Automated parameter tuning based on RMS errors for nonequispaced FFTs

Franziska Nestler

In this paper we study the error behavior of the well known fast Fourier transform for nonequispaced data (NFFT) with respect to the $\mathcal{L}_2$-norm. We compare the arising errors for different window functions and show that the accuracy of the algorithm can be significantly improved by modifying the shape of the window function. Based on the considered error estimates for different window functions we are able to state an easy and efficient method to tune the involved parameters automatically. The numerical examples show that the optimal parameters depend on the given Fourier coefficients, which are assumed not to be of a random structure or roughly of the same magnitude but rather subject to a certain decrease.

Key words and phrases : nonequispaced fast Fourier transform, nonuniform fast Fourier transform, NFFT, NUFFT

2000 AMS Mathematics Subject Classification : 65T

1 Introduction

A broad variety of mathematical algorithms and applications depend on the calculation of the nonequispaced discrete Fourier transform, which is a generalization of the discrete Fourier transform to nonequispaced nodes. Especially, its fast approximate realization called nonequispaced fast Fourier transform (NFFT) or rather nonuniform fast Fourier transform (NUFFT) [5, 1, 25, 27, 23, 9, 16] led to the development of a large number of fast numerical algorithms.

Basically, the NFFT, which is an approximate algorithm, consists of three steps. Using a so called window function, the given coefficients are at first deconvolved in Fourier domain. The result is transformed into spatial domain by an FFT and a discrete convolution with the window function is the final step. Thereby, the window function is chosen such that it is well localized in spatial as well as in Fourier domain. Given this property, the deconvolution step can be realized very efficiently and the resulting aliasing errors can be kept small.

In this paper we investigate the occurrent errors measured in the $\mathcal{L}_2$-norm. In some numerical examples, we evaluate these errors for different window functions. We show that the accuracy of the algorithm can be improved by modifying the shape parameter of the window function and that the optimal value of this shape parameter very much depends on the given set of Fourier coefficients. If the input signals are assumed to be random and uncorrelated,

a prediction of the optimal shape parameter is possible for certain window functions. As an example, for the Gaussian window function a convenient choice of the shape parameter has already been derived in [25] as well as in [4]. Further windows for which the question concerning the optimal choice of the shape parameter is also interesting, as for example Kaiser-Bessel functions [8, 12], have been suggested in the literature.

However, there are many applications, where the given Fourier coefficients are not of a random structure. As an example, the NFFT can be used in order to evaluate sums of the form

$$f(y_k) := \sum_{j=1}^{N} \alpha_j K(y_k - x_j), \quad k = 1, \ldots, M,$$

where the nodes y_k, x_j, the coefficients α_j and a certain smooth kernel function K is given, see [22] for instance. The method, which is widely known as NFFT based fast summation, is based on approximating the kernel function by a trigonometric polynomial, where the corresponding Fourier coefficients are naturally subject to a certain decay. For the fast NFFT based Gauss transform [18] we have an exponential decrease of the Fourier coefficients, for instance. The NFFT based fast summation is also applied for the computation of the Coulomb potentials and forces in particle systems, where the kernel function is given by $K(r) = r^{-1}$. This problem can also be considered subject to periodic boundary conditions, where the analytical Fourier coefficients are known and also underlie an exponential decrease, see [6, 11, 3, 20].

Thus, in our numerical examples we consider certain sets of decreasing Fourier coefficients and show that an appropriate modification of the window's shape parameter leads to substantially better results. In other words, we optimize the shape of the window function based on the given Fourier coefficients, instead of analyzing a worst case error which is independent from the given coefficients, see [13] for instance. For the Kaiser-Bessel window function the variability of the shape parameter was also considered in [7], but an adaption was not done depending on the given Fourier coefficients.

In our tests we additionally compare the errors between two different deconvolution approaches. We also consider the $\mathcal{L}_2$-optimized deconvolution, which has already been considered in [4, Appendix A] or [13], and also give numerical evidence that only small improvements are possible by applying this optimized deconvolution scheme. Based on the error estimates, we are able to state an easy and efficient method to tune all parameters involved in the univariate NFFT algorithm. Note that it has already been observed that in some applications the NFFT with very small oversampling factors [28] or even without oversampling [3] leads to very precise approximations. The results presented in this paper confirm that in some cases an oversampling is in fact not needed.

We remark that an overall tuning, which in addition optimizes the set of parameters with respect to runtime, should depend on the used hardware. In addition, the runtime behavior regarding the window evaluation is different for the individual window functions, unless the evaluation is based on interpolation tables. The overall performance may also depend on the used hardware as well as on the applied variant of the NFFT (multithreaded NFFT [26], NFFT on GPUs [17], parallel NFFT [20]). Anyway, in order to develop optimal runtime models, a precise prediction of the arising errors as well as automated parameter tuning methods, as discussed in this paper, are essential.

The outline of this paper is as follows. In Section 2 we give a short introduction to the NFFT. We start by introducing the necessary notations and then give a formula for the computation of the approximation error in the $\mathcal{L}_2$-norm. In Section 3 we introduce different

window functions and show how the corresponding aliasing errors can be estimated. We compare two different deconvolution approaches and point out how the choice of the window's shape parameter can influence the goodness of the approximation. Therefore we consider some univariate examples. A comparison to measured approximation errors is done in Section 4. Based on these error estimates we describe an automatic parameter tuning for the univariate case in Section 5. We continue with some remarks concerning the multivariate case and conclude with a short summary.

2 NFFT

In the following we give a short introduction to the NFFT in d dimensions. At first, we will introduce the necessary notations.

For some $\boldsymbol{M} = (M_1, \ldots, M_d) \in 2\mathbb{N}^d$ we define the index set $\mathcal{I}_{\boldsymbol{M}}$ by

$$\mathcal{I}_{\boldsymbol{M}} := \left\{-\tfrac{M_1}{2}, \ldots, \tfrac{M_1}{2} - 1\right\} \times \cdots \times \left\{-\tfrac{M_d}{2}, \ldots, \tfrac{M_d}{2} - 1\right\}.$$

For two vectors $\boldsymbol{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$ and $\boldsymbol{y} = (y_1, \ldots, y_d) \in \mathbb{R}^d$ we define the component wise product by $\boldsymbol{x} \odot \boldsymbol{y} := (x_1 y_1, \ldots, x_d y_d) \in \mathbb{R}^d$ as well as the inner product via $\boldsymbol{x} \cdot \boldsymbol{y} := x_1 y_1 + \cdots + x_d y_d \in \mathbb{R}$. For a vector $\boldsymbol{x} \in \mathbb{R}^d$ with non vanishing components we set $\boldsymbol{x}^{-1} := (x_1^{-1}, \ldots, x_d^{-1}) \in \mathbb{R}^d$.

Let some arbitrary nodes $\boldsymbol{x}_j \in \mathbb{T}^d$, where $\mathbb{T} := \mathbb{R}/\mathbb{Z} \simeq [-1/2, 1/2)$ and $j = 1, \ldots, N$, be given. We are now interested in a fast evaluation of a given trigonometric polynomial in the unequally spaced points $\boldsymbol{x}_j$, i.e., we want to compute the sums

$$f(\boldsymbol{x}_j) := \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{f}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}_j}, \quad j = 1, \ldots, N, \tag{2.1}$$

where the Fourier coefficients $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$, $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}$, are also given.

The well known NFFT algorithm can be used to evaluate sums of the form (2.1) very efficiently with $\mathcal{O}(|\mathcal{I}_{\boldsymbol{M}}| \log |\mathcal{I}_{\boldsymbol{M}}| + N)$ arithmetic operations. In the following, we will give an overview of the main steps.

The basic idea is to approximate the function f by a sum of translates of a one-periodic function $\tilde{\varphi}$, i.e.,

$$f(\boldsymbol{x}) \approx \tilde{f}(\boldsymbol{x}) := \sum_{\boldsymbol{l} \in \mathcal{I}_{\boldsymbol{\sigma} \odot \boldsymbol{M}}} g_{\boldsymbol{l}} \tilde{\varphi}\left(\boldsymbol{x} - \boldsymbol{l} \odot (\boldsymbol{\sigma} \odot \boldsymbol{M})^{-1}\right), \tag{2.2}$$

where we denote by $\boldsymbol{\sigma} \geq 1$ (component wise) the oversampling factor and the coefficients $g_{\boldsymbol{l}} \in \mathbb{C}$ are by now unknown. In other words, the approximate function values are obtained by computing a discrete convolution of a given window function with some coefficients $g_{\boldsymbol{l}}$, which have to be determined. In the following we denote the oversampled grid size shortly by $\boldsymbol{M}_{\mathrm{o}} := \boldsymbol{\sigma} \odot \boldsymbol{M}$. The function $\tilde{\varphi}$ is the periodization of a window function φ, which is constructed based on a univariate function $\tilde{\varphi}_{\mathrm{1d}}$ via a tensor product scheme, i.e.,

$$\tilde{\varphi}(\boldsymbol{x}) := \sum_{\boldsymbol{r} \in \mathbb{Z}^d} \varphi(\boldsymbol{x} + \boldsymbol{r}), \text{ where } \varphi(\boldsymbol{y}) = \prod_{j=1}^{d} \varphi_{\mathrm{1d}}(y_j) \text{ for } \boldsymbol{y} = (y_1, \ldots, y_d) \in \mathbb{R}^d. \tag{2.3}$$

A transformation of $\tilde{f}$ into Fourier space gives

$$\tilde{f}(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_{\mathrm{o}}}} \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}) \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}} + \sum_{\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}} \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_{\mathrm{o}}}} \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}) \mathrm{e}^{-2\pi \mathrm{i} (\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_{\mathrm{o}}) \cdot \boldsymbol{x}}, \tag{2.4}$$

where we denote by

$$c_{\boldsymbol{k}}(\tilde{\varphi}) := \int_{\mathbb{T}^d} \tilde{\varphi}(\boldsymbol{x}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{x}} \mathrm{d}\boldsymbol{x} = \int_{\mathbb{R}^d} \varphi(\boldsymbol{x}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{x}} \mathrm{d}\boldsymbol{x} = \hat{\varphi}(\boldsymbol{k})$$

the Fourier coefficients of $\tilde{\varphi}$ and the discrete Fourier coefficients $\hat{g}_{\boldsymbol{k}}$ are given by

$$\hat{g}_{\boldsymbol{k}} = \sum_{\boldsymbol{l} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} g_{\boldsymbol{l}} \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k}\cdot(\boldsymbol{l} \odot {\boldsymbol{M}_\mathrm{o}}^{-1})}.$$

For the following considerations we assume that we have $c_{\boldsymbol{k}}(\tilde{\varphi}) \in \mathbb{R}$. The idea is now to choose the coefficients $\hat{g}_{\boldsymbol{k}}$ appropriately. Then, the coefficients $g_{\boldsymbol{l}}$ in (2.2) can be computed by a d-variate (inverse) FFT

$$g_{\boldsymbol{l}} = \frac{1}{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|} \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \hat{g}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\cdot(\boldsymbol{l} \odot {\boldsymbol{M}_\mathrm{o}}^{-1})} \tag{2.5}$$

and the evaluation of (2.2) gives the approximate function values $\tilde{f}(\boldsymbol{x}_j) \approx f(\boldsymbol{x}_j)$.

However, the evaluation of the sums (2.2) might be computationally demanding unless φ is compactly supported on a comparably small domain or at least sufficiently small outside of it. In the latter case we replace the window function φ by a truncated version

$$\varphi_\mathrm{t}(\boldsymbol{x}) := \varphi(\boldsymbol{x}) \cdot \prod_{j=1}^{d} \chi_{[-\frac{m}{\sigma_j M_j}, \frac{m}{\sigma_j M_j}]}(x_j) = \begin{cases} \varphi(\boldsymbol{x}) & : \boldsymbol{x} \in \bigotimes_{j=1}^{d} [-\frac{m}{\sigma_j M_j}, \frac{m}{\sigma_j M_j}], \\ 0 & : \text{else}, \end{cases}$$

and approximate f by

$$f(\boldsymbol{x}) \approx \tilde{f}(\boldsymbol{x}) := \sum_{\boldsymbol{l} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} g_{\boldsymbol{l}} \tilde{\varphi}_\mathrm{t}\left(\boldsymbol{x} - \boldsymbol{l} \odot {\boldsymbol{M}_\mathrm{o}}^{-1}\right),$$

where now only $(2m+1)^d \ll |\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|$ summands are not equal to zero. In the following we will refer to m as the support parameter.

It is an interesting question how to choose the unknown coefficients $\hat{g}_{\boldsymbol{k}}$. A comparison of (2.1) and (2.4), where we have to replace $\tilde{\varphi}$ by $\tilde{\varphi}_\mathrm{t}$ in the case that φ is not compactly supported, gives $f(\boldsymbol{x}) - \tilde{f}(\boldsymbol{x}) =$

$$\sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{x}} \hat{f}_{\boldsymbol{k}} - \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\cdot\boldsymbol{x}} \left(\hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) + \sum_{\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}} \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{-2\pi \mathrm{i} (\boldsymbol{r} \odot \boldsymbol{M}_\mathrm{o})\cdot\boldsymbol{x}} \right) \tag{2.6}$$

and

$$\left| f(\boldsymbol{x}) - \tilde{f}(\boldsymbol{x}) \right| \leq \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \left| \hat{f}_{\boldsymbol{k}} - \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \right| + \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}} \setminus \mathcal{I}_{\boldsymbol{M}}} \left| \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \right| + \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \sum_{\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}} \left| \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \right|.$$

Thus, at first glance it seems advantageous to set

$$\hat{g}_{\boldsymbol{k}} := \begin{cases} \hat{d}_{\boldsymbol{k}} \hat{f}_{\boldsymbol{k}} & : \boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}} \\ 0 & : \text{else} \end{cases}, \tag{2.7}$$

where we define

$$\hat{d}_{\boldsymbol{k}} := \frac{1}{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}, \tag{2.8}$$

cf. [23] for instance.

Remark 2.1. For some $\boldsymbol{x} \in \mathbb{T}^d$ let the operator $\mathcal{T}_{\boldsymbol{x}} : \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}}|} \to \mathbb{C}$ be defined by

$$\mathcal{T}_{\boldsymbol{x}} : \left[\hat{f}_{\boldsymbol{k}}\right]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \mapsto \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{f}_{\boldsymbol{k}} \mathrm{e}^{2\pi \mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}} = f(\boldsymbol{x}).$$

As described above, $\mathcal{T}_{\boldsymbol{x}}$ is approximated by $\mathcal{T}_{\boldsymbol{x}} \approx \mathcal{B}_{\boldsymbol{x}}\mathcal{F}\mathcal{D}$, where

$$\begin{aligned}
\mathcal{D} : \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}}|} &\to \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|}, \quad \left[\hat{f}_{\boldsymbol{k}}\right]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \mapsto \left[\hat{g}_{\boldsymbol{k}}\right]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \quad \text{via (2.7)} \\
\mathcal{F} : \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|} &\to \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|}, \quad \left[\hat{g}_{\boldsymbol{k}}\right]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \mapsto \left[g_{\boldsymbol{l}}\right]_{\boldsymbol{l}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \quad \text{via (2.5)} \\
\mathcal{B}_{\boldsymbol{x}} : \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|} &\to \mathbb{C}, \quad \left[g_{\boldsymbol{l}}\right]_{\boldsymbol{l}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \mapsto \sum_{\boldsymbol{l}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} g_{\boldsymbol{l}} \tilde{\varphi}_\mathrm{t}(\boldsymbol{x} - \boldsymbol{l} \odot \boldsymbol{M}_\mathrm{o}^{-1}).
\end{aligned}$$

In other words, the NFFT computes

$$f(\boldsymbol{x}_j) \approx \mathcal{B}_{\boldsymbol{x}_j}\mathcal{F}\mathcal{D}\hat{\boldsymbol{f}} =: \tilde{f}(\boldsymbol{x}_j)$$

for all $j = 1, \ldots, N$. The three steps can be explained as follows. The step from Fourier space to spatial domain is done via the ordinary inverse FFT denoted by $\mathcal{F}$, see (2.5). Since the function values are finally approximated by computing a discrete convolution ($\mathcal{B}_{\boldsymbol{x}_j}$), the given Fourier coefficients have to be scaled appropriately ($\mathcal{D} : \hat{f}_{\boldsymbol{k}} \mapsto \hat{g}_{\boldsymbol{k}}$), which we refer to as a deconvolution, see also [4].

The efficient computation of

$$h(\boldsymbol{k}) = \sum_{j=1}^{N} f_j \mathrm{e}^{2\pi \mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}_j}$$

for all $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}$ corresponds to applying the adjoint operator, which reads as

$$\big(h(\boldsymbol{k})\big)_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} = \sum_{j=1}^{N} \mathcal{T}_{\boldsymbol{x}_j}^* f_j \approx \sum_{j=1}^{N} \mathcal{D}^*\mathcal{F}^*\mathcal{B}_{\boldsymbol{x}_j}^* f_j.$$

□

We want to determine for which coefficients $\hat{d}_{\boldsymbol{k}}$ the $\mathcal{L}_2$-norm or root mean square (RMS) error

$$\left\|f - \tilde{f}\right\|_2^2 := \int_{\mathbb{T}^d} |f(\boldsymbol{x}) - \tilde{f}(\boldsymbol{x})|^2 \mathrm{d}\boldsymbol{x} = \int_{\mathbb{T}^d} |\mathcal{T}_{\boldsymbol{x}}\hat{\boldsymbol{f}} - \mathcal{B}_{\boldsymbol{x}}\mathcal{F}\mathcal{D}\hat{\boldsymbol{f}}|^2 \mathrm{d}\boldsymbol{x}.$$

is minimized. In the following Lemma we present the optimal deconvolution coefficients, see also [4, Appendix A] or [13], for instance. The coefficients are also optimal for the adjoint problem.

Lemma 2.1. *For $\hat{\boldsymbol{f}} \in \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}}|}$ and $y \in \mathbb{C}$ we have*

$$\operatorname*{arg\,min}_{\hat{d}_{\boldsymbol{k}}\in\mathbb{R}} \int_{\mathbb{T}^d} |\mathcal{T}_{\boldsymbol{x}}\hat{\boldsymbol{f}} - \mathcal{B}_{\boldsymbol{x}}\mathcal{F}\mathcal{D}\hat{\boldsymbol{f}}|^2 \mathrm{d}\boldsymbol{x} = \operatorname*{arg\,min}_{\hat{d}_{\boldsymbol{k}}\in\mathbb{R}} \int_{\mathbb{T}^d} \|\mathcal{T}_{\boldsymbol{x}}^* y - \mathcal{F}^*\mathcal{D}^*\mathcal{B}_{\boldsymbol{x}}^* y\|^2 \mathrm{d}\boldsymbol{x},$$

where we denote by $\|\cdot\|$ the Euclidean norm in $\mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}}|}$. The optimal deconvolution coefficients are given by

$$\hat{d}_{\boldsymbol{k}} = \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}^2(\tilde{\varphi}_\mathrm{t})}. \tag{2.9}$$

For the NFFT approximation error measured in the $\mathcal{L}_2$-norm we obtain

$$\left\|f-\tilde{f}\right\|_2^2 = \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \left|\hat{f}_{\boldsymbol{k}}\right|^2 \frac{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})} \qquad \text{for } \hat{d}_{\boldsymbol{k}} := \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}, \tag{2.10}$$

$$\left\|f-\tilde{f}\right\|_2^2 = \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \left|\hat{f}_{\boldsymbol{k}}\right|^2 \sum_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} \frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})} \qquad \text{for } \hat{d}_{\boldsymbol{k}} := \frac{1}{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}. \tag{2.11}$$

Proof. The term $f(\boldsymbol{x})-\tilde{f}(\boldsymbol{x}) = \mathcal{T}_{\boldsymbol{x}}\hat{\boldsymbol{f}} - \mathcal{B}_{\boldsymbol{x}}\mathcal{F}\mathcal{D}\hat{\boldsymbol{f}}$ is given in (2.6). We obtain

$$\left\|f-\tilde{f}\right\|_2^2 = \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \left|\hat{f}_{\boldsymbol{k}} - \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})\right|^2 + \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}\setminus\mathcal{I}_{\boldsymbol{M}}} |\hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})|^2 + \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \sum_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} \hat{g}^2_{\boldsymbol{k}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}).$$

In order to minimize this error we have to set $\hat{g}_{\boldsymbol{k}} = 0$ for $\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}\setminus\mathcal{I}_{\boldsymbol{M}}$, i.e., the choice of $\hat{g}_{\boldsymbol{k}}$ via (2.7) is reasonable. We get

$$\left\|f-\tilde{f}\right\|_2^2 = \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \left|\hat{f}_{\boldsymbol{k}}\right|^2 \left(1-\hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})\right)^2 + \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \left|\hat{f}_{\boldsymbol{k}}\right|^2 \sum_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} \hat{d}^2_{\boldsymbol{k}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}). \tag{2.12}$$

Now, it is easy to determine the optimal coefficients $\hat{d}_{\boldsymbol{k}}$ by differentiating with respect to $\hat{d}_{\boldsymbol{k}}$ and setting the result equal to zero for each $\boldsymbol{k}$. We obtain

$$\begin{aligned} 0 &= -2c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})(1-\hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})) + 2\hat{d}_{\boldsymbol{k}} \sum_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \\ &= -2c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) + 2\hat{d}_{\boldsymbol{k}} \sum_{\boldsymbol{r}\in\mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}). \\ \iff \hat{d}_{\boldsymbol{k}} &= \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}. \end{aligned}$$

Inserting the obtained coefficients $\hat{d}_{\boldsymbol{k}}$ into (2.12) we obtain (2.10). Following the standard approach (2.8) we end up with (2.11).

For the adjoint problem we easily compute

$$\int_{\mathbb{T}^d} |\mathcal{T}^* y - \mathcal{D}^*\mathcal{F}^*\mathcal{B}^*_{\boldsymbol{x}} y|^2 \mathrm{d}\boldsymbol{x} = |y|^2 \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \left(1-\hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})\right)^2 + |y|^2 \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \sum_{\boldsymbol{r}\in\mathbb{Z}^d\setminus\{\boldsymbol{0}\}} \hat{d}^2_{\boldsymbol{k}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}).$$

Obviously, we obtain the same optimal deconvolution coefficients $\hat{d}_{\boldsymbol{k}}$. □

Remark 2.2. The resulting error measured in the $\mathcal{L}_2$-norm, see (2.10) or (2.11), strongly depends on the given Fourier coefficients $\hat{f}_{\boldsymbol{k}}$. The window specific error terms are supposed to show significantly different dependencies on the indices $\boldsymbol{k}$ for differing window functions. Thus, the optimal window function may also strongly depend on the properties of the given Fourier coefficients.

As an example, an optimization of the window function in terms of a worst case analysis, see [13] for instance, seems ideally suited in the case that the Fourier coefficients are all of a

comparable size or of a random structure. In contrast, if the Fourier coefficients have a very special structure, e.g. they are known to be subject to a very rapid decrease, we suppose that a more adapted optimization of the window function can lead to substantially better results. □

In the typical case that $|c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})| \ll |c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})|$ for $\boldsymbol{r} \neq \boldsymbol{0}$ the two deconvolution approaches (2.8) and (2.9) are more or less equivalent since

$$\frac{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{\boldsymbol{r}\in\mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})} \approx \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})} = \frac{1}{c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}.$$

The NFFT algorithm as described above requires $\mathcal{O}(|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}| \log |\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}| + m^d N)$ arithmetic operations and can be summarized as follows.

Algorithm 1 (NFFT)**.**
Input: nodes $\boldsymbol{x}_j \in \mathbb{T}^d$ $(j = 1, \ldots, N)$, coefficients $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$ $(\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}, \boldsymbol{M} \in 2\mathbb{N}^d)$, oversampling factor $\boldsymbol{\sigma} \in \mathbb{R}^d$, $\boldsymbol{\sigma} \geq \boldsymbol{1}$.

i) (De-)convolution in Fourier domain:
Define the factors $\hat{d}_{\boldsymbol{k}} \in \mathbb{C}$ for all $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}$, e.g., as given in (2.8) or (2.9).
Set $\hat{g}_{\boldsymbol{k}} := \hat{d}_{\boldsymbol{k}} \hat{f}_{\boldsymbol{k}}$ for all $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}$ and $\hat{g}_{\boldsymbol{k}} := 0$ for $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}} \setminus \mathcal{I}_{\boldsymbol{M}}$.
Complexity: $\mathcal{O}(|\mathcal{I}_{\boldsymbol{M}}|)$.

ii) Use the (inverse) FFT for the computation of the coefficients

$$g_{\boldsymbol{l}} = \frac{1}{|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} \hat{g}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\cdot(\boldsymbol{l}\odot {\boldsymbol{M}_\mathrm{o}}^{-1})}, \boldsymbol{l} \in \mathcal{I}_{\boldsymbol{M}_\mathrm{o}}.$$

Complexity: $\mathcal{O}(|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}| \log |\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|)$.

iii) Convolution in spatial domain: Compute

$$f(\boldsymbol{x}_j) \approx \tilde{f}(\boldsymbol{x}_j) := \sum_{\boldsymbol{l}\in\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}} g_{\boldsymbol{l}} \tilde{\varphi}_\mathrm{t}\left(\boldsymbol{x}_j - \boldsymbol{l} \odot {\boldsymbol{M}_\mathrm{o}}^{-1}\right)$$

for all $j = 1, \ldots, N$. Complexity: $\mathcal{O}(m^d N)$.

Output: $\tilde{f}(\boldsymbol{x}_j) \approx f(\boldsymbol{x}_j)$ for $j = 1, \ldots, N$. □

Using a matrix-vector notation, i.e., considering the matrix representations of the operators introduced in Remark 2.1, we write

$$\boldsymbol{f} = \boldsymbol{A}\hat{\boldsymbol{f}} \approx \boldsymbol{B}\boldsymbol{F}\boldsymbol{D}\hat{\boldsymbol{f}} =: \tilde{\boldsymbol{f}},$$

where we define the vectors $\boldsymbol{f} := [f(\boldsymbol{x}_j)]_{j=1}^N \in \mathbb{C}^N$, $\tilde{\boldsymbol{f}} := [\tilde{f}(\boldsymbol{x}_j)]_{j=1}^N \in \mathbb{C}^N$ and $\hat{\boldsymbol{f}} := [\hat{f}_{\boldsymbol{k}}]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \in \mathbb{C}^{|\mathcal{I}_{\boldsymbol{M}}|}$. The matrix $\boldsymbol{D}$ is a diagonal matrix with entries $\hat{d}_{\boldsymbol{k}}$ and 0, $\boldsymbol{F}$ is the matrix representing the d-dimensional inverse FFT of size $|\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}|$ and $\boldsymbol{B}$ is a block band matrix, assumed that the nodes $\boldsymbol{x}_j$ are correspondingly ordered, with entries $\tilde{\varphi}_\mathrm{t}\left(\boldsymbol{x}_j - \boldsymbol{l} \odot {\boldsymbol{M}_\mathrm{o}}^{-1}\right)$.

The matrix-vector form of the adjoint NFFT is simply obtained by adjoining the matrix representing the NFFT algorithm, i.e., we have

$$\boldsymbol{h} = \boldsymbol{A}^* \boldsymbol{f} \approx \boldsymbol{D}^* \boldsymbol{F}^* \boldsymbol{B}^* \boldsymbol{f},$$

where $\boldsymbol{h} := [h(\boldsymbol{k})]_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}}$. Thus, the derivation of the algorithm is straightforward, see [23, 16].

3 Window functions and error estimates

There are many possible choices for an NFFT window function. In this section we aim to derive accurate bounds for the above derived error in the $\mathcal{L}_2$-norm, which can be evaluated in a fast way. For simplicity we restrict our considerations to the univariate case.

In order to predict the resulting NFFT approximation error measured in the $\mathcal{L}_2$-norm, see (2.10) and (2.11), we have to evaluate the sums

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi}_{\mathrm{t}})}{c^2_k(\tilde{\varphi}_{\mathrm{t}})}$$

for all $k = -M/2, \ldots, M/2 - 1$. In most cases these sums are not known analytically. Thus, it seems reasonable to derive upper bounds by estimating the infinite remainders. Based on this, an upper bound for the overall error is obtained.

In order to get very precise error bounds we only consider window functions for which the Fourier coefficients of the truncated window are known analytically. We will see that the derived error bounds enable a very precise prediction of the occurrent errors. This can be applied in order to develop an automatic parameter tuning, as we describe later on in more detail. The numerical experiments presented in this section have been done with MATLAB.

3.1 Cardinal B-Splines

We define the centered cardinal B-splines by

$$B_1(x) := \begin{cases} 1 & : x \in [-1/2, 1/2), \\ 0 & : \text{else}, \end{cases}$$
$$B_{n+1}(x) := (B_n * B_1)(x),$$

where we denote by $*$ the convolution operator on $\mathbb{R}$. The cardinal B-spline of order n is compactly supported with $\mathrm{supp} B_n = [-n/2, n/2]$ and the Fourier transform is given by

$$\hat{B}_n(\xi) = \mathrm{sinc}^n(\pi\xi),$$

where we define the sinc function

$$\mathrm{sinc}(x) := \begin{cases} \frac{\sin x}{x} & : x \neq 0, \\ 1 & : x = 0. \end{cases}$$

If we define the window function φ via

$$\varphi(x) := B_{2m}(\sigma M x) \quad \text{with} \quad \mathrm{supp}(\varphi) = \left[-\tfrac{m}{\sigma M}, \tfrac{m}{\sigma M}\right],$$

cf. [1, 22], the corresponding Fourier coefficients read as

$$c_k(\tilde{\varphi}) = \frac{1}{\sigma M} \mathrm{sinc}^{2m}\left(\tfrac{\pi k}{\sigma M}\right).$$

In the following we denote by

$$\Phi_n(x) := \sum_{r\in\mathbb{Z}} B_n(r) x^r = B_n(0) + \sum_{r=1}^{\infty} B_n(r)(x^r + x^{-r})$$

the well know Euler-Frobenius functions [24]. The Fourier coefficients of the cardinal B-spline $\hat{B}_n(k)$ fulfill the relation, see [2, page 135] and [21],

$$\sum_{r\in\mathbb{Z}} |\hat{B}_n(k+r)|^2 = \Phi_{2n}(\mathrm{e}^{-2\pi\mathrm{i}k}) = B_{2n}(0) + 2\sum_{r=1}^{\infty} B_{2n}(r)\cos(2\pi kr). \tag{3.1}$$

Applying (3.1) we end up with

$$\sum_{r\in\mathbb{Z}} c^2_{k+r\sigma M}(\tilde{\varphi}) = \frac{1}{\sigma^2 M^2}\sum_{r\in\mathbb{Z}} \hat{B}^2_{2m}\left(\tfrac{k}{\sigma M}+r\right) = \frac{1}{\sigma^2 M^2}\Phi_{4m}\left(\mathrm{e}^{-2\pi\mathrm{i}k/\sigma M}\right).$$

Especially for the introduced B-spline window it is easy to derive an upper bound for the sums

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})}.$$

Utilizing $\sin^2(x+r\pi) = \sin^2 x$ and estimating the infinite sum by an integral we obtain [25]

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})} = \sum_{r\in\mathbb{Z}\setminus\{0\}} \left(\frac{\frac{k}{\sigma M}}{\frac{k}{\sigma M}+r}\right)^{4m} \leq \frac{8m}{4m-1}\left(\frac{\frac{|k|}{\sigma M}}{\frac{|k|}{\sigma M}-1}\right)^{4m}, \tag{3.2}$$

which can be evaluated in a numerically stable way. In contrast, the evaluation via

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})} = \frac{\Phi_{4m}\left(\mathrm{e}^{-2\pi\mathrm{i}k/\sigma M}\right) - \operatorname{sinc}^{4m}\left(\frac{\pi k}{\sigma M}\right)}{\operatorname{sinc}^{4m}\left(\frac{\pi k}{\sigma M}\right)}$$

seems numerically unstable. The error terms obtained by using the $\mathcal{L}_2$-optimized coefficients $\hat{d}_k$ can be estimated by

$$\begin{aligned}
\frac{\sum\limits_{r\in\mathbb{Z}\setminus\{0\}} c^2_{k+r\sigma M}(\tilde{\varphi})}{\sum\limits_{r\in\mathbb{Z}} c^2_{k+r\sigma M}(\tilde{\varphi})} &= \frac{1}{\sum\limits_{r\in\mathbb{Z}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})}} \sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})} \\
&= \frac{\operatorname{sinc}^{4m}\left(\frac{\pi k}{\sigma M}\right)}{\Phi_{4m}\left(\mathrm{e}^{-2\pi\mathrm{i}k/\sigma M}\right)} \sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})} \\
&< \frac{\operatorname{sinc}^{4m}\left(\frac{\pi k}{\sigma M}\right)}{\Phi_{4m}\left(\mathrm{e}^{-2\pi\mathrm{i}k/\sigma M}\right)} \frac{8m}{4m-1}\left(\frac{\frac{|k|}{\sigma M}}{\frac{|k|}{\sigma M}-1}\right)^{4m}.
\end{aligned} \tag{3.3}$$

In general the single error terms for the optimized deconvolution approach are smaller than those obtained by the standard approach by the factors

$$\frac{1}{\sum\limits_{r\in\mathbb{Z}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})}} = \frac{1}{1+\sum\limits_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})}} \leq 1.$$

For the B-spline window we have equality for $k=0$, see (3.2). A lower bound of the prefactors is given by

$$\frac{1}{1+\sum\limits_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma M}(\tilde{\varphi})}{c^2_k(\tilde{\varphi})}} \geq \frac{1}{1+\frac{8m}{4m-1}\left(\frac{|k|/\sigma M}{|k|/\sigma M-1}\right)^{4m}} \overset{\frac{|k|}{\sigma M}=\frac{1}{2}}{\geq} \frac{1}{1+\frac{8m}{4m-1}} = \frac{4m-1}{12m-1} \approx \frac{1}{3},$$

i.e., we can at best reduce the overall error by a factor of 3.

Especially if oversampling is applied, i.e., $\sigma > 1$, or if the coefficients $\hat{f}_k$ are comparatively small for $|k|/\sigma M \approx 1/2$, we expect only insignificant improvements by using the optimized coefficients (2.9). We illustrate this by the following example.

Example 3.1. We compare the results of the above described error estimates for two different choices of $\hat{f}_k$. We set $M := 64$ and choose

$$\hat{f}_k := \frac{1}{1+k^2} \quad \text{for } k \in \mathcal{I}_{64} \tag{3.4}$$

as a first example and

$$\hat{f}_k := \mathrm{e}^{-(k/5)^2} \quad \text{for } k \in \mathcal{I}_{64} \tag{3.5}$$

in a second test. Note the difference between the two examples. The coefficients $\hat{f}_k$ as given in (3.5) tend to zero exponentially fast, i.e., for large values of k the factors $\hat{f}_k$ only have an insignificant influence on the overall error. In contrast, the coefficients given in (3.4) tend to zero very slowly.

In Figure 3.1 we plot the estimate of $\|f - \tilde{f}\|_2^2$ with respect to m for the two different settings. We have

$$\|f-\tilde{f}\|_2^2 < \begin{cases} \dfrac{8m}{4m-1} \displaystyle\sum_{k\in\mathcal{I}_M} |\hat{f}_k|^2 \left(\dfrac{\frac{|k|}{\sigma M}}{\frac{|k|}{\sigma M} - 1} \right)^{4m} & : \hat{d}_k \text{ via (2.8)}, \\ \dfrac{8m}{4m-1} \displaystyle\sum_{k\in\mathcal{I}_M} |\hat{f}_k|^2 \dfrac{\left(\operatorname{sinc}\frac{\pi k}{\sigma M}\right)^{4m}}{\Phi_{4m}\left(\mathrm{e}^{-2\pi \mathrm{i} k/\sigma M}\right)} \left(\dfrac{\frac{|k|}{\sigma M}}{\frac{|k|}{\sigma M} - 1} \right)^{4m} & : \hat{d}_k \text{ via (2.9)}. \end{cases} \tag{3.6}$$

For the coefficients (3.4) we observe that the error can be somewhat reduced by using the optimized coefficients (2.9) in the case $\sigma = 1$. Already for a small oversampling factor $\sigma = 5/4$ the errors are almost the same. In the case that the very fast decreasing coefficients (3.5) are given, already for $\sigma = 1$ no difference between the two approaches can be seen. □

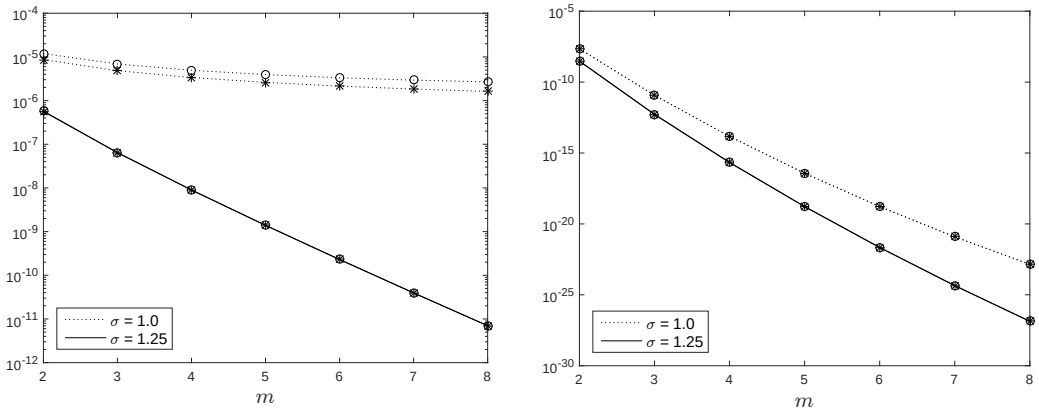

Figure 3.1: Estimated errors (3.6) for different values of m and σ. We set $\hat{f}_k := (1+k^2)^{-1}$, $k \in \mathcal{I}_{64}$, (left) and $\hat{f}_k := \mathrm{e}^{-(k/5)^2}$, $k \in \mathcal{I}_{64}$, (right). We plot the error terms for the two different approaches to set the deconvolution coefficients $\hat{d}_k$ (variant 1 (o): define $\hat{d}_k$ by (2.8), variant 2 (∗): set $\hat{d}_k$ as given in (2.9)).

3.2 Modified B-spline window

We introduce a shape parameter $b \in {}^1\!/\!_2\,\mathbb{N} = \{{}^1\!/\!_2, 1, {}^3\!/\!_2, 2, \dots\}$ and define the modified B-spline window function φ by

$$\varphi(x) := B_{2b}\left(\frac{\sigma M b}{m} x\right).$$

As for the standard B-spline window we have $\operatorname{supp}(\varphi) = [-{}^m\!/\!_{\sigma M}, {}^m\!/\!_{\sigma M}]$, but we also allow a different order of the B-spline, which is $2b$. The Fourier coefficients of the periodic version $\tilde{\varphi}$ are given by

$$c_k(\tilde{\varphi}) = \frac{m}{\sigma M b} \operatorname{sinc}^{2b}\left(\frac{m\pi k}{\sigma M b}\right).$$

For $b = m$ we are in the case of the standard B-spline window and can apply the error estimation described in the previous section.

In the following we restrict our considerations to the case $b > {}^m\!/\!_2$, since

- for $b \leq {}^m\!/\!_2$ some Fourier coefficients may be equal to zero. As an example, if we set $b := {}^m\!/\!_2$, $\sigma = 1$ then $c_{M/2+r\sigma M}(\tilde{\varphi}) = 0$ for all $r \in \mathbb{Z}$, i.e., the deconvolution coefficients (2.8) and (2.9) are not defined.
- In addition it is easy to see that for $b = {}^m\!/\!_2$ we obtain a larger error than for the standard case, where $b = m$. For each $r \in \mathbb{Z} \setminus \{0\}$ we have

$$\frac{c_{k+r\sigma M}^2(\tilde{\varphi})}{c_k^2(\tilde{\varphi})} \overset{\frac{m}{b} \in \mathbb{N}}{=} \left(\frac{\frac{k}{\sigma M}}{\frac{k}{\sigma M} + r}\right)^{4b} \overset{b = \frac{m}{2}}{=} \left(\frac{\frac{k}{\sigma M}}{\frac{k}{\sigma M} + r}\right)^{2m} \geq \left(\frac{\frac{k}{\sigma M}}{\frac{k}{\sigma M} + r}\right)^{4m}.$$

If $b > {}^m\!/\!_2$ and $b \neq m$ we cannot exploit the periodicity of $\sin^2(\cdot)$. We proceed as follows. For some $R_k \in \mathbb{N}$ we obtain

$$\begin{aligned}\sum_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi}) &= \frac{m^2}{\sigma^2 M^2 b^2} \sum_{r \in \mathbb{Z}\setminus\{0\}} \operatorname{sinc}^{4b}\left(\tfrac{m\pi}{b}\left[\tfrac{k}{\sigma M} + r\right]\right) \\ &\leq \frac{m^2}{\sigma^2 M^2 b^2}\left(\sum_{0<|r|\leq R_k} \operatorname{sinc}^{4b}\left(\tfrac{m\pi}{b}\left[\tfrac{k}{\sigma M} + r\right]\right) + \sum_{|r|>R_k} \frac{1}{\left(\frac{m\pi}{b}\right)^{4b}\left(\frac{k}{\sigma M} + r\right)^{4b}}\right),\end{aligned}$$

where

$$\begin{aligned}\sum_{|r|>R_k} \frac{1}{\left(\frac{k}{\sigma M} + r\right)^{4b}} &< \int_{R_k}^{\infty} \frac{\mathrm{d}r}{\left(\frac{k}{\sigma M} + r\right)^{4b}} + \int_{R_k}^{\infty} \frac{\mathrm{d}r}{\left(\frac{k}{\sigma M} - r\right)^{4b}} \\ &= \frac{1}{4b-1}\left(\frac{1}{\left(\frac{k}{\sigma M} + R_k\right)^{4b-1}} - \frac{1}{\left(\frac{k}{\sigma M} - R_k\right)^{4b-1}}\right).\end{aligned}$$

In summary we have

$$\sum_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi}) < s\left(\tfrac{k}{\sigma M}\right),$$

where we set

$$s\left(\tfrac{k}{\sigma M}\right) := \frac{m^2}{\sigma^2 M^2 b^2}\left(\sum_{0<|r|\leq R_k} \operatorname{sinc}^{4b}\left(\tfrac{m\pi}{b}\left[\tfrac{k}{\sigma M} + r\right]\right) + \frac{\left(\frac{k}{\sigma M} + R_k\right)^{1-4b} - \left(\frac{k}{\sigma M} - R_k\right)^{1-4b}}{\left(\frac{m\pi}{b}\right)^{4b}(4b-1)}\right)$$

for some $R_k \in \mathbb{N}$. In order to get a precise estimate we start with $R_k := 1$ and increase R_k step by step until the remainder

$$\frac{m^2}{\sigma^2 M^2 b^2} \frac{\left(\frac{k}{\sigma M} + R_k\right)^{1-4b} - \left(\frac{k}{\sigma M} - R_k\right)^{1-4b}}{\left(\frac{m\pi}{b}\right)^{4b} (4b-1)}$$

is comparatively small.

In order to estimate the sums

$$\frac{\sum\limits_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi})}{\sum\limits_{r \in \mathbb{Z}} c_{k+r\sigma M}^2(\tilde{\varphi})}$$

we use (3.3) in the case $b = m$. Otherwise we exploit that a function of the form $f(y) = \frac{y}{c^2+y}$ is monotonically increasing and obtain

$$\frac{\sum\limits_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi})}{\sum\limits_{r \in \mathbb{Z}} c_{k+r\sigma M}^2(\tilde{\varphi})} < \frac{s\left(\frac{k}{\sigma M}\right)}{c_k^2(\tilde{\varphi}) + s\left(\frac{k}{\sigma M}\right)}. \tag{3.7}$$

Example 3.2. We consider the case $m = 4$. For different shape parameters $b \in {}^1\!/_2\,\mathbb{N}$, we plot the above derived estimates of the terms

$$\frac{\sum\limits_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi})}{\sum\limits_{r \in \mathbb{Z}} c_{k+r\sigma M}^2(\tilde{\varphi})} = \frac{\sum\limits_{r \in \mathbb{Z}\setminus\{0\}} \operatorname{sinc}^{4b}\left(\frac{m\pi}{b}\left[\frac{k}{\sigma M} + r\right]\right)}{\sum\limits_{r \in \mathbb{Z}} \operatorname{sinc}^{4b}\left(\frac{m\pi}{b}\left[\frac{k}{\sigma M} + r\right]\right)} \tag{3.8}$$

in Figure 3.2. Since we only expect small differences between the two deconvolution approaches as well as for overview purposes we only plot the error terms for the $\mathcal{L}_2$ optimized deconvolution variant.

It seems not reasonable to use a shape parameter $b > m$ since the error terms are larger than in the case $b = m$ for all k. Depending on the given coefficients $\hat{f}_k$ and the chosen oversampling factor, a shape parameter $b \in \{{}^5\!/_2, 3, {}^7\!/_2, 4\}$ is supposed to be optimal. □

Example 3.3. We consider again the two sets of Fourier coefficients as given in (3.4) as well as (3.5). We estimate the quadratic error $\|f - \tilde{f}\|_2^2$ as described above for $m \in \{4, 8\}$, where we choose different shape parameters $b \in {}^1\!/_2\,\mathbb{N}$ and oversampling factors σ. We plot the results in Figure 3.3. For the slowly decreasing factors (3.4) a shape parameter $b < m$ is optimal in most cases, i.e., we obtain smaller errors than for the standard B-spline window, where $b = m$. In contrast, for the very fast decreasing coefficients (3.5) the minimal error is in most cases obtained by setting $b = m$. As expected, we only see small differences between the two deconvolution schemes. □

In summary, we conjecture that in general a shape parameter $b \in {}^1\!/_2\mathbb{N} : {}^m\!/_2 < b \leq m$ is optimal, i.e., we can reduce the error in certain cases by using a B-spline of a lower order while keeping the support constant. The optimal value for the shape parameter strongly depends on the given Fourier coefficients. The numerical examples show that especially for slowly decreasing or also randomly structured coefficients the error is reduced significantly by adjusting the order of the B-spline appropriately.

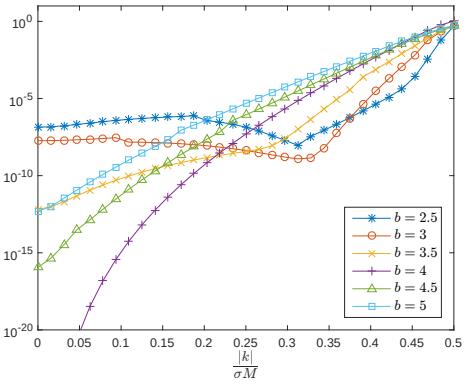

Figure 3.2: Estimated error terms (3.8) for $m = 4$ and different shape parameters b. We plot the results with respect to $\frac{|k|}{\sigma M} \in [0, 1/2]$.

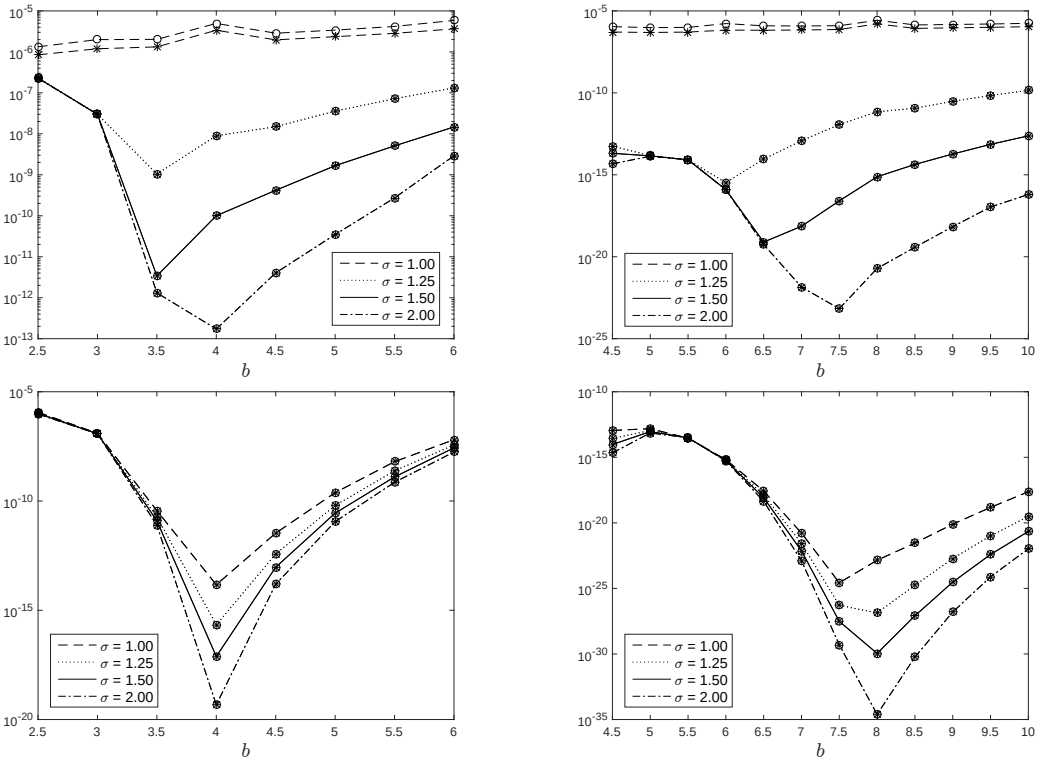

Figure 3.3: Estimate of $\|f - \tilde{f}\|_2^2$ for the modified B-spline window, where we consider the sets of Fourier coefficients $\hat{f}_k$ given in (3.4) as well as (3.5) (top down). We plot the error with respect to the shape parameter b, where we set $m = 4$ (left) and $m = 8$ (right), for different oversampling factors σ. We compare the error terms for the two different approaches to set $\hat{d}_k$ (variant 1 (o): define $\hat{d}_k$ by (2.8), variant 2 (*): set $\hat{d}_k$ as given in (2.9)).

3.3 Bessel window

In the following we consider a window function which is constructed based on the Kaiser-Bessel NFFT window as introduced in [22, Appendix]. In order to get a window function φ with compact support we interchange the roles of time and frequency domain. We refer to the resulting function as the Bessel (I_0) window function, cf. [19], which is also found under the name Kaiser-Bessel function in the literature [15, 8, 12].

For some shape parameter $b > 0$ we define the Bessel window function by

$$\varphi(x) := \frac{1}{2} \begin{cases} I_0\left(b\sqrt{m^2 - \sigma^2 M^2 x^2}\right) & : |x| \leq \frac{m}{\sigma M}, \\ 0 & : \text{else}, \end{cases}$$

where I_0 denotes the modified zero-order Bessel function. The corresponding Fourier coefficients are of the form

$$c_k(\tilde{\varphi}) = \frac{1}{\sigma M} \begin{cases} \dfrac{\sinh\left(m\sqrt{b^2 - 4\pi^2 k^2/(\sigma^2 M^2)}\right)}{\sqrt{b^2 - 4\pi^2 k^2/(\sigma^2 M^2)}} & : |k| \leq \frac{\sigma M b}{2\pi}, \\ m \operatorname{sinc}\left(m\sqrt{4\pi^2 k^2/(\sigma^2 M^2) - b^2}\right) & : \text{else}, \end{cases}$$

and tend to zero only with order 1, since φ is not continuous in $x = \pm m/\sigma M$.

The error sums $\sum_{r \in \mathbb{Z}\setminus\{0\}} c^2_{k+r\sigma M}(\tilde{\varphi})$ can be estimated as follows. For $R_k > \frac{|k|}{\sigma M} + \frac{b}{2\pi}$ we have

$$\begin{aligned} \sum_{r \in \mathbb{Z}\setminus\{0\}} c^2_{k+r\sigma M}(\tilde{\varphi}) &= \sum_{1<|r|\leq R_k} c^2_{k+r\sigma M}(\tilde{\varphi}) + \frac{m^2}{\sigma^2 M^2} \sum_{r=R_k+1}^{\infty} \operatorname{sinc}^2\left(m\sqrt{4\pi^2(|k|/\sigma M \pm r)^2 - b^2}\right) \\ &\leq \sum_{1<|r|\leq R_k} c^2_{k+r\sigma M}(\tilde{\varphi}) + \frac{1}{\sigma^2 M^2} \sum_{r=R_k+1}^{\infty} \frac{1}{4\pi^2(|k|/\sigma M \pm r)^2 - b^2} \\ &< \sum_{1<|r|\leq R_k} c^2_{k+r\sigma M}(\tilde{\varphi}) + \frac{1}{\sigma^2 M^2} \int_{R_k}^{\infty} \frac{\mathrm{d}r}{4\pi^2(|k|/\sigma M \pm r)^2 - b^2}, \end{aligned}$$

where the integrals can be computed by

$$\begin{aligned} \int_{R_k}^{\infty} \frac{\mathrm{d}r}{4\pi^2(|k|/\sigma M \pm r)^2 - b^2} &= \frac{1}{2\pi} \left(\int_{2\pi(|k|/\sigma M + R_k)}^{\infty} + \int_{-\infty}^{2\pi(|k|/\sigma M - R_k)} \right) \frac{\mathrm{d}r}{r^2 - b^2} \\ &= \frac{\ln\left|\frac{2\pi(|k|/\sigma M - R_k) - b}{2\pi(|k|/\sigma M - R_k) + b}\right| + \ln\left|\frac{2\pi(|k|/\sigma M + R_k) + b}{2\pi(|k|/\sigma M + R_k) - b}\right|}{4\pi b}. \end{aligned} \tag{3.9}$$

In order to get a precise estimate we suggest to proceed as follows, cf. Section 3.2. We start with $R_k := \left\lceil \frac{|k|}{\sigma M} + \frac{b}{2\pi} \right\rceil$ and increase R_k step by step until the remainder (3.9) is comparatively small.

If we define the coefficients $\hat{d}_k$ via (2.8) then

$$s(k/\sigma M) := \sum_{1<|r|\leq R_k} c^2_{k+r\sigma M}(\tilde{\varphi}) + \frac{1}{\sigma^2 M^2} \frac{\ln\left|\frac{2\pi(|k|/\sigma M - R_k) - b}{2\pi(|k|/\sigma M - R_k) + b}\right| + \ln\left|\frac{2\pi(|k|/\sigma M + R_k) + b}{2\pi(|k|/\sigma M + R_k) - b}\right|}{4\pi b}$$

gives an upper bound for the corresponding error terms. In the case that we use the optimized coefficients (2.9) we apply (3.7).

Example 3.4. We investigate the behavior of the error in the $\mathcal{L}_2$-norm with respect to b. By default, the shape parameter b is set to [22, Appendix]

$$b_0 := 2\pi(1 - 1/2\sigma).$$

If we choose $\sigma \in \{1, 5/4\}$, for instance, we obtain the following standard values for b.

$$b_0 = \begin{cases} \pi \approx 3.14 & : \sigma = 1 \,, \\ \frac{6}{5}\pi \approx 3.77 & : \sigma = 5/4 \,. \end{cases}$$

In this example we consider again the two different sets of Fourier coefficients given in (3.4) and (3.5), respectively.

For the slowly decreasing coefficients (3.4) the estimated errors are plotted in Figure 3.4, first row. For $\sigma = 1$ we can see a small difference in the errors obtained for the two different approaches to set $\hat{d}_k$. The optimal values for b nearly coincide with the suggested default values b_0. The results for the coefficients (3.5) are depicted below. The predicted errors for the two variants to set $\hat{d}_k$ are nearly the same. On the other hand, we obtain considerably different optimal shape parameters. □

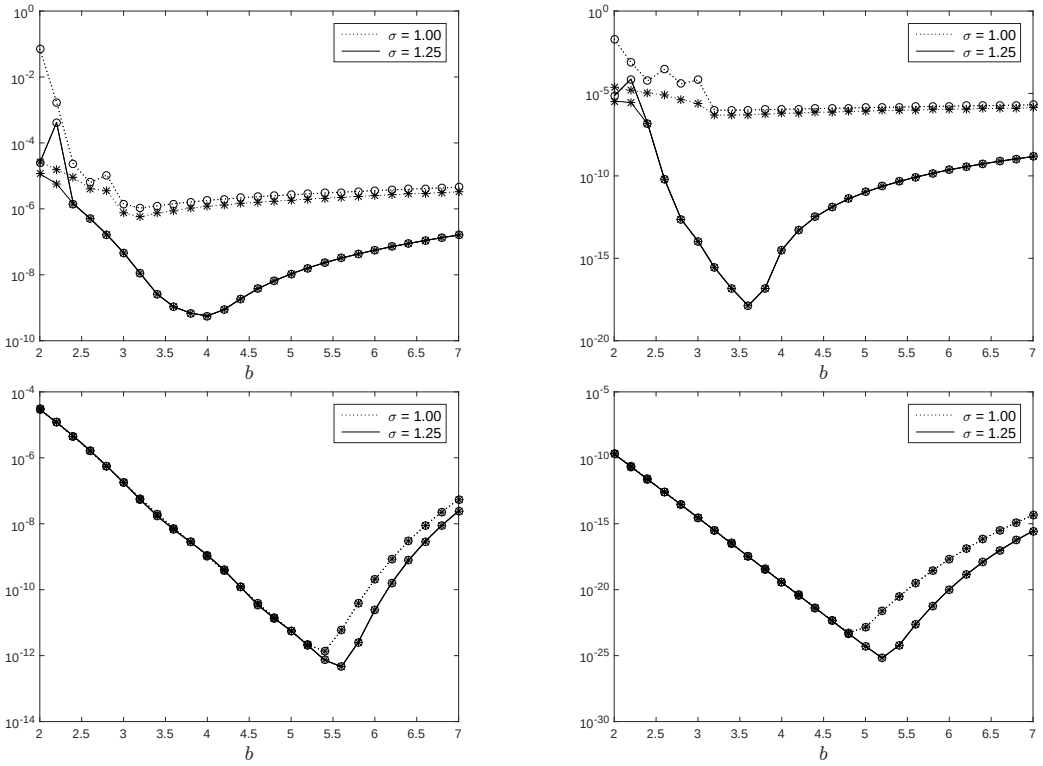

Figure 3.4: Estimate of $\|f - \tilde{f}\|_2^2$ for the Bessel window, where we consider the sets of Fourier coefficients $\hat{f}_k$ as given in (3.4) as well as (3.5) (top down). We plot the error with respect to the shape parameter b, where we set $m = 3$ (left) and $m = 6$ (right), for $\sigma \in \{1, 5/4\}$. We compare the obtained errors for the two variants to set the coefficients $\hat{d}_k$ (variant 1 (∘): define $\hat{d}_k$ by (2.8), variant 2 (∗): set $\hat{d}_k$ as given in (2.9)).

3.4 Gaussian window function

For some shape parameter $b > 0$ we define the Gaussian window function by [5, 22, 9]

$$\varphi(x) = \frac{1}{\sqrt{\pi b}} \mathrm{e}^{-\sigma^2 M^2 x^2 / b}.$$

The Fourier coefficients of the periodic version $\tilde{\varphi}$ are given by

$$c_k(\tilde{\varphi}) = \frac{1}{\sigma M} \mathrm{e}^{-b\pi^2 k^2/(\sigma^2 M^2)}$$

and tend to zero exponentially fast in k. The Fourier coefficients of the truncated version $\tilde{\varphi}_\mathrm{t} := \tilde{\varphi} \cdot \tilde{\chi}_{[-m/\sigma M, m/\sigma M]}$ can be expressed by

$$\begin{aligned}
c_k(\tilde{\varphi}_\mathrm{t}) = \hat{\varphi}_\mathrm{t}(k) &= \frac{1}{\sqrt{\pi b}} \int_{-m/\sigma M}^{m/\sigma M} \mathrm{e}^{-\sigma^2 M^2 x^2/b^2} \mathrm{e}^{2\pi \mathrm{i} k x} \mathrm{d}x \\
&= \frac{\mathrm{e}^{-b\pi^2 k^2/(\sigma^2 M^2)}}{\sqrt{\pi b}} \int_{-m/\sigma M}^{m/\sigma M} \mathrm{e}^{-\left[\sigma M x/\sqrt{b} - \mathrm{i}\pi k \sqrt{b}(\sigma M)\right]^2} \mathrm{d}x \\
&= \frac{\mathrm{e}^{-b\pi^2 k^2/(\sigma^2 M^2)}}{2\sigma M} \left[\operatorname{erf}\left(\tfrac{m}{\sqrt{b}} + \mathrm{i}\tfrac{\pi k \sqrt{b}}{\sigma M} \right) + \operatorname{erf}\left(\tfrac{m}{\sqrt{b}} - \mathrm{i}\tfrac{\pi k \sqrt{b}}{\sigma M} \right) \right] \\
&= \frac{\mathrm{e}^{-b\pi^2 k^2/(\sigma^2 M^2)}}{\sigma M} \operatorname{Re}\left[\operatorname{erf}\left(\tfrac{m}{\sqrt{b}} + \mathrm{i}\tfrac{\pi k \sqrt{b}}{\sigma M} \right) \right],
\end{aligned}$$

where we denote by erf the well known error function. In contrast to $c_k(\tilde{\varphi})$ the Fourier coefficients of the truncated Gaussian tend to zero only with order 1, since the corresponding window function φ_t is not continuous in $x = \pm m/\sigma M$.

Example 3.5. We consider the following three possibilities to set $\hat{d}_k$ and give some comparisons for the Gaussian window function.

$$\|f - \tilde{f}\|_2^2 = \sum_{k \in \mathcal{I}_M} \left| \hat{f}_k \right|^2 \cdot \begin{cases} \sum\limits_{r \in \mathbb{Z}\setminus\{0\}} \frac{c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t})}{c_k^2(\tilde{\varphi}_\mathrm{t})} & : \hat{d}_k \text{ via (2.8)}, \\ \frac{\sum\limits_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t})}{\sum\limits_{r \in \mathbb{Z}} c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t})} & : \hat{d}_k \text{ via (2.9).}, \\ \left(1 - \frac{c_k(\tilde{\varphi}_\mathrm{t})}{c_k(\tilde{\varphi})}\right)^2 + \sum\limits_{r \in \mathbb{Z}\setminus\{0\}} \frac{c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t})}{c_k^2(\tilde{\varphi})} & : \hat{d}_k = \frac{1}{c_k(\tilde{\varphi})}. \end{cases}$$

Commonly, the NFFT deconvolution step is done by using the Fourier coefficients $c_k(\tilde{\varphi})$ of the non truncated Gaussian. Since the convolution in spatial domain is actually done with a truncated Gaussian, it seems reasonable to use the Fourier coefficients of the truncated function, see Section 2. In order to compare the two approaches we also take into consideration the variant to set $\hat{d}_k := c_k^{-1}(\tilde{\varphi})$.

Estimating the remainder of the series

$$\sum_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t}) = \sum_{0<|r|\le R_k} c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t}) + \sum_{|r|>R_k} c_{k+r\sigma M}^2(\tilde{\varphi}_\mathrm{t})$$

after truncation proves to be rather difficult, which is due to the presence of the complex valued error function. Thus, we approximate the series by the first part, where we increase

R_k step by step as long as the sum grows significantly. In order to evaluate the complex valued error function in MATLAB we use the Faddeeva package by Steven G. Johnson, see [14] for the implementation in C.

In the following we evaluate the above error terms for the Fourier coefficients given in (3.4) as well as in (3.5) for different shape parameters b. Again, we see that the optimal value for the shape parameter depends on both, the given coefficients $\hat{f}_k$ and the parameters used within the NFFT (m, σ, definition of $\hat{d}_k$).

In many applications the shape parameter b is chosen as follows

$$b_0 := \frac{2\sigma}{2\sigma - 1}\frac{m}{\pi},$$

see [25, 4, 9]. For example, if we choose $m \in \{3, 6\}$ combined with $\sigma \in \{1, 5/4\}$ we obtain the values listed in Table 3.1.

	$\sigma = 1$	$\sigma = 5/4$
$m = 3$	$b_0 \approx 1.91$	$b_0 \approx 1.59$
$m = 6$	$b_0 \approx 3.82$	$b_0 \approx 3.18$

Table 3.1: Value of b_0 for different combinations of m and σ.

We evaluate the quadratic error $\|f - \tilde{f}\|_2^2$ for the given Fourier coefficients as described above for different parameter sets. For fixed m and σ the results are plotted in Figure 3.5. In most cases the optimal shape parameter differs from the standard values, which are given in Table 3.1. Which value for b is optimal, obviously depends on the given Fourier coefficients $\hat{f}_k$ as well as on the chosen oversampling factor σ.

In general, better results are obtained by using the Fourier coefficients of the truncated Gauss window. For the coefficients (3.4) we obtain slightly better results by using the optimized factors (2.9) in the case $\sigma = 1$. If the very rapidly decreasing coefficients (3.5) are given, the two variants (2.8) and (2.9) again produce almost the same errors, already for $\sigma = 1$. □

3.5 Comparison

In the previous section we investigated the error of the NFFT in the $\mathcal{L}_2$-norm. Some concrete examples showed that, especially in the case that the given Fourier coefficients do not decrease rapidly and $\sigma \approx 1$, the accuracy can be somewhat improved by using the optimized coefficients (2.9) within the NFFT (Algorithm 1).

The Bessel and also the Gaussian window function depend on a shape parameter b, which we aim to choose optimal for given Fourier coefficients. We also introduced the modified B-spline window involving a shape parameter, which allows us to choose the order of the B-spline independently from the support parameter m. The presented numerical examples show that in many cases the optimal values for the shape parameter differ from the corresponding default values, which are widely used. Thus, depending on the given Fourier coefficients the resulting errors can be significantly reduced by adjusting the shape parameter appropriately.

For a comparison of all introduced window functions, we consider again the two sets of Fourier coefficients (3.4) and (3.5). As described in the previous paragraphs, we estimate the quadratic error (2.10) for $m \in \{2, \ldots, 8\}$ and $\sigma \in \{1, 5/4\}$. We plot the results in Figure 3.6, where for each m we choose the shape parameter b for which the predicted error is

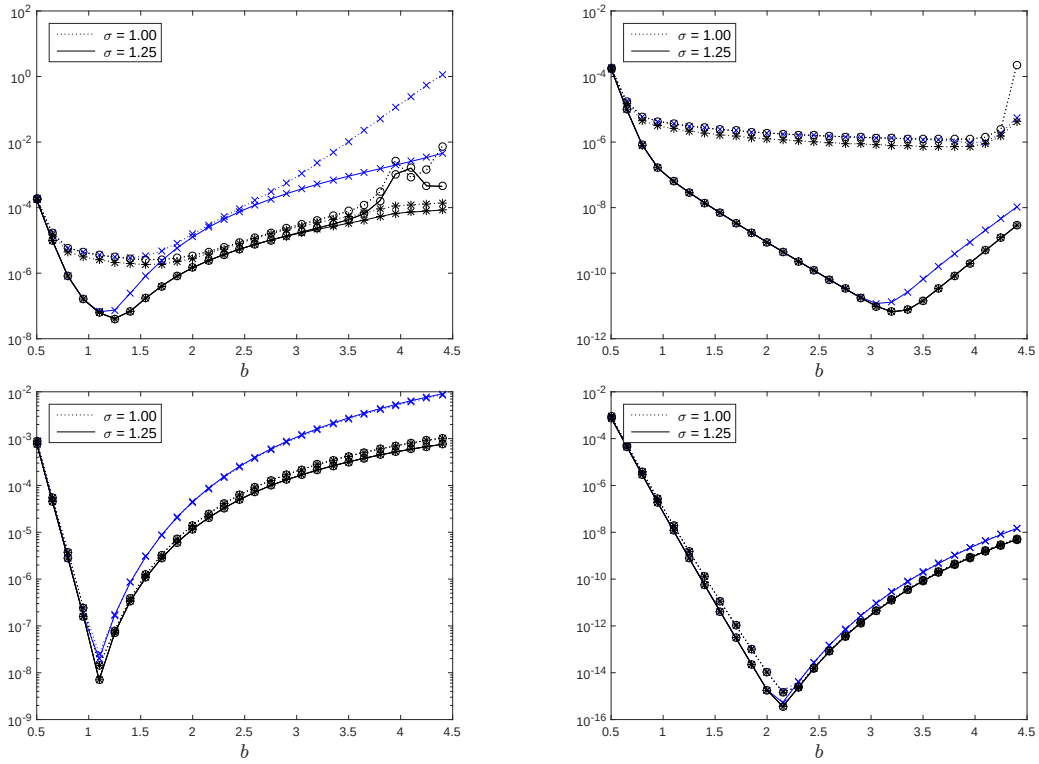

Figure 3.5: Estimate of $\|f - \tilde{f}\|_2^2$ for the Gaussian window, where we consider the sets of Fourier coefficients $\hat{f}_k$ as given is (3.4) as well as (3.5) (top down). We plot the error with respect to the shape parameter b, where we set $m = 3$ (left) and $m = 6$ (right), for $\sigma \in \{1, 5/4\}$. We compare the obtained errors for the three variants to set the coefficients $\hat{d}_k$ (variant 1 (o): $\hat{d}_k$ via (2.8), variant 2 (∗): $\hat{d}_k$ via (2.9), variant 3 (x): $\hat{d}_k := c_k^{-1}(\tilde{\varphi})$).

minimal, see Figure 3.7 for the optimal shape parameters b_{opt}. We can see that the optimal shape parameters adopt significantly different values for the two considered sets of Fourier coefficients.

The Bessel window function yields the smallest error in most cases. For the fast decreasing coefficients (3.5) the B-spline window performs better in the case that the support parameter m is very small.

4 Verification of the theoretical estimates

We use a simple implementation of the univariate NFFT with the introduced window functions in MATLAB in order to verify the theoretical estimates.

We revisit the examples presented in Section 3 and compare the theoretical error estimates with experimental measurements. We compute the sums

$$\tilde{f}(x_j) \approx f(x_j) = \sum_{k \in \mathcal{I}_M} \hat{f}_k \mathrm{e}^{-2\pi \mathrm{i} k x_j}, \quad j = 1, \ldots, N,$$

where we consider the two sets of Fourier coefficients (3.4) and (3.5), for $N = 500$ randomly

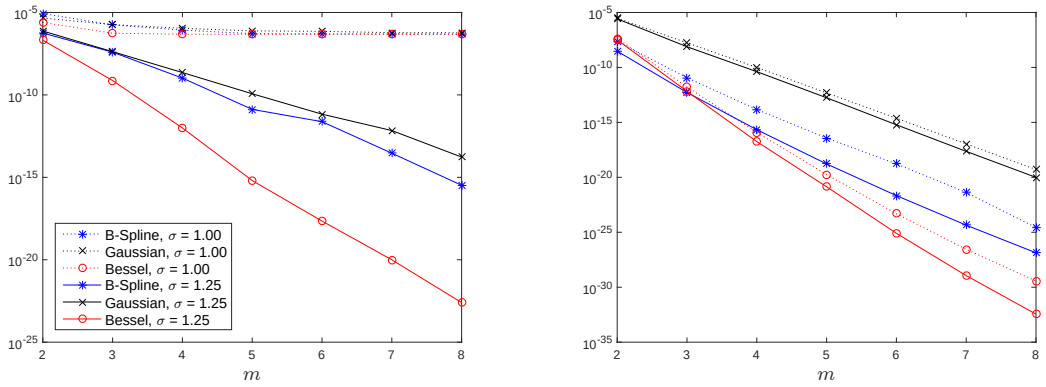

Figure 3.6: Estimated quadratic errors (2.10) for the sets of Fourier coefficients (3.4) and (3.5) (from left to right) with respect to m. We used different window functions and oversampling factors (see legend).

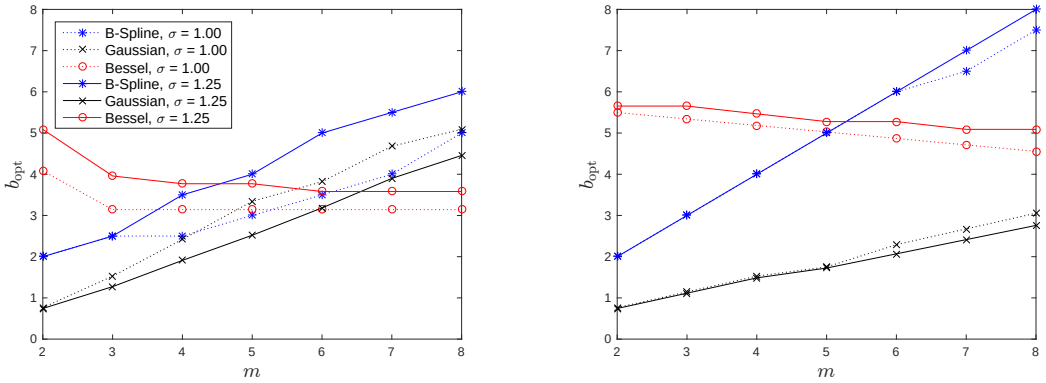

Figure 3.7: Corresponding optimal values for the shape parameter b, see Figure 3.6

chosen nodes $x_j \in \mathbb{T}, j = 1, \ldots, N$. Instead of the quadratic error $\|\tilde{f} - f\|_2^2$ we compute

$$\Delta f^2 := \frac{1}{N} \sum_{j=1}^{N} \left[f(x_j) - \tilde{f}(x_j) \right]^2 . \tag{4.1}$$

Since we have seen in the last section that the optimized deconvolution scheme (2.9) does not serve clear benefits compared to the standard approach (2.8), we only run our algorithm applying the standard deconvolution (`o`). We plot the predicted as well as the measured errors in the following figures. The predicted errors are represented by the line plots, whereas the circles (`o`) mark the measured errors. For the Gaussian window function we also plot the results obtained by using the Fourier coefficients of the non truncated function, i.e., we set $\hat{d}_k := c_k^{-1}(\tilde{\varphi})$. In this case the predicted errors are represented by blue lines and the measured errors are marked by the crosses (`x`).

We can see that the theoretical estimates match quite well with the measured errors (4.1). Of course, the measured errors are in some cases somewhat smaller than the predicted errors, which is due to the fact that the computation of the predicted errors is based on computing upper bounds of the involved error terms, see Section 3.

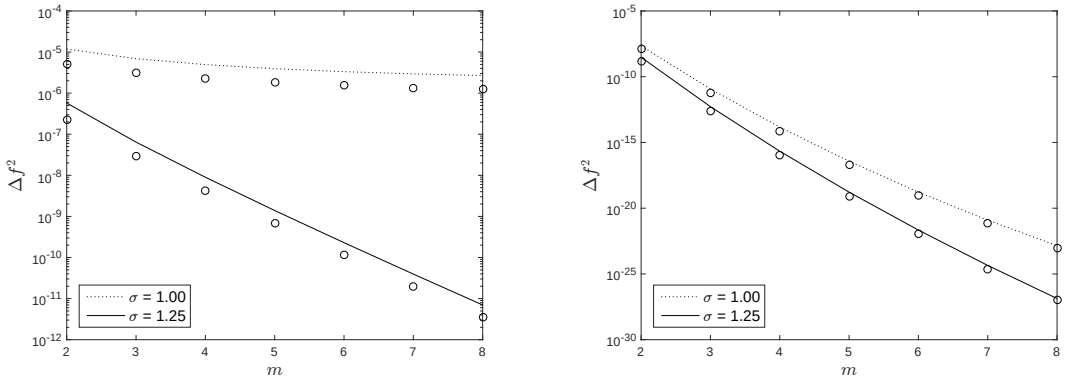

Figure 4.1: Predicted errors in the $\mathcal{L}_2$-norm (lines) as well as the measured errors (4.1) (marked by o) for the Standard B-spline window. We consider the Fourier coefficients as given in (3.4) as well as (3.5) (from left to right).

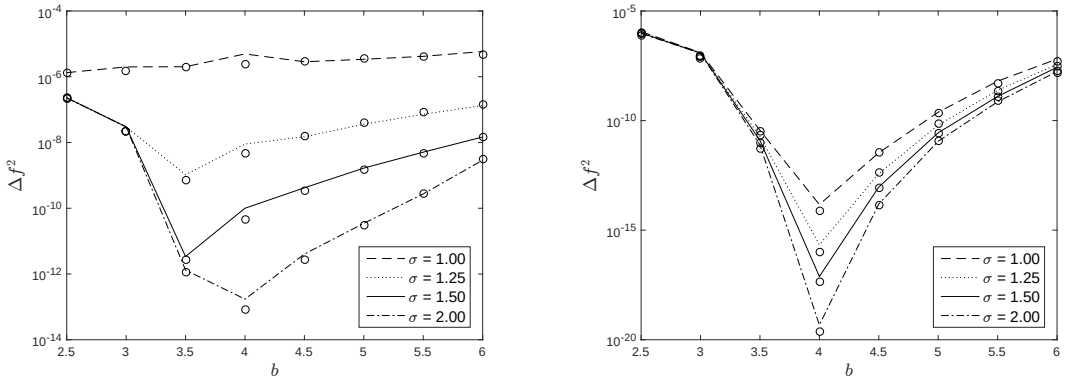

Figure 4.2: Modified B-spline window, Fourier coefficients (3.4) on the left and (3.5) on the right, $m = 4$.

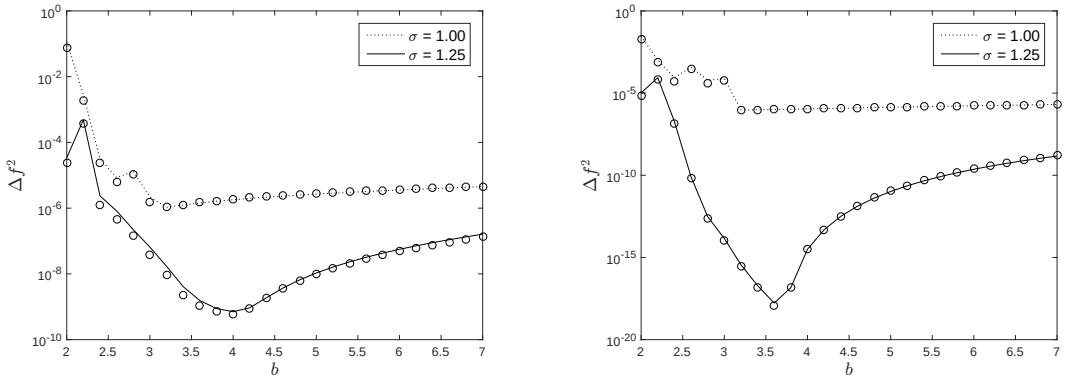

Figure 4.3: Bessel window, Fourier coefficients (3.4), $m := 3$ (left) and $m := 6$ (right).

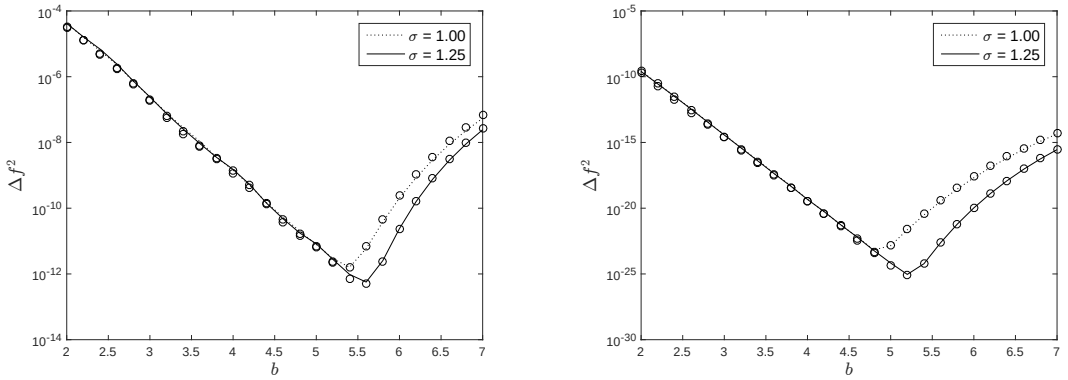

Figure 4.4: Bessel window, Fourier coefficients (3.5), $m := 3$ (left) and $m := 6$ (right).

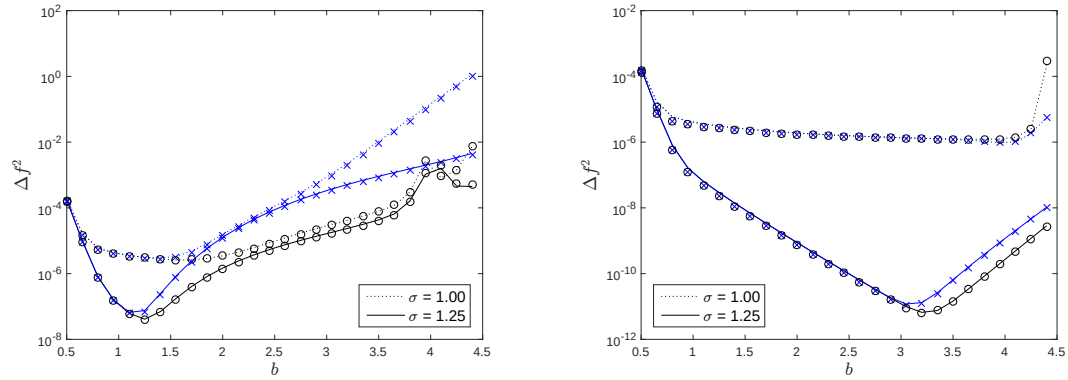

Figure 4.5: Gaussian window, Fourier coefficients (3.4), $m := 3$ (left) and $m = 6$ (right).

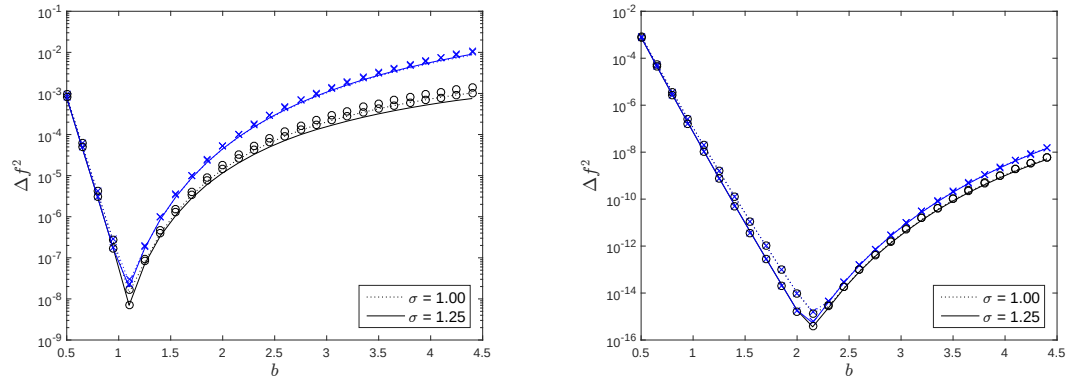

Figure 4.6: Gaussian window, Fourier coefficients (3.5), $m := 3$ (left) and $m = 6$ (right).

5 Parameter tuning

Based on the very accurate error estimates we are able to construct a tuning algorithm for the shape parameter b. In the following we denote by

$$\Delta f^2_{\approx}(\text{window}, b, m, \sigma) \begin{cases} \approx \|f - \tilde{f}\|_2^2 \quad \text{as described in Section 3} & : b > 0, \\ = +\infty & : b \leq 0, \end{cases}$$

the predicted quadratic error, which can be computed for the different window functions as described in Section 3.

For the (modified) B-spline window it seems that a shape parameter $b \in {}^1\!/\!_2\mathbb{N} \cap ({}^m\!/\!_2, m]$ is always optimal. Thus, we suggest to tune the shape parameter as follows.

Algorithm 2 (Shape parameter tuning for B-splines)**.**
We start with $b = m$ and compute the predicted NFFT approximation error in the $\mathcal{L}_2$-norm. Then, we reduce the value of b by ${}^1\!/\!_2$ as long as the predicted error decreases and $b > {}^m\!/\!_2$.

Input: Fourier coefficients $\hat{f}_k$, $k \in \mathcal{I}_M$, support parameter m, oversampling factor σ.

i) Set $b_{\text{opt}} := m$ and $\Delta f^2_{\approx,\text{opt}} := \Delta f^2_{\approx}(\text{B-spline}, b_{\text{opt}}, m, \sigma)$.

ii) Set $b_{\text{next}} := m - {}^1\!/\!_2$ and $\Delta f^2_{\approx,\text{next}} := \Delta f^2_{\approx}(\text{B-spline}, b_{\text{next}}, m, \sigma)$.

iii) While $\Delta f^2_{\approx,\text{next}} < \Delta f^2_{\approx,\text{opt}}$ and $b_{\text{next}} > {}^m\!/\!_2$:
   a) Set $b_{\text{opt}} := b_{\text{next}}$ and $\Delta f^2_{\approx,\text{opt}} := \Delta f^2_{\approx,\text{next}}$.
   b) Set $b_{\text{next}} := b_{\text{opt}} - {}^1\!/\!_2$ and $\Delta f^2_{\approx,\text{next}} := \Delta f^2_{\approx}(\text{B-spline}, b_{\text{next}}, m, \sigma)$.

Output: Optimal shape parameter b_{opt} and predicted quadratic error $\Delta f^2_{\approx,\text{opt}}$. □

For the Bessel and the Gaussian window function we proceed as follows. We suppose that in most cases the error depends on the shape parameter b as depicted in Figure 3.4, for instance. That is, that the error increases with growing distance $|b - b_{\text{opt}}|$ from the unique optimal shape paramter b_{opt}. Based on this assumption we suggest a simple search algorithm in order to tune the shape parameter b, see Algorithm 3.

Algorithm 3 (Shape parameter tuning for Bessel and Gauss window)**.**
We start with $b = b_0$, i.e., we choose the default shape parameter, and compute the NFFT approximation error in the $\mathcal{L}_2$-norm. Additionally we compute the errors for $b_{\text{left}} := b_0 - {}^{b_0}\!/\!_2$ as well as $b_{\text{right}} := b_0 + {}^{b_0}\!/\!_2$, i.e., we choose the step size $d := {}^{b_0}\!/\!_2$ in both directions. From these three shape parameters we choose the one yielding the smallest approximation error for the next iteration step. If the old and new b-value coincide, we choose a smaller step size $d \mapsto {}^d\!/\!_2$. We repeat this until approximately the same errors are obtained for b_{left}, b and b_{right}.

Input: Fourier coefficients $\hat{f}_k$, $k \in \mathcal{I}_M$, support parameter m, oversampling factor σ, desired window function (Bessel or Gauss).

i) Set $b_{\text{opt}} := b_0$, $d := {}^{b_0}\!/\!_2$ and $\Delta f^2_{\approx,\text{opt}} := \Delta f^2_{\approx}(\text{window}, b_{\text{opt}}, m, \sigma)$.

ii) Set $b_{\text{left}} := b_{\text{opt}} - d$ and $\Delta f^2_{\approx,\text{left}} := \Delta f^2_{\approx}(\text{window}, b_{\text{left}}, m, \sigma)$.

iii) Set $b_{\text{right}} := b_{\text{opt}} + d$ and $\Delta f^2_{\approx,\text{right}} := \Delta f^2_{\approx}(\text{window}, b_{\text{right}}, m, \sigma)$.

iv) Until $\max\{\Delta f^2_{\approx,\text{left}}, \Delta f^2_{\approx,\text{opt}}, \Delta f^2_{\approx,\text{right}}\} \approx \min\{\Delta f^2_{\approx,\text{left}}, \Delta f^2_{\approx,\text{opt}}, \Delta f^2_{\approx,\text{right}}\}$
   a) $b_{\text{opt}} := \underset{b_{\text{left}}, b_{\text{opt}}, b_{\text{right}}}{\arg\min} \{\Delta f^2_{\approx,\text{left}}, \Delta f^2_{\approx,\text{opt}}, \Delta f^2_{\approx,\text{right}}\}$.
   b) $\Delta f^2_{\approx,\text{opt}} := \min\{\Delta f^2_{\approx,\text{left}}, \Delta f^2_{\approx,\text{opt}}, \Delta f^2_{\approx,\text{right}}\}$.
   c) If the old and the new b_{opt} coincide, choose a smaller step size (set $d := {}^d\!/\!_2$).

d) Set $b_{\text{left}} := b_{\text{opt}} - d$ and $\Delta f^2_{\approx,\text{left}} := \Delta f^2_{\approx}(\text{window}, b_{\text{left}}, m, \sigma)$.

e) Set $b_{\text{right}} := b_{\text{opt}} + d$ and $\Delta f^2_{\approx,\text{right}} := \Delta f^2_{\approx}(\text{window}, b_{\text{right}}, m, \sigma)$.

Output: Optimal shape parameter b_{opt} and predicted quadratic error $\Delta f^2_{\approx,\text{opt}}$. □

Example 5.1. We use Algorithm 3 in order to tune the optimal shape parameters for different parameter settings. The results, see Tables 5.1 and 5.2, coincide with the experimentally determined optimal shape parameters, which we plot in Figure 3.7. For the predicted approximation errors see Figure 3.6. □

	$\sigma = 1$	$\sigma = 5/4$
$m = 2$	4.0743	5.0364
$m = 3$	3.1416	4.0350
$m = 4$	3.1539	3.8067
$m = 5$	3.1907	3.7294
$m = 6$	3.2398	3.7340
$m = 7$	3.2398	3.6705
$m = 8$	3.3379	3.6862

	$\sigma = 1$	$\sigma = 5/4$
$m = 2$	5.5101	5.5776
$m = 3$	5.3751	5.6015
$m = 4$	5.2094	5.4597
$m = 5$	5.0437	5.3622
$m = 6$	4.8781	5.2462
$m = 7$	4.7063	5.1358
$m = 8$	4.5406	5.0216

Table 5.1: Tuned optimal shape parameters b_{opt} for the Fourier coefficients (3.4) on the left hand side and for the Fourier coefficients (3.5) on the right hand side.
Window function: Bessel. See Figure 3.6 for the obtained errors.

	$\sigma = 1$	$\sigma = 5/4$
$m = 2$	0.7759	0.7626
$m = 3$	1.6114	1.2434
$m = 4$	2.4669	1.8900
$m = 5$	3.3323	2.5697
$m = 6$	3.7600	3.2453
$m = 7$	4.5956	3.8297
$m = 8$	4.9338	4.4762

	$\sigma = 1$	$\sigma = 5/4$
$m = 2$	0.7709	0.7647
$m = 3$	1.1116	1.1004
$m = 4$	1.4672	1.4382
$m = 5$	1.8278	1.7822
$m = 6$	2.1934	2.1324
$m = 7$	2.5763	2.4878
$m = 8$	2.9692	2.8474

Table 5.2: Tuned optimal shape parameters b_{opt} for the Fourier coefficients (3.4) on the left hand side and for the Fourier coefficients (3.5) on the right hand side.
Window function: (truncated) Gaussian. See Figure 3.6 for the obtained errors.

Another task concerning the parameter tuning is to find a parameter set for which the computational time is as minimal as possible, assuming that a certain accuracy has to be achieved. Of course, the optimal set of parameters regarding computation time may very much depend on the processor which is used for the calculations.

In order to reach a required accuracy we could apply a tuning algorithm of the form 2 or 3 in order to determine sufficiently large oversampling factors σ and appropriate shape parameters b for different values of the support parameter m. The optimal parameter set can then be determined by a comparison between measured computation times on the used computer. A corresponding tuning could be roughly of the following form.

Algorithm 4 (Accuracy tuning)**.**
Input: Fourier coefficients $\hat{f}_k$, set of window functions, list of support parameters $m_1 \leq \cdots \leq m_n$, set of oversampling factors $1 = \sigma_1 \leq \cdots \leq \sigma_{\max}$, required accuracy $\epsilon > 0$.

i) Compute the required oversampling factors:
For each window function and for each support parameter m_j, $j = 1, \ldots, n$, set

$$\sigma_{\min}(m_j) := \min \left\{ \sigma \in \{\sigma_1, \ldots, \sigma_{\max}\} : \Delta f_{\approx}(\text{window}, b_{\text{opt}}, m_j, \sigma) \leq \epsilon \right\},$$

if this minimum exists. Thereby, use Algorithm 2 or rather Algorithm 3 in order to tune the shape parameter b in each case.

ii) For all obtained possible sets of parameters find the optimal one regarding runtime by running a simple test scenario.

□

Example 5.2. We consider the two sets of Fourier coefficients as given in (3.4) and (3.5). For a given required accuracy ϵ we compare the tuned parameters for the Bessel and the B-spline window, see Tables 5.3 and 5.4. Thereby, we computed for each m the required oversampling factor by

$$\sigma_{\min}(m) := \min \left\{ \sigma \in \left\{ 1 + \tfrac{s}{16}, s = 1, \ldots, 16 \right\} : \Delta f_{\approx}(\text{window}, b_{\text{opt}}, m, \sigma) \leq \epsilon \right\},$$

i.e., we set $\sigma_{\max} := 2$.

The results show that for the Bessel window a smaller oversampling factor is needed compared to the B-spline window in order to obtain the given accuracy. Different combinations of the parameters m and σ are possible. Which one is the optimal with respect to runtime will depend on the used hardware.

Note that if the window function is evaluated based on interpolation tables, see [16] for instance, the runtime needed for the evaluation is independent from the window function φ. In our example we would achieve better runtimes by using the Bessel window. □

	Bessel			B-spline		
	$\sigma_{\min}$	b_{opt}	$\Delta f_{\approx}$	$\sigma_{\min}$	b_{opt}	$\Delta f_{\approx}$
$m = 4$	1.5000	4.24	9.62e-08	–	–	–
$m = 5$	1.1875	3.59	4.82e-08	1.6250	4.5	7.88e-08
$m = 6$	1.1250	3.45	1.51e-08	1.4375	5.0	3.77e-08
$m = 7$	1.0625	3.30	2.63e-08	1.3125	5.5	3.39e-08
$m = 8$	1.0625	3.23	1.92e-08	1.1875	5.5	9.21e-08

Table 5.3: Computed parameter sets and predicted errors for the Fourier coefficients (3.4). We set the required accuracy to $\epsilon := 10^{-7}$.

The multivariate case

In the multivariate case the prediction of the error is somewhat more complicated. The computation of the error sums

$$\sum_{\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}} \frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\text{o}}}(\tilde{\varphi}_{\text{t}})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_{\text{t}})} \quad \text{and} \quad \frac{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^d \setminus \{\boldsymbol{0}\}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\text{o}}}(\tilde{\varphi}_{\text{t}})}{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^d} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\text{o}}}(\tilde{\varphi}_{\text{t}})}$$

	Bessel			B-spline		
	$\sigma_{\min}$	b_{opt}	$\Delta f_{\approx}$	$\sigma_{\min}$	b_{opt}	$\Delta f_{\approx}$
$m = 4$	–	–	–	–	–	–
$m = 5$	1.0625	5.14	8.35e-11	1.4375	5.0	7.99e-11
$m = 6$	1.0000	4.88	2.21e-12	1.1250	6.0	6.90e-11
$m = 7$	1.0000	4.71	5.28e-14	1.0000	6.5	2.05e-11
$m = 8$	1.0000	4.54	1.87e-15	1.0000	7.5	5.19e-13

Table 5.4: Computed parameter sets and predicted errors for the Fourier coefficients (3.5). We set the required accuracy to $\epsilon := 10^{-10}$.

is more or less straight forward since the window function is constructed based on a tensor product approach (2.3) and thus the terms can be separated with respect to the single dimensions.

Obviously, the evaluation of the estimates (2.11) and (2.10) are easy and especially possible in an efficient way, if the coefficients $\hat{f}_{\boldsymbol{k}}$ are also of a tensor product structure, i.e., we have

$$\hat{f}_{\boldsymbol{k}} = \prod_{j=1}^{d} \hat{g}_{k_j}.$$

In this case all necessary computations can be separated with respect to the d dimensions and an efficient tuning of the involved parameters can be realized quite similar to the univariate case. As an example, coefficients of the form $\mathrm{e}^{-\alpha\|\boldsymbol{k}\|^2} = \prod_{j=1}^{d} \mathrm{e}^{-\alpha k_j^2}$ satisfy this condition.

However, in many cases the given Fourier coefficients are not of a tensor product structure or rather not even given analytically. In this case an approximation or estimation of the form

$$\hat{f}_{\boldsymbol{k}} \approx \prod_{j=1}^{d} \hat{g}_{k_j} \quad \text{or} \quad \hat{f}_{\boldsymbol{k}} \leq \prod_{j=1}^{d} \hat{g}_{k_j}$$

might be necessary in order to enable an efficient prediction of the occurrent errors.

As an example, if the Fourier coefficients are given by $\hat{f}_{\boldsymbol{k}} := (1+\|\boldsymbol{k}\|^2)^{-1}$, an approximation of the form

$$\frac{1}{x} \approx \sum_{j=1}^{n} r_j \mathrm{e}^{-w_j x} \tag{5.1}$$

over a sufficiently large interval $[1, \ell)$ gives

$$\frac{1}{1+\|\boldsymbol{k}\|^2} \approx \sum_{j=1}^{n} r_j \mathrm{e}^{-w_j} \mathrm{e}^{-w_j\|\boldsymbol{k}\|^2} =: \sum_{j=1}^{n} \tilde{r}_j \mathrm{e}^{-w_j\|\boldsymbol{k}\|^2},$$

where the single summands are now of a tensor product structure. One possible tool for computing an approximation of the form (5.1) is the Remez algorithm, see [10] for instance.

There are some multivariate applications where the underlying Fourier coefficients are known analytically. As one example we refer to the three dimensional periodic coulomb problem, where the electrostatic potentials and forces of a set of charges in the three dimensional space are of interest, see [6]. The well known P^{2}NFFT algorithm [20] combines the adjoint NFFT and the NFFT to evaluate the Coulomb potentials and forces very efficiently.

The underlying Fourier coefficients are of the form $\sim \|\boldsymbol{k}\|^{-2}\mathrm{e}^{-\alpha\|\boldsymbol{k}\|^2}$. In order to develop an efficient parameter tuning, the Fourier coefficients have to be approximated by a tensor product expression, e.g. via applying an approximation of the form (5.1).

6 Summary

In this paper we revisited the error formulas for the well established NFFT algorithm. We showed how we can achieve a very precise prediction of the approximation errors measured in the $\mathcal{L}_2$-norm for different window functions. Thereby, we concentrated on the univariate case, where a straight and efficient evaluation of the correspondent error sums is possible.

In our numerical examples we compared two different deconvolution approaches and modified the shape of the considered window functions. The results show that only minimal improvements can be obtained by applying the $\mathcal{L}_2$-optimized deconvolution scheme. Additionally, the examples show that, especially in the case that the Fourier coefficients are subject to a certain decrease, an appropriate modification of the window's shape parameter can lead to significantly smaller approximation errors. For the well established B-spline window function we introduced a modified version, which also contains a shape parameter. In our examples we could achieve considerable improvements compared to the classical B-spline window in a setting where the Fourier coefficients decreased only moderately. However, a comparison between the different window functions showed that the Bessel window is in most cases the best choice and that an appropriate tuning of the shape parameter is essential.

For the univariate case we suggest an easy parameter tuning. Given a required accuracy, different combinations of the involved parameters are possible. Which set of parameters is optimal with respect to the computation time may depend on the used hardware. A corresponding tuning method could be based on the derived error estimates as well as on the mentioned shape parameter tuning.

The prediction of the approximation errors in the multivariate case holds some more difficulties. Only if the Fourier coefficients are of a tensor product structure, an efficient computation of the error terms is more or less straight forward. However, in most applications we have a multivariate setting. One example is the NFFT based fast Ewald summation, which is used in the area of particle simulation for the computation of the Coulomb potentials and forces in charged particle systems. In a subsequent paper we aim to apply the derived error estimates in order to serve a comparison between different window functions as well as to develop a more precise tuning of the involved parameters for this particular application.

Acknowledgments

The author gratefully acknowledges support by the German Research Foundation (DFG), project PO 711/12-1.

References

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

[2] C.K. Chui: *An Introduction to Wavelets.* Academic Press, Boston, 1992.

[3] M. Deserno and C. Holm: *How to mesh up Ewald sums. I. A theoretical and numerical comparison of various particle mesh routines.* J. Chem. Phys., 109:7678 – 7693, 1998.

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

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

[6] P.P. Ewald: *Die Berechnung optischer und elektrostatischer Gitterpotentiale.* Ann. Phys., 369:253–287, 1921.

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

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

[9] L. Greengard and J.Y. Lee: *Accelerating the nonuniform fast Fourier transform.* SIAM Rev., 46:443 – 454, 2004.

[10] W. Hackbusch: *Entwicklungen nach Exponentialsummen.* Techn. rep., Max Planck Institute for Mathematics in the Sciences, 2005. `http://www.mis.mpg.de/de/publications/andere-reihen/tr/report-0405.html`.

[11] R.W. Hockney and J.W. Eastwood: *Computer simulation using particles.* Taylor & Francis, Inc., Bristol, PA, USA, 1988.

[12] J.I. Jackson, C.H. Meyer, D.G. Nishimura, and A. Macovski: *Selection of a convolution function for Fourier inversion using gridding.* IEEE Trans. Med. Imag., 10:473 – 478, 1991.

[13] M. Jacob: *Optimized least-square nonuniform Fast Fourier Transform.* IEEE Trans. Signal Process., 57:2165 – 2177, 2009.

[14] S. Johnson, A. Carvellino, and J. Wuttke: *libcerf, numeric library for complex error functions.* `http://apps.jcns.fz-juelich.de/libcerf`.

[15] J.F. Kaiser: *Digital filters.* In F.F. Kuo and J.F. Kaiser (eds.): *System analysis by digital computer.* Wiley, New York, 1966.

[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] S. Kunis and S. Kunis: *The nonequispaced FFT on graphics processing units.* PAMM, Proc. Appl. Math. Mech., 12, 2012.

[18] S. Kunis, D. Potts, and G. Steidl: *Fast Gauss transform with complex parameters using NFFTs.* J. Numer. Math., 14:295 – 303, 2006.

[19] M. Pippig: *Massively Parallel, Fast Fourier Transforms and Particle-Mesh Methods.* Dissertation. Technische Universität Chemnitz, Faculty of Mathematics, 2015.

[20] M. Pippig and D. Potts: *Parallel three-dimensional nonequispaced fast Fourier transforms and their application to particle simulation.* SIAM J. Sci. Comput., 35:C411 – C437, 2013.

[21] G. Plonka and M. Tasche: *On the computation of periodic spline wavelets.* Appl. Comput. Harmon. Anal., 2:1 – 14, 1995.

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

[23] 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 (eds.): *Modern Sampling Theory: Mathematics and Applications*, pp. 247 – 270, Boston, MA, USA, 2001. Birkhäuser.

[24] I.J. Schoenberg: *Cardinal interpolation and spline functions.* J. Approx. Theory, 2(2):167 – 206, 1969.

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

[26] T. Volkmer: *OpenMP parallelization in the NFFT software library.* Preprint 2012-07, Faculty of Mathematics, Technische Universität Chemnitz, 2012.

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

[28] Z. Yang and M. Jacob: *Mean square optimal NUFFT approximation for efficient non-Cartesian MRI reconstruction.* J. Mag. Reson., (242):126–135, 2014.