l-k)\ \forall l, k$, which means $N \geq M$.

For the second matrix product we have

$$\boldsymbol{A}\boldsymbol{A}^* = \left[\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \mathrm{e}^{2\pi\mathrm{i}k\frac{j}{N}}\, \mathrm{e}^{-2\pi\mathrm{i}l\frac{j}{N}}\right]_{j,\, h=-\frac{N}{2}}^{\frac{N}{2}-1} = \left[\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \mathrm{e}^{2\pi\mathrm{i}k(j-h)/N}\right]_{j,\, h=-\frac{N}{2}}^{\frac{N}{2}-1}.$$

Here one can also immediately see the main diagonal because for $j = h$ we have $(\boldsymbol{AA}^*)_{j,j} = \sum_{j=-\frac{M}{2}}^{\frac{M}{2}-1} \mathrm{e}^0 = M$. It arises analogously that for the remaining entries we have

$$\begin{aligned}(\boldsymbol{AA}^*)_{j,h} &= \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \left(\mathrm{e}^{2\pi\mathrm{i}(j-h)/N}\right)^k = \sum_{k=0}^{M-1} \left(\mathrm{e}^{2\pi\mathrm{i}(j-h)/N}\right)^k \cdot \left(\mathrm{e}^{2\pi\mathrm{i}(j-h)/N}\right)^{-M/2} \\ &= \frac{\left(\mathrm{e}^{2\pi\mathrm{i}(j-h)/N}\right)^M - 1}{\mathrm{e}^{2\pi\mathrm{i}(j-h)/N} - 1} \cdot \mathrm{e}^{-\pi\mathrm{i}(j-h)M/N}.\end{aligned}$$

Now we claim $N \mid M(j-h)\ \forall j \neq h$ so that we receive only zeros again. But by definition we have $N \nmid (j-h)$, so we need $N \mid M$, i. e., we obtain $\boldsymbol{AA}^* = M\boldsymbol{I}_N$ for $M \geq N$ with $N \mid M$.

Accordingly, in these special cases we are given an inversion of the NFFT since by composition of the two matrices and application to a vector we are able to retrieve the original vector only multiplied by a certain known constant.

Therefore, we seek to use this result and look for a good approximation of the inversion in the general case. This should be done by modification of the matrix $\boldsymbol{B}$ so that we receive an approximation of the form $\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^* \approx M\boldsymbol{I}_N$ similar to the equispaced case. For that purpose, the entries of this matrix $\boldsymbol{B}$ should be calculated such that its sparse structure with at most $(2m+1)$ entries per row and consequently the arithmetic complexity of the algorithms is preserved. A matrix $\boldsymbol{B}$ satisfying this property we call **(2m+1)-sparse**.

Having this in mind we give an outline how to handle problems (3.1) and (3.2).

(1) To solve (3.1) our aim is to compute a sparse matrix $\boldsymbol{B}^*$ from given nodes $x_j$ such that by application of Algorithm 2.5 we obtain a fast inverse NFFT.

We suppose we are given the approximation $\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^* \approx M\boldsymbol{I}_N$. Then it would also be true that

$$\frac{1}{M}\,\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f} \approx \boldsymbol{f} \quad \forall \boldsymbol{f} \in \mathbb{C}^N. \tag{3.3}$$

If we set

$$\check{\boldsymbol{f}} := \frac{1}{M}\,\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f},$$

we are able to rewrite the above approximation (3.3) as $\boldsymbol{A}\check{\boldsymbol{f}} \approx \boldsymbol{f}$. Since we already know that $\boldsymbol{A}\hat{\boldsymbol{f}} = \boldsymbol{f}$, this would mean that we have $\check{\boldsymbol{f}} \approx \hat{\boldsymbol{f}}$, which could be interpreted as a reconstruction of the Fourier coefficients by modifying the adjoint NFFT.

To achieve a good approximation we want $\check{\boldsymbol{f}}$ to be as close as possible by $\hat{\boldsymbol{f}}$. This works best if we optimize the approximation $\boldsymbol{A}\check{\boldsymbol{f}} \approx \boldsymbol{f}$. In other words, we aim to solve the optimization problem

$$\underset{\check{\boldsymbol{f}}\in\mathbb{C}^M}{\text{Minimize}} \quad \|\boldsymbol{A}\check{\boldsymbol{f}} - \boldsymbol{f}\|_2. \tag{3.4}$$

Using the definition of $\check{\boldsymbol{f}}$ this norm can be estimated by

$$\begin{aligned}\|M\boldsymbol{A}\check{\boldsymbol{f}} - M\boldsymbol{f}\|_2 &= \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f} - M\boldsymbol{f}\|_2 \\ &= \|(\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N)\boldsymbol{f}\|_2 \\ &\le \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}\,\|\boldsymbol{f}\|_2.\end{aligned}$$

Because $\boldsymbol{f}$ is given we minimize the expression above by solving

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2.$$

(2) To solve (3.2) we aim to compute a sparse matrix $\boldsymbol{B}$ from given nodes $x_j$ such that by application of Algorithm 2.4 we obtain a fast inverse adjoint NFFT.

Again we suppose $\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* \approx M\boldsymbol{I}_N$. It can be seen that this is equivalent to the transpose

$$\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^* \approx M\,\boldsymbol{I}_N \quad \text{and} \quad \frac{1}{M}\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\,(\boldsymbol{A}^*\boldsymbol{f}) \approx \boldsymbol{f} \quad \forall \boldsymbol{f}\in\mathbb{C}^N,$$

respectively. Because we know that $\boldsymbol{h} = \boldsymbol{A}^*\boldsymbol{f}$ holds, this could be interpreted as a reconstruction of the function values by modifying the NFFT.

To achieve a good approximation we want to solve the optimization problem

$$\underset{\boldsymbol{f}\in\mathbb{C}^N}{\text{Minimize}} \quad \|\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{h} - M\boldsymbol{f}\|_2,$$

where the norm could be estimated as follows.

$$\begin{aligned}\|\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{h} - M\boldsymbol{f}\|_2 &= \|\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{f} - M\boldsymbol{f}\|_2 \\ &= \|(\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^* - M\boldsymbol{I}_N)\boldsymbol{f}\|_2 \\ &\le \|\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}\,\|\boldsymbol{f}\|_2.\end{aligned}$$

Because $\boldsymbol{f}$ is the wanted solution we minimize the expression above by solving the optimization problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2.$$

So, all in all, this would mean with the chosen approach we are able to generate an inverse NFFT as well as an inverse adjoint NFFT by modifying the matrix $\boldsymbol{B}^*$ and $\boldsymbol{B}$, respectively, and applying Algorithm 2.5 and Algorithm 2.4 with these modified matrices.

*Remark* 3.1 We investigate below if the reconstruction error can be reduced by appropriate choice of the entries of the matrix $\boldsymbol{B}^*$, cf. [19]. There they analyzed the minimization of the Frobenius norm $\|\boldsymbol{A}-\boldsymbol{BFD}\|_{\mathrm{F}}$ regarding a sparse matrix $\boldsymbol{B}$ to achieve an error as small as possible for the NFFT. In contrast, we study the minimization of $\|\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2$ to achieve a minimum error for the inverse NFFT as well as the minimization of $\|\boldsymbol{BFDA}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2$ to achieve a minimum error for the inverse adjoint NFFT. $\diamond$

## 3.2 Inverse NFFT - underdetermined case

For solving the problem (3.1) we consider the matrix $\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^*$ for given nodes $x_j \in \mathbb{T},\ j=1,\dots,N$. Apparently, we have

$$\boldsymbol{D}^*\boldsymbol{F}^* = \left[\frac{1}{M_\sigma \hat{w}(-k)}\,\mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\right]_{k=-\frac{M}{2},\,l=-\frac{M_\sigma}{2}}^{\frac{M}{2}-1,\,\frac{M_\sigma}{2}-1}.$$

Multiplying by the matrix $\boldsymbol{A}$ yields

$$\boldsymbol{AD}^*\boldsymbol{F}^* = \left[\frac{1}{M_\sigma}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\frac{1}{\hat{w}(-k)}\,\mathrm{e}^{2\pi\mathrm{i}k\left(x_j-\frac{l}{M_\sigma}\right)}\right]_{j=1,\,l=-\frac{M_\sigma}{2}}^{N,\,\frac{M_\sigma}{2}-1}. \tag{3.5}$$

By defining the "inverse window function"

$$K(x) = \frac{1}{M_\sigma}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\frac{1}{\hat{w}(-k)}\,\mathrm{e}^{2\pi\mathrm{i}kx} \tag{3.6}$$

we receive

$$\boldsymbol{K} := \boldsymbol{AD}^*\boldsymbol{F}^* = \left(K\left(x_h - \tfrac{l}{M_\sigma}\right)\right)_{h=1,\,l=-\frac{M_\sigma}{2}}^{N,\ \frac{M_\sigma}{2}-1}.$$

Having a closer look at the matrix $\boldsymbol{B}^*$ it becomes apparent that there are only a few nonzero entries. Thus, we are going to study the window function $\tilde{w}_m$ for further simplification. For the window $w_m$ we have

$$\operatorname{supp}(w_m) = \left[-\tfrac{m}{M_\sigma}, \tfrac{m}{M_\sigma}\right],$$

i. e., for the 1-periodic version $\tilde{w}_m(x) := \sum_{z\in\mathbb{Z}} w_m(x+z)$ we have

$$\begin{aligned}\tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) \neq 0 &\iff \exists\, z \in \mathbb{Z}:\ w_m\Big(x_j - \tfrac{l}{M_\sigma} + z\Big) \neq 0\\ &\iff \exists\, z \in \mathbb{Z}:\ -\tfrac{m}{M_\sigma} \leq x_j - \tfrac{l}{M_\sigma} + z \leq \tfrac{m}{M_\sigma}\\ &\iff \exists\, z \in \mathbb{Z}:\ -m \leq M_\sigma x_j - l + M_\sigma z \leq m.\end{aligned}$$

By defining the set

$$I_{M_\sigma,m}(x_j) := \Big\{l \in \big\{-\tfrac{M_\sigma}{2},\dots,\tfrac{M_\sigma}{2}-1\big\} : \exists\, z \in \mathbb{Z} \text{ with} - m \leq M_\sigma x_j - l + M_\sigma z \leq m\Big\} \tag{3.7}$$

we can therefore write

$$\begin{aligned}(\boldsymbol{K}\boldsymbol{B}^*)_{h,j} &= (\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*)_{h,j}\\ &= \sum_{l\in I_{M_\sigma,m}(x_j)} \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(-k)}\, \mathrm{e}^{2\pi\mathrm{i}k\left(x_h - \frac{l}{M_\sigma}\right)}\, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\\ &= \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \mathrm{e}^{2\pi\mathrm{i}kx_h} \left(\sum_{l\in I_{M_\sigma,m}(x_j)} \frac{1}{M_\sigma \hat{w}(-k)}\, \mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\right).\end{aligned}$$

Hence, we have

$$\begin{aligned}&\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2 =\\ &= \sum_{h=1}^{N}\sum_{j=1}^{N} \left| \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \mathrm{e}^{2\pi\mathrm{i}kx_h} \underbrace{\left(\sum_{l\in I_{M_\sigma,m}(x_j)} \frac{1}{M_\sigma \hat{w}(-k)}\, \mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\right)}_{= \frac{1}{M_\sigma \hat{w}(-k)} \sum_{l\in I_{M_\sigma,m}(x_j)} \mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\, \tilde{w}_m\left(x_j - \frac{l}{M_\sigma}\right)} - N\delta_{hj} \right|^2.\end{aligned} \tag{3.8}$$

Based on the definition of the Frobenius norm of a matrix $\boldsymbol{A} \in \mathbb{R}^{k\times n}$ and the definition of the Euclidean norm of a vector $\boldsymbol{x} \in \mathbb{R}^n$ we obtain for $\boldsymbol{a}_j$ being columns of a matrix $\boldsymbol{A} \in \mathbb{R}^{k\times n}$ that

$$\|\boldsymbol{A}\|_F^2 = \sum_{i=1}^{k}\sum_{j=1}^{n} |a_{ij}|^2 = \sum_{j=1}^{n} \|\boldsymbol{a}_j\|_2^2.$$

In our setting this means that (3.8) can be rewritten by dint of

$$\boldsymbol{T}_j = \left(\mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\right)_{k=-\frac{M}{2},\, l\in I_{M_\sigma,m}(x_j)}^{\frac{M}{2}-1}, \qquad \boldsymbol{b}_j = \left(\tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)\right)_{l\in I_{M_\sigma,m}(x_j)},$$

$$\text{and} \quad \boldsymbol{e}_j = (\delta_{hj})_{h=1}^{N},$$

as

$$\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2 = \sum_{j=1}^{N} \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j\boldsymbol{b}_j - M\boldsymbol{e}_j\|_2^2.$$

This expression gets minimal if and only if $\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j\boldsymbol{b}_j - M\boldsymbol{e}_j\|_2^2$ gets minimal for all $j = 1, \ldots, N$. As a result, we obtain the optimization problem

$$\underset{\tilde{\boldsymbol{b}}_j \in \mathbb{R}^{2m+1}}{\text{Minimize}} \ \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j\tilde{\boldsymbol{b}}_j - M\boldsymbol{e}_j\|_2^2, \quad j = 1, \ldots, N, \tag{3.9}$$

because the columns of the matrix $\boldsymbol{B}^*$ contain at most $(2m+1)$ nonzeros. This corresponds to a least squares problem of the form $\|\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y}\|_2^2 \to \min$. To solve it we consider

$$\|\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y}\|_2^2 = (\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y})^*(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y}).$$

By means of the first-order optimality condition

$$\nabla\|\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y}\|_2^2 = 2\boldsymbol{C}^*(\boldsymbol{C}\boldsymbol{x} - \boldsymbol{y}) \overset{!}{=} 0,$$

we receive the normal equations

$$\boldsymbol{C}^*\boldsymbol{C}\boldsymbol{x} = \boldsymbol{C}^*\boldsymbol{y}.$$

These are always solvable and they are unambiguously solvable if and only if $\boldsymbol{C}$ has full column rank. Then the matrix $\boldsymbol{C}^*\boldsymbol{C}$ is regular and we obtain the solution

$$\boldsymbol{x} = (\boldsymbol{C}^*\boldsymbol{C})^{-1}\boldsymbol{C}^*\boldsymbol{y}.$$

Thus, if the matrix $\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j \in \mathbb{C}^{N\times(2m+1)}$ has full rank the solution of problem (3.9) is given by

$$\tilde{\boldsymbol{b}}_j = \left[(\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j)^*\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j\right]^{-1}(\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j)^*M\boldsymbol{e}_j, \quad j = 1, \ldots, N. \tag{3.10}$$

For generating a matrix again it must be pointed out that the vectors $\tilde{\boldsymbol{b}}_j$ only contain the nonzeros of the matrix. Hence, attention should be paid to the periodicity which one can also see in the structure of the matrix $\boldsymbol{B}$. This leads to Algorithm 3.3.

*Remark* 3.2 Whether the matrix $\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j$ has full rank only depends on the matrix $\boldsymbol{A}$. The conditions when this one has full rank can be found e. g. in [15] and [18]. $\diamond$

**Algorithm 3.3** (Optimization of the matrix $\boldsymbol{B}^*$)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \dots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$. Furthermore, we are given the matrices $\boldsymbol{A}, \boldsymbol{D}^*$ and $\boldsymbol{F}^*$.

1. For $j = 1, \dots, N$:

   Determine the set $I_{M_\sigma,m}(x_j)$, cf. (3.7). $\mathcal{O}(1)$

   Determine the matrix $\boldsymbol{T}_j = \left(\mathrm{e}^{-2\pi\mathrm{i}k\frac{l}{M_\sigma}}\right)_{k=-\frac{M}{2},\, l\in I_{M_\sigma,m}(x_j)}^{\frac{M}{2}-1}$. $\mathcal{O}(1)$

   Compute $\mathcal{O}(NM)$

$$\boldsymbol{K}_j := \boldsymbol{A}\boldsymbol{D}^*\boldsymbol{T}_j = \left[\frac{1}{M_\sigma}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{\hat{w}(-k)}\,\mathrm{e}^{2\pi\mathrm{i}k\left(x_h - \frac{l}{M_\sigma}\right)}\right]_{h=1,\, l\in I_{M_\sigma,m}(x_j)}^{N}. \tag{3.11}$$

   Solve the normal equations for $\boldsymbol{K}_j \in \mathbb{C}^{N\times(2m+1)}$, i. e., compute $\tilde{\boldsymbol{b}}_j$, cf. (3.10). $\mathcal{O}(N)$

2. Compose $\tilde{\boldsymbol{B}}^*$ column-wise of the vectors $\tilde{\boldsymbol{b}}_j$ observing the periodicity. $\mathcal{O}(N)$

Output: optimized matrix $\tilde{\boldsymbol{B}}^*$

Complexity: $\mathcal{O}(N^2M)$

It can be seen that this algorithm is of high complexity $\mathcal{O}(N^2M)$, so our next step is to improve these computational costs. The most costly step is the computation of the matrix $\boldsymbol{K}_j$. Thus, we aim to accelerate this process.

We already know from Section 2.3 that sums of the form

$$f_j = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k\,\mathrm{e}^{2\pi\mathrm{i}ky_j}$$

can be computed in $\mathcal{O}(M\log M + N)$ arithmetic operations for given nodes $y_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \dots, N,$ and coefficients $\hat{f}_k \in \mathbb{C}$, $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$, see Algorithm 2.4. If we have a look at the matrix $\boldsymbol{K}_j$ it becomes apparent that we can compute its entries by dint of the NFFT with coefficients

$$\hat{f}_k = \frac{1}{M_\sigma \hat{w}(-k)}, \quad k = -\frac{M}{2}, \dots, \frac{M}{2} - 1, \tag{3.12}$$

and nodes

$$y_{h,l} := x_h - \tfrac{l}{M_\sigma}, \quad h = 1, \dots, N, \; l \in I_{M_\sigma,m}(x_j),$$

which are at most $N(2m+1)$ many. If we put the columns of $\boldsymbol{K}_j$ one below the other into a vector, we are able to compute these entries only using one NFFT of length $N(2m+1)$. In so doing, we have to reshape the obtained vector into a matrix afterwards.

Another point to mention is that the coefficients $\hat{f}_k$ are the same for the computation of all matrices $\boldsymbol{K}_j, j = 1, \dots, N$. This is to say, we can precompute step 1 and step 2 of Algorithm 2.4 since there is no information needed about the corresponding nodes. Only the last step of Algorithm 2.4 has to be performed separately for every $j = 1, \dots, N$. Hence, we receive the following algorithm.

**Algorithm 3.4** (Fast optimization of the matrix $\boldsymbol{B}^*$)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \dots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$. Furthermore, we are given the oversampling factor $\sigma_2 \geq 1$ and the cut-off parameter $m_2$ for an NFFT.

```
1. Compute step 1 and step 2 of Algorithm 2.4 with $\hat{f}_k$ in (3.12).        $\mathcal{O}(M \log M)$
2. For $j = 1, \dots, N$:
      Determine the set $I_{M_\sigma,m}(x_j)$, cf. (3.7).        $\mathcal{O}(1)$
      Perform step 3 of Algorithm 2.4 for the vector of nodes
            $\boldsymbol{y} := \left(y_1^T, \dots, y_s^T\right)^T$
      for $y_n$ being the columns of the matrix $\boldsymbol{Y} := (y_{h,l})_{h=1,\, l \in I_{M_\sigma,m}(x_j)}^N$.        $\mathcal{O}(N)$
      Reshape the obtained vector into the matrix $\boldsymbol{K}_j \in \mathbb{C}^{N \times (2m+1)}$.        $\mathcal{O}(N)$
      Solve the normal equations for $\boldsymbol{K}_j$, i.e., compute $\tilde{\boldsymbol{b}}_j$, cf. (3.10).        $\mathcal{O}(N)$
3. Compose $\tilde{\tilde{\boldsymbol{B}}}^*$ column-wise of the vectors $\tilde{\boldsymbol{b}}_j$ observing the periodicity.        $\mathcal{O}(N)$
```

Output: optimized matrix $\tilde{\tilde{\boldsymbol{B}}}^*$

Complexity: $\mathcal{O}(N^2 + M \log M)$

### Numerical results

Now we have a look at some numerical examples.

**Example 3.5** First of all, we want to verify that the optimization was successful. Therefore, we compare the norms

$$\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}, \tag{3.13}$$

where $\boldsymbol{B}^*$ denotes the original matrix from the adjoint NFFT as well as

$$\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\tilde{\boldsymbol{B}}^* - M\boldsymbol{I}_N\|_{\mathrm{F}} \quad \text{and} \quad \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\tilde{\tilde{\boldsymbol{B}}}^* - M\boldsymbol{I}_N\|_{\mathrm{F}} \tag{3.14}$$

with the optimized matrices $\tilde{\boldsymbol{B}}^*$ and $\tilde{\tilde{\boldsymbol{B}}}^*$ generated by our Algorithms 3.3 and 3.4.

(i) As mentioned in [14, 4, 1] we examine as a first example what happens if we choose so-called jittered equispaced nodes

$$x_j = -\frac{1}{2} + \frac{j-1}{N} + \frac{1}{4N}\theta, \quad j = 1,\dots,N, \text{ with } \theta \sim U(0,1), \tag{3.15}$$

where $U(0,1)$ denotes the uniform distribution on the interval $(0,1)$. Here we choose $N = 128$ and consider the norms (3.13) and (3.14) for $M = 2^c$ with $c = 4,\dots,12$.

In Figure 3.1 one can find the comparison of the norms for B-Splines of different order, cf. (2.8), and for different values of the oversampling factor $\sigma$ for the adjoint NFFT. There it can be seen that the minimization was really successful especially for large values of $M$ compared to $N$. Then it is obvious that the norm of the original matrix is still getting bigger while the norms using the optimized matrices get much smaller. It also becomes apparent that the optimization was less successful in case we have $M < N$ because until the point where these sizes are equal all curves rise and are quite close to each other. In this case our optimization is not able to improve the approximation. But in fact, this is not really surprising because then we try to approximate the identity by a low rank matrix since $\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^* \in \mathbb{C}^{N\times N}$ has at most rank $M$.

This tendency can be seen for all tested B-Spline orders as well as for different oversampling factors for the adjoint NFFT. And also for other window functions like the Gaussian, cf. (2.9), illustrated in Figure 3.2, sinc functions, cf. (2.10), presented in Figure 3.3 and the Kaiser-Bessel window, cf. (2.11), depicted in Figure 3.4 one can see the same behavior of the considered norms. It can also be seen that the norms using the two optimized matrices $\tilde{\boldsymbol{B}}^*$ and $\tilde{\tilde{\boldsymbol{B}}}^*$ are quite indistinguishable.

In Algorithm 3.4 we chose the Kaiser-Bessel window, an oversampling of $\sigma_2 = 2.0$ and a cut-off $m_2$ twice as large as the cut-off parameter $m$ from the adjoint NFFT to calculate the NFFT. With these chosen parameters we are able to receive a high accuracy in computation. But also lower choices for the parameters are possible, see Example 3.6.

(ii) Now we repeat the same example only changing the arrangement of the nodes. What we consider next are the Chebyshev nodes

$$x_j = \frac{1}{2}\cos\left(\frac{2(N-j)+1}{2N}\,\pi\right), \quad j = 1,\dots,N. \tag{3.16}$$

The corresponding results are found in Figure 3.5. Here only the outcome for B-Splines is presented but for the other windows the graphs look almost the same. For these nodes one can see that the optimization problem is much more complicated since the minimization only works if we have $M \gg N$. If we have a look at the same setting from above one can see that for $N = 128$ the optimization was not too effective because the Frobenius norm is still rising for growing $M$. Choosing $N = 16$ instead the norm could again be minimized. However, the gap between $M$ and $N$ has to be huge to get results similar to those from above.

(iii) Again we only change the nodes, so now we choose logarithmically spaced nodes

$$x_j = \left(\frac{6}{5}\right)^{j-N} - \frac{1}{2}, \quad j = 1,\dots,N, \tag{3.17}$$

like suggested in [14]. Figure 3.6 shows the results for B-Splines and again the other window functions are left out because the curves look similar.

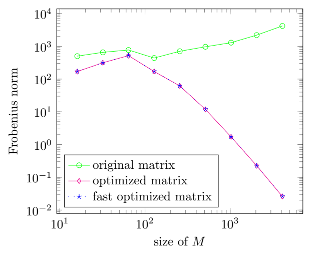
In this case one can also see the same behavior as for Chebyshev nodes, namely that the optimization can only be successful if we have a large difference between $M$ and $N$ but now even worse than for Chebyshev nodes. ◇

**Example 3.6** Next we examine the accuracy of Algorithm 3.4 in comparison to Algorithm 3.3. Since we now have another computation of an NFFT in Algorithm 3.4 in addition to the adjoint NFFT for the inversion we are able to choose parameters for this additional NFFT properly such that the fast computation provides matrices minimizing the norms as well as the slow algorithm does. Here we are free to choose another window function as well as another cut-off $m_2$ and oversampling factor $\sigma_2$.

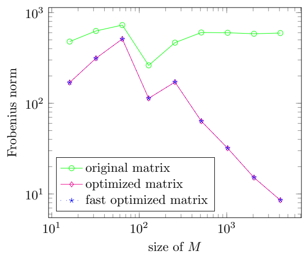
We stick to the norms in (3.14) and consider the relative error

$$\frac{\left|\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\tilde{\boldsymbol{B}}^* - M\boldsymbol{I}_N\|_{\mathrm{F}} - \|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\tilde{\tilde{\boldsymbol{B}}}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}\right|}{\|\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\tilde{\boldsymbol{B}}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}} \tag{3.18}$$

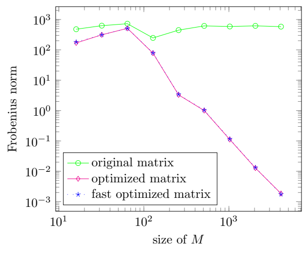
for different sizes of the oversampling factor $\sigma_2$ and growing cut-off parameter $m_2$. We decided for the Kaiser-Bessel function because it is known that this leads to the best results, cf. [11].



(a) $m = 2$ and $\sigma = 1.0$

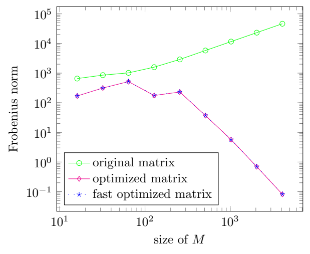


(b) $m = 2$ and $\sigma = 2.0$

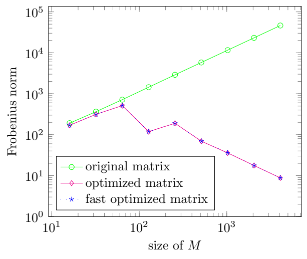


(c) $m = 4$ and $\sigma = 2.0$

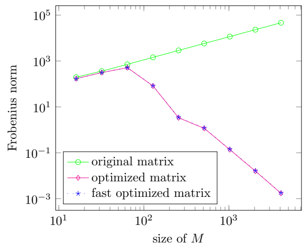
Figure 3.1: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \ldots, 12$ using B-Spline window functions.



(a) $m = 2$ and $\sigma = 1.0$

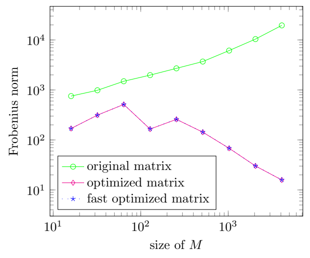


(b) $m = 2$ and $\sigma = 2.0$

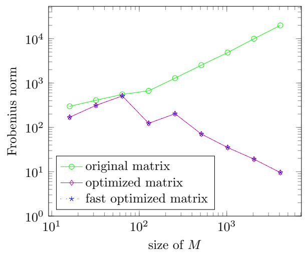


(c) $m = 4$ and $\sigma = 2.0$

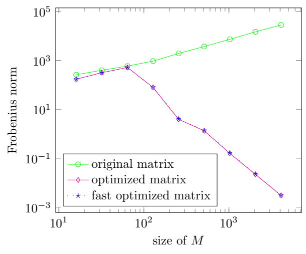
Figure 3.2: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \ldots, 12$ using Gaussian window functions.



(a) $m = 2$ and $\sigma = 1.2$

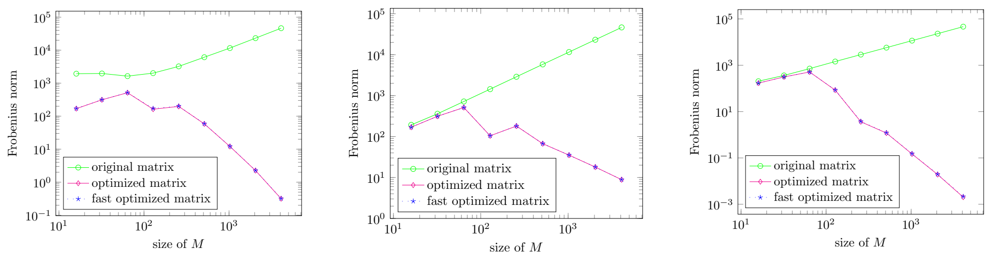


(b) $m = 2$ and $\sigma = 2.0$



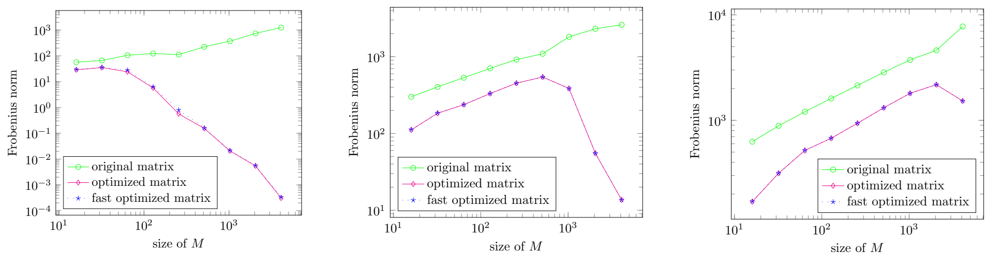
(c) $m = 4$ and $\sigma = 2.0$

Figure 3.3: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \ldots, 12$ using powers of the sinc function.



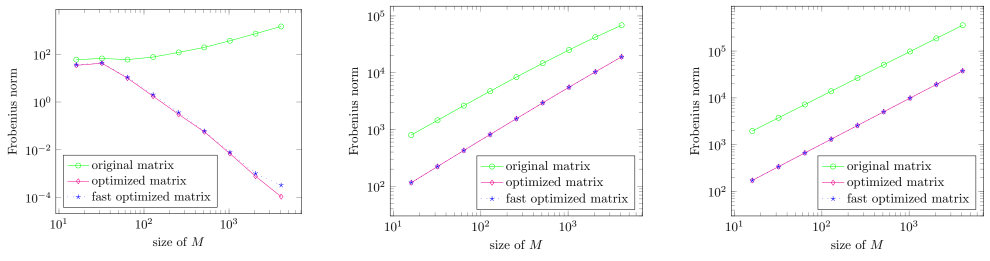
(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.4: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \dots, 12$ using Kaiser-Bessel windows.



(a) $N = 16$ (b) $N = 64$ (c) $N = 128$

Figure 3.5: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for different numbers of Chebyshev nodes for $M = 2^c$ with $c = 4, \dots, 12$ using the B-Spline of order 4 with $\sigma = 1.0$.



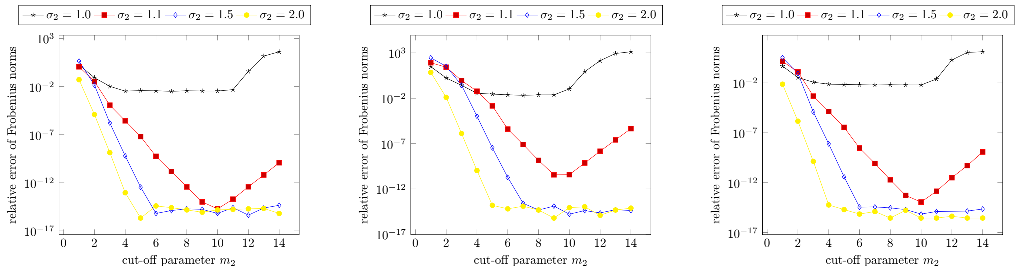
(a) $N = 16$ (b) $N = 64$ (c) $N = 128$

Figure 3.6: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for different numbers of logarithmic nodes for $M = 2^c$ with $c = 4, \dots, 12$ using the B-Spline of order 4 with $\sigma = 1.0$.

Figure 3.7 displays the observed relative errors (3.18) for the inversion with B-Splines and Kaiser-Bessel windows for $N = 128$ jittered equispaced nodes and $M = 256$ Fourier coefficients with $\sigma = 1.0$ and therefore $M_\sigma = M$. We recognize that the error is smallest when we choose the oversampling $\sigma_2$ as high as possible and a cut-off $m_2$ slightly bigger than the double of the cut-off $m$ from the inversion. Whereas it can be seen that the results are never that good if one chooses $\sigma_2 = 1.0$ and get even worse for high cut-off $m_2$. Taking an oversampling factor $\sigma > 1$ for the inversion instead the outcomes are nearly the same.

For the other known window functions, here exemplified for the Kaiser-Bessel window, the behavior is nearly the same except that the best results may be obtained for higher values of the cut-off $m_2$.

For Chebyshev nodes or logarithmically spaced nodes one can receive similar results. This is why this should be left out here. ◇



(a) B-Spline with $m = 2$ (b) B-Spline with $m = 4$ (c) Kaiser-Bessel with $m = 2$

Figure 3.7: Comparison of the relative errors (3.18) for different window functions for the inversion with $\sigma = 1.0$ for $N = 128$ jittered equispaced nodes and $M = 256$ Fourier coefficients for growing cut-off parameter $m_2$ and different oversampling factors $\sigma_2$.

## Further simplification

Regarding Algorithm 3.3 and Algorithm 3.4 we have currently seen that the inversion can be done similarly for every of our known window functions. Therefore, we want to use this fact and simplify the algorithms even more by replacing the window function by the Dirichlet kernel

$$D_{\frac{M}{2}-1}(x) = \sum_{k=-\frac{M}{2}+1}^{\frac{M}{2}-1} \mathrm{e}^{2\pi \mathrm{i} kx} = \frac{\sin((M-1)\pi x)}{\sin(\pi x)}, \tag{3.19}$$

i. e., we set $\hat{w}(k) = 1$ for all $k = -\frac{M}{2}+1, \dots, \frac{M}{2}-1$ and the last nonzero entry $\frac{1}{\hat{w}(\frac{M}{2})}$ shall be set to zero. Hence, the entries of the matrix $\boldsymbol{K}_j$ in (3.11) can explicitly be stated as seen in (3.19) and therefore the term $M \log M$ in the computational costs of Algorithm 3.4 can be eliminated. Thus, we obtain the following algorithm.

**Algorithm 3.7** (Fast optimization of the matrix $\boldsymbol{B}^*$ using the Dirichlet kernel)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \ldots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$.

```
1. For $j = 1, \ldots, N$:
       Determine the set $I_{M_\sigma,m}(x_j)$, cf. (3.7).                         $\mathcal{O}(1)$
       Compute                                                                      $\mathcal{O}(N)$
           $\boldsymbol{K}_j = \left[\frac{1}{M_\sigma} D_{\frac{M}{2}-1}\left(x_h - \frac{l}{M_\sigma}\right)\right]_{h=1,\, l \in I_{M_\sigma,m}(x_j)}^{N}.$
       Solve the normal equations for $\boldsymbol{K}_j$, i. e., compute $\tilde{\boldsymbol{b}}_j$, cf. (3.10).   $\mathcal{O}(N)$
2. Compose $\tilde{\tilde{\boldsymbol{B}}}^*$ column-wise of the vectors $\tilde{\boldsymbol{b}}_j$ observing the periodicity.   $\mathcal{O}(N)$
```

Output: optimized matrix $\tilde{\tilde{\boldsymbol{B}}}^*$

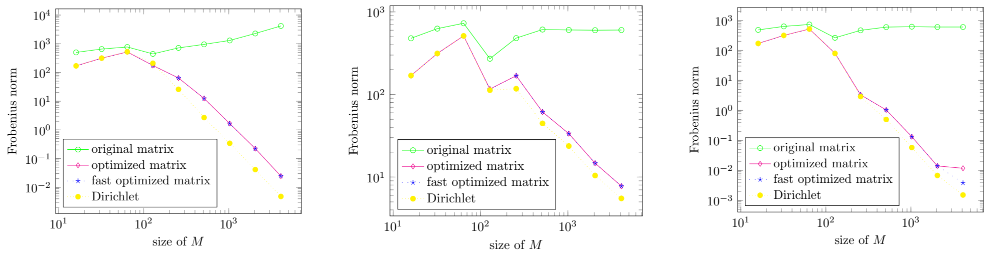
Complexity: $\mathcal{O}(N^2)$

**Example 3.8** Again we do the same experiment as in Example 3.5, now comparing Algorithms 3.3 and 3.4 using B-Spline window functions and our new Algorithm 3.7 with the Dirichlet kernel $D_{\frac{M}{2}-1}(x)$. Figure 3.8 displays the corresponding outcomes. It is obvious that the minimization using the Dirichlet kernel works as well as the original optimization does or even better for large $M$.

Next we have a look at the run-times of our algorithms. Here we choose the B-Spline of order 4 and $\sigma = 1.0$ for the inversion. For the NFFT in Algorithm 3.4 we choose the Kaiser-Bessel window with the double cut-off parameter $m_2 = 4$ and an oversampling of $\sigma_2 = 2.0$.

Having a look at Figure 3.9 it can be seen that the computational costs of Algorithm 3.3 rise rapidly if $M$ grows whereas those of Algorithm 3.4 only increase slowly for growing $M$. Though, for small sizes of $M$ it is obvious that Algorithm 3.3 is even faster than our improved Algorithm 3.4. This arises because many precomputations have to be done in Algorithm 3.4 whereas the original algorithm only evaluates the matrix which can be done quickly if $M$ is small. Hence, the fast Algorithm 3.4 is most effective if we have a large number of Fourier coefficients. Having a look at the run-time of our latest Algorithm 3.7, it can be seen that this step again reduced the run-time considerably since now this algorithm is the fastest for all sizes of $M$.

This results can also be seen for differently spaced nodes as well as for other parameters of the inversion. What can also change the time needed for computation is the number of nodes and the parameters chosen for the fast computation via NFFT. The higher the oversampling or the cut-off parameter are the longer it will take to calculate the optimized matrix. However, this additional tests shall be left out. $\diamond$



(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.8: Comparison of Frobenius norms (3.13) and (3.14) of the original matrix $\boldsymbol{B}$ with the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 using B-Spline window functions as well as the optimized matrix $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.7 using the Dirichlet kernel for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \dots, 12$.

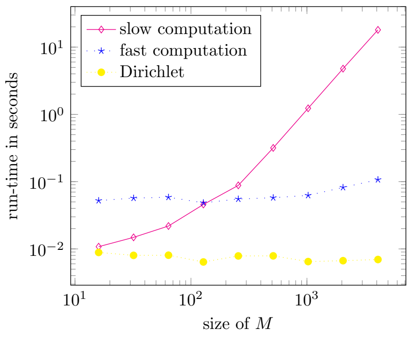


Figure 3.9: Comparison of run-times of Algorithms 3.3 and 3.4 using the B-Spline of order 4 as well as Algorithm 3.7 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \dots, 12$ without oversampling.

**Example 3.9** We have already seen that the minimization of the norm was really successful in the underdetermined case. Now our purpose is to check if also the inverse NFFT, see problem (3.1), can be approximated. Like mentioned in (3.3) the approximation $\frac{1}{M}\boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f} \approx \boldsymbol{f}$ implies that $\hat{\boldsymbol{f}} \approx \check{\boldsymbol{f}} = \frac{1}{M}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f}$. Therefore, we examine the norm $\|\boldsymbol{A}\check{\boldsymbol{f}} - \boldsymbol{f}\|_2$ mentioned in (3.4) for specific $\boldsymbol{f} \in \mathbb{C}^N$. To this end, we choose a trigonometric polynomial

$$f(x) = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \hat{f}_k \, \mathrm{e}^{2\pi \mathrm{i} kx} \tag{3.20}$$

with random coefficients $\hat{f}_k \in (1, 100)$, $k = -\frac{M}{2}, \ldots, \frac{M}{2} - 1$, and compute the vector $\boldsymbol{f} = (f(x_j))_{j=1}^N$ at nonequispaced nodes $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \ldots, N$. Then we consider the approximation

$$\check{\boldsymbol{f}} = \left(\check{f}_k\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} = \frac{1}{M}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f} \tag{3.21}$$

with the original matrix $\boldsymbol{B}^*$ from the adjoint NFFT as well as

$$\check{\check{\boldsymbol{f}}} = \left(\check{\check{f}}_k\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} = \frac{1}{M}\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{O}^*\boldsymbol{f} \tag{3.22}$$

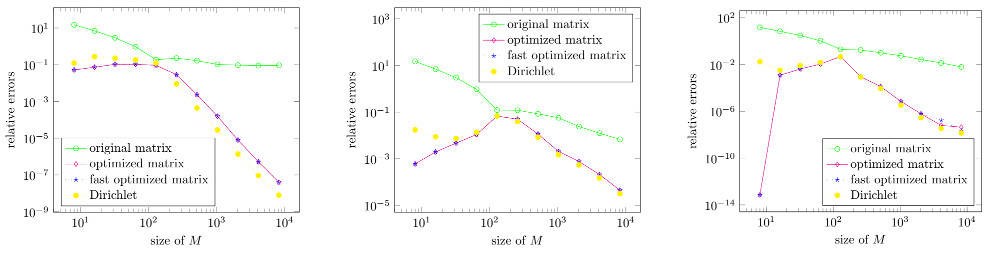
where the matrix $\boldsymbol{O}$ denotes a placeholder for the optimized matrix $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3, $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 and $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.7. Given these approximations we study the corresponding relative errors

$$\frac{\|\boldsymbol{A}\check{\boldsymbol{f}} - \boldsymbol{f}\|_2}{\|\boldsymbol{f}\|_2} \quad \text{and} \quad \frac{\|\boldsymbol{A}\check{\check{\boldsymbol{f}}} - \boldsymbol{f}\|_2}{\|\boldsymbol{f}\|_2}. \tag{3.23}$$

We remark that $\boldsymbol{A}\boldsymbol{A}^* \not\approx M\boldsymbol{I}_N$ for nonequispaced nodes so that we do not expect a small error for $\|\boldsymbol{A}\check{\boldsymbol{f}} - \boldsymbol{f}\|_2$. We only use it for comparative purposes.

For this test we take $N = 128$ nodes and compute the relative errors for $M = 2^c$ with $c = 3, \ldots, 13$. The results for jittered equispaced nodes, see (3.15), using the B-Spline window function are depicted in Figure 3.10. There it can be seen that again the optimization works best if we are in the underdetermined setting $M > N$. Then the errors of all optimized matrices get much smaller than the original ones while the smallest error can be achieved using Algorithm 3.7 with the Dirichlet kernel. If we have $M < N$ instead then our latest algorithm is somewhat worse than Algorithms 3.3 and 3.4 but in this setting the optimization was not quite successful anyway. Another point to mention is that in case we use oversampling, the errors are even worse for high numbers of $M$ while there are improvements for $M < N$.

For the other window functions the outcomes are essentially the same and hence they are omitted here. In case we are given Chebyshev nodes, see (3.16), or logarithmically spaced nodes, see (3.17), we refer to the appropriate results in Example 3.5 since the relative errors show the same descent as the matrix norms mentioned there. ◇



(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.10: Comparison of relative errors (3.23) using the original matrix $\boldsymbol{B}$, the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 using B-Spline window functions as well as the optimized matrix $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.7 using the Dirichlet kernel for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 3, \dots, 13$.

**Example 3.10** Finally, we figure out if this approach allows us to perform an inverse NFFT for a given function $f$. To this end, we choose the trigonometric function

$$f(x) = \cos^2(\pi x^2)\,\sin(10x^2), \quad x \in \left[-\tfrac{1}{2}, \tfrac{1}{2}\right), \tag{3.24}$$

like suggested in [14]. Now we aim to approximate the Fourier coefficients $c_k(f)$. For this purpose, we consider the function

$$g(x) := \int_{-\frac{1}{2}}^{\frac{1}{2}} f(y)\,\tilde{w}(y-x)\,\mathrm{d}y. \tag{3.25}$$

Hence, by means of the convolution operator we can write $g = f * \tilde{w}(-\bullet)$. Then the convolution theorem implies $c_k(g) = c_k(f)\,c_{-k}(\tilde{w})$ such that we have $c_k(f) = \frac{c_k(g)}{c_{-k}(\tilde{w})}$. If we now suppose we are not given the function $g$ but only evaluations at points $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, we can use a quadrature rule with weights $\frac{1}{M}$ so that we obtain the approximation

$$g(x) \approx \frac{1}{M} \sum_{j=1}^{N} f(x_j)\,\tilde{w}(x_j - x) = \frac{1}{M}\,\tilde{g}(x) \tag{3.26}$$

with the function $\tilde{g}(x)$ as in (2.12). Then we can approximate $c_k(g)$ by $\frac{1}{M} c_k(\tilde{g})$ and therefore we have the approximation

$$c_k(f) = \frac{c_k(g)}{c_{-k}(\tilde{w})} \approx \frac{c_k(\tilde{g})}{M \cdot c_{-k}(\tilde{w})} \overset{(2.13)}{=} \frac{1}{M}\tilde{h}_k = \left(\tfrac{1}{M}\,\boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f}\right)_k \overset{(3.21)}{=} \check{f}_k.$$

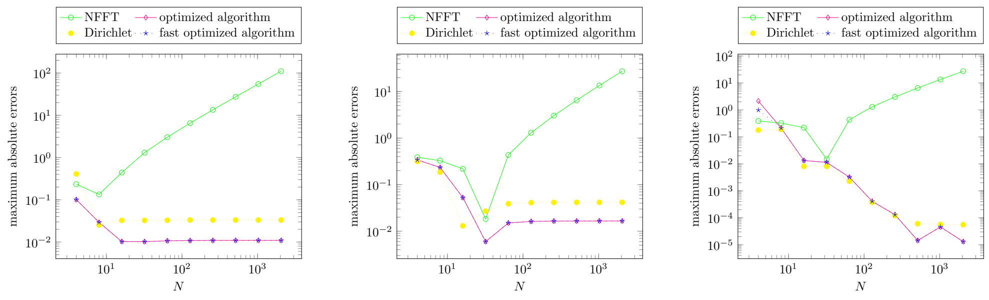
Thus, we consider for $\boldsymbol{f} = (f(x_j))_{j=1}^N$ the approximations $\check{\boldsymbol{f}}$ and $\check{\check{\boldsymbol{f}}}$ of the Fourier coefficients, see (3.21) and (3.22), and compare them to the exact Fourier coefficients of $f$ which we can compute analytically via the formula

$$c_k(f) := \int_{-\frac{1}{2}}^{\frac{1}{2}} f(x)\,\mathrm{e}^{-2\pi \mathrm{i} kx}\,\mathrm{d}x, \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1.$$

For this example we reconstruct $M = 32$ Fourier coefficients from function values $f(x_j)$ given for jittered equispaced nodes $x_j$ once in an overdetermined and once in an underdetermined setting, namely $N = 128$ and $N = 8$. To this end, we choose the B-Spline window of order 4 as well as the Dirichlet kernel without oversampling for computing the approximations.

Figure 3.12 depicts on the one hand side the reconstruction, i. e., the exact Fourier coefficients $c_k(f)$ compared to the computed approximations $\check{f}_k$ and $\check{\check{f}}_k$, and on the other hand side the pointwise errors $|c_k(f) - \check{f}_k|$ as well as $|c_k(f) - \check{\check{f}}_k|$, $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$, for each of the developed algorithms. There it can be seen that the algorithms work best in the overdetermined case, namely $M < N$. But since we could minimize the norm the most in the underdetermined case we also managed computing a better approximation for $M > N$. The approximations could further be improved for $M < N$ by making use of a higher oversampling or larger cut-off whereas it may be worsened for $M > N$. This can be seen in Figure 3.11. There the maximum absolute errors $\max_{k=-\frac{N}{2},\dots,\frac{N}{2}-1} |c_k(f) - \check{f}_k|$ and $\max_{k=-\frac{N}{2},\dots,\frac{N}{2}-1} |c_k(f) - \check{\check{f}}_k|$ are displayed for all the algorithms using different parameters.

When using other window functions or considering Chebyshev nodes and logarithmically spaced nodes the results look quite the same. Hence, all these tests shall be omitted here. ◇

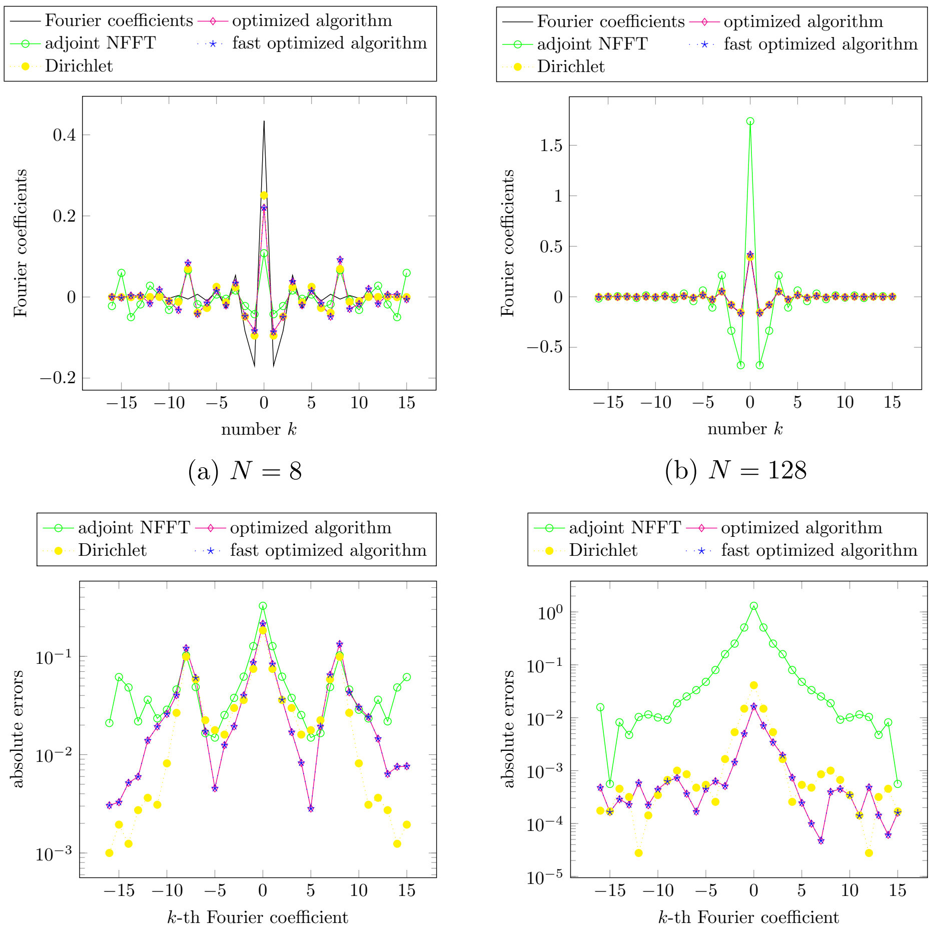


(a) $M = 8, m = 2$ and $\sigma = 1.0$

(b) $M = 32, m = 2$ and $\sigma = 1.0$

(c) $M = 32, m = 4$ and $\sigma = 2.0$

Figure 3.11: Comparison of maximum absolute errors of Algorithms 2.5, 3.3 and 3.4 using the B-Spline window as well as Algorithm 3.7 using the Dirichlet kernel for jittered equispaced nodes and $N = 2^c$ with $c = 2, \dots, 11$.



(a) $N = 8$ (b) $N = 128$

(c) $N = 8$ (d) $N = 128$

Figure 3.12: Reconstruction of the Fourier coefficients (top) and pointwise errors (bottom) for $M = 32$ Fourier coefficients, generated by Algorithms 2.5, 3.3 and 3.4 using the B-Spline of order 4 as well as Algorithm 3.7 using the Dirichlet kernel without oversampling for jittered equispaced nodes.

## 3.3 Inverse adjoint NFFT - overdetermined case

Now we want to solve the problem (3.2), so we search an approximation of the form $\boldsymbol{BK}^* = \boldsymbol{BFDA}^* \approx M\boldsymbol{I}_N$. Therefore, we consider the optimization problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}\,:\,\boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{BFDA}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2.$$

This is equivalent to the transposed problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|(\boldsymbol{BFDA}^*)^* - M\boldsymbol{I}_N^*\|_{\mathrm{F}}^2,$$

i. e., to

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{AD}^*\boldsymbol{F}^*\boldsymbol{B}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}^2.$$

However, this is exactly the problem we already discussed in Section 3.2 and hence could be solved likewise.

## Numerical results

Here some examples analogous to those mentioned above are discussed.

**Example 3.11** First of all, we check if the optimization was successful, cf. Example 3.5. Therefore, we examine the norms

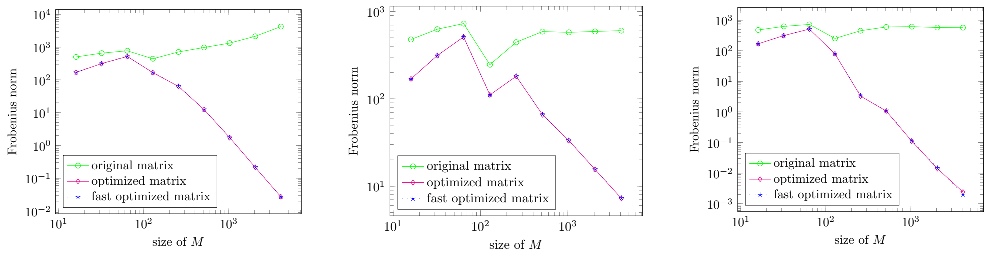
$$\|\boldsymbol{BFDA}^* - M\boldsymbol{I}_N\|_{\mathrm{F}} \tag{3.27}$$

as well as

$$\|\tilde{\boldsymbol{B}}\boldsymbol{FDA}^* - M\boldsymbol{I}_N\|_{\mathrm{F}} \quad \text{and} \quad \|\tilde{\tilde{\boldsymbol{B}}}\boldsymbol{FDA}^* - M\boldsymbol{I}_N\|_{\mathrm{F}}, \tag{3.28}$$

where $\boldsymbol{B}$ denotes the original matrix of the NFFT and $\tilde{\boldsymbol{B}}, \tilde{\tilde{\boldsymbol{B}}}$ denote the adjoint of the matrices computed by Algorithm 3.3 and Algorithm 3.4, respectively.

In Figure 3.13 the outcome for $N = 128$ jittered equispaced nodes using B-Spline window functions is displayed. Like already mentioned above, this optimization problem is the same as in Section 3.2 and therefore these pictures are exactly the same as in Figure 3.1. Hence, we refer to Example 3.5 for further results. ◇



(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.13: Comparison of Frobenius norms (3.27) and (3.28) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 4, \ldots, 12$ using B-Spline window functions.

**Example 3.12** Analogously to Example 3.9, we test if the inverse adjoint NFFT in (3.2) can successfully be computed. Thus, we also choose a trigonometric polynomial with random coefficients $\hat{f}_k \in (1, 100)$, $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$, and determine the vector $\boldsymbol{f} = (f(x_j))_{j=1}^N$ for $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$. We know that $\boldsymbol{BFDA}^* \approx M\boldsymbol{I}_N$ implies $\frac{1}{M}\boldsymbol{BFD}(\boldsymbol{A}^*\boldsymbol{f}) \approx \boldsymbol{f}$ so that we consider the approximations

$$\tilde{\boldsymbol{f}} = \left(\tilde{f}_j\right)_{j=1}^N = \frac{1}{M}\boldsymbol{BFD}(\boldsymbol{A}^*\boldsymbol{f}) \tag{3.29}$$

with the original matrix $\boldsymbol{B}$ from the NFFT as well as

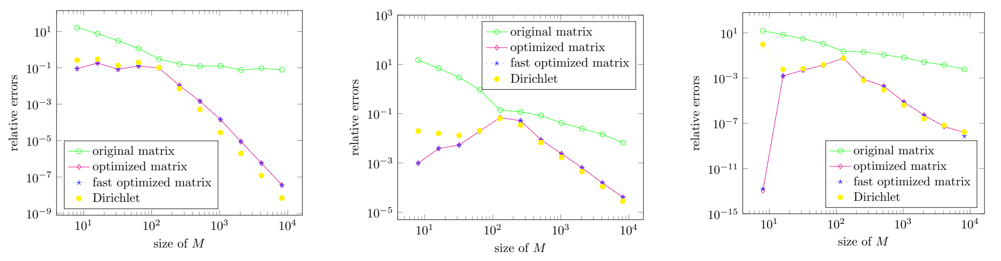
$$\tilde{\tilde{\boldsymbol{f}}} = \left(\tilde{\tilde{f}}_j\right)_{j=1}^N = \frac{1}{M}\boldsymbol{OFD}(\boldsymbol{A}^*\boldsymbol{f}), \tag{3.30}$$

where $\boldsymbol{O}$ is a placeholder for the optimized matrix $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3, $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 and $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.7. Given these approximations we study the corresponding relative errors

$$\frac{\|\tilde{\boldsymbol{f}} - \boldsymbol{f}\|_2}{\|\boldsymbol{f}\|_2} \quad \text{and} \quad \frac{\|\tilde{\tilde{\boldsymbol{f}}} - \boldsymbol{f}\|_2}{\|\boldsymbol{f}\|_2}. \tag{3.31}$$

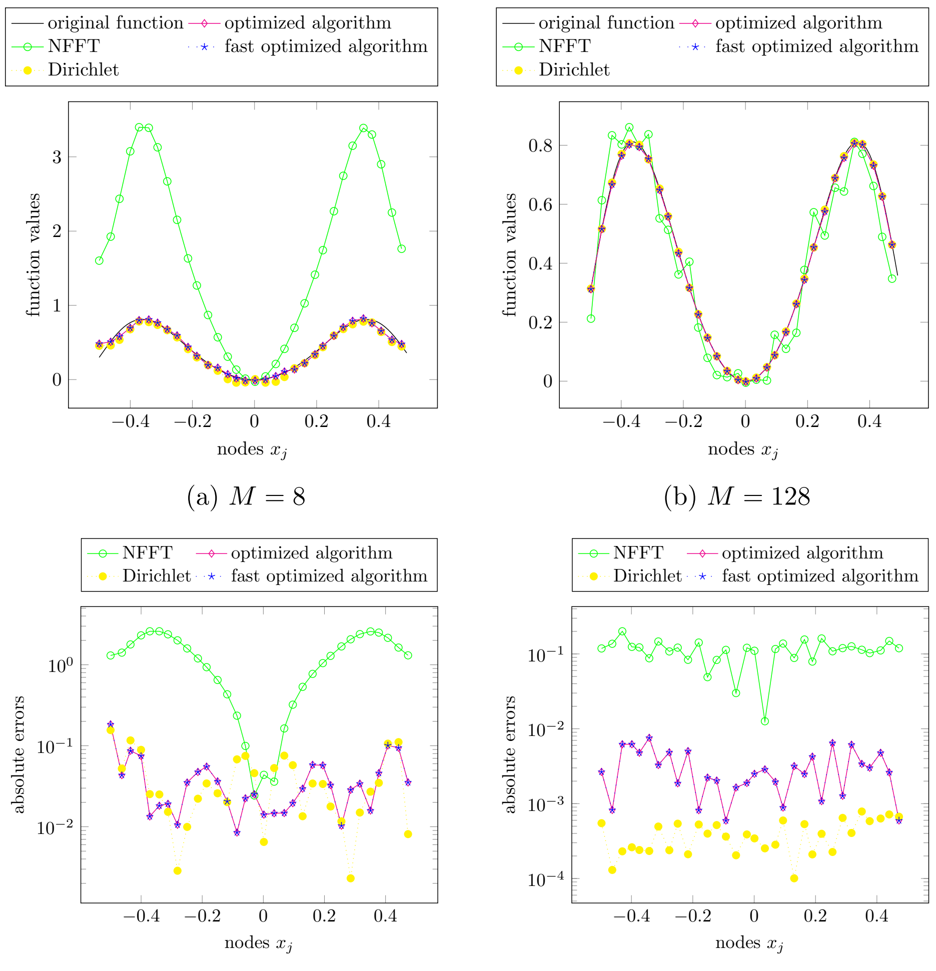
Like in Example 3.9 we choose $N = 128$ and $M = 2^c$ with $c = 3, \dots, 13$.

The results for jittered equispaced nodes using the B-Spline window function can be found in Figure 3.14 where almost the same behavior as in Example 3.9 can be seen. Again the graphs for other window functions or nodes are left out. ◇



(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.14: Comparison of relative errors (3.31) using the original matrix $\boldsymbol{B}$, the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.3 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.4 using B-Spline window functions as well as the optimized matrix $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.7 using the Dirichlet kernel for $N = 128$ jittered equispaced nodes and $M = 2^c$ with $c = 3, \dots, 13$.



(a) $M = 8$ (b) $M = 128$

(c) $M = 8$ (d) $M = 128$

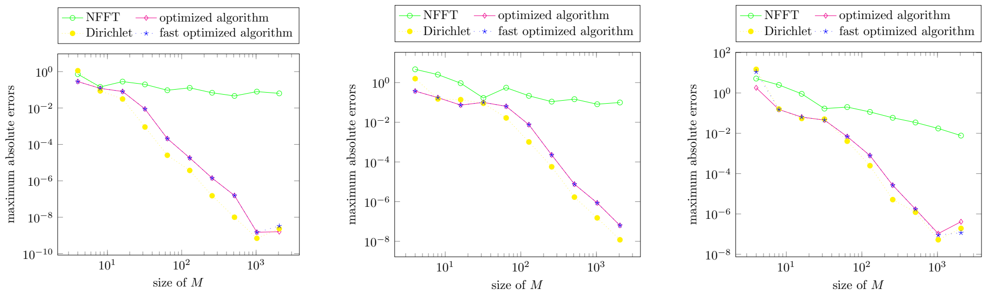
Figure 3.15: Reconstruction of function values (top) and pointwise errors (bottom) for $N = 32$ nodes, generated by Algorithms 2.4, 3.3 and 3.4 using the B-Spline of order 4 as well as Algorithm 3.7 using the Dirichlet kernel without oversampling for jittered equispaced nodes.

**Example 3.13** Like in Example 3.10 we are going to check if we are able to perform an inverse adjoint NFFT for the trigonometric function (3.24). In this case we consider the approximations (3.29) and (3.30) for $\boldsymbol{f} = (f(x_j))_{j=1}^{N}$ and compare them to the function values for $N = 32$ jittered equispaced nodes $x_j$. Similar to Example 3.10 we consider the reconstruction, in this case the approximations $\tilde{\boldsymbol{f}}$ and $\tilde{\tilde{\boldsymbol{f}}}$ compared to the real function values $\boldsymbol{f}$ as well as the pointwise errors $|\tilde{f}_j - f(x_j)|$

and $|\tilde{\tilde{f}}_j - f(x_j)|$, $j = 1, \dots, N$.

The corresponding results can be found in Figure 3.15. There we see that our new algorithms lead to better approximations in both, the underdetermined and the overdetermined case but obviously they work best in the setting $M > N$. This is to be seen in Figure 3.16. There the maximum absolute errors $\max_{j=1,\dots,N} |f_j - \tilde{f}_j|$ and $\max_{j=1,\dots,N} |f_j - \tilde{\tilde{f}}_j|$ are displayed for fixed $N$ and growing $M$. It becomes apparent that the errors can be minimized for having a large gap between $M$ and $N$.

Further results are left out for the same reasons as in Example 3.10. ◇



(a) $N = 8, m = 2$ and $\sigma = 1.0$

(b) $N = 32, m = 2$ and $\sigma = 1.0$

(c) $N = 32, m = 4$ and $\sigma = 2.0$

Figure 3.16: Comparison of maximum absolute errors of Algorithms 2.4, 3.3 and 3.4 using the B-Spline window as well as Algorithm 3.7 using the Dirichlet kernel for jittered equispaced nodes and $M = 2^c$ with $c = 2, \dots, 11$.

## 3.4 Inverse NFFT - overdetermined case

Previously, we studied $\boldsymbol{K}\boldsymbol{B}^*$ and $\boldsymbol{B}\boldsymbol{K}^*$, where

$$\boldsymbol{K} = \boldsymbol{A}\boldsymbol{D}^*\boldsymbol{F}^* \in \mathbb{C}^{N \times M_\sigma} \quad \text{and} \quad \boldsymbol{B} \in \mathbb{R}^{N \times M_\sigma}.$$

Considering the problem $\boldsymbol{A}\hat{\boldsymbol{f}} = \boldsymbol{f}$ for given $\boldsymbol{f} \in \mathbb{C}^N$, mentioned in (3.1), we already introduced algorithms for the underdetermined case $M > N$ in Section 3.2. However, often we are given $N > M$ nonequispaced samples and search a corresponding trigonometric polynomial of degree $M$. Hence, we look for another approach and therefore investigate $\boldsymbol{B}^*\boldsymbol{K}$ next. Initially, we consider the function

$$\tilde{g}(x) = \sum_{j=1}^{N} f_j \, \tilde{w}_m(x_j - x).$$

Evaluating $\tilde{g}(x)$ at equispaced points $x = \frac{l}{M_\sigma}$, $l = -\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1$, yields

$$\tilde{g}\Big(\tfrac{l}{M_\sigma}\Big) = \sum_{j=1}^{N} f_j \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) \,. \tag{3.32}$$

That means, by defining the vector

$$\tilde{\boldsymbol{g}} \coloneqq \Big(\tilde{g}\Big(\tfrac{l}{M_\sigma}\Big)\Big)_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} ,$$

(3.32) can be expressed in matrix-vector notation as $\tilde{\boldsymbol{g}} = \boldsymbol{B}^* \boldsymbol{f}$. Furthermore, we know from (2.13) that the approximation of the adjoint NFFT can be written as

$$\begin{aligned}
\tilde{h}_k &= \frac{1}{M_\sigma \hat{w}(-k)} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \sum_{j=1}^{N} f_j \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) \, \mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma} \\
&\overset{(3.32)}{=} \frac{1}{M_\sigma \hat{w}(-k)} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \tilde{g}\Big(\tfrac{l}{M_\sigma}\Big) \, \mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma}, \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1,
\end{aligned} \tag{3.33}$$

and thereby we have $(\tilde{h}_k)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} =: \tilde{\boldsymbol{h}} = \boldsymbol{D}^* \boldsymbol{F}^* \tilde{\boldsymbol{g}}$. Now we claim $\tilde{\boldsymbol{h}} \overset{!}{\approx} \hat{\boldsymbol{f}}$. Thus, it follows

$$\tilde{\boldsymbol{g}} \overset{(3.32)}{=} \boldsymbol{B}^* \boldsymbol{f} = \boldsymbol{B}^* \boldsymbol{A} \hat{\boldsymbol{f}} \overset{!}{\approx} \boldsymbol{B}^* \boldsymbol{A} \tilde{\boldsymbol{h}} \overset{(3.33)}{=} \boldsymbol{B}^* \boldsymbol{A} \boldsymbol{D}^* \boldsymbol{F}^* \tilde{\boldsymbol{g}},$$

i. e., we seek $\boldsymbol{B}^*$ as solution of the optimization problem

$$\underset{\boldsymbol{B} \in \mathbb{R}^{N \times M_\sigma} :\, \boldsymbol{B}\, (2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{B}^* \boldsymbol{A} \boldsymbol{D}^* \boldsymbol{F}^* - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 = \|\boldsymbol{B}^* \boldsymbol{K} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 \,.$$

This is equivalent to the transposed problem

$$\underset{\boldsymbol{B} \in \mathbb{R}^{N \times M_\sigma} :\, \boldsymbol{B}\, (2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{K}^* \boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 = \|\boldsymbol{F} \boldsymbol{D} \boldsymbol{A}^* \boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 \,. \tag{3.34}$$

By means of definitions (3.5) and (2.15) we obtain

$$\boldsymbol{F} \boldsymbol{D} \boldsymbol{A}^* \boldsymbol{B} = \left[ \sum_{j=1}^{N} \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)} \, \mathrm{e}^{-2\pi \mathrm{i} k \left(x_j - \frac{s}{M_\sigma}\right)} \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) \right]_{s,\, l = -\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} . \tag{3.35}$$

Analogously to (3.7), we define the set

$$I_{M_\sigma, m}(l) \coloneqq \{ j \in \{1, \dots, N\} : \exists\, z \in \mathbb{Z} \text{ with} - m \le M_\sigma x_j - l + M_\sigma z \le m \} \,. \tag{3.36}$$

Hence, we can rewrite the optimization problem (3.34) by analogy with Section 3.2 as

$$\|\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 = \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \|\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l\boldsymbol{b}_l - \boldsymbol{e}_l\|_2^2,$$

where

$$\boldsymbol{b}_l := \left(\tilde{w}_m\left(x_j - \tfrac{l}{M_\sigma}\right)\right)_{j\in I_{M_\sigma,m}(l)}, \quad \boldsymbol{H}_l := \left(\mathrm{e}^{-2\pi\mathrm{i}kx_j}\right)_{k=-\frac{M}{2},\, j\in I_{M_\sigma,m}(l)}^{\frac{M}{2}-1}$$

and $\boldsymbol{e}_l$ denote the columns of the identity matrix $\boldsymbol{I}_{M_\sigma}$. Thereby we receive the optimization problems

$$\underset{\tilde{\boldsymbol{b}}_l\in\mathbb{R}^{2m+1}}{\text{Minimize}} \ \|\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l\tilde{\boldsymbol{b}}_l - \boldsymbol{e}_l\|_2^2, \quad l = -\tfrac{M_\sigma}{2}, \dots, \tfrac{M_\sigma}{2} - 1.$$

If the matrix $\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l \in \mathbb{C}^{M_\sigma\times|I_{M_\sigma,m}(l)|}$ has full rank the solution of the problem (3.34) is given by

$$\tilde{\boldsymbol{b}}_l = \left((\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l)^*\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l\right)^{-1}(\boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l)^*\boldsymbol{e}_l, \quad l = -\tfrac{M_\sigma}{2}, \dots, \tfrac{M_\sigma}{2} - 1. \tag{3.37}$$

In so doing, we obtain an approximation of the Fourier coefficients by

$$\hat{\boldsymbol{f}} \approx \tilde{\boldsymbol{h}} \overset{(3.33)}{=} \boldsymbol{D}^*\boldsymbol{F}^*\tilde{\boldsymbol{g}} \overset{(3.32)}{=} \boldsymbol{D}^*\boldsymbol{F}^*\boldsymbol{B}^*\boldsymbol{f}. \tag{3.38}$$

In other words, this approach yields another way to invert the NFFT by also modifying the adjoint NFFT. Thus, we obtain Algorithm 3.15.

*Remark* 3.14 This approach could also be deduced from the explanations in Section 3.1 if we suppose $\sigma = 1.0$ and therefore $M_\sigma = M$. Here we consider $\boldsymbol{A}^*\boldsymbol{A}$ instead of $\boldsymbol{A}\boldsymbol{A}^*$, cf. Section 3.2. Again we want to find a generalization of this approximation by modifying the matrix $\boldsymbol{B}$, i. e., we look for an approximation of the form

$$\boldsymbol{A}^*\boldsymbol{B}\boldsymbol{F}\boldsymbol{D} \approx \boldsymbol{I}_M.$$

Multiplying right-hand by the inverse of $\boldsymbol{D}$ yields

$$\boldsymbol{A}^*\boldsymbol{B}\boldsymbol{F} \approx \boldsymbol{D}^{-1}.$$

Since $\boldsymbol{F} \in \mathbb{C}^{M_\sigma\times M}$ is quadratic in this setting we can multiply right-hand by the inverse of $\boldsymbol{F}$ such that

$$\boldsymbol{A}^*\boldsymbol{B} \approx \boldsymbol{D}^{-1}\boldsymbol{F}^{-1}.$$

Now we can multiply left-hand by $\boldsymbol{D}$ and afterwards left-hand by $\boldsymbol{F}$ so that we receive

$$\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{B} \approx \boldsymbol{I}_{M_\sigma},$$

which leads to the optimization problem (3.34) as well. ◇

**Algorithm 3.15** (Optimization of the matrix $\boldsymbol{B}$)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \ldots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$. Furthermore, we are given the matrices $\boldsymbol{A}^*, \boldsymbol{D}$ and $\boldsymbol{F}$.

```
1. For $l = -\frac{M_\sigma}{2}, \ldots, \frac{M_\sigma}{2} - 1$:
     Determine the set $I_{M_\sigma,m}(l)$, cf. (3.36).                                   $\mathcal{O}(N)$
     Determine the matrix $\boldsymbol{H}_l := \left(\mathrm{e}^{-2\pi\mathrm{i}kx_j}\right)_{k=-\frac{M}{2},\, j\in I_{M_\sigma,m}(l)}^{\frac{M}{2}-1}$.     $\mathcal{O}(N)$
     Compute                                                                              $\mathcal{O}(M^2N)$
       $$\boldsymbol{K}_l^* =: \boldsymbol{F}\boldsymbol{D}\boldsymbol{H}_l = \left[\frac{1}{M_\sigma}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\frac{1}{\hat{w}(k)}\,\mathrm{e}^{2\pi\mathrm{i}k\left(\frac{l}{M_\sigma}-x_j\right)}\right]_{l=-\frac{M_\sigma}{2},\, j\in I_{M_\sigma,m}(l)}^{\frac{M_\sigma}{2}-1}. \quad (3.39)$$
     Solve the normal equations for $\boldsymbol{K}_l^* \in \mathbb{C}^{M_\sigma\times|I_{M_\sigma,m}(l)|}$, i. e.,
     compute $\tilde{\boldsymbol{b}}_l$, cf. (3.37).                                       $\mathcal{O}(N^3 + N^2M)$
2. Compose $\tilde{\boldsymbol{B}}$ column-wise of the vectors $\tilde{\boldsymbol{b}}_l$ observing the periodicity.   $\mathcal{O}(M)$
```

Output: optimized matrix $\tilde{\boldsymbol{B}}$

Complexity: $\mathcal{O}(M^3N + N^2M^2 + N^3M)$

*Remark* 3.16 Here we cannot tell anything about the dimensions of $\boldsymbol{K}_l^*$ in general since the size of the set $I_{M_\sigma,m}(l)$ depends on several parameters. Without further information we have to assume that this size will be depending on $N$.

As soon as the parameters are given one can calculate the dimensions of $\boldsymbol{K}_l^*$. For example for jittered equispaced nodes it can be seen that the size of $I_{M_\sigma,m}(l)$ fulfills the relation $|I_{M_\sigma,m}(l)| = \frac{2mN}{M_\sigma}$. But also for other nodes we recognized that this size rises with growing $m$ and decreases for bigger $M_\sigma$. Assuming we are given jittered equispaced nodes, the complexity related to $N$ can be eliminated and we end up with arithmetic costs of $\mathcal{O}(M^3)$. ◇

We also wish to accelerate this algorithm like shown in Section 3.2. The problem is that in general the most costly step is not the computation of the matrix $\boldsymbol{K}_l^*$ but solving the normal equations. This arises since the matrix $\boldsymbol{K}_l\boldsymbol{K}_l^*$ is not that small anymore. But considering Remark 3.16 we can assume that there are settings where the matrix dimensions also get small so that it is reasonable to think about a faster computation of the matrix $\boldsymbol{K}_l^*$.

Analogously to Section 3.2, we are able to calculate the entries of the matrix $\boldsymbol{K}_l^*$, see (3.39), by dint of an NFFT with the $M$ coefficients

$$\hat{f}_k = \frac{1}{M_\sigma \hat{w}(k)}, \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1, \tag{3.40}$$

and the nodes

$$y_{l,j} := \frac{l}{M_\sigma} - x_j, \quad l = -\tfrac{M_\sigma}{2}, \dots, \tfrac{M_\sigma}{2} - 1, \; j \in I_{M_\sigma,m}(l),$$

which are at most $M_\sigma N$ many. Here we also compute only one NFFT by writing the columns of $\boldsymbol{K}_l^*$ one below the other. Therefore, we obtain a vector including all entries of $\boldsymbol{K}_l^*$ which we have to reshape afterwards. This leads to the following algorithm.

**Algorithm 3.17** (Fast optimization of the matrix $\boldsymbol{B}$)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \dots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$.

1. Compute step 1 and step 2 of Algorithm 2.4 with $\hat{f}_k$ in (3.40). $\mathcal{O}(M \log M)$
2. For $l = -\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1$:

   Determine the set $I_{M_\sigma,m}(l)$, cf. (3.36). $\mathcal{O}(N)$

   Perform step 3 of Algorithm 2.4 for the vector of nodes

   $$\boldsymbol{y} := \left(y_1^T, \dots, y_s^T\right)^T$$

   for $y_n$ being the columns of the matrix $\boldsymbol{Y} := (y_{l,j})_{l=-\frac{M_\sigma}{2},\, j \in I_{M_\sigma,m}(l)}^{\frac{M_\sigma}{2}-1}$. $\mathcal{O}(N)$

   Reshape the obtained vector into the matrix $\boldsymbol{K}_l^* \in \mathbb{C}^{M_\sigma \times |I_{M_\sigma,m}(l)|}$. $\mathcal{O}(N)$

   Solve the normal equations for $\boldsymbol{K}_l^*$, i. e., compute $\tilde{\boldsymbol{b}}_l$, cf. (3.37). $\mathcal{O}(N^3 + N^2 M)$
3. Compose $\tilde{\tilde{\boldsymbol{B}}}$ column-wise of the vectors $\tilde{\boldsymbol{b}}_l$ observing the periodicity. $\mathcal{O}(M)$

Output: optimized matrix $\tilde{\tilde{\boldsymbol{B}}}$

Complexity: $\mathcal{O}(M \log M + N^2 M^2 + N^3 M)$

*Remark* 3.18 If we assume again we are given nodes which are somewhat uniformly distributed, like for instance jittered equispaced nodes, we can get rid of the complexity related to $N$ and end up with arithmetic costs of $\mathcal{O}(M^2)$. ◇

**Example 3.19** Like in Example 3.5 we want to verify at first that the optimization was successful. On that account, we compare the norms

$$\|\boldsymbol{FDA}^*\boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}} \tag{3.41}$$

where $\boldsymbol{B}$ denotes the original matrix from the NFFT as well as

$$\|\boldsymbol{FDA}^*\tilde{\boldsymbol{B}} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}} \quad \text{and} \quad \|\boldsymbol{FDA}^*\tilde{\tilde{\boldsymbol{B}}} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}} \tag{3.42}$$

with the optimized matrices $\tilde{\boldsymbol{B}}$ and $\tilde{\tilde{\boldsymbol{B}}}$ generated by our Algorithms 3.15 and 3.17.

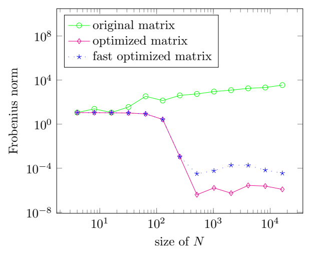
(i) Again we examine at first what happens for jittered equispaced nodes, see (3.15). Here we choose $M = 128$ and consider the norms (3.41) and (3.42) for $N = 2^c$ nodes with $c = 2, \dots, 14$.

    In Figure 3.17 one can find the comparison of the norms for B-Splines of different order, cf. (2.8), and for different oversampling for the inversion. Without oversampling it can be seen that the minimization was very successful especially for large values of $N$ compared to $M$. There it is obvious that the norm of the original matrix is still rising while the norms using the optimized matrices decrease. For the relation $N < M$ we have somewhat the same result as in Example 3.5, namely that the minimization was not that successful in this setting. Like already discussed before this results from the fact that the considered matrix $\boldsymbol{FDA}^*\boldsymbol{B}$ is of low rank. Having a look at the graphs with high oversampling we recognize that the norms of the optimized matrices remain stable for all sizes of $N$. Furthermore, in that case the two lines of the optimized algorithms are indistinguishable.

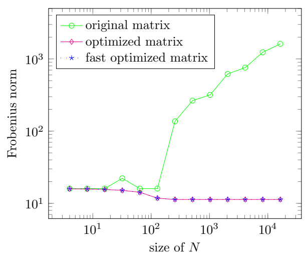
    Also for other window functions like the Gaussian, cf. (2.9), illustrated in Figure 3.18, sinc functions, cf. (2.10), presented in Figure 3.19 and the Kaiser-Bessel window, cf. (2.11), depicted in Figure 3.20 one can see the same behavior of the considered norms. Only one big difference can be seen for the sinc function with the parameter choice $m = 2$ and $\sigma = 1.2$. In this case the minimization does not work as well as for the other window functions since the norm does not get smaller.

    In order to calculate the NFFT in Algorithm 3.17 we chose the Kaiser-Bessel window, an oversampling of $\sigma_2 = 2.0$ and $m_2$ twice as large as the cut-off parameter $m$ from the inversion to calculate the NFFT in Algorithm 3.17 to achieve results comparable to Example 3.5. But one could also choose a larger cut-off to receive more accuracy for growing $N$, see Example 3.20.

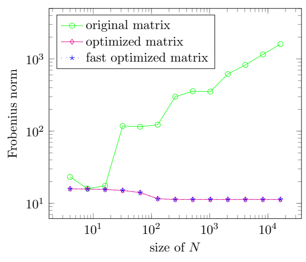
(ii) Next we repeat the same example using Chebyshev nodes, cf. (3.16). The corresponding results can be found in Figure 3.21. It can be seen that these graphs look nearly the same as for jittered equispaced nodes.



(a) $m = 2$ and $\sigma = 1.0$

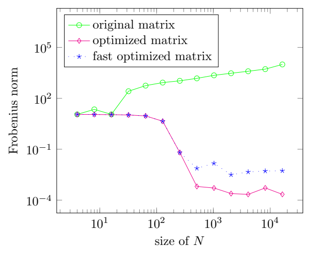


(b) $m = 2$ and $\sigma = 2.0$

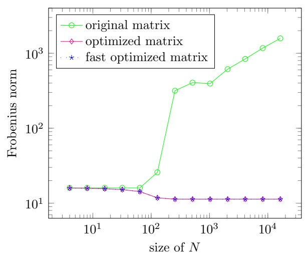


(c) $m = 4$ and $\sigma = 2.0$

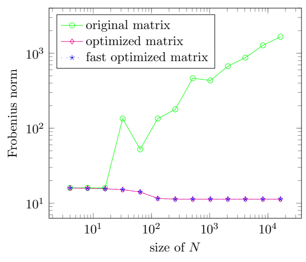
Figure 3.17: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \ldots, 14$ using B-Spline window functions.



(a) $m = 2$ and $\sigma = 1.0$

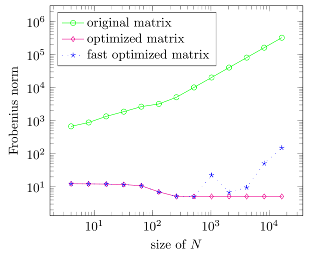


(b) $m = 2$ and $\sigma = 2.0$

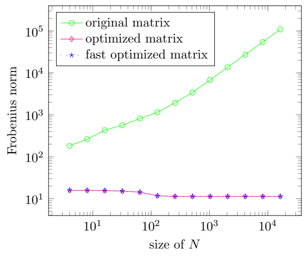


(c) $m = 4$ and $\sigma = 2.0$

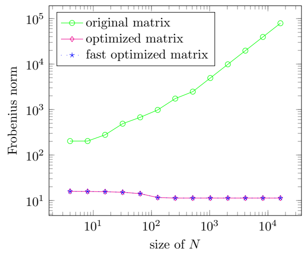
Figure 3.18: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \ldots, 14$ using Gaussian window functions.



(a) $m = 2$ and $\sigma = 1.2$

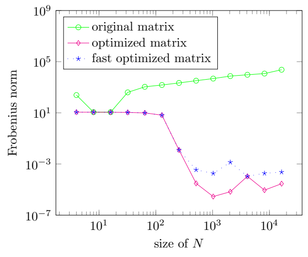


(b) $m = 2$ and $\sigma = 2.0$

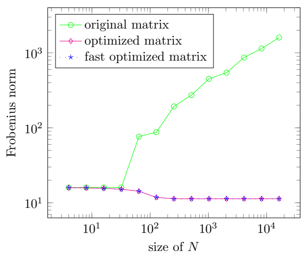


(c) $m = 4$ and $\sigma = 2.0$

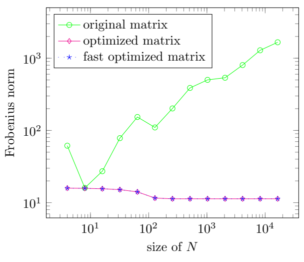
Figure 3.19: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \ldots, 14$ using powers of the sinc function.



(a) $m = 2$ and $\sigma = 1.0$

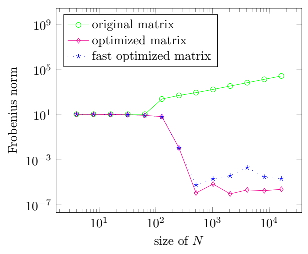


(b) $m = 2$ and $\sigma = 2.0$

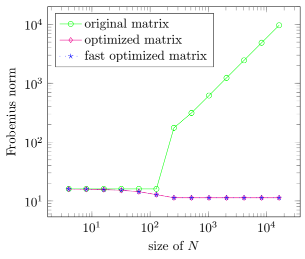


(c) $m = 4$ and $\sigma = 2.0$

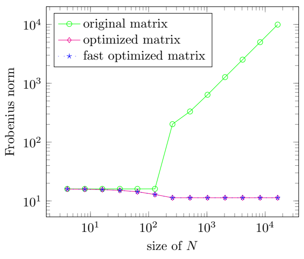
Figure 3.20: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \ldots, 14$ using Kaiser-Bessel windows.



(a) $m = 2$ and $\sigma = 1.0$

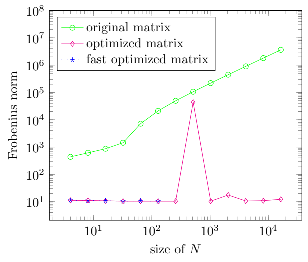


(b) $m = 2$ and $\sigma = 2.0$

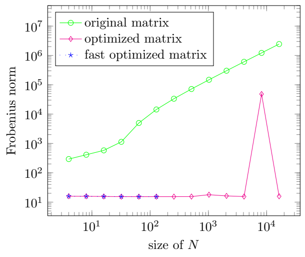


(c) $m = 4$ and $\sigma = 2.0$

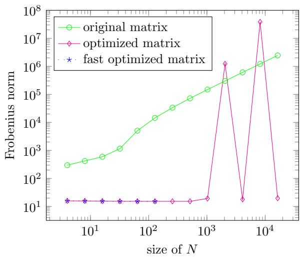
Figure 3.21: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ Chebyshev nodes with $c = 2, \ldots, 14$ using B-Spline window functions.



(a) $m = 2$ and $\sigma = 1.0$



(b) $m = 2$ and $\sigma = 2.0$



(c) $m = 4$ and $\sigma = 2.0$

Figure 3.22: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$ as well as the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 for $M = 128$ and $N = 2^c$ logarithmic nodes with $c = 2, \ldots, 14$ using B-Spline window functions.

Here only the outcome for B-Splines is presented but for the other windows the curves look quite the same.

(iii) Now we have a look at logarithmically spaced nodes, cf. (3.17). Figure 3.22 shows the results for B-Splines and again the other window functions are left out because the graphs look similar.

For this sampling pattern one can see a different behavior of the matrix norms. While the graphs start the same as for other nodes, it can be recognized that for $N > M$ there is no line for the fast algorithm. The reason is that the matrices needed for calculation get extremely bad conditioned so that the computation via Algorithm 3.17 is not possible anymore. But having a look at the condition number of $\boldsymbol{A}$ it becomes apparent that this problem is simply to ill-posed since $\mathrm{cond}(\boldsymbol{A}) > 10^{17}$ for all $N > M$. Therefore, the norms cannot be made small like for other samplings. $\diamond$

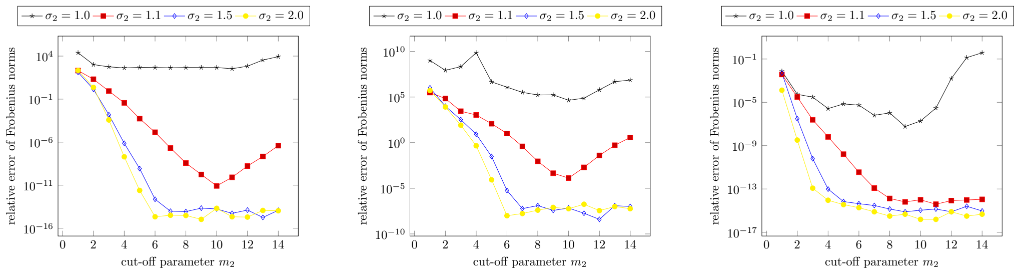
**Example 3.20** Like in Example 3.6 we investigate the accuracy of Algorithm 3.17 in comparison to Algorithm 3.15. Analogously, we consider the relative error

$$\frac{\left| \|\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\tilde{\boldsymbol{B}} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}} - \|\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\tilde{\tilde{\boldsymbol{B}}} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}} \right|}{\|\boldsymbol{F}\boldsymbol{D}\boldsymbol{A}^*\tilde{\boldsymbol{B}} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}} \tag{3.43}$$

for different sizes of the oversampling factor $\sigma_2$ and increasing cut-off parameter $m_2$ when using the Kaiser-Bessel window for the computation of the NFFT. Here we choose $N = 256$ jittered equispaced nodes, $M = 128$ and test for different choices of the parameters when using the B-Spline window for the inversion. The corresponding results are depicted in Figure 3.23. There it can be seen that the smallest errors are achieved for high oversampling $\sigma_2 = 2.0$ and a cut-off $m_2$ much higher than $m$. Another point to mention is that a larger cut-off $m$ for the inversion does not lead to higher accuracy whereas high oversampling $\sigma$ can improve the results. For the other window functions we obtained comparable results and hence the graphs shall be omitted here. $\diamond$

**Example 3.21** Now we have a look at the run-times of Algorithm 3.15 and Algorithm 3.17. For this purpose, we consider again $N = 2^c$ jittered equispaced nodes with $c = 2, \dots, 14$ and use the B-Spline of order 4 without oversampling for the inversion and the Kaiser-Bessel window with $m_2 = 4$ and $\sigma_2 = 2.0$ for the NFFT. The outcomes for $M = 128$ and $M = 1024$ are displayed in Figure 3.24. There it can be seen that for small sizes of $M$ the optimized algorithm is as slow as the original Algorithm 3.15 or even needs more time. But if we choose a higher number $M$ of Fourier coefficients, Algorithm 3.17 is much faster.

Similar results can also be achieved for other parameters of the inversion. $\diamond$



(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 4$ and $\sigma = 1.0$ (c) $m = 2$ and $\sigma = 2.0$

Figure 3.23: Comparison of the relative errors (3.43) using B-Spline window functions for the inversion with $N = 256$ jittered equispaced nodes and $M = 128$ for growing cut-off parameter $m_2$ and different oversampling $\sigma_2$.

### Further simplification

Like already shown in Section 3.2 we can simplify the computation of the matrix $\boldsymbol{K}_l^*$ even further. Again we want to incorporate the Dirichlet kernel (3.19) so that the entries of the matrix $\boldsymbol{K}_l^*$ in (3.39) can explicitly be stated and therefore the term $M \log M$ in the computational costs of Algorithm 3.17 can be eliminated. Hence, we obtain the following algorithm.

**Algorithm 3.22** (Fast optimization of the matrix $\boldsymbol{B}$ using the Dirichlet kernel)

For $N \in \mathbb{N}$ let $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right), j = 1, \dots, N$, be given nodes as well as $M \in 2\mathbb{N}$, $\sigma \geq 1$ and $M_\sigma = \sigma M$.

1. For $l = -\frac{M_\sigma}{2}, \dots, \frac{M_\sigma}{2} - 1$:

   Determine the set $I_{M_\sigma,m}(l)$, cf. (3.36). $\mathcal{O}(N)$

   Compute $\mathcal{O}(MN)$

$$\boldsymbol{K}_l^* = \left[\frac{1}{M_\sigma} D_{\frac{M}{2}-1}\left(\frac{l}{M_\sigma} - x_j\right)\right]_{l=-\frac{M_\sigma}{2}, j \in I_{M_\sigma,m}(l)}^{\frac{M_\sigma}{2}-1}.$$

   Solve the normal equations for $\boldsymbol{K}_l^*$, i. e., compute $\tilde{\boldsymbol{b}}_l$, cf. (3.37). $\mathcal{O}(N^3 + N^2 M)$

2. Compose $\tilde{\tilde{\boldsymbol{B}}}$ column-wise of the vectors $\tilde{\boldsymbol{b}}_l$ observing the periodicity. $\mathcal{O}(M)$

Output: optimized matrix $\tilde{\tilde{\boldsymbol{B}}}$

Complexity: $\mathcal{O}(M^2 N + N^2 M^2 + N^3 M)$

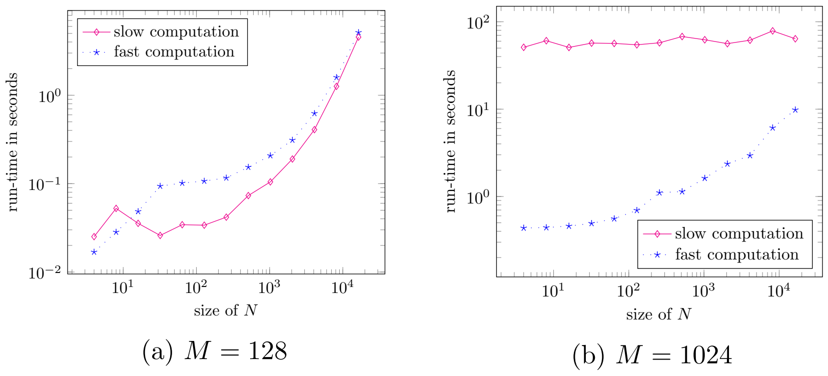


Figure 3.24: Comparison of the run-times of Algorithm 3.15 and Algorithm 3.17 for $N = 2^c$ jittered equispaced nodes with $c = 2, \dots, 14$ using the B-Spline of order 4 without oversampling for the inversion and the Kaiser-Bessel window with $m_2 = 4$ and $\sigma_2 = 2.0$ for the NFFT in Algorithm 3.17.

**Example 3.23** Again we do a test similar to Example 3.19 now comparing Algorithms 3.15 and 3.17 using B-Spline window functions as well as our new Algorithm 3.22. In Figure 3.25 the corresponding outcomes are displayed. It is obvious that the minimization using the Dirichlet kernel works as well as the original optimization does or even better.

While the computational costs could not be scaled down, it can be seen that this step now reduced the run-time for all sizes of $M$, represented in Figure 3.26. $\diamond$

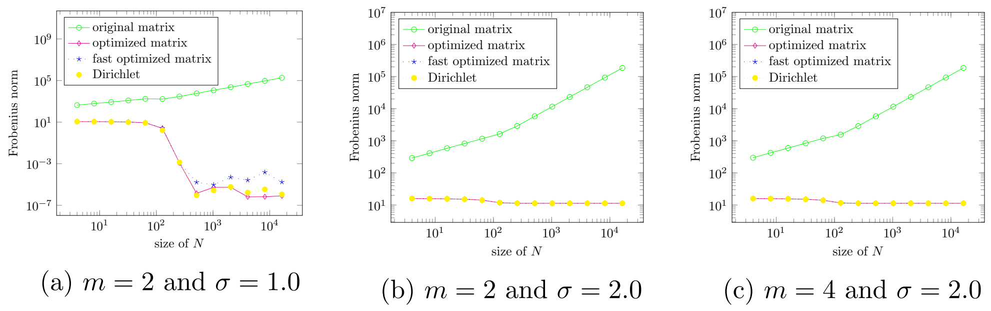


Figure 3.25: Comparison of Frobenius norms (3.41) and (3.42) of the original matrix $\boldsymbol{B}$, the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 using B-Spline window functions as well as the optimized matrix generated by Algorithm 3.22 using the Dirichlet kernel for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \dots, 14$.

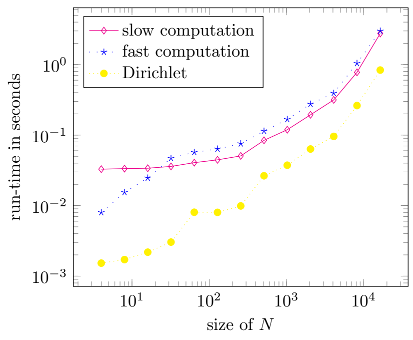


Figure 3.26: Comparison of run-times of Algorithms 3.15 and 3.17 using the B-Spline of order 4 as well as Algorithm 3.22 for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 2, \dots, 14$ without oversampling.

**Example 3.24** In the previous examples we found out that the minimization of the norm was quite successful in the overdetermined case. Like in Example 3.9 we are now interested in calculating an inverse NFFT, see problem (3.1). Hence, we choose again a trigonometric polynomial, cf. (3.20), with random coefficients $\hat{f}_k \in (1, 100)$, $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$, and compute the vector $\boldsymbol{f} = (f(x_j))_{j=1}^N$ at nonequispaced nodes $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$. Since it was already explained in (3.38) that we are now given the approximation $\hat{\boldsymbol{f}} \approx \boldsymbol{D}^* \boldsymbol{F}^* \boldsymbol{B}^* \boldsymbol{f}$ we consider

$$\check{\boldsymbol{f}} = \boldsymbol{D}^* \boldsymbol{F}^* \boldsymbol{B}^* \boldsymbol{f} \tag{3.44}$$

with the original matrix $\boldsymbol{B}^*$ from the adjoint NFFT as well as

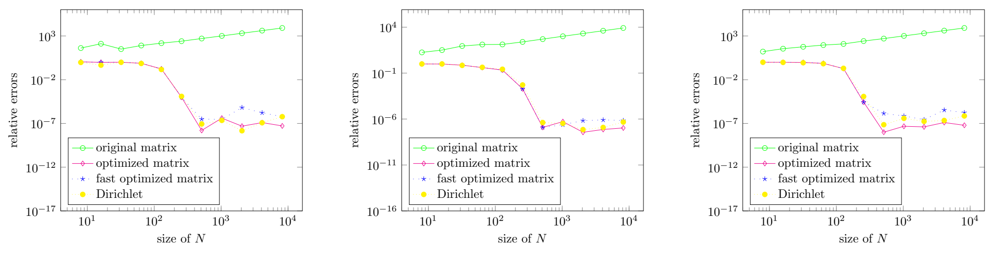
$$\check{\check{\boldsymbol{f}}} = \boldsymbol{D}^* \boldsymbol{F}^* \boldsymbol{O}^* \boldsymbol{f} \tag{3.45}$$

where the matrix $\boldsymbol{O}$ denotes a place marker for the optimized matrix $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15, $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 and $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.22. Given these approximations we study the corresponding relative errors

$$\frac{\|\check{\boldsymbol{f}} - \hat{\boldsymbol{f}}\|_2}{\|\hat{\boldsymbol{f}}\|_2} \quad \text{and} \quad \frac{\|\check{\check{\boldsymbol{f}}} - \hat{\boldsymbol{f}}\|_2}{\|\hat{\boldsymbol{f}}\|_2}. \tag{3.46}$$

For this test we use $M = 128$ and compute the relative errors for $N = 2^c$ nodes with $c = 3, \dots, 13$. The results for jittered equispaced nodes using the B-Spline window function are displayed in Figure 3.27. There it can be seen that the optimization works best if we are in the overdetermined setting $N > M$. Then the errors of all optimized matrices get much smaller than the original one. If we have $N < M$ instead all algorithms do not optimize as much as in the converse setting. Another point to mention is that in case we use oversampling the outcomes are better than the Frobenius norms in Figure 3.17 suggest.

For other window functions the results are nearly the same except that for sinc functions the best method is always Algorithm 3.22. However, the graphs shall be omitted here. Given Chebyshev nodes the results are similar, but given logarithmically spaced nodes instead, this test is quite useless since we know from Figure 3.22 that the optimization does not work. Thus, both are left out. $\diamond$

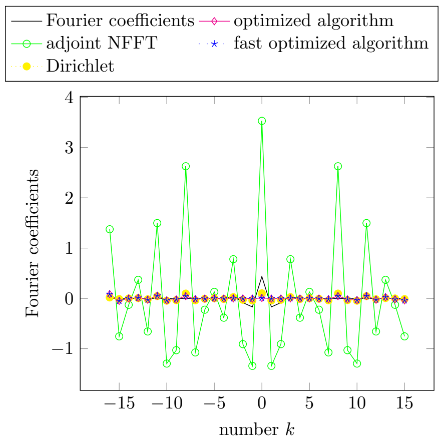


(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

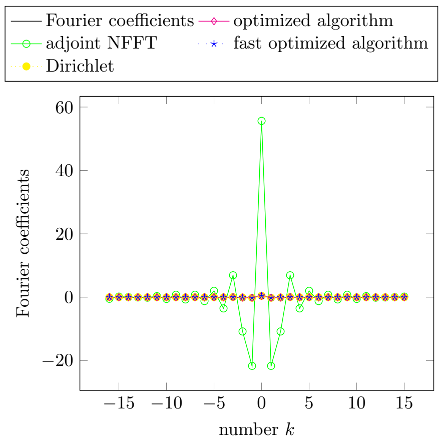
Figure 3.27: Comparison of relative errors (3.46) using the original matrix $\boldsymbol{B}$, the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 using B-Spline window functions as well as the optimized matrix generated by Algorithm 3.22 using the Dirichlet kernel for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 3, \dots, 13$.

**Example 3.25** Again we have a look at the trigonometric function (3.24), cf. Example 3.10. Similarly, we want to compare our approximations (3.44) and (3.45) to the exact Fourier coefficients $c_k(f)$ but now we approximate the function (3.25) with a quadrature rule using weights 1 so that we receive the approximation $g(x) \approx \tilde{g}(x)$ instead of (3.26) and hence $c_k(f) \approx \check{f}_k$. Considering the reconstruction and the errors like mentioned in Example 3.10 we obtain the results displayed in Figure 3.28. There we can see that we are not able to reconstruct the Fourier coefficients in the underdetermined case $M > N$ whereas the approximation in the overdetermined setting $N > M$ is even better than in Example 3.10. These results could even be improved by making use of a higher oversampling factor like it can be seen in Figure 3.29. There the maximum absolute errors $\max_{k=-\frac{N}{2},\dots,\frac{N}{2}-1} |c_k(f) - \check{f}_k|$ and $\max_{k=-\frac{N}{2},\dots,\frac{N}{2}-1} |c_k(f) - \check{\check{f}}_k|$ are depicted.

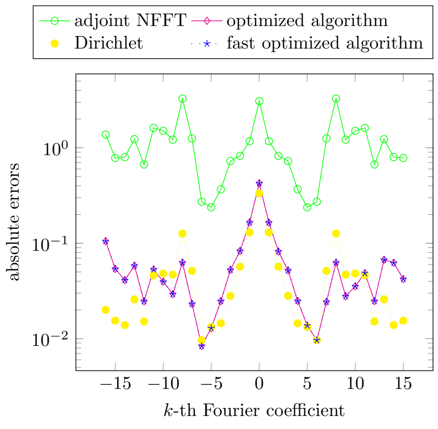
Using Chebyshev nodes instead we see quite the same outcomes and even for logarithmically spaced nodes the approximation can largely be improved in the tested setting. However, all these tests are left out here. $\diamond$



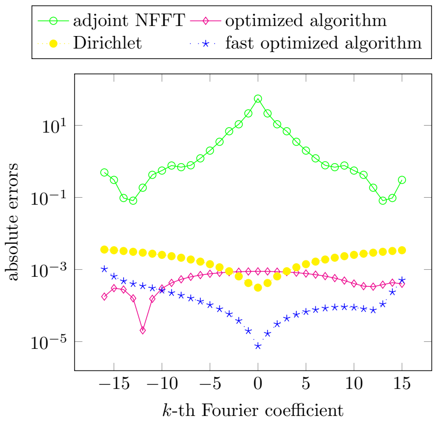
(a) $N = 8$



(b) $N = 128$



(c) $N = 8$



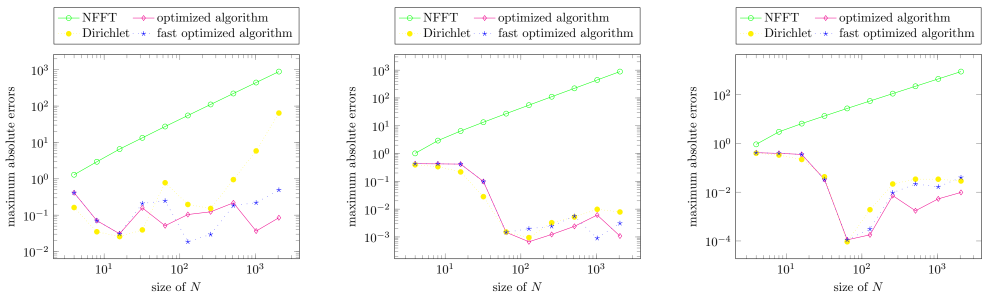
(d) $N = 128$

Figure 3.28: Reconstruction of the Fourier coefficients (top) and pointwise errors (bottom) for $M = 32$ Fourier coefficients, generated by Algorithms 2.5, 3.15 and 3.17 using the B-Spline of order 4 as well as Algorithm 3.22 using the Dirichlet kernel without oversampling for jittered equispaced nodes.

## 3.5 Inverse adjoint NFFT - underdetermined case

Now we have a look at the last remaining matrix product $\boldsymbol{K}^*\boldsymbol{B}$ and the corresponding optimization problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}\,:\,\boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{K}^*\boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2\,.$$



(a) $M = 8, m = 2$ and $\sigma = 1.0$

(b) $M = 32, m = 2$ and $\sigma = 1.0$

(c) $M = 32, m = 4$ and $\sigma = 2.0$

Figure 3.29: Comparison of maximum absolute errors of Algorithms 2.5, 3.15 and 3.17 using the B-Spline window as well as Algorithm 3.22 using the Dirichlet kernel for jittered equispaced nodes and $N = 2^c$ with $c = 2, \dots, 11$.

However, we have already solved this problem in Section 3.4 to find a solution for the transposed problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\ (2m+1)\text{-sparse}}{\text{Minimize}} \qquad \|\boldsymbol{B}^*\boldsymbol{K} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2 .$$

Hence, we only examine the actual approximation. Due to the minimization we have

$$\boldsymbol{FDA}^*\boldsymbol{B} = \boldsymbol{K}^*\boldsymbol{B} \approx \boldsymbol{I}_{M_\sigma}.$$

Because $\boldsymbol{B} \in \mathbb{R}^{N\times M_\sigma}$ is a rectangular matrix and therefore not invertible we multiply this approximation by a right-inverse of $\boldsymbol{B}$, i. e., a matrix $\boldsymbol{B}' \in \mathbb{R}^{M_\sigma\times N}$ that holds $\boldsymbol{BB}' = \boldsymbol{I}_N$, and receive

$$\boldsymbol{FDA}^* \approx \boldsymbol{B}'.$$

Multiplying this by our wanted vector $\boldsymbol{f}$ yields

$$\boldsymbol{FDA}^*\boldsymbol{f} \approx \boldsymbol{B}'\boldsymbol{f},$$

which can be rewritten by means of $\boldsymbol{A}^*\boldsymbol{f} = \boldsymbol{h}$ as

$$\boldsymbol{FDh} \approx \boldsymbol{B}'\boldsymbol{f}.$$

Finally, we multiply left-hand by $\boldsymbol{B}$ which results in the approximation

$$\boldsymbol{BFDh} \approx \boldsymbol{f}$$

and thus provides another method to invert the adjoint NFFT by modifying the NFFT.

**Example 3.26** Since we know that we are in the transposed setting of Section 3.4 and have already seen that in this case the matrix norms are exactly the same, cf. Example 3.11, we refer to Example 3.19 for results with respect to the matrix norms.

Analogously to Example 3.24, we test if we are able to successfully approximate the inverse adjoint NFFT in (3.2). Hence, we also choose a trigonometric polynomial with random coefficients $\hat{f}_k \in (1, 100)$, $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$, and determine the vector $\boldsymbol{f} = (f(x_j))_{j=1}^N$ for $x_j \in \left[-\frac{1}{2}, \frac{1}{2}\right)$, $j = 1, \dots, N$. We consider the approximations

$$\tilde{\boldsymbol{f}} = \boldsymbol{BFD}(\boldsymbol{A}^* \boldsymbol{f}) = \boldsymbol{BFDh} \tag{3.47}$$

with the original matrix $\boldsymbol{B}$ from the NFFT as well as

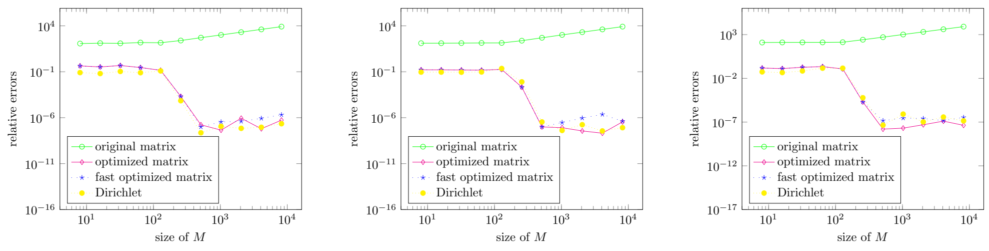
$$\tilde{\tilde{\boldsymbol{f}}} = \boldsymbol{OFD}(\boldsymbol{A}^* \boldsymbol{f}) = \boldsymbol{OFDh}, \tag{3.48}$$

where $\boldsymbol{O}$ is a place marker for the optimized matrix $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15, $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 and $\tilde{\tilde{\tilde{\boldsymbol{B}}}}$ generated by Algorithm 3.22. Given these approximations we examine the relative errors

$$\frac{\|\boldsymbol{A}^* \tilde{\boldsymbol{f}}\|_2}{\|\boldsymbol{h}\|_2} \quad \text{and} \quad \frac{\|\boldsymbol{A}^* \tilde{\tilde{\boldsymbol{f}}}\|_2}{\|\boldsymbol{h}\|_2} \tag{3.49}$$

Like in Example 3.24 we choose $M = 128$ and $N = 2^c$ with $c = 3, \dots, 13$.

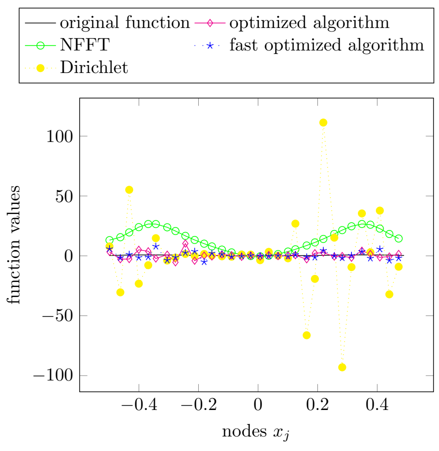
The results for jittered equispaced nodes using the B-Spline window function can be found in Figure 3.30. There quite the same behavior can be seen as in Example 3.24. Again the graphs for other window functions or nodes are left out. ◇



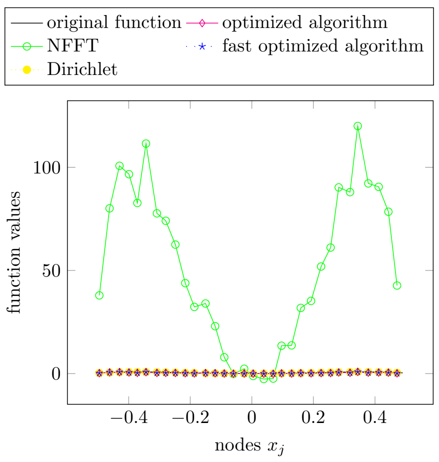
(a) $m = 2$ and $\sigma = 1.0$ (b) $m = 2$ and $\sigma = 2.0$ (c) $m = 4$ and $\sigma = 2.0$

Figure 3.30: Comparison of relative errors (3.49) using the original matrix $\boldsymbol{B}$, the optimized matrices $\tilde{\boldsymbol{B}}$ generated by Algorithm 3.15 and $\tilde{\tilde{\boldsymbol{B}}}$ generated by Algorithm 3.17 using B-Spline window functions as well as the optimized matrix generated by Algorithm 3.22 using the Dirichlet kernel for $M = 128$ and $N = 2^c$ jittered equispaced nodes with $c = 3, \dots, 13$.

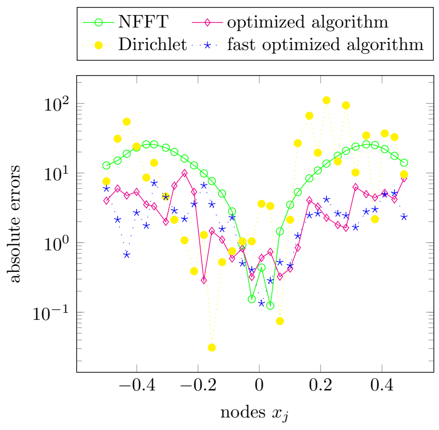
**Example 3.27** Finally, we do the same test as in Example 3.13, namely the reconstruction of the trigonometric function (3.24). For this purpose, we consider the approximations (3.47) and (3.48) and compare them to the function values for jittered equispaced nodes. The results are depicted in Figure 3.31. These graphs show that our optimization was not successful for $N > M$ since there is no reasonable chance to approximate the function values in this setting. But also in the overdetermined case $N < M$ the approximations are not that good as in Example 3.13. Moreover, Figure 3.32 shows that the errors cannot be made smaller for this approach. $\diamond$



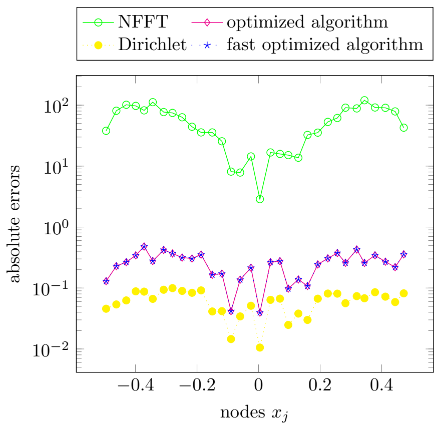
(a) $M = 8$



(b) $M = 128$

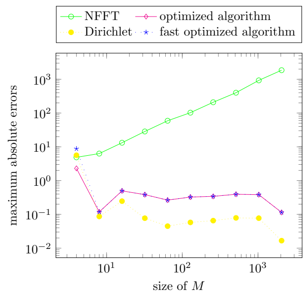


(c) $M = 8$

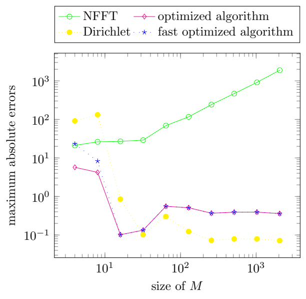


(d) $M = 128$

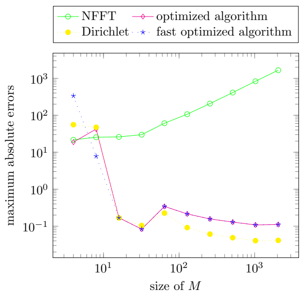
Figure 3.31: Reconstruction of function values (top) and pointwise errors (bottom) for $N = 32$ nodes, generated by Algorithms 2.4, 3.15 and 3.17 using the B-Spline of order 4 as well as Algorithm 3.22 using the Dirichlet kernel without oversampling for jittered equispaced nodes.



(a) $N = 8, m = 2$ and $\sigma = 1.0$



(b) $N = 32, m = 2$ and $\sigma = 1.0$



(c) $N = 32, m = 4$ and $\sigma = 2.0$

Figure 3.32: Comparison of maximum absolute errors of Algorithms 2.4, 3.15 and 3.17 using the B-Spline window as well as Algorithm 3.22 using the Dirichlet kernel for jittered equispaced nodes and $M = 2^c$ with $c = 2, \dots, 11$.

# 4 Frames

In this chapter we will have a look at the concept of frames and discuss another approach for inverting the NFFT based on frame theory. Besides the main ideas of frames and the basic information about the approximation of the inverse frame operator, a link to the ideas explained in Chapter 3 will be provided.

## 4.1 Basic information

Let $\mathcal{H}$ be a Hilbert space. The main characteristic of a basis $\{\varphi_j\}_{j=1}^{\infty}$ of $\mathcal{H}$ is that every element $f \in \mathcal{H}$ can be uniquely represented as a linear combination of the basis elements, i. e.,

$$f = \sum_{j=1}^{\infty} c_j(f)\, \varphi_j. \tag{4.1}$$

However, the constraints for being a basis are very restrictive. We need linear independence and often we also claim orthogonality. Frequently these requirements make it hard to construct bases with additional properties.

For this reason, we will have a look at a generalization of the concept of a basis. A frame is also a sequence $\{\varphi_j\}_{j=1}^{\infty}$ in $\mathcal{H}$ which allows to represent each element $f \in \mathcal{H}$ via (4.1). Even if the coefficients $c_j(f)$ are not necessarily unique anymore this definition is much more flexible. Thus, we will introduce the essential characteristics of frames below, which are basically adapted from explanations in [3] and [14].

**Definition 4.1** A sequence $\{\varphi_j\}_{j=1}^{\infty} \subset \mathcal{H}$ is called **frame** for $\mathcal{H}$ if there exist two constants $A, B > 0$ such that

$$A\|f\|^2 \leq \sum_{j=1}^{\infty} |\langle f, \varphi_j \rangle|^2 \leq B\|f\|^2 \quad \forall f \in \mathcal{H}.$$

The numbers $A$ and $B$ are named **frame bounds** and they are not unique.

(i) A frame $\{\varphi_j\}_{j=1}^{\infty}$ is referred to as **tight** if one can choose $A = B$.

(ii) If the frame condition is violated for a frame $\{\varphi_j\}_{j=1}^{\infty}$ as soon as an arbitrary element is removed, the frame is termed **exact**. $\diamond$

For illustration we examine a few examples next.

**Example 4.2** Let $\{e_j\}_{j=1}^{\infty}$ be an orthonormal basis for $\mathcal{H}$.

(i) By repeating each element of $\{e_j\}_{j=1}^{\infty}$ twice we receive the tight frame

$$\{\varphi_j\}_{j=1}^{\infty} = \{e_1, e_1, e_2, e_2, \dots\}$$

with $A = B = 2$.

(ii) If we only repeat $e_1$ we obtain the frame

$$\{\varphi_j\}_{j=1}^{\infty} = \{e_1, e_1, e_2, e_3, \dots\}$$

with frame bounds $A = 1$ and $B = 2$.

(iii) Now we consider

$$\{\varphi_j\}_{j=1}^{\infty} = \left\{e_1, \frac{1}{\sqrt{2}}e_2, \frac{1}{\sqrt{2}}e_2, \frac{1}{\sqrt{3}}e_3, \frac{1}{\sqrt{3}}e_3, \frac{1}{\sqrt{3}}e_3, \dots\right\},$$

i. e., the $k$-fold repetition of the vector $\frac{1}{\sqrt{k}}e_k$. This is a tight frame because for every element $f \in \mathcal{H}$ we have

$$\sum_{j=1}^{\infty} |\langle f, \varphi_j\rangle|^2 = \sum_{k=1}^{\infty} k\, |\langle f, \tfrac{1}{\sqrt{k}}e_k\rangle|^2 = \sum_{k=1}^{\infty} |\langle f, e_k\rangle|^2.$$

Together with Parsevals equation

$$\sum_{k=1}^{\infty} |\langle f, e_k\rangle|^2 = \|f\|^2, \quad f \in \mathcal{H},$$

this yields $A = B = 1$. ◇

**Definition 4.3** The operator

$$S\colon \mathcal{H} \to \mathcal{H}, \quad Sf = \sum_{j=1}^{\infty} \langle f, \varphi_j\rangle \varphi_j,$$

is called **frame operator**. ◇

**Lemma 4.4** *Let $\{\varphi_j\}_{j=1}^{\infty}$ be a frame with frame operator $S$. Then the following holds.*

*(i) $S$ is bounded, invertible, self-adjoint and positive.*

*(ii) $\{S^{-1}\varphi_j\}_{j=1}^{\infty}$ is a frame with frame operator $S^{-1}$.*

*Proof.* see [3], Lemma 5.1.5.

**Theorem 4.5** *(frame decomposition)*
*If $\{\varphi_j\}_{j=1}^{\infty}$ is a frame with frame operator $S$, then*

$$f = \sum_{j=1}^{\infty} \langle f, S^{-1}\varphi_j \rangle \varphi_j = \sum_{j=1}^{\infty} \langle f, \varphi_j \rangle S^{-1}\varphi_j \quad \forall f \in \mathcal{H}. \tag{4.2}$$

*In other words, every element in $\mathcal{H}$ can be represented as a linear combination of the elements of the frame.*

*Proof.* With Definition 4.3 and Lemma 4.4 we have

$$f = SS^{-1}f = \sum_{j=1}^{\infty} \langle S^{-1}f, \varphi_j \rangle \varphi_j = \sum_{j=1}^{\infty} \langle f, S^{-1}\varphi_j \rangle \varphi_j.$$

The second notation in (4.2) we obtain from the relation $f = S^{-1}Sf$. □

**Definition 4.6** A frame $\{\psi_j\}_{j=1}^{\infty}$ is named **dual frame** of $\{\varphi_j\}_{j=1}^{\infty}$ if we have

$$f = \sum_{j=1}^{\infty} \langle f, \psi_j \rangle \varphi_j \quad \forall f \in \mathcal{H}.$$

This holds for example for the frame $\{\psi_j := S^{-1}\varphi_j\}_{j=1}^{\infty}$ which is termed **canonical dual frame**. The numbers $\langle f, S^{-1}\varphi_j \rangle$ and $\langle f, \varphi_j \rangle$, respectively, are referred to as **frame coefficients**. ◇

The frame decomposition (4.2) is one of the most important results of the frame theory. Though, in practice it is mostly difficult (or even impossible) to apply this decomposition directly. The reason for this is that $\mathcal{H}$ is in general an infinite dimensional Hilbert space what makes it hard to invert the frame operator. Hence, it is necessary to be able to approximate the frame operator $S^{-1}$.

## 4.2 Approximation of the inverse frame operator

For approximating the inverse frame operator we use the method from [13] by analogy with [14]. This method is based on the projection onto a finite dimensional subspace of $\mathcal{H}$. For this purpose, we need the term of an admissible frame. Therefore, we primarily consider the following definition, cf. [14].

**Definition 4.7** A frame $\{\psi_l\}_{l=-\infty}^{\infty}$ is called **admissible frame** with respect to a frame $\{\varphi_j\}_{j=1}^{\infty}$ if the following conditions hold.

(i) There exist positive constants $c_0$ and $t > 1$ such that

$$|\langle \psi_j, \psi_l \rangle| \le c_0(1 + |j - l|)^{-t}, \quad j, l \in \mathbb{Z}.$$

(ii) There exist positive constants $c_1$ and $s > \frac{1}{2}$ such that

$$|\langle \varphi_j, \psi_l \rangle| \le c_1(1 + |j - l|)^{-s}, \quad j \in \mathbb{N},\, l \in \mathbb{Z}.$$

◇

We suppose $\{\psi_l\}_{l=-\infty}^{\infty}$ is an admissible frame with respect to the frame $\{\varphi_j\}_{j=1}^{\infty}$. As shown in [13], the dual frame $\{S^{-1}\varphi_j\}_{j=1}^{\infty}$ can then be approximated by

$$S^{-1}\varphi_j \approx \tilde{\varphi}_j := \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} p_{l,j}\, \psi_l, \quad j = 1, \dots, N, \tag{4.3}$$

where $\mathbf{\Phi}^\dagger =: [p_{l,j}]_{l=-\frac{M_\sigma}{2},\, j=1}^{\frac{M_\sigma}{2}-1,\, N}$ is the Moore-Penrose pseudoinverse of the matrix

$$\mathbf{\Phi} := [\langle \varphi_j, \psi_l \rangle]_{j=1,\, l=-\frac{M_\sigma}{2}}^{N,\, \frac{M_\sigma}{2}-1}. \tag{4.4}$$

In doing so, the matrix dimensions have to fulfill the condition $N \ge M_\sigma + cM_\sigma^{\frac{1}{2s-1}}$ with a constant $c > 0$. Given this approximation of the dual frame, inserting (4.3) in (4.2) and cutting off the infinite sum yields the approximation

$$f \approx \tilde{f} := \sum_{j=1}^{N} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \langle f, \varphi_j \rangle\, p_{l,j}\, \psi_l. \tag{4.5}$$

## 4.3 Linking the frame-theoretical approach to the iNFFT

Now we aim to find a link between the frame approximation (4.5) and the inversion of the NFFT from Chapter 3. As recommended in [14], the frames

$$\{\varphi_j(k) := \mathrm{e}^{-2\pi \mathrm{i} k x_j},\, j \in \mathbb{N}\} \tag{4.6}$$

and

$$\left\{\psi_l(k) := \frac{\mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma}}{M_\sigma \hat{w}(-k)},\, l \in \mathbb{Z}\right\} \tag{4.7}$$

shall be chosen for $k \in \mathbb{R}$, where $x_j \in [-\frac{1}{2}, \frac{1}{2})$ denote the nonequispaced nodes. Note that we changed time and frequency domain to match our notations in Chapter 2. Thereby we obtain the scalar products

$$\langle \varphi_j, \psi_l \rangle_{L_2} = \int_{-\infty}^{\infty} \varphi_j(k)\, \overline{\psi_l(k)}\, \mathrm{d}k = \int_{-\infty}^{\infty} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i} k \left(x_j - \frac{l}{M_\sigma}\right)}\, \mathrm{d}k. \tag{4.8}$$

For $\frac{1}{\hat{w}(k)} \in L_1(\mathbb{R})$ we consider the Fourier transform

$$\int_{-\infty}^{\infty} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i} k y}\, \mathrm{d}k =: F_{L_2}(y).$$

In combination with (4.8) this yields

$$\langle \varphi_j, \psi_l \rangle_{L_2} = F_{L_2}\left(x_j - \tfrac{l}{M_\sigma}\right).$$

Hence, the matrix $\mathbf{\Phi}$ from (4.4) is of the form

$$\mathbf{\Phi}_{L_2} = \left( F_{L_2}\left(x_j - \tfrac{l}{M_\sigma}\right) \right)_{j=1,\, l=-\frac{M_\sigma}{2}}^{N,\ \frac{M_\sigma}{2}-1}.$$

Considering a discrete version of the frames instead, i. e., let (4.6) and (4.7) be only defined for $k \in \mathbb{Z}$, we receive the scalar product

$$\langle \varphi_j, \psi_l \rangle_{\ell_2} = \sum_{k=-\infty}^{\infty} \varphi_j(k)\, \overline{\psi_l(k)} = \sum_{k=-\infty}^{\infty} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i} k \left(x_j - \frac{l}{M_\sigma}\right)} =: F_{\ell_2}\left(x_j - \tfrac{l}{M_\sigma}\right).$$

By truncating the infinite sum we obtain an approximation of the matrix $\mathbf{\Phi}$ from (4.4) by

$$\mathbf{\Phi}_{\ell_2} = \left( \overline{K\left(x_j - \tfrac{l}{M_\sigma}\right)} \right)_{j=1,\, l=-\frac{M_\sigma}{2}}^{N,\ \frac{M_\sigma}{2}-1} \tag{4.9}$$

with the kernel as seen in (3.6), i. e.,

$$\overline{K(x)} = \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i} k x}.$$

In the following explanations we decide to choose $\mathbf{\Phi} = \mathbf{\Phi}_{\ell_2}$.

*Remark* 4.8 In general, we do not have admissible frames for our known window functions $w$ because of the factor $\frac{1}{\hat{w}(k)}, k = -\infty, \dots, \infty$, cf. Remark 2.3. Only if we consider finite frames the appropriate conditions can be satisfied.

In addition, it must be pointed out that for other sampling patterns than the jittered equispaced nodes it was already mentioned in [4] that the admissibility condition may not hold or even the conditions for constituting a frame may fail, cf. [14].

◇

### 4.3.1 Theoretical results

Now we consider the frame approximation (4.5) again. Our aim is to show that the inversion of the NFFT illustrated in Chapter 3 can also be expressed by means of a frame-theoretical approach, i. e., by approximating a function $\hat{f}$ in frequency domain, cf. (2.7), and subsequently sampling at equispaced points $k = -\frac{M}{2}, \dots, \frac{M}{2} - 1$.

The frame approximation of the function $\hat{f}$ sought-after is given by

$$\hat{f} \approx \tilde{\hat{f}} = \sum_{j=1}^{N} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \langle \hat{f}, \varphi_j \rangle_{\ell_2} \, p_{l,j} \, \psi_l \tag{4.10}$$

with $p_{l,j}$ as defined in (4.4). Hence, we are acquainted with two different methods to compute the Fourier coefficients $\hat{f}_k$ from given data $\langle \hat{f}, \varphi_j \rangle =: f_j$; the frame approximation (4.10) as well as the adjoint NFFT (2.13). In what follows, we suppose that we can achieve a reconstruction via frames. Utilizing this, we are going to modify the adjoint NFFT so that we can use this simple method to invert the NFFT. Thus, we are looking for an approximation of the form

$$\tilde{h}_k \approx \tilde{\hat{f}}(k) \approx \hat{f}_k, \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1.$$

To compare the adjoint NFFT and the frame approximation we firstly rewrite the approximation (2.13) from Algorithm 2.5 by analogy to [14]. This yields

$$\begin{aligned} \tilde{h}_k &= \frac{1}{M_\sigma} \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \underbrace{\sum_{j=1}^{N} f_j \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big)}_{=: c_l} \underbrace{\frac{\mathrm{e}^{-2\pi \mathrm{i} k l / M_\sigma}}{\hat{w}(-k)}}_{= M_\sigma \psi_l(k)} \\ &= \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} c_l \, \psi_l(k), \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1, \end{aligned} \tag{4.11}$$

with coefficients vector

$$\boldsymbol{c} = \left( \sum_{j=1}^{N} f_j \, \tilde{w}_m\Big(x_j - \tfrac{l}{M_\sigma}\Big) \right)_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} = \boldsymbol{B}^* \boldsymbol{f}, \tag{4.12}$$

where

$$\boldsymbol{f} := (f_j)_{j=1}^{N} = \Big( \langle \hat{f}, \varphi_j \rangle_{\ell_2} \Big)_{j=1}^{N}.$$

Likewise we can rewrite the frame approximation (4.10), cf. [14], as

$$\tilde{\tilde{h}}_k := \tilde{\tilde{f}}(k) = \sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} d_l\, \psi_l(k), \quad k = -\tfrac{M}{2}, \dots, \tfrac{M}{2} - 1, \tag{4.13}$$

where

$$\boldsymbol{d} := (d_l)_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} = \boldsymbol{\Phi}^\dagger \boldsymbol{f}. \tag{4.14}$$

Furthermore, we define the vectors

$$\tilde{\boldsymbol{h}} := \left(\tilde{h}_k\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \quad \text{and} \quad \tilde{\tilde{\boldsymbol{h}}} := \left(\tilde{\tilde{h}}_k\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1}$$

as well as the matrix

$$\boldsymbol{\Psi} := (\psi_l(k))_{k=-\frac{M}{2},\, l=-\frac{M_\sigma}{2}}^{\frac{M}{2}-1,\, \frac{M_\sigma}{2}-1}.$$

Therefore, (4.11) and (4.13) can be represented by

$$\tilde{\boldsymbol{h}} = \boldsymbol{\Psi}\boldsymbol{c} \quad \text{and} \quad \tilde{\tilde{\boldsymbol{h}}} = \boldsymbol{\Psi}\boldsymbol{d}. \tag{4.15}$$

Hence, now we can estimate the difference between both approximations.

**Theorem 4.9** *(cf. Theorem 2.4. in [14])*
*Let*

$$\hat{\boldsymbol{w}} := \left(\frac{1}{\hat{w}(-k)}\right)_{k=-\frac{M}{2}}^{\frac{M}{2}-1}$$

*be a vector satisfying $\|\hat{\boldsymbol{w}}\|_2 < \infty$. Then the following estimates hold.*

*(i) For $M_\sigma < N$ we have*

$$\left\|\tilde{\boldsymbol{h}} - \tilde{\tilde{\boldsymbol{h}}}\right\|_2 \le \frac{1}{\sqrt{M_\sigma}}\, \|\hat{\boldsymbol{w}}\|_2\, \|\boldsymbol{\Phi}\boldsymbol{B}^* - \boldsymbol{I}_N\|_F\, \|\boldsymbol{\Phi}^\dagger \boldsymbol{f}\|_2, \tag{4.16}$$

*(ii) For $M_\sigma > N$ we have*

$$\left\|\tilde{\boldsymbol{h}} - \tilde{\tilde{\boldsymbol{h}}}\right\|_2 \le \frac{1}{\sqrt{M_\sigma}}\, \|\hat{\boldsymbol{w}}\|_2\, \|\boldsymbol{B}^*\boldsymbol{\Phi} - \boldsymbol{I}_{M_\sigma}\|_F\, \|\boldsymbol{\Phi}^\dagger \boldsymbol{f}\|_2. \tag{4.17}$$

*where $\boldsymbol{B}^*$ denotes the adjoint matrix of* (2.15) *and $\boldsymbol{\Phi} := \boldsymbol{\Phi}_{\ell_2}$ is given as in* (4.9).

*Proof.* By analogy with [14] Definition (4.15) combined with the definition of the Frobenius norm implies

$$\begin{aligned}
\left\|\tilde{\boldsymbol{h}}-\tilde{\tilde{\boldsymbol{h}}}\right\|_2 &= \|\boldsymbol{\Psi c}-\boldsymbol{\Psi d}\|_2 = \|\boldsymbol{\Psi}(\boldsymbol{c}-\boldsymbol{d})\|_2 \leq \|\boldsymbol{\Psi}\|_{\mathrm{F}}\,\|\boldsymbol{c}-\boldsymbol{d}\|_2 \\
&= \sqrt{\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1}|\psi_l(k)|^2}\cdot\|\boldsymbol{c}-\boldsymbol{d}\|_2 \\
&= \sqrt{\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1}\left|\frac{1}{M_\sigma\hat{w}(-k)}\,\mathrm{e}^{-2\pi\mathrm{i}kl/M_\sigma}\right|^2}\cdot\|\boldsymbol{c}-\boldsymbol{d}\|_2 \\
&= \frac{1}{M_\sigma}\sqrt{\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\left|\frac{1}{\hat{w}(-k)}\right|^2\sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1}\underbrace{\left|\mathrm{e}^{-2\pi\mathrm{i}kl/M_\sigma}\right|^2}_{\leq 1}}\cdot\|\boldsymbol{c}-\boldsymbol{d}\|_2 \\
&\leq \frac{1}{M_\sigma}\sqrt{M_\sigma}\cdot\sqrt{\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1}\left|\frac{1}{\hat{w}(-k)}\right|^2}\cdot\|\boldsymbol{c}-\boldsymbol{d}\|_2 \\
&= \frac{1}{\sqrt{M_\sigma}}\,\|\hat{\boldsymbol{w}}\|_2\,\|\boldsymbol{c}-\boldsymbol{d}\|_2\,.
\end{aligned}$$

Now we consider the norm $\|\boldsymbol{c}-\boldsymbol{d}\|_2$ separately.

(i) For $M_\sigma < N$ we have by (4.12) and (4.14) that

$$\begin{aligned}
\boldsymbol{c}-\boldsymbol{d} &= \big(\boldsymbol{B}^*-\boldsymbol{\Phi}^\dagger\big)\boldsymbol{f} = \big(\boldsymbol{B}^*-(\boldsymbol{\Phi}^*\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^*\big)\boldsymbol{f} \\
&= \big((\boldsymbol{\Phi}^*\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^*\big)(\boldsymbol{\Phi B}^*-\boldsymbol{I}_N)\boldsymbol{f}.
\end{aligned}$$

This leads to

$$\begin{aligned}
\|\boldsymbol{c}-\boldsymbol{d}\|_2 &\leq \|\boldsymbol{\Phi B}^*-\boldsymbol{I}_N\|_{\mathrm{F}}\,\|\big((\boldsymbol{\Phi}^*\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^*\big)\boldsymbol{f}\|_2 \\
&\leq \|\boldsymbol{\Phi B}^*-\boldsymbol{I}_N\|_{\mathrm{F}}\,\|\boldsymbol{\Phi}^\dagger\boldsymbol{f}\|_2.
\end{aligned}$$

(ii) In analogy, for $M_\sigma > N$ we have that

$$\begin{aligned}
\boldsymbol{c}-\boldsymbol{d} &= \big(\boldsymbol{B}^*-\boldsymbol{\Phi}^\dagger\big)\boldsymbol{f} = \big(\boldsymbol{B}^*-\boldsymbol{\Phi}^*(\boldsymbol{\Phi\Phi}^*)^{-1}\big)\boldsymbol{f} \\
&= (\boldsymbol{B}^*\boldsymbol{\Phi}-\boldsymbol{I}_{M_\sigma})\big(\boldsymbol{\Phi}^*(\boldsymbol{\Phi\Phi}^*)^{-1}\big)\boldsymbol{f}
\end{aligned}$$

and thereby

$$\begin{aligned}
\|\boldsymbol{c}-\boldsymbol{d}\|_2 &\leq \|\boldsymbol{B}^*\boldsymbol{\Phi}-\boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}\,\|\big(\boldsymbol{\Phi}^*(\boldsymbol{\Phi\Phi}^*)^{-1}\big)\boldsymbol{f}\|_2 \\
&\leq \|\boldsymbol{B}^*\boldsymbol{\Phi}-\boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}\,\|\boldsymbol{\Phi}^\dagger\boldsymbol{f}\|_2.
\end{aligned}$$

□

### 4.3.2 Optimization

Now we aim to use the result above to modify the adjoint NFFT such that we achieve an inversion of the NFFT. Therefore, we want to minimize the distance shown in (4.16) and (4.17), respectively. To this end, we suppose we are given nodes $x_j$ as well as frames $\{\varphi_j\}$ and $\{\psi_l\}$ and thereby the matrix $\boldsymbol{\Phi} = \boldsymbol{\Phi}_{\ell_2}$. Then our purpose is to improve the approximation of the adjoint NFFT by modifying the matrix $\boldsymbol{B}^*$.

#### Connection to the first approach

At first we consider the case $M_\sigma < N$. By dint of the ideas above the optimization problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{\Phi}\boldsymbol{B}^* - \boldsymbol{I}_N\|_{\mathrm{F}}^2 \tag{4.18}$$

arises from the estimate (4.16). For solving this problem we will have a closer look at the matrix $\boldsymbol{\Phi}\boldsymbol{B}^*$. Definitions (2.15) and (4.9) yield

$$\boldsymbol{\Phi}\boldsymbol{B}^* = \left[\sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)} \,\mathrm{e}^{-2\pi\mathrm{i}k\left(x_j - \frac{l}{M_\sigma}\right)}\, \tilde{w}_m\!\left(x_h - \tfrac{l}{M_\sigma}\right)\right]_{j,h=1}^{N} . \tag{4.19}$$

In addition, we consider analogously to (3.35)

$$\boldsymbol{BFDA}^* = \left[\sum_{l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1} \sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)} \,\mathrm{e}^{-2\pi\mathrm{i}k\left(x_j - \frac{l}{M_\sigma}\right)}\, \tilde{w}_m\!\left(x_h - \tfrac{l}{M_\sigma}\right)\right]_{h,j=1}^{N} . \tag{4.20}$$

Comparing the matrices (4.19) and (4.20) it can be recognized that (4.19) is exactly the transposed of (4.20), i. e.,

$$\boldsymbol{\Phi}\boldsymbol{B}^* = (\boldsymbol{BFDA}^*)^T = (\boldsymbol{BK}^*)^T .$$

Since the optimization problem (4.18) is thereby equivalent to the transposed problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{BFDA}^* - \boldsymbol{I}_N\|_{\mathrm{F}}^2,$$

(4.18) can be solved like already seen in Section 3.3. It may be recognized that the objective is a slightly different one than in Chapter 3 since now we are looking for an approximation of the form $\boldsymbol{BFDA}^* \approx \boldsymbol{I}_N$ instead of $\boldsymbol{BFDA}^* \approx M\boldsymbol{I}_N$. However, in other words, only a constant is missing which does not change the method.

### Connection to the second approach

For $M_\sigma > N$ the ideas above lead to the optimization problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{B}^*\boldsymbol{\Phi} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2. \tag{4.21}$$

Again we have a closer look at the appropriate matrix

$$\boldsymbol{B}^*\boldsymbol{\Phi} = \left[\sum_{j=1}^{N}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i}k\left(x_j - \frac{s}{M_\sigma}\right)}\, \tilde{w}_m\!\left(x_j - \frac{l}{M_\sigma}\right)\right]_{l,\,s=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1}. \tag{4.22}$$

Furthermore, we also consider the matrix from (3.35)

$$\boldsymbol{FDA}^*\boldsymbol{B} = \left[\sum_{j=1}^{N}\sum_{k=-\frac{M}{2}}^{\frac{M}{2}-1} \frac{1}{M_\sigma \hat{w}(k)}\, \mathrm{e}^{-2\pi \mathrm{i}k\left(x_j - \frac{s}{M_\sigma}\right)}\, \tilde{w}_m\!\left(x_j - \frac{l}{M_\sigma}\right)\right]_{s,\,l=-\frac{M_\sigma}{2}}^{\frac{M_\sigma}{2}-1}.$$

Once again, a comparison of the matrices (4.22) and (3.35) yields that they are equal except for transposition, i. e.,

$$\boldsymbol{B}^*\boldsymbol{\Phi} = \left(\boldsymbol{FDA}^*\boldsymbol{B}\right)^T = \left(\boldsymbol{K}^*\boldsymbol{B}\right)^T.$$

Because the optimization problem (4.21) is hence equivalent to the transposed problem

$$\underset{\boldsymbol{B}\in\mathbb{R}^{N\times M_\sigma}:\ \boldsymbol{B}\,(2m+1)\text{-sparse}}{\text{Minimize}} \quad \|\boldsymbol{FDA}^*\boldsymbol{B} - \boldsymbol{I}_{M_\sigma}\|_{\mathrm{F}}^2,$$

(4.21) can be solved like already discussed in Section 3.5.

Therefore, we have shown that both frame-theoretical attempts can be traced back to the methods for inverting the NFFT which were already introduced in Chapter 3.

# Summary

In the present thesis we developed a new direct method for computing an inverse NFFT, i. e., for the reconstruction of the Fourier coefficients $\hat{f}_k$ from given function values $f_j$. Furthermore, a solution for the adjoint problem, the reconstruction of function values $f_j$ from given data $h_k$, was proposed. Therefore, these problems were split up to overdetermined and underdetermined cases which then could be solved separately.

The main idea for finding a solution in each of these cases was the minimization of a certain Frobenius norm. Hence, the appropriate matrices were studied thoroughly so that the solution of the optimization problem could be deduced by means of the least squares method. This already led to first algorithms.

Moreover, these algorithms were improved and efficient methods for computation were presented. As a first approach we were able to reduce the computational costs by approximating the entries of the corresponding matrices via NFFT. Though, in numerical experiments it became apparent that the developed inversion works the same for all common window functions. Therefore, we managed to improve the complexity even further in a second step by making use of the Dirichlet kernel instead of the window functions. So, all in all, we ended up with precomputational algorithms of complexity $\mathcal{O}(N^2)$ and $\mathcal{O}(M^2)$, respectively, whereas the algorithms for the inversion require only $\mathcal{O}(M \log M + N)$ arithmetic operations.

These algorithms were finally tested and compared to the original NFFT algorithms. There it could be seen that each algorithm substantially minimizes the norm it was supposed to and that the fastest method is even the best. Regarding the reconstruction of a given function we also showed that each algorithm leads to much better approximations especially in the setting it was developed for.

Last but not least, we investigated another approach for inverting the NFFT. This was done by dint of the frame approximation which could be used to approximate a function $\hat{f}$ in frequency domain and subsequently sample at equispaced points. This procedure could then be compared to the adjoint NFFT so that the last-mentioned could be modified to achieve a good approximation. In so doing, we found out that the thereby obtained approaches can be traced back to the methods for the inversion of the NFFT which were introduced before.

# Bibliography

[1] A. P. Austin and L. N. Trefethen. Trigonometric interpolation and quadrature in perturbed points. *SIAM J. Numer. Anal.*, 55:2113–2122, 2017.

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

[3] O. Christensen. *An introduction to frames and Riesz bases (Second Edition).* Applied and Numerical Harmonic Analysis. Birkhäuser Basel, 2016.

[4] J. Davis, A. Gelb, and G. Song. A high-dimensinoal inverse frame operator approximation technique. *SIAM Journal on Numerical Analysis*, 54(4):2282–2301, 2016.

[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] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data II. *Appl. Comput. Harmon. Anal.*, 2:85–100, 1995.

[8] H. G. Feichtinger, K. Gröchenig, and T. Strohmer. Efficient numerical methods in non-uniform sampling theory. *Numer. Math.*, 69:423–440, 1995.

[9] J. A. Fessler and B. P. Sutton. Nonuniform fast Fourier transforms using min-max interpolation. *IEEE Trans. Signal Process.*, 51:560–574, 2003.

[10] G. B. Folland. *Fourier Analysis and its Applications.* Brooks/Cole, Albany, 1992.

[11] K. Fourmont. Non equispaced fast Fourier transforms with applications to tomography. *J. Fourier Anal. Appl.*, 9:431–450, 2003.

[12] C. Gasquet and P. Witomski. *Fourier Analysis and Applications: Filtering, Numerical Computation, Wavelets.* Springer, 1999.

[13] A. Gelb and G. Song. Approximating the inverse frame operator from localized frames. *Appl. Comput. Harm. Anal.*, 35(1):94–110, 2013.

[14] A. Gelb and G. Song. A frame theoretic approach to the nonuniform fast Fourier transform. *SIAM J. Numer. Anal.*, 52(3):1222–1242, 2014.

[15] K. Gröchenig. Reconstruction algorithms in irregular sampling. *Math. Comput.*, 59:181–194, 1992.

[16] J. Keiner, S. Kunis, and D. Potts. Using NFFT3 - a software library for various nonequispaced fast Fourier transforms. *ACM Trans. Math. Software*, 36:Article 19, 1–30, 2009.

[17] M. Kircheis. Die direkte inverse NFFT. Bachelorarbeit, Fakultät für Mathematik, Technische Universität Chemnitz, 2017.

[18] S. Kunis and D. Potts. Stability results for scattered data interpolation by trigonometric polynomials. *SIAM J. Sci. Comput.*, 29:1403–1419, 2007.

[19] A. Nieslony and G. Steidl. Approximate factorizations of Fourier matrices with nonequispaced knots. *Linear Algebra Appl.*, 266:337–351, 2003.

[20] D. Potts. Schnelle Fourier-Transformationen für nichtäquidistante Daten und Anwendungen. Habilitation, Universität zu Lübeck, `http://www.tu-chemnitz.de/~potts`, 2003.

[21] D. Potts and G. Steidl. Fast summation at nonequispaced knots by NFFTs. *SIAM J. Sci. Comput.*, 24:2013–2037, 2003.

[22] 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, MA, USA, 2001. Birkhäuser.

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

[24] G. Steidl and M. Tasche. *Schnelle Fourier–Transformation–Theorie und Anwendungen.* 8 Lehrbriefe der FernUniv. Hagen, 1997.

[25] C. F. Van Loan. *Computational Frameworks for the Fast Fourier Transform.* SIAM, Philadelphia, PA, USA, 1992.

[26] M. W. Wong. *Discrete Fourier Analysis.* Birkhäuser Basel, 2011.