Parameter tuning for the NFFT based fast Ewald summation

Franziska Nestler

The computation of the Coulomb potentials and forces in charged particle systems under 3d-periodic boundary conditions is possible in an efficient way by utilizing the Ewald summation formulas and applying the fast Fourier transform (FFT). In this paper we consider the particle-particle NFFT (P^2NFFT) approach, which is based on the fast Fourier transform for nonequispaced data (NFFT) and compare the error behaviors regarding different window functions, which are used in order to approximate the given continuous charge distribution by a mesh based charge density. While typically B-splines are applied in the scope of particle mesh methods, we consider for the first time also an approximation by Bessel functions. We show how the resulting root mean square errors in the forces can be predicted precisely and efficiently. The results show that if the parameters are tuned appropriately the Bessel window function can keep up with the B-spline window and is in many cases even the better choice with respect to computational costs.

Key words and phrases : Ewald summation, particle methods, nonequispaced fast Fourier transform, NFFT, P3M, P2NFFT, ScaFaCoS

2000 AMS Mathematics Subject Classification : 65T

1. Introduction

In this paper we consider the computation of the Coulomb potentials and forces in charged particle systems subject to 3d-periodic boundary conditions. Unfortunately, the underlying infinite sums, which have to be evaluated, are very slowly and even conditionally convergent.

Nevertheless, there are already quite a lot methods for the efficient evaluation of the Coulomb potentials and forces. Most of them, as for instance [17, 5, 10, 6, 24], are based on the so called Ewald summation approach [11], which splits the badly converging sum into two rapidly converging parts, where the underlying order of summation takes a central role. The one part is a sum in spatial domain and can be evaluated efficiently. The other part is a sum in Fourier space. An efficient evaluation of this part is possible by applying the fast Fourier transform (FFT). Thereby, the sticking point is that the particles are not distributed on a

franziska.nestler@mathematik.tu-chemnitz.de
Technische Universutät Chemnitz, Faculty of Mathematics, 09107 Chemnitz, Germany

regular grid. Thus, the present continuous charge distribution has at first to be approximated by a regular grid based charge density before the FFT can be applied.

Algorithms which are based on such an approximation are commonly known as particle mesh methods [17, 5, 10, 6, 28, 29]. Most methods use B-splines in order to perform this grid based approximation step, as for example the well-established particle-particle particle-mesh (P^3M) method [17, 6]. Also approximations via a Gaussian have already been considered, see [24]. The particle-particle NFFT (P^2NFFT) approach, which was suggested in [16, 28], is based on the FFT for nonequispaced data (NFFT) and allows the usage of various types of approximating window functions, as for example also (Kaiser-)Bessel functions besides B-splines and Gaussian. In this context we remark that in a variety of applications the results strongly depend on which window function is applied. As an example, in the field of magnetic resonance imaging Kaiser-Bessel functions seem most suitable, see [12].

Note that [3] serves a detailed comparison between different efficient methods for the 3d-periodic Coulomb problem. The results show that the P^3M and P^2NFFT solvers rank among the best methods in this field. We further remark that the P^2NFFT method has already been generalized to mixed periodic boundary conditions, see [26].

It is widely unknown how the error can be estimated a priori for the different window functions. How do we have to set the NFFT parameters and which window function do we have to choose in order to reach a certain accuracy, while keeping the computational costs as small as possible? In order to answer this question, we consider the root mean square error (rms) in the forces, which is a common error measure in the field of molecular dynamics simulations, and draw some comparisons between different window functions.

The outline of the paper is as follows. In Section 2 we give an introduction to the NFFT. In Sections 3 we consider to the Coulomb problem for 3d-periodic boundary conditions and introduce the corresponding Ewald formulas. We also discuss the estimation of the rms force error which results from the truncation of the Ewald sums. In Section 4 we describe the concept of the P^2NFFT method, discuss how the rms errors caused by the applied NFFT approximations can be estimated a priori and draw some comparisons between different window functions, where we concentrate on the B-spline as well as the Bessel window function, based on the results presented in [25]. In addition, we present an efficient method to tune all involved parameters automatically. An overall tuning, which in addition optimizes the set of parameters with respect to runtime, should depend on the used hard- and software. Thus, we may tune the method with respect to runtime for small particle systems by comparing the runtimes obtained for different parameter combinations in order to apply the found optimal set of parameters also to larger systems. We demonstrate the proposed tuning with the help of some examples, for which we use the ScaFaCoS library [1]. In Section 5 we finish with some conclusions.

2. The nonequispaced FFT

In the following we give a short introduction to the NFFT in three dimensions. At first, we introduce some notations which we will stick to in the whole paper. For some $\boldsymbol{M} = (M_1, M_2, M_3) \in 2\mathbb{N}^3$ we define the index set $\mathcal{I}_{\boldsymbol{M}}$ by

$$\mathcal{I}_{\boldsymbol{M}} := \bigotimes_{j=1}^{3} \{-M_j/2, \ldots, M_j/2 - 1\}.$$

For two vectors $\boldsymbol{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ and $\boldsymbol{y} = (y_1, y_2, y_3) \in \mathbb{R}^3$ we define the component wise product by $\boldsymbol{x} \odot \boldsymbol{y} := (x_1 y_1, x_2 y_2, x_3 y_3) \in \mathbb{R}^3$ as well as the inner product via $\boldsymbol{x} \cdot \boldsymbol{y} := x_1 y_1 + x_2 y_2 + x_3 y_3 \in \mathbb{R}$. For a vector $\boldsymbol{x} \in \mathbb{R}^3$ with non vanishing components we define $\boldsymbol{x}^{-1} := (x_1^{-1}, x_2^{-1}, x_3^{-1}) \in \mathbb{R}^3$.

Let the coefficients $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$ for $k \in \mathcal{I}_{\boldsymbol{M}}, \boldsymbol{M} \in 2\mathbb{N}^3$, and some arbitrary nodes $\boldsymbol{x}_j \in \mathbb{T}^3$, 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 the trigonometric polynomial

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

at the given nodes $\boldsymbol{x}_j$, i.e., we want to compute

$$f(\boldsymbol{x}_j) = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{f}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}_j}, \quad j = 1, \ldots, N. \tag{2.2}$$

The straightforward and exact algorithm for the evaluation of such sums is called NDFT and requires $\mathcal{O}(N|\mathcal{I}_{\boldsymbol{M}}|)$ arithmetical operations. The NFFT algorithm [9, 4, 35, 38, 31, 14, 21] can be used in order to approximate sums of the form (2.2) very efficiently with only $\mathcal{O}(|\mathcal{I}_{\boldsymbol{M}}| \log |\mathcal{I}_{\boldsymbol{M}}| + N)$ arithmetic operations. In the following we will give an overview of the main steps.

In principle, the function f is approximated by a sum of translates of a one-periodic function $\tilde{\varphi}$, i.e.,

$$f(\boldsymbol{x}) \approx \tilde{f}(\boldsymbol{x}) := \sum_{\boldsymbol{l} \in \mathcal{I}_{\boldsymbol{\sigma} \odot \boldsymbol{M}}} g_{\boldsymbol{l}} \tilde{\varphi}\left(\boldsymbol{x} - \boldsymbol{l} \odot (\boldsymbol{\sigma} \odot \boldsymbol{M})^{-1}\right), \tag{2.3}$$

where we denote by $\boldsymbol{\sigma} \geq 1$ (component wise) the oversampling factor. 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 via a tensor product scheme, i.e., we set

$$\tilde{\varphi}(\boldsymbol{x}) := \sum_{\boldsymbol{r} \in \mathbb{Z}^3} \varphi(\boldsymbol{x} + \boldsymbol{r}), \text{ where } \varphi(\boldsymbol{y}) = \prod_{j=1}^{3} \varphi_j(y_j) \text{ for } \boldsymbol{y} = (y_1, y_2, y_3) \in \mathbb{R}^3. \tag{2.4}$$

Thereby, $\varphi_j(\cdot)$ are univariate functions. A transformation of $\tilde{f}$ into Fourier space gives

$$\tilde{f}(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_{\mathrm{o}}}} \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}) \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}} + \sum_{\boldsymbol{r} \in \mathbb{Z}^3 \setminus \{\boldsymbol{0}\}} \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}_{\mathrm{o}}}} \hat{g}_{\boldsymbol{k}} c_{\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}) \mathrm{e}^{-2\pi \mathrm{i} (\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_{\mathrm{o}}) \cdot \boldsymbol{x}}, \tag{2.5}$$

where we denote by

$$c_{\boldsymbol{k}}(\tilde{\varphi}) := \int_{\mathbb{T}^3} \tilde{\varphi}(\boldsymbol{x}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{n} \cdot \boldsymbol{x}} \mathrm{d}\boldsymbol{x} = \int_{\mathbb{R}^3} \varphi(\boldsymbol{x}) \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{n} \cdot \boldsymbol{x}} \mathrm{d}\boldsymbol{x} = \hat{\varphi}(\boldsymbol{k})$$

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

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

The Idea is now to choose the coefficients $\hat{g}_{\boldsymbol{k}}$ appropriately. Then, the coefficients $g_{\boldsymbol{l}}$ in (2.3) can be computed via the inverse FFT

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

and the evaluation of (2.3) gives the approximate function values $\tilde{f}(\boldsymbol{x}_j) \approx f(\boldsymbol{x}_j)$.

However, the evaluation of the sums (2.3) might be computationally demanding unless φ is compactly supported on a comparable small domain or at least sufficiently small outside of it. In the latter case we replace the window function $\tilde{\varphi}$ by a truncated version

$$\varphi_{\mathrm{t}}(\boldsymbol{x}) := \varphi(\boldsymbol{x}) \cdot \prod_{j=1}^{3} \chi_{[-\frac{m}{\sigma_j M_j}, \frac{m}{\sigma_j M_j}]}(x_j) = \begin{cases} \varphi(\boldsymbol{x}) & : \boldsymbol{x} \in \bigotimes_{j=1}^{3} [-\frac{m}{\sigma_j M_j}, \frac{m}{\sigma_j M_j}], \\ 0 & : \text{else}, \end{cases}$$

and approximate f by

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

Thereby, we refer to $m \in \mathbb{N}$ as the support parameter. Note, that we could use different values for m in the single dimensions, but for simplicity we choose the same for all three dimensions.

Comparing (2.2) and (2.5) shows that it is reasonable to set

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

where we define

$$\hat{d}_{\boldsymbol{k}} := \frac{1}{c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}. \tag{2.6}$$

Optimizing the error with respect to the $\mathcal{L}_2$-norm shows that

$$\hat{d}_{\boldsymbol{k}} := \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^3} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})} \tag{2.7}$$

is the optimal choice of the coefficients $\hat{d}_{\boldsymbol{k}}$, see [8, 19, 25].

Applying (2.1) and (2.5), where we have to replace $\tilde{\varphi}$ by $\tilde{\varphi}_{\mathrm{t}}$ in the case that φ is not compactly supported, we obtain the general representation of the resulting error

$$f(\boldsymbol{x}) - \tilde{f}(\boldsymbol{x}) = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{f}_{\boldsymbol{k}} \left[1 - \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}}) \right] \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}} + \sum_{\boldsymbol{r} \in \mathbb{Z}^3 \setminus \{\boldsymbol{0}\}} \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \hat{f}_{\boldsymbol{k}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}}) \mathrm{e}^{-2\pi \mathrm{i} (\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}) \cdot \boldsymbol{x}}. \tag{2.8}$$

Finally, for the two different deconvolution approaches (2.6) and (2.7) we obtain the following expressions for the error measured in the $\mathcal{L}_2$-norm

$$\left\| f - \tilde{f} \right\|_2^2 = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \left| \hat{f}_{\boldsymbol{k}} \right|^2 \sum_{\boldsymbol{r} \in \mathbb{Z}^3 \setminus \{\boldsymbol{0}\}} \frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})} \qquad \text{for } \hat{d}_{\boldsymbol{k}} := \frac{1}{c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}, \tag{2.9}$$

$$\left\| f - \tilde{f} \right\|_2^2 = \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \left| \hat{f}_{\boldsymbol{k}} \right|^2 \frac{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^3 \setminus \{\boldsymbol{0}\}} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})}{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^3} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})} \qquad \text{for } \hat{d}_{\boldsymbol{k}} := \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}{\sum\limits_{\boldsymbol{r} \in \mathbb{Z}^3} c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})}. \tag{2.10}$$

We summarize the NFFT algorithm as follows.

Algorithm 2.1 (NFFT)**.**
Input: nodes $\boldsymbol{x}_j \in \mathbb{T}^3$ $(j = 1, \ldots, N)$, coefficients $\hat{f}_{\boldsymbol{k}} \in \mathbb{C}$ $(\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}, \boldsymbol{M} \in 2\mathbb{N}^3)$, oversampling factor $\boldsymbol{\sigma} \in \mathbb{R}^3$, $\boldsymbol{\sigma} \geq \mathbf{1}$.

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 (2.6) or (2.7).
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: $\mathcal{O}(m^3 N)$.

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

The problem of evaluating sums of the form

$$h(\boldsymbol{k}) := \sum_{j=1}^{N} f_j \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}_j}, \quad \boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}},$$

where for each $j = 1, \ldots, N$ a coefficient $f_j \in \mathbb{C}$ is given, can be treated very similarly. We refer to the method for the exact evaluation of the sums $h(\boldsymbol{k})$ to the adjoint NDFT. The corresponding fast algorithm is known as the adjoint NFFT. Note that the matrix-vector form of the adjoint NDFT is simply obtained by adjoining the matrix representing the NDFT. Thus, the derivation of the fast algorithm for the adjoint problem is straightforward, see [31, 21], and the error can be written as

$$h(\boldsymbol{k}) - \tilde{h}(\boldsymbol{k}) = \sum_{j=1}^{N} f_j [1 - \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})] \mathrm{e}^{2\pi \mathrm{i} \boldsymbol{k} \cdot \boldsymbol{x}_j} - \sum_{\boldsymbol{r} \in \mathbb{Z}^3 \setminus \{\mathbf{0}\}} \sum_{j=1}^{N} f_j \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi \mathrm{i} (\boldsymbol{k} + \boldsymbol{r} \odot \boldsymbol{M}_\mathrm{o}) \cdot \boldsymbol{x}_j}. \tag{2.11}$$

Window functions

In the following we consider different window functions and show how the error sums

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

in the univariate setting can be estimated, compare to (2.9) and (2.10) for three dimensions. For details about the derivation of those estimates we refer to [25] and references therein.

We will come back to the three dimensional case in Section 4, where the NFFT based Ewald summation is considered.

In this paper we restrict our considerations to the B-spline window and the Bessel I_0 window function, which showed the best performance in [25]. The considered window functions are compactly supported in spatial domain, i.e., we have $\varphi_{\mathrm{t}} = \varphi$. Note that the Gaussian window function was outperformed by the other windows in the examples presented in [25]. Especially in the case of rapidly decreasing Fourier coefficients, which we also have for to the Coulomb problem, the B-spline and the Bessel window function produced much smaller approximation errors.

B-spline window

For $b_j \in {}^1\!/\!_2\,\mathbb{N}$, $j = 1, \ldots, 3$, we define the B-spline window in three variables by [4, 30, 25]

$$\varphi(\boldsymbol{x}) := \prod_{j=1}^{3} B_{2b_j}\left(\frac{\sigma_j M_j b_j x_j}{m}\right).$$

We have $\operatorname{supp}\varphi = \bigotimes_{j=1}^{3}[-{}^m\!/\!_{\sigma_j M_j}, {}^m\!/\!_{\sigma_j M_j}]$ and the corresponding Fourier coefficients are given by

$$c_k(\tilde{\varphi}_j) = \frac{m}{\sigma_j M_j b_j}\operatorname{sinc}^{2b_j}\left(\frac{m\pi k}{\sigma_j M_j b_j}\right).$$

For $b_j = m$ we are in the setting of the standard cardinal B-spline window [4, 30]. In this case we have

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} \frac{c^2_{k+r\sigma_j M_j}(\tilde{\varphi}_j)}{c^2_k(\tilde{\varphi}_j)} < \frac{8m}{4m-1}\left(\frac{|k|}{|k|-\sigma_j M_j}\right)^{4m},$$

$$\sum_{r\in\mathbb{Z}} c^2_{k+r\sigma_j M_j}(\tilde{\varphi}_j) = \frac{1}{\sigma_j^2 M_j^2}\Phi_{4m}\left(\mathrm{e}^{-2\pi\mathrm{i}k/\sigma_j M_j}\right), \tag{2.12}$$

where we denote by Φ_n the well known Euler-Frobenius functions [34]. If $b_j \neq m$ and ${}^m\!/\!_{b_j} \notin \mathbb{Z}$ we obtain with some $R \in \mathbb{N}$

$$\sum_{r\in\mathbb{Z}\setminus\{0\}} c^2_{k+r\sigma_j M_j}(\tilde{\varphi}_j) \leq \sum_{0<|r|\leq R} c^2_{k+r\sigma_j M_j}(\tilde{\varphi}_j) + \frac{m^2}{\sigma_j^2 M_j^2 b_j^2}\frac{\left(\frac{|k|}{\sigma_j M_j}+R\right)^{1-4b_j} - \left(\frac{|k|}{\sigma_j M_j}-R\right)^{1-4b_j}}{\left(\frac{m\pi}{b_j}\right)^{4b_j}(4b_j-1)},$$

see [25].

Bessel window

We consider a window function which is constructed based on the Kaiser-Bessel window, which was introduced in [30, 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, which is also found under the name Kaiser-Bessel function in the literature [20, 12, 18]. We define the Bessel window by

$$\varphi(\boldsymbol{x}) := \prod_{j=1}^{3} \varphi_j(x_j),$$

where for the shape parameters $b_j > 0$, $j = 1, \ldots, 3$,

$$\varphi_j(x) := \begin{cases} I_0\left(b_j\sqrt{m^2 - \sigma_j^2 M_j^2 x^2}\right) & : x \in [-m/\sigma_j M_j, m/\sigma_j M_j], \\ 0 & : \text{else.} \end{cases}$$

Typically, the standard shape parameter

$$\boldsymbol{b} := 2\pi(1 - (2\boldsymbol{\sigma})^{-1}) \tag{2.13}$$

is used. However, in [25] we showed that a modification of the shape parameter can lead to significant improvements in terms of the arising approximation errors. Thereby, it very much depends on the given coefficients $\hat{f}_{\boldsymbol{k}}$ which shape parameter is optimal with respect to accuracy.

The Fourier coefficients of the Bessel window read as

$$c_k(\tilde{\varphi}_j) = \frac{1}{\sigma_j M_j} \begin{cases} \dfrac{\sinh\left(m\sqrt{b_j^2 - 4\pi^2 k^2/(\sigma_j^2 M_j^2)}\right)}{\sqrt{b_j^2 - 4\pi^2 k^2/(\sigma_j^2 M_j^2)}} & : |k| \leq \frac{\sigma_j M_j b_j}{2\pi}, \\ m\,\mathrm{sinc}\left(m\sqrt{4\pi^2 k^2/(\sigma_j^2 M_j^2) - b_j^2}\right) & : \text{else.} \end{cases}$$

For some $R \in \mathbb{N} : R > \frac{|k|}{\sigma_j M_j} + \frac{b_j}{2\pi}$ we have [25]

$$\sum_{r \in \mathbb{Z}\setminus\{0\}} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_j) \leq \sum_{0<|r|\leq R} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_j) + \frac{\ln\left(\frac{2\pi(|k|/\sigma_j M_j - R) - b_j}{2\pi(|k|/\sigma_j M_j - R) + b_j}\right) + \ln\left(\frac{2\pi(|k|/\sigma_j M_j + R) + b_j}{2\pi(|k|/\sigma_j M_j + R) - b_j}\right)}{4\pi b \sigma_j^2 M_j^2}.$$

3. Ewald summation and rms errors

We consider a system of N charges q_j distributed in a box of the size $L_1 \times L_2 \times L_3$, where $L_1, L_2, L_3 \in \mathbb{R}_+$. The electrostatic potential for each particle j subject to 3d–periodic boundary conditions is defined as

$$\phi_j = \sum_{\boldsymbol{n} \in \mathbb{Z}^3} {\sum_{i=1}^{N}}' \frac{q_i}{\|\boldsymbol{x}_{ij} + \boldsymbol{L} \odot \boldsymbol{n}\|}, \tag{3.1}$$

where the prime indicates that for $\boldsymbol{n} = \boldsymbol{0}$ the terms with $i = j$ are omitted and the vector $\boldsymbol{L} \in \mathbb{R}_+^3$ is defined by $\boldsymbol{L} = (L_1, L_2, L_3)$. In the following we assume that the system is electrical neutral, i.e., we have

$$\sum_{j=1}^{N} q_j = 0. \tag{3.2}$$

Note that if (3.2) is fulfilled, the infinite sum (3.1) is only conditional convergent, i.e., the values of the potentials strongly depend on the underlying order of summation. In general, a so called spherical limit is considered, see [22], for instance.

In molecular dynamics simulations one is also interested in calculating the forces acting on the particles, which are given by

$$\boldsymbol{F}_j := -q_j \nabla_{\boldsymbol{x}_j} \phi_j. \tag{3.3}$$

As already indicated in the introduction, the so called Ewald summation technique is the basis for a variety of efficient algorithms in this field. The basic idea behind the Ewald summation approach can be explained as follows. It makes use of the trivial identity

$$\frac{1}{r} = \frac{\operatorname{erf}(\alpha r)}{r} + \frac{\operatorname{erfc}(\alpha r)}{r}, \tag{3.4}$$

where $\alpha > 0$ is named Ewald or splitting parameter, $\operatorname{erf}(x) := \frac{2}{\sqrt{\pi}} \int_0^x \mathrm{e}^{-t^2} \mathrm{d}t$ is the error function and $\operatorname{erfc}(x) := 1 - \operatorname{erf}(x)$ is the complementary error function. Based on (3.4) the potential is split into two rapidly converging parts. The complementary error function $\operatorname{erfc}(x)$ tends to zero exponentially fast as x grows. Thus, the corresponding part is absolutely converging and can be evaluated directly by truncating the infinite sum. The second part is still long ranged and conditionally convergent, but for the error function we have

$$\lim_{r \to 0} \frac{\operatorname{erf}(\alpha r)}{r} = \frac{2\alpha}{\sqrt{\pi}},$$

i.e., we do not have a singularity in this part. As a result, we can transform the remaining infinite sum into a rapidly converging sum in Fourier space, where the underlying order of summation is of importance.

If the spherical summation order is applied, we obtain [11, 23]

$$\phi_j = \phi_j^{\mathrm{S}} + \phi_j^{\mathrm{L}} + \phi_j^{\mathrm{self}}. \tag{3.5}$$

Thereby we define the short range part

$$\phi_j^{\mathrm{S}} := \sum_{\boldsymbol{n} \in \mathbb{Z}^3} {\sum_{i=1}^{N}}' q_i \frac{\operatorname{erfc}(\alpha \| \boldsymbol{x}_{ij} + \boldsymbol{L} \odot \boldsymbol{n} \|)}{\| \boldsymbol{x}_{ij} + \boldsymbol{L} \odot \boldsymbol{n} \|},$$

the long range part

$$\phi_j^{\mathrm{L}} := \frac{1}{\pi V} \sum_{\boldsymbol{k} \in \mathbb{Z}^3 \setminus \{\boldsymbol{0}\}} \frac{\mathrm{e}^{-\pi^2 \| \boldsymbol{k} \odot \boldsymbol{L}^{-1} \|^2 / \alpha^2}}{\| \boldsymbol{k} \odot \boldsymbol{L}^{-1} \|^2} S(\boldsymbol{k}) \, \mathrm{e}^{-2\pi \mathrm{i} (\boldsymbol{k} \odot \boldsymbol{L}^{-1}) \cdot \boldsymbol{x}_j},$$

where we set

$$S(\boldsymbol{k}) := \sum_{i=1}^{N} q_i \mathrm{e}^{2\pi \mathrm{i} (\boldsymbol{k} \odot \boldsymbol{L}^{-1}) \cdot \boldsymbol{x}_i}$$

and denote by $V := L_1 L_2 L_3$ the volume of the box, and the self potential

$$\phi_j^{\mathrm{self}} := -\frac{2\alpha}{\sqrt{\pi}} q_j.$$

We are also interested in the computation of the forces $\boldsymbol{F}_j$ acting on the particles, which we define in (3.3). In most applications, the forces are computed by applying the differentiation operator directly to the Ewald representation (3.5) of the potentials, i.e., also the force splits into a short range and a long range part

$$\boldsymbol{F}_j = \boldsymbol{F}_j^{\mathrm{S}} + \boldsymbol{F}_j^{\mathrm{L}},$$

where the short range part is given by

$$\boldsymbol{F}_j^{\mathrm{S}} := -q_j \sum_{\boldsymbol{n}\in\mathcal{S}} \sum_{i=1}^{N}{}' q_i \left(\frac{2\alpha}{\sqrt{\pi}} \mathrm{e}^{-\alpha^2 \|\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}\|^2} + \frac{\operatorname{erfc}(\alpha\|\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}\|)}{\|\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}\|} \right) \frac{\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}}{\|\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}\|^2}.$$

The differentiation in the long range part can be performed easily in Fourier space, which results in

$$\boldsymbol{F}_j^{\mathrm{L}} := \frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \frac{\mathrm{e}^{-\pi^2\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2/\alpha^2}}{\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2} (\boldsymbol{k}\odot\boldsymbol{L}^{-1}) S(\boldsymbol{k}) \mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\cdot\boldsymbol{x}_j}.$$

The rms error in the forces

$$\Delta F := \sqrt{\frac{1}{N}\sum_{j=1}^{N} \|\boldsymbol{F}_j - \boldsymbol{F}_{j,\approx}\|^2},$$

where $\boldsymbol{F}_{j,\approx}$ denotes an approximately computed version of the exact force $\boldsymbol{F}_j$, is commonly taken as a measure of accuracy. The estimation of the rms force error is discussed in the following sections.

3.1. Rms force error in the short range part

Since the complementary error function erfc rapidly tends to zero, the real space parts of the potentials as well as the forces can be computed approximately by direct evaluation, i.e., all distances $\|\boldsymbol{x}_{ij}+\boldsymbol{L}\odot\boldsymbol{n}\|$ larger than an appropriate cutoff radius r_{cut} are ignored. Note that if we assume a sufficiently homogenous particle distribution, each particle only interacts with a fixed number of neighbors and the short range parts can be computed with a linked cell algorithm [13] in $\mathcal{O}(N)$ arithmetic operations.

An analysis of the rms errors in the energy as well as the forces for cubic box shapes was investigated in [22] for the first time. The authors give an estimate of the rms force error in the near field, which can be easily generalized for non cubic boxes. By [22, equation (18)] we obtain

$$\Delta F^{\mathrm{S}} = \sqrt{\frac{1}{N}\sum_{j=1}^{N} \left\|\boldsymbol{F}_j^{\mathrm{S}} - \boldsymbol{F}_{j,\approx}^{\mathrm{S}}\right\|^2} \approx \frac{2Q}{\sqrt{r_{\mathrm{cut}} N V}} \mathrm{e}^{-\alpha^2 r_{\mathrm{cut}}^2}, \tag{3.6}$$

where we define

$$Q := \sum_{j=1}^{N} q_j^2. \tag{3.7}$$

Remark 3.1. Consider two different particle systems with numbers of particles $N_{1,2}$, corresponding box volumes $V_{1,2}$ and sums of squared charge values $Q_{1,2}$. It is easy to see that, if

$$\frac{N_1}{V_1} = \frac{N_2}{V_2} \quad \text{and} \quad \frac{Q_1}{V_1} = \frac{Q_2}{V_2} \tag{3.8}$$

is fulfilled, the expected rms force errors in the short range parts are equal, provided that the same values for α and r_{cut} are used. □

3.2. Rms force error in the long range part

The Fourier coefficients

$$\hat{\psi}(\boldsymbol{k}) := \begin{cases} \dfrac{\mathrm{e}^{-\pi^2\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2/\alpha^2}}{\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2} & : \boldsymbol{k}\neq\boldsymbol{0} \\ 0 & : \boldsymbol{k}=\boldsymbol{0} \end{cases}$$

tend to zero exponentially fast as $\|\boldsymbol{k}\|\to\infty$ so that we can set

$$\begin{aligned} \phi_j^{\mathrm{L}} \approx \phi_{\mathrm{t},j}^{\mathrm{L}} &:= \frac{1}{\pi V}\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{\psi}(\boldsymbol{k})S(\boldsymbol{k})\mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\cdot\boldsymbol{x}_j} \\ \boldsymbol{F}_j^{\mathrm{L}} \approx \boldsymbol{F}_{\mathrm{t},j}^{\mathrm{L}} &:= \frac{2\mathrm{i}q_j}{V}\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})S(\boldsymbol{k})\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})} \end{aligned} \tag{3.9}$$

for $\boldsymbol{M}\in 2\mathbb{N}^3$ large enough, which leads to a truncation error.

In the following we introduce a general approach to the estimation of rms errors. This approach has already been discussed and applied by several authors, see [7, 37] for instance.

Given the charges q_j and the positions $\boldsymbol{x}_j$, we consider a vector valued expression (error etc.) of the form

$$\boldsymbol{\varepsilon}_j := q_j\sum_{\substack{i=1\\ i\neq j}}^{N} q_i\boldsymbol{\chi}_{ij}, \tag{3.10}$$

where each vector $\boldsymbol{\chi}_{ij}$ only depends on the positions $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. We assume that the contributions from different particles are uncorrelated, which implies

$$\left\langle \boldsymbol{\chi}_{ij}^*\cdot\boldsymbol{\chi}_{ik}\right\rangle = \delta_{jk}\left\langle \boldsymbol{\chi}_{ij}^*\cdot\boldsymbol{\chi}_{ij}\right\rangle =: \delta_{jk}\chi^2.$$

Thereby, the angular brackets denote that the average over all possible configurations is considered. Of course, this assumption is not always true but should at least be reasonable for random particle distributions. We obtain

$$\left\langle\|\boldsymbol{\varepsilon}_j\|^2\right\rangle = q_j^2\sum_{i\neq j}\sum_{k\neq j} q_iq_k\left\langle\boldsymbol{\chi}_{ij}^*\cdot\boldsymbol{\chi}_{kj}\right\rangle = q_j^2\chi^2\sum_{i=1}^{N} q_i^2 = q_j^2\chi^2Q,$$

where Q is defined as in (3.7), and finally

$$\sqrt{\frac{1}{N}\sum_{j=1}^{N}\|\boldsymbol{\varepsilon}_j\|^2} \approx \frac{\chi Q}{\sqrt{N}}. \tag{3.11}$$

In the following, we will see that the truncation errors regarding the long range parts of the forces are of the form (3.10). Thus, the corresponding rms error can be estimated by (3.11).

Spherical coordinates

The rms force error in the long range part for cubic box shapes was also considered in [22]. In the following we will see that estimating the rms force error by the above described approach

ends up with a very similar result. We now assume that the particles are distributed in a cubic box of edge length $L > 0$, i.e., we set $\boldsymbol{L} = (L, L, L)$. We have

$$\begin{aligned}\boldsymbol{F}_j^{\mathrm{L}} &\approx \frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})S(\boldsymbol{k})\mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\cdot\boldsymbol{x}_j} \\ &= \frac{2\mathrm{i}q_j}{L^4} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \boldsymbol{k}\hat{\psi}(\boldsymbol{k})S(\boldsymbol{k})\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}_j/L},\end{aligned}$$

where now also the three components of the vector $\boldsymbol{M} \in 2\mathbb{N}^3$ are assumed to coincide, i.e. $\boldsymbol{M} := (M, M, M)$ with $M \in 2\mathbb{N}$. The corresponding error is given by

$$\Delta\boldsymbol{F}_j^{\mathrm{L}} := \frac{2\mathrm{i}q_j}{L^4} \sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_{\boldsymbol{M}}} \boldsymbol{k}\hat{\psi}(\boldsymbol{k})S(\boldsymbol{k})\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}_j/L} = q_j \sum_{i=1}^{N} q_i\boldsymbol{\chi}_{ij}$$

with

$$\boldsymbol{\chi}_{ij} := \frac{2\mathrm{i}}{L^4} \sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_{\boldsymbol{M}}} \boldsymbol{k}\hat{\psi}(\boldsymbol{k})\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot\boldsymbol{x}_{ij}/L}.$$

Utilizing symmetry properties we see that $\chi_{ii} = \boldsymbol{0}$, i.e., the error $\Delta\boldsymbol{F}_j^{\mathrm{L}}$ is of the form (3.10). We easily estimate

$$\begin{aligned}\chi^2 &= \frac{1}{L^3}\int_{L\mathbb{T}^3}\frac{1}{L^3}\int_{L\mathbb{T}^3}\left|\boldsymbol{\chi}_{ij}\right|^2\mathrm{d}\boldsymbol{x}_j\mathrm{d}\boldsymbol{x}_i = \frac{4}{L^8}\sum_{\boldsymbol{k}\in\mathbb{Z}^3\setminus\mathcal{I}_{\boldsymbol{M}}}\|\boldsymbol{k}\|^2\hat{\psi}(\boldsymbol{k})^2 \\ &\approx \frac{4}{L^4}\int_{M/2}^{\infty}\frac{\mathrm{e}^{-2\pi^2r^2/(\alpha^2L^2)}}{r^2}r^2\mathrm{d}r\int_0^{\pi}\sin\theta\mathrm{d}\theta\int_0^{2\pi}\mathrm{d}\phi \\ &= \frac{16\pi}{L^4}\cdot\frac{\alpha L}{2^{3/2}\sqrt{\pi}}\operatorname{erfc}\left(\tfrac{\pi M}{\sqrt{2}\alpha L}\right) = \frac{8\pi\alpha}{\sqrt{2\pi}L^3}\operatorname{erfc}\left(\tfrac{\pi M}{\sqrt{2}\alpha L}\right)\end{aligned} \tag{3.12}$$

and the rms error in the long range part is given by

$$\Delta F^{\mathrm{L}} := \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left\|\Delta\boldsymbol{F}_j^{\mathrm{L}}\right\|^2} \approx \frac{\chi Q}{\sqrt{N}}.$$

Note that the error is for simplicity approximated by using a radial cutoff. Nevertheless, the restriction to an index set $\mathcal{I}_{\boldsymbol{M}}$ is reasonable in order to approximate the remaining finite sums by the NFFT algorithms, see Section 4. Thus, the actual truncation error resulting from substituting $\mathbb{Z}^3$ by $\mathcal{I}_{\boldsymbol{M}}$ is supposed to be somewhat smaller than the presented estimate.

For M large enough we can simplify (3.12) by applying the asymptotic expansion of the complementary error function [2, number 7.1.23] and obtain

$$\chi^2 \approx \frac{8\sqrt{\pi}\alpha}{L^3\sqrt{2}}\cdot\frac{\sqrt{2}\alpha L}{\pi^{3/2}M}\mathrm{e}^{-\pi^2M^2/(2\alpha^2L^2)} = \frac{8\alpha^2}{L^2\pi M}\mathrm{e}^{-\pi^2M^2/(2\alpha^2L^2)},$$

which for $M := 2K$ almost coincides with [22, (32)]. In summary, we obtain

$$\Delta F^{\mathrm{L}} \approx \frac{\alpha\sqrt{8}Q}{L\sqrt{\pi MN}}\mathrm{e}^{-\pi^2M^2/(4\alpha^2L^2)}.$$

Ellipsoidal coordinates

In this section we generalize the error estimate as presented above to the case of non cubic box shapes using ellipsoidal coordinates. The vector valued errors $\boldsymbol{\chi}_{ij}$ can now be written as

$$\boldsymbol{\chi}_{ij} = \frac{2\mathrm{i}}{V} \sum_{\boldsymbol{k} \in \mathbb{Z}^3 \setminus \mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k} \odot \boldsymbol{L}^{-1}) \hat{\psi}(\boldsymbol{k}) \mathrm{e}^{2\pi \mathrm{i} (\boldsymbol{k} \odot \boldsymbol{L}^{-1}) \cdot \boldsymbol{x}_{ij}}.$$

We obtain

$$\chi^2 = \frac{1}{V} \int_{\mathcal{B}} \frac{1}{V} \int_{\mathcal{B}} |\boldsymbol{\chi}_{ij}|^2 \mathrm{d}\boldsymbol{x}_j \mathrm{d}\boldsymbol{x}_i = \frac{4}{V^2} \sum_{\mathbb{Z}^3 \setminus \mathcal{I}_{\boldsymbol{M}}} \frac{\mathrm{e}^{-2\pi^2 \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 / \alpha^2}}{\|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2},$$

where $\mathcal{B} := L_1 \mathbb{T} \times L_2 \mathbb{T} \times L_3 \mathbb{T}$ denotes the box in which the particles are distributed.

In the following we assume that for some $\beta > 0$ the vectors $\boldsymbol{M}$ and $\boldsymbol{L}$ fulfill the relation

$$\boldsymbol{M} = \beta \boldsymbol{L},$$

i.e., the numbers of grid points, which are used in each dimension, are scaled accordingly to the boxes' side lengths. Based on this assumption, we continue by approximating the infinite sum by integrating over all

$$\boldsymbol{k} \in \mathbb{R} : \frac{k_1^2}{(M_1/2)^2} + \frac{k_2^2}{(M_2/2)^2} + \frac{k_3^2}{(M_3/2)^2} > 1,$$

i.e., we exclude the ellipsoid with the semi-axis lengths $M_1/2, M_2/2, M_3/2$. The domain can be parameterized by

$$\begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} r \frac{M_1}{2} \sin\theta \cos\phi \\ r \frac{M_2}{2} \sin\theta \sin\phi \\ r \frac{M_3}{2} \cos\theta \end{pmatrix},$$

where $r \in (1, \infty)$, $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$. With

$$\|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 = \frac{k_1^2}{L_1^2} + \frac{k_2^2}{L_2^2} + \frac{k_3^2}{L_3^2} = \frac{\beta^2}{4} \left(\frac{k_1^2}{(M_1/2)^2} + \frac{k_2^2}{(M_2/2)^2} + \frac{k_3^2}{(M_3/2)^2} \right) = \frac{\beta^2 r^2}{4}$$

we obtain

$$\begin{aligned} \chi^2 &\approx \frac{16}{V^2} \frac{M_1 M_2 M_3}{8\beta^2} \int_1^\infty \mathrm{e}^{-\pi^2 r^2 \beta^2 / 2\alpha^2} \mathrm{d}r \int_0^\pi \sin\theta \mathrm{d}\theta \int_0^{2\pi} \mathrm{d}\phi \\ &= \frac{8\pi}{V^2} \frac{M_1 M_2 M_3}{\beta^2} \int_1^\infty \mathrm{e}^{-\pi^2 r^2 \beta^2 / 2\alpha^2} \mathrm{d}r \\ &= \frac{8\pi}{V^2} \frac{M_1 M_2 M_3}{\beta^2} \sqrt{\frac{\alpha^2}{2\pi\beta^2}} \operatorname{erfc}\left(\frac{\pi\beta}{\sqrt{2}\alpha} \right) = \frac{8\pi\alpha}{V\sqrt{2\pi}} \operatorname{erfc}\left(\frac{\pi\beta}{\sqrt{2}\alpha} \right). \end{aligned}$$

In the special case that we have $M_1 = M_2 = M_3 = M$ and $L_1 = L_2 = L_3 = L$, which also means $\beta = M/L$, we end up with (3.12). Again, if the cutoff parameters are large enough we obtain by the asymptotic expansion of the complementary error function

$$\chi^2 \approx \frac{8\pi\alpha}{V\sqrt{2\pi}} \frac{\sqrt{2}\alpha}{\pi^{3/2}\beta} \mathrm{e}^{-\pi^2\beta^2/2\alpha^2} = \frac{8\alpha^2}{V\pi\beta} \mathrm{e}^{-\pi^2\beta^2/2\alpha^2}.$$

Substituting

$$\beta = \frac{1}{\sqrt{3}} \|\boldsymbol{M} \odot \boldsymbol{L}^{-1}\|$$

we end up with

$$\chi^2 \approx \frac{8\sqrt{3}\alpha^2}{V\pi\|\boldsymbol{M} \odot \boldsymbol{L}^{-1}\|} \mathrm{e}^{-\pi^2\|\boldsymbol{M}\odot\boldsymbol{L}^{-1}\|^2/6\alpha^2}$$

and

$$\Delta F^{\mathrm{L}} \approx \frac{2\sqrt{2}\sqrt[4]{3}\alpha Q}{\sqrt{NV\pi\|\boldsymbol{M} \odot \boldsymbol{L}^{-1}\|}} \mathrm{e}^{-\pi^2\|\boldsymbol{M}\odot\boldsymbol{L}^{-1}\|^2/12\alpha^2}.$$

Remark 3.2. Using the parameter β, the rms force error in the long range part can be written as

$$\Delta F^{\mathrm{L}} \approx \frac{2\sqrt{2}\alpha Q}{\sqrt{VN\pi\beta}} \mathrm{e}^{-\pi^2\beta^2/4\alpha^2}. \tag{3.13}$$

It is easy to see that for two different particle systems fulfilling (3.8) the expected rms force errors in the long range parts are equal, provided that the same values for α and β are used.

As an example, consider the case that the long range parts of the forces for a system with $N_1 = 100$ particles distributed in the box $\mathcal{B}_1 := \mathbb{T}^3$ with charges $q_j = (-1)^j$, $j = 1, \ldots, N_1$, is approximated by using the cutoff $\boldsymbol{M}_1 = (32, 32, 32)$.

Then, the same expected long range part rms error is obtained for a particle system composed of $N_2 = 800$ particles distributed in the box $\mathcal{B}_2 := 2\mathbb{T}^3$ with charges $q_j = (-1)^j$, $j = 1, \ldots, N_2$, by using the far field cutoff $\boldsymbol{M}_2 = (64, 64, 64)$, provided that the same splitting parameter α is chosen. □

3.3. Parameter tuning

The above derived error estimates allow a very precise prediction of the occurrent rms errors when calculating the forces via the Ewald formulas.

In the following we consider a concrete particle system and compare the predicted rms force errors with the actually obtained errors for different parameter settings.

Example 3.3. We consider a so called cloud wall system consisting of $N = 600$ charges $q_j = (-1)^j$ in a box with edge length vector $\boldsymbol{L} = (20, 10, 10)$. The cloud wall system consists of a diffusive particle cloud surrounding two oppositely charged walls and was proposed in [3] as a test system because of its significant long range part.

According to the box shape we applied different far field cutoffs $\boldsymbol{M} = (2M, M, M)$ with $M \in 2\mathbb{N}$ to approximate the long range parts of the forces, where the summation was done over the full mesh $\mathcal{I}_{\boldsymbol{M}}$. The near field computations were done by inserting different cutoffs $r_{\mathrm{cut}} \in \{4.0, 4.5, 5.0\}$. We applied the ScaFaCoS software library [1] for the computation of the forces, where we used the NDFT as well as the adjoint NDFT in order to compute the Fourier sums exactly, and estimated the resulting rms force errors with the available reference data.

For relatively large values of α as well as for large mesh sizes $\boldsymbol{M}$, the actual error in the far field is somewhat overestimated by the derived upper bound. This is supposed to be due to the fact that the algorithm uses the full mesh $\mathcal{I}_{\boldsymbol{M}}$ instead of the supposed ellipsoidal cutoff scheme. However, we see that the achieved error behavior is described very well by the stated estimates. □

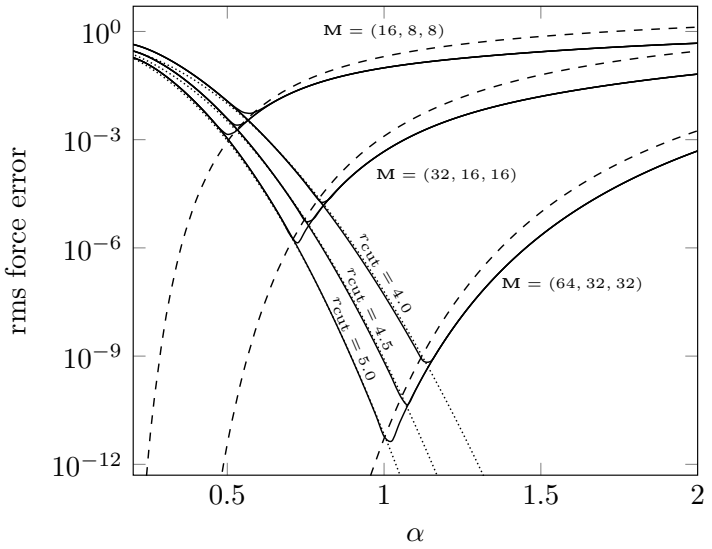

Figure 3.1: Achieved rms force errors (solid) for different far field cutoffs $\boldsymbol{M} = (2M, M, M)$ with $M \in \{8, 16, 32\}$ and different near field cutoffs $r_{\text{cut}} \in \{4.0, 4.5, 5.0\}$. We also plot the estimates for the rms force error in the near field (dotted) as well as for the far field with elliptical cutoff (dashed), see (3.6) and (3.13). (Test case: cloud wall system with $N = 600$ particles in a box with edge lengths $\boldsymbol{L} = (20, 10, 10)$.)

Based on the derived error estimates we may tune the parameters as follows. Note that a first tuning approach is discussed in [22]. A tuning similar to Algorithm 3.1 is already used within the ScaFaCoS library [1] and can also be modified in order to tune the accuracy with respect to the absolute rms potential error, see [24].

Algorithm 3.1 (Ewald tuning)**.**
Input: required accuracy $\varepsilon > 0$, near field cutoff radius $0 < r_{\text{cut}} \leq \min(L_1, L_2, L_3)$.

i) Compute α via (3.6): Claiming

$$\Delta F^{\text{S}} \approx \frac{2Q}{\sqrt{r_{\text{cut}} NV}} \mathrm{e}^{-\alpha^2 r_{\text{cut}}^2} = \frac{\varepsilon}{2}$$

we obtain

$$\alpha = \frac{1}{r_{\text{cut}}} \sqrt{\ln\left(\frac{4Q}{\varepsilon\sqrt{r_{\text{cut}} NV}}\right)}.$$

ii) Compute β via (3.13): Inserting the above computed value for α we choose $\boldsymbol{M}$ such that also the far field error is approximately of the size $\varepsilon/2$. We use the error estimate for the elliptical cutoff scheme (3.13), i.e., we set

$$\frac{2\sqrt{2}\alpha Q}{\sqrt{NV\pi\beta}} \mathrm{e}^{-\pi^2\beta^2/(4\alpha^2)} = \frac{\varepsilon}{2}$$

and obtain

$$\frac{\pi^2\beta^2}{\alpha^2} \mathrm{e}^{\pi^2\beta^2/\alpha^2} = \frac{2^{10}\alpha^2 Q^4}{N^2 V^2 \varepsilon^4} \iff \beta = \frac{\alpha}{\pi} \sqrt{\mathrm{W}\left(\frac{2^{10}\alpha^2 Q^4}{N^2 V^2 \varepsilon^4}\right)},$$

where we denote by $\mathrm{W}(\cdot)$ the well known Lambert W function, which is implicitly defined by

$$\mathrm{W}(x)\mathrm{e}^{\mathrm{W}(x)} = x.$$

iii) Set $\boldsymbol{M} := 2\left\lceil \frac{\beta}{2}\boldsymbol{L} \right\rceil \in 2\mathbb{N}^3$ (round up component wise).

□

Assuming that the near field and the far field part of the error are independent of each other we have

$$\Delta F \approx \sqrt{(\Delta F_{\approx}^{\mathrm{S}})^2 + (\Delta F_{\approx}^{\mathrm{L}})^2} \approx \frac{\varepsilon}{\sqrt{2}} < \varepsilon.$$

Thus, we expect that the overall rms force error is indeed bounded above by the given (absolute) accuracy.

Example 3.4. We applied the above described tuning for the calculation of the forces in the cloud wall system we already considered in the Example above. We started with different near field cutoffs r_{cut} as well as required accuracies and approximated the potentials as well as the forces by using the ScaFaCoS library [1], where we applied the NDFT as well as the adjoint NDFT in order to compute the Fourier sums. In the table we list the tuned values for the splitting parameter α and for the mesh size $\boldsymbol{M}$. Additionally, we give the corresponding obtained rms force errors ΔF, which are computed by the given reference data. We see that the required accuracy is achieved in each case. □

r_{cut}	ε	α	M_1	M_2	M_3	ΔF
3.0	1.0e-04	1.0244	42	22	22	4.3568e-05
3.0	1.0e-06	1.2495	60	30	30	4.0693e-07
3.0	1.0e-08	1.4397	80	40	40	3.9889e-09
3.0	1.0e-10	1.6077	100	50	50	3.8680e-11
4.0	1.0e-04	0.7625	30	16	16	4.4317e-05
4.0	1.0e-06	0.9323	46	24	24	4.1773e-07
4.0	1.0e-08	1.0756	60	30	30	3.9316e-09
4.0	1.0e-10	1.2020	74	38	38	3.7935e-11
5.0	1.0e-04	0.6063	24	12	12	5.2145e-05
5.0	1.0e-06	0.7428	36	18	18	5.0385e-07
5.0	1.0e-08	0.8579	48	24	24	5.2132e-09
5.0	1.0e-10	0.9593	60	30	30	4.6912e-11

Table 3.1: Tuned parameters (see Algorithm 3.1) as well as achieved rms force errors ΔF (computed with the available reference data) for different combinations of the near field cutoff r_{cut} and the required accuracy ε. (Test case: cloud wall system with $N = 600$ particles in a box with edge lengths $\boldsymbol{L} = (20, 10, 10)$.)

We have seen that the rms force error can be predicted very precisely. Additionally, given some near field cutoff radius r_{cut} and a required accuracy ε, the splitting parameter α and the far field cutoff $\boldsymbol{M}$ can be tuned such that this accuracy is achieved. Based on the used hard- and software, some specific value for r_{cut} will be optimal with respect to runtime.

However, the computation of the long range part requires $\mathcal{O}(N^2)$ arithmetic operations. In order to enable a more efficient computation we apply the NFFT algorithms, as we describe in the following section.

4. NFFT based fast Ewald summation and rms errors

We consider the efficient evaluation of the truncated long range parts of the potentials

$$\phi_{\mathrm{t},j}^{\mathrm{L}} := \frac{1}{\pi V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{\psi}(\boldsymbol{k}) S(\boldsymbol{k}) \mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\cdot\boldsymbol{x}_j}.$$

The sums

$$S(\boldsymbol{k}) = \sum_{i=1}^{N} q_i \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\cdot\boldsymbol{x}_i} = \sum_{i=1}^{N} q_i \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}, \quad \boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}},$$

can be approximated by the adjoint NFFT, $S(\boldsymbol{k}) \approx \tilde{S}(\boldsymbol{k})$, $\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}$. After a multiplication with the Fourier coefficients $\hat{\psi}(\boldsymbol{k})$ we can compute the outer sums

$$\frac{1}{\pi V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \hat{\psi}(\boldsymbol{k}) \tilde{S}(\boldsymbol{k}) \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}, \quad j = 1,\dots N,$$

via the NFFT.

The fast computation of the truncated versions of the forces' long range parts $\boldsymbol{F}_{\mathrm{t},j}^{\mathrm{L}}$, as defined in (3.9), can be done in an analog manner. Note that the outer sums have to be computed by a vector valued NFFT, i.e., three one-dimensional NFFTs are needed. The above described method (P^{2}NFFT) is part of the publicly available ScaFaCoS library [1]. The NFFT and the adjoint NFFT are computed by using the parallel FFT (PFFT) software [27]. See [36] for another implementation of the described algorithm.

In the following we derive the expression of the corresponding rms error for arbitrary window functions φ. At first we approximate the sums $S(\boldsymbol{k})$ by $\tilde{S}(\boldsymbol{k})$ via the adjoint NFFT

$$\boldsymbol{F}_{\mathrm{t},j}^{\mathrm{L}} \approx \boldsymbol{F}_{\mathrm{nffth},j}^{\mathrm{L}} := \frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1}) \hat{\psi}(\boldsymbol{k}) \tilde{S}(\boldsymbol{k}) \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}.$$

In the second step we approximate $\boldsymbol{F}_{\mathrm{nffth},j}^{\mathrm{L}}$ via a vector valued 3d-NFFT

$$\boldsymbol{F}_{\mathrm{nffth},j}^{\mathrm{L}} \approx \tilde{\boldsymbol{F}}_{\mathrm{nffth},j}^{\mathrm{L}}.$$

Lemma 4.1. *Let an electrical neutral system* (3.2) *of N charges $q_j\in\mathbb{R}$ at positions $\boldsymbol{x}_j\in\mathcal{B} := L_1\mathbb{T}\times L_2\mathbb{T}\times L_3\mathbb{T}$ be given. Suppose that the truncated long range parts of the forces $\boldsymbol{F}_{\mathrm{t},j}^{\mathrm{L}}$ as defined in* (3.9) *are computed via the NFFT based method by using the oversampled mesh $\mathcal{I}_{\boldsymbol{M}_\mathrm{o}}$ and a symmetric window function φ. Then, the resulting error in the forces can for each $j=1,\dots,N$ be written in the form*

$$\boldsymbol{F}_{\mathrm{t},j}^{\mathrm{L}} - \tilde{\boldsymbol{F}}_{\mathrm{nffth},j}^{\mathrm{L}} = q_j \sum_{\substack{i=1\\ i\neq j}}^{N} q_i \boldsymbol{\chi}_{ij}.$$

Thereby, the quadratic mean of the error terms $\boldsymbol{\chi}_{ij}$ has the lower bound

$$\chi^2 := \frac{1}{V}\int_{\mathcal{B}}\frac{1}{V}\int_{\mathcal{B}}\left|\boldsymbol{\chi}_{ij}\right|^2 \mathrm{d}\boldsymbol{x}_j \mathrm{d}\boldsymbol{x}_i \geq \chi^2_{\mathrm{opt}},$$

where

$$\chi^2_{\mathrm{opt}} := \frac{4}{V^2}\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}}\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2\hat{\psi}(\boldsymbol{k})^2\left(1-\left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3}\frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}\right]^{-2}\right). \tag{4.1}$$

The corresponding optimal NFFT deconvolution coefficients are given by (2.7)*. If the deconvolution coefficients are set as in* (2.6)*, the expected quadratic error reads as*

$$\chi^2 = \frac{4}{V^2}\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}}\|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2\hat{\psi}(\boldsymbol{k})^2\left(\left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3}\frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}\right]^2-1\right) =: \chi^2_{\mathrm{standard}}. \tag{4.2}$$

Proof. See Appendix A. ■

We denote by

$$\Delta F^{\mathrm{L}}_{\mathrm{fast}} := \sqrt{\frac{1}{N}\sum_{j=1}^{N}\left\|\boldsymbol{F}^{\mathrm{L}}_{\mathrm{t}}-\tilde{\boldsymbol{F}}^{\mathrm{L}}_{\mathrm{nffth},j}\right\|^2} \approx \frac{\chi Q}{\sqrt{N}} \tag{4.3}$$

the resulting RMS error.

Remark 4.2. Note the difference to the derivation of the optimal influence function by Hockney and Eastwood [17, Section 8-3-3]. The optimal influence function is derived by considering the approximation of

$$\sum_{\boldsymbol{k}\in\mathbb{Z}^3}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\left(\sum_{i=1}^{N}q_i\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right)\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}$$

by transforming the continuous charge distribution into a grid based charge density. In contrast, we already start with the truncated sum

$$\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}}(\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\left(\sum_{i=1}^{N}q_i\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right)\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}.$$

However, an appropriate comparison of the obtained errors or rather the optimal deconvolution coefficients shows the equivalence of the P^{3}M and the P^{2}NFFT method, where we have to set the oversampling factor $\boldsymbol{\sigma} := (1,1,1)$ and apply the B-spline window function. In other words, the P^{3}M method is a special case of the P^{2}NFFT. □

4.1. Efficient computation of the resulting rms errors

In the following we will discuss how the above derived expressions for the rms force error can be estimated efficiently. In three dimensions we use a tensor product approach (2.4) in order to construct the window function φ by only using univariate functions.

Thus, also the Fourier coefficients $c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})$, $\boldsymbol{k} \in \mathbb{Z}^3$, are of a tensor product structure

$$c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}}) = \prod_{j=1}^{3} c_{k_j}(\tilde{\varphi}_{j,\mathrm{t}}).$$

In order to estimate the rms force error efficiently, we separate the computations regarding the three dimensions by using an approximation of the form

$$\frac{1}{x} \approx \sum_{i=1}^{n} r_i \mathrm{e}^{-w_i x} \quad \text{for } x \in [1, \ell),$$

where $\ell \gg 1$ should be chosen large enough. Such an approximation can be obtained with the help of the well known ESPRIT algorithm [33], see [32] for instance, or by using the Remez algorithm, which was also used by the authors of [15].

For $\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}$ and $L_{\max} := \max\{L_1, L_2, L_3\}$ we have

$$1 \le x := L_{\max}^2 \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 \le L_{\max}^2 \left(\tfrac{M_1^2}{4L_1^2} + \tfrac{M_2^2}{4L_2^2} + \tfrac{M_3^2}{4L_3^2}\right) = \frac{3\beta^2 L_{\max}^2}{4}.$$

Thus, we should choose $\ell \ge L_{\max}^2 \left(\tfrac{M_1^2}{4L_1^2} + \tfrac{M_2^2}{4L_2^2} + \tfrac{M_3^2}{4L_3^2}\right)$.

Example 4.3. For quadratic box shapes, i.e., $\boldsymbol{L} := (L, L, L)$ and $\boldsymbol{M} := (M, M, M)$ we have

$$L_{\max}^2 \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 \le \|\boldsymbol{k}\|^2 \le \frac{3M^2}{4} < 2 \cdot 10^5 \quad \forall\, M \le 512.$$

In [15] the authors provide an approximation with only $n = 11$ exponential terms with

$$\max_{x \in [1, 2\cdot 10^5)} \left| \frac{1}{x} - \sum_{i=1}^{11} r_i \mathrm{e}^{-w_i x} \right| \le 7 \cdot 10^{-6}.$$

For the numerical experiments considered in this paper, it is sufficient to use this approximation, as we will see later. For larger values of M, another approximation has to be selected. Of course, in the case $M \ll 512$, we could also use even shorter approximation sums. □

Now, we have for $\boldsymbol{k} \neq \boldsymbol{0}$

$$\begin{aligned} \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 &= \frac{\mathrm{e}^{-2\pi^2 \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2/\alpha^2}}{\|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2} = \frac{L_{\max}^2 \mathrm{e}^{-2\pi^2 \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2/\alpha^2}}{\frac{L_{\max}^2 k1^2}{L_1^2} + \frac{L_{\max}^2 k2^2}{L_2^2} + \frac{L_{\max}^2 k3^2}{L_3^2}} \\ &\approx L_{\max}^2 \sum_{i=1}^{n} r_i \mathrm{e}^{-(2\pi^2/\alpha^2 + w_i L_{\max}^2)(k_1^2/L_1^2 + k_2^2/L_2^2 + k_3^2/L_3^2)}. \end{aligned} \tag{4.4}$$

Assume that we know the upper bounds

$$\sum_{r \in \mathbb{Z} \setminus \{0\}} c_{k + r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}}) \le s_j(k), \tag{4.5}$$

where $k \in \mathcal{I}_{M_j}$. Then

$$\left(\sum_{r \in \mathbb{Z}} c_{k + r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}}) \right)^2 \le c_k^4(\tilde{\varphi}_{j,\mathrm{t}}) + 2c_k^2(\tilde{\varphi}_{j,\mathrm{t}}) s_j(k) + s_j^2(k) =: c_k^4(\tilde{\varphi}_{j,\mathrm{t}}) + \tilde{s}_j(k)$$

and for $\boldsymbol{k} = (k_1, k_2, k_3) \in \mathcal{I}_{\boldsymbol{M}}$ we obtain

$$\left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3} \frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})}\right]^2 \leq \prod_{j=1}^{3}\left[1+\frac{\tilde{s}_j(k_j)}{c^4_{k_j}(\tilde{\varphi}_{j,\mathrm{t}})}\right]$$
$$= 1+\sum_{j=1}^{3}\frac{\tilde{s}_j(k_j)}{c^4_{k_j}(\tilde{\varphi}_{j,\mathrm{t}})} + \sum_{j_1<j_2}\frac{\tilde{s}_{j_1}(k_{j_1})}{c^4_{k_{j_1}}(\tilde{\varphi}_{j_1,\mathrm{t}})}\frac{\tilde{s}_{j_2}(k_{j_2})}{c^4_{k_{j_2}}(\tilde{\varphi}_{j_2,\mathrm{t}})} + \prod_{j=1}^{3}\frac{\tilde{s}_j(k_j)}{c^4_{k_j}(\tilde{\varphi}_{j,\mathrm{t}})}.$$

For $i = 1, \ldots, n$ and $j = 1, \ldots, 3$ we define the sums

$$S_{0,i,j} := \sum_{k\in\mathcal{I}_{M_j}} \mathrm{e}^{-(2\pi^2/\alpha^2+w_iL^2_{\max})k^2/L_j^2}$$
$$S_{1,i,j} := \sum_{k\in\mathcal{I}_{M_j}} \frac{\tilde{s}_j(k)}{c^4_k(\tilde{\varphi}_{j,\mathrm{t}})}\mathrm{e}^{-(2\pi^2/\alpha^2+w_iL^2_{\max})k^2/L_j^2}$$

and obtain for $\chi^2_{\mathrm{standard}}$, as defined in (4.2), by applying (4.4)

$$\chi^2_{\mathrm{standard}} \leq \frac{4}{V^2}\sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2\hat{\psi}(\boldsymbol{k})^2\left(\prod_{j=1}^{3}\left[1+\frac{\tilde{s}_j(k_j)}{c^4_{k_j}(\tilde{\varphi}_{j,\mathrm{t}})}\right]-1\right)$$
$$\approx \frac{4L^2_{\max}}{V^2}\sum_{i=1}^{n} r_i\left(\sum_{j_1,j_2,j_3}(S_{0,i,j_1}S_{0,i,j_2}S_{1,i,j_3}+S_{1,i,j_1}S_{1,i,j_2}S_{0,i,j_3})+S_{1,i,j_1}S_{1,i,j_2}S_{0,i,j_3}\right)$$
$$-\frac{4L^2_{\max}}{V^2}\left(\prod_{j=1}^{3}\left[1+\frac{\tilde{s}_j(0)}{c^4_0(\tilde{\varphi}_{j,\mathrm{t}})}\right]-1\right)\sum_{i=1}^{n} r_i, \tag{4.6}$$

i.e., we can estimate the error with $\mathcal{O}(n(M_1+M_2+M_3))$ arithmetic operations.

For the computation of χ^2_{opt} as defined in (4.1) we define the sums

$$R_{0,i,j} := \sum_{k\in\mathcal{I}_{M_j}} \frac{c^4_k(\tilde{\varphi}_{j,\mathrm{t}})}{c^4_k(\tilde{\varphi}_{j,\mathrm{t}})+\tilde{s}_j(k)}\mathrm{e}^{-(2\pi^2/\alpha^2+w_iL^2_{\max})k^2/L_j^2}$$
$$R_{1,i,j} := \sum_{k\in\mathcal{I}_{M_j}} \frac{\tilde{s}_j(k)}{c^4_k(\tilde{\varphi}_{j,\mathrm{t}})+\tilde{s}_j(k)}\mathrm{e}^{-(2\pi^2/\alpha^2+w_iL^2_{\max})k^2/L_j^2}$$

and obtain

$$\begin{aligned}
\chi^2_{\mathrm{opt}} &\le \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left(1 - \prod_{j=1}^{3} \frac{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}})}{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}}) + \tilde{s}_j(k_j)}\right) \\
&= \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \prod_{j=1}^{3} \frac{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}})}{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}}) + \tilde{s}_j(k_j)} \left(\prod_{j=1}^{3}\left[1 + \frac{\tilde{s}_j(k_j)}{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}})}\right] - 1\right) \\
&\approx \frac{4L_{\max}^2}{V^2} \sum_{i=1}^{n} r_i \left(\sum_{j_1,j_2,j_3} (R_{0,i,j_1}R_{0,i,j_2}R_{1,i,j_3} + R_{1,i,j_1}R_{1,i,j_2}R_{0,i,j_3}) + R_{1,i,j_1}R_{1,i,j_2}R_{0,i,j_3}\right) \\
&\quad - \frac{4L_{\max}^2}{V^2} \left(1 - \prod_{j=1}^{3} \frac{c_0^4(\tilde{\varphi}_{j,\mathrm{t}})}{c_0^4(\tilde{\varphi}_{j,\mathrm{t}}) + \tilde{s}_j(0)}\right) \sum_{i=1}^{n} r_i. \qquad (4.7)
\end{aligned}$$

In the case that the sums

$$\sum_{r\in\mathbb{Z}} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}})$$

can be computed exactly (standard B-splines, for instance) we obtain

$$\begin{aligned}
\chi^2_{\mathrm{opt}} &\le \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \prod_{j=1}^{3} \frac{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}})}{\left(\sum_{r\in\mathbb{Z}} c_{k_j+r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}})\right)^2} \left(\prod_{j=1}^{3}\left[1 + \frac{\tilde{s}_j(k_j)}{c_{k_j}^4(\tilde{\varphi}_{j,\mathrm{t}})}\right] - 1\right) \\
&\approx \frac{4L_{\max}^2}{V^2} \sum_{i=1}^{n} r_i \left(\sum_{j_1,j_2,j_3} (Q_{0,i,j_1}Q_{0,i,j_2}Q_{1,i,j_3} + Q_{1,i,j_1}Q_{1,i,j_2}Q_{0,i,j_3}) + Q_{1,i,j_1}Q_{1,i,j_2}Q_{0,i,j_3}\right) \\
&\quad - \frac{4L_{\max}^2}{V^2} \prod_{j=1}^{3} \frac{c_0^4(\tilde{\varphi}_{j,\mathrm{t}})}{\left(\sum_{r\in\mathbb{Z}} c_{k_j+r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}})\right)^2} \left(\prod_{j=1}^{3}\left[1 + \frac{\tilde{s}_j(0)}{c_0^4(\tilde{\varphi}_{j,\mathrm{t}})}\right] - 1\right) \sum_{i=1}^{n} r_i, \qquad (4.8)
\end{aligned}$$

where

$$\begin{aligned}
Q_{0,i,j} &:= \sum_{k\in\mathcal{I}_{M_j}} \frac{c_k^4(\tilde{\varphi}_{j,\mathrm{t}})}{\left(\sum_{r\in\mathbb{Z}} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}})\right)^2} \mathrm{e}^{-(2\pi^2/\alpha^2 + w_i L_{\max}^2)k^2/L_j^2} \\
Q_{1,i,j} &:= \sum_{k\in\mathcal{I}_{M_j}} \frac{\tilde{s}_j(k)}{\left(\sum_{r\in\mathbb{Z}} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_{j,\mathrm{t}})\right)^2} \mathrm{e}^{-(2\pi^2/\alpha^2 + w_i L_{\max}^2)k^2/L_j^2}.
\end{aligned}$$

4.2. B-spline vs. Bessel window

We consider the the two window functions we introduced in Section 2 and evaluate the occurrent rms errors, as described above, for some test scenarios.

B-spline window

For the B-spline window the upper bounds $s_j(k)$ as introduced by (4.5) are given in

$$s_j(k) = c_k^2(\tilde{\varphi}_j) \cdot \frac{8m}{4m-1} \left(\frac{|k|}{|k| - \sigma_j M_j} \right)^{4m} \tag{4.9}$$

in the case $b_j = m$. The sums

$$\sum_{r \in \mathbb{Z}} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}_j)$$

are known exactly, see (2.12), so that we can apply (4.8) in order to approximate the $\mathcal{L}_2$-optimized rms error. On the other hand, if $b_j \neq m$ and $m/b_j \notin \mathbb{Z}$ we obtain with some appropriate $R \in \mathbb{N}$

$$s_j(k) = \sum_{0<|r|\leq R} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}) + \frac{m^2}{\sigma_j^2 M_j^2 b_j^2} \frac{\left(\frac{|k|}{\sigma_j M_j} + R\right)^{1-4b_j} - \left(\frac{|k|}{\sigma_j M_j} - R\right)^{1-4b_j}}{\left(\frac{m\pi}{b_j}\right)^{4b_j} (4b_j - 1)}, \tag{4.10}$$

as provided in Section 2.

Example 4.4. We consider a cubic box shape with $\boldsymbol{L} = (L, L, L)$, where we set $L = 10$, and compute the quadratic means χ_{opt} as well as χ_{standard} for different values of α. Thereby, we use different values for m, $\boldsymbol{M} := (M, M, M)$ and $\boldsymbol{\sigma} := (\sigma, \sigma, \sigma)$, i.e., the same oversampling factor σ is used for all three dimensions. Thus, we also use the same shape parameter b in each dimension, i.e., we set

$$\boldsymbol{b} := (b, b, b) \tag{4.11}$$

for $b \in 1/2\mathbb{N}$.

For each configuration, we computed the predicted errors (4.6) and (4.7)/(4.8) (depending on the shape parameter) for all

$$b \in 1/2\mathbb{N} : m/2 < b \leq m,$$

as suggested in [25], and picked out the smallest error as well as the corresponding optimal shape parameter b_{opt}. The results are plotted in Figures 4.1, 4.2 and 4.3. For comparison we also plot the errors obtained by using the standard B-spline window, i.e. $b := m$.

It can be seen, that there are only insignificant differences between the two errors χ_{opt} and χ_{standard}, which is supposed to be due to the rapid decrease of the Fourier coefficients $\|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 \hat{\tilde{\psi}}(\boldsymbol{k})^2$. Furthermore, the usage of a shape parameter $b \neq m$ allows only small improvements. □

Bessel window

For the Bessel I_0 window the upper bounds $s_j(k)$ for $k \in \mathcal{I}_{M_j}$ are obtained as described in Section 2, i.e., we set

$$s_j(k) = \sum_{0<|r|\leq R} c_{k+r\sigma_j M_j}^2(\tilde{\varphi}) + \frac{\ln\left(\frac{2\pi(|k|/\sigma_j M_j - R) - b_j}{2\pi(|k|/\sigma_j M_j - R) + b_j}\right) + \ln\left(\frac{2\pi(|k|/\sigma_j M_j + R) + b_j}{2\pi(|k|/\sigma_j M_j + R) - b_j}\right)}{4\pi b \sigma_j^2 M_j^2}. \tag{4.12}$$

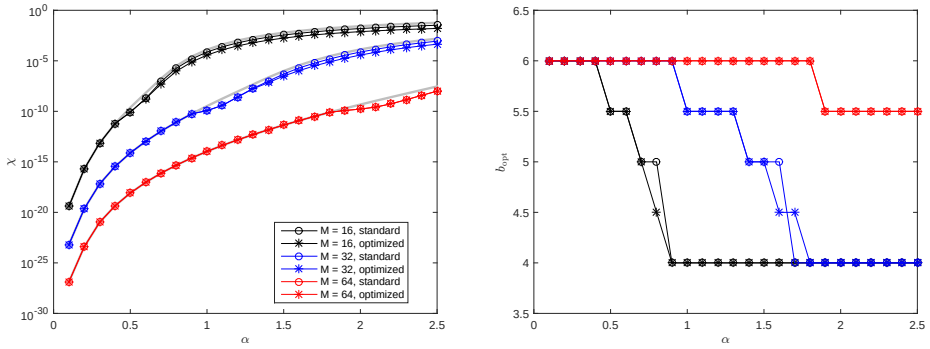

Figure 4.1: Approximate values for χ_{opt} (4.7)/(4.8) ($*$) and χ_{standard} (4.6) ($\circ$) for different far field cutoffs M with respect to the splitting parameter α (left). Thereby we choose the support parameter $m = 6$ and the oversampling factor $\sigma = 1$. The corresponding optimal shape parameters b_{opt} are given in the plot on the right hand side. On the left hand side we also plot the results obtained by using the standard B-spline window (gray), where $b := m$.

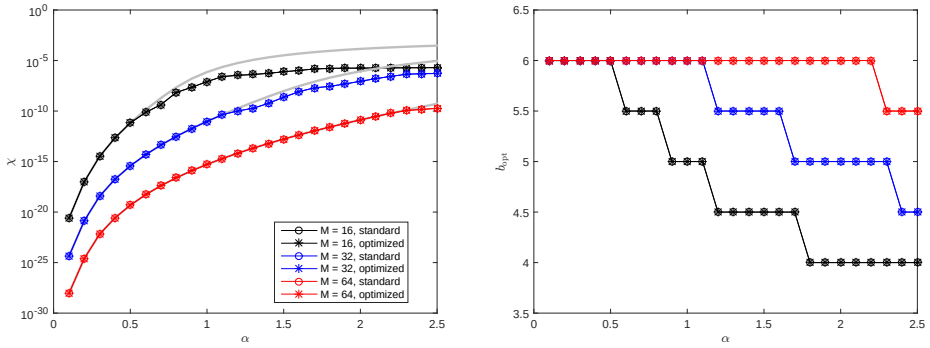

Figure 4.2: Approximate values for χ_{opt} (4.7)/(4.8) ($*$) and χ_{standard} (4.6) ($\circ$) for different far field cutoffs M with respect to the splitting parameter α (left). Thereby we choose the support parameter $m = 6$ and the oversampling factor $\sigma = 1.25$. The corresponding optimal shape parameters b_{opt} are given in the plot on the right hand side. On the left hand side we also plot the results obtained by using the standard B-spline window (gray), where $b := m$.

for some appropriate $R > \frac{|k|}{\sigma_j M_j} + \frac{b_j}{2\pi}$.

We suggest the following approach to tune the shape parameter $\boldsymbol{b}$ automatically, similar to [25, Alg. 5.2]. For simplicity we again assume that approximately the same oversampling factor is applied in all three dimensions, i.e., we have

$$\boldsymbol{\sigma} \approx (\sigma, \sigma, \sigma)$$

where $\sigma \geq 1$. Thus, it is reasonable to use also the same shape parameter $b > 0$ in the single dimensions, i.e., we define $\boldsymbol{b}$ via (4.11). Note that the oversampled grid size has to be set via

$$M_{\text{o},j} := 2 \left\lceil \tfrac{\sigma}{2} M_j \right\rceil,$$

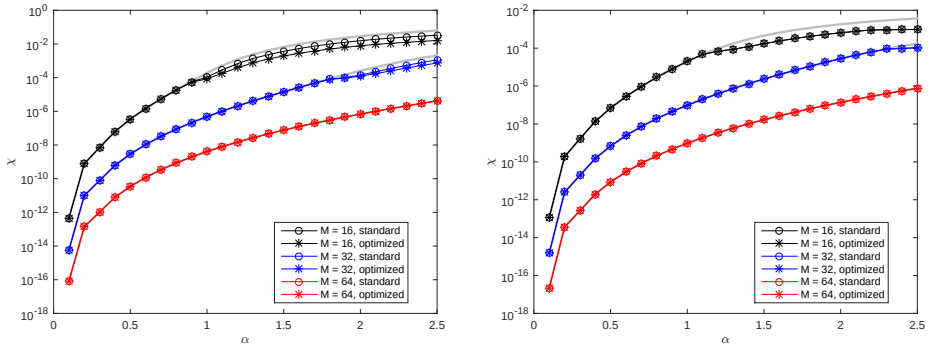

Figure 4.3: Approximate values for χ_{opt} (4.7)/(4.8) (∗) and χ_{standard} (4.6) (○) for different far field cutoffs M with respect to the splitting parameter α. Thereby we choose the support parameter $m = 3$ combined with the oversampling factors $\sigma = 1$ (left) and $\sigma = 1.25$ (right). For each configuration we use the optimal shape parameter b_{opt}. For comparison, we also plot the results obtained by using the standard B-spline window (gray), where $b := m$.

since we need $\boldsymbol{M}_{\text{o}} \in 2\mathbb{N}^3$. Thus, the oversampling factors $\sigma_j := M_{\text{o},j}/M_j$, which are actually applied in the single dimensions, may slightly differ from σ.

In Algorithm 4.1 we denote by

$$\Delta F_{\text{fast}}(b, m, \boldsymbol{\sigma})$$

the predicted rms error (4.3), which depends on the chosen shape parameter b, the support parameter m, the oversampling factor $\boldsymbol{\sigma}$ as well as the applied deconvolution scheme. Based on the given sets of parameters the resulting error terms χ can be approximated by (4.6) and respectively (4.7).

Algorithm 4.1 (Shape parameter tuning for the Bessel window)**.**
Input: Splitting parameter α, box size vector $\boldsymbol{L}$, far field cutoff $\boldsymbol{M}$, support parameter m, oversampling factor σ, which is to be applied in all three dimensions.

i) Compute the actual applied oversampling factor: $M_{\text{o},j} := 2\left\lceil \frac{\sigma}{2} M_j \right\rceil$, $\sigma_j := \frac{M_{\text{o},j}}{M_j}$.

ii) Set $b_{\text{opt}} := 2\pi(1 - 1/2\sigma)$, i.e., take the standard shape parameter (2.13) as a first guess for the optimal shape parameter.

iii) Choose a start step size d, e.g. $d := \frac{b_{\text{opt}}}{2} = \pi(1 - 1/2\sigma)$.

iv) Compute $\Delta F^2_{\text{fast,opt}} := \Delta F^2_{\text{fast}}(b_{\text{opt}}, m, \boldsymbol{\sigma})$.

v) Set $b_{\text{left}} := b_{\text{opt}} - d$ and $\Delta F^2_{\text{fast,left}} := \Delta F^2_{\text{fast}}(b_{\text{left}}, m, \boldsymbol{\sigma})$.

vi) Set $b_{\text{right}} := b_{\text{opt}} + d$ and $\Delta F^2_{\text{fast,right}} := \Delta F^2_{\text{fast}}(b_{\text{right}}, m, \boldsymbol{\sigma})$.

vii) Until $\max\{\Delta F^2_{\text{fast,left}}, \Delta F^2_{\text{fast,opt}}, \Delta F^2_{\text{fast,right}}\} \approx \min\{\Delta F^2_{\text{fast,left}}, \Delta F^2_{\text{fast,opt}}, \Delta F^2_{\text{fast,right}}\}$

- If $\min\{\Delta F^2_{\text{fast,left}}, \Delta F^2_{\text{fast,opt}}, \Delta F^2_{\text{fast,right}}\} = \Delta F^2_{\text{fast,opt}}$:

  a) Set $d := d/2$, i.e., choose a smaller step size.

  Else:

a) $b_{\text{opt}} := \underset{b_{\text{left}}, b_{\text{opt}}, b_{\text{right}}}{\arg\min} \{\Delta F^2_{\text{fast,left}}, \Delta F^2_{\text{fast,opt}}, \Delta F^2_{\text{fast,right}}\}$.

b) $\Delta F^2_{\text{fast,opt}} := \min\{\Delta F^2_{\text{fast,left}}, \Delta F^2_{\text{fast,opt}}, \Delta F^2_{\text{fast,right}}\}$.

- Set $b_{\text{left}} := b_{\text{opt}} - d$ and $\Delta F^2_{\text{fast,left}} := \Delta F^2_{\text{fast}}(b_{\text{left}}, m, \boldsymbol{\sigma})$.
- Set $b_{\text{right}} := b_{\text{opt}} + d$ and $\Delta F^2_{\text{fast,right}} := \Delta F^2_{\text{fast}}(b_{\text{right}}, m, \boldsymbol{\sigma})$.

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

Example 4.5. We again consider the case of a cubic box shape with $\boldsymbol{L} = (L, L, L)$, where we set $L = 10$, and compute the quadratic means χ_{opt} as well as χ_{standard} for different values of α, now for the Bessel window function. Thereby, we tune the shape parameter b, which is applied in all three dimensions, by the suggested tuning Algorithm 4.1. We plot the resulting values of the error terms over α in Figures 4.4, 4.5 and 4.6.

As already observed for the B-spline window, we cannot see significant differences between χ_{opt} as well as χ_{standard} for the Bessel window as well. The tuned optimal shape parameters b_{opt} adopt very different values for the different splitting parameters α. A comparison to the results obtained by using the standard shape parameter (2.13) yields significant improvements of the resulting rms errors, similar as in the numerical examples presented in [25], but is omitted here for overview purposes. □

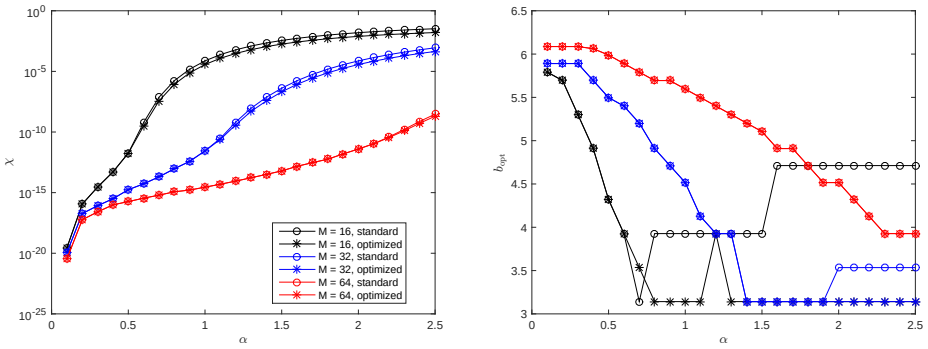

Figure 4.4: Approximate values for χ_{opt} (4.7) (∗) and χ_{standard} (4.6) (o) for different far field cutoffs M with respect to the splitting parameter α (left). Thereby we choose the support parameter $m = 6$ and the oversampling factor $\sigma = 1$. The corresponding optimal shape parameters b_{opt} are given in the plot on the right hand side. (window function: Bessel)

Comparison

We reconsider the results of the previous Examples 4.4 as well as 4.5 and compare the obtained error terms χ_{standard} for the two window functions. For the Bessel window we used the optimal shape parameters as obtained in Example 4.5. In the case of B-splines we only consider the standard B-spline window, i.e., the case $b = m$.

In Figures 4.7 and 4.8 we directly compare the obtained error sums for the two window functions. For large enough splitting parameters α the Bessel window function produces much smaller errors than the B-spline window. This is especially the case for small mesh sizes and

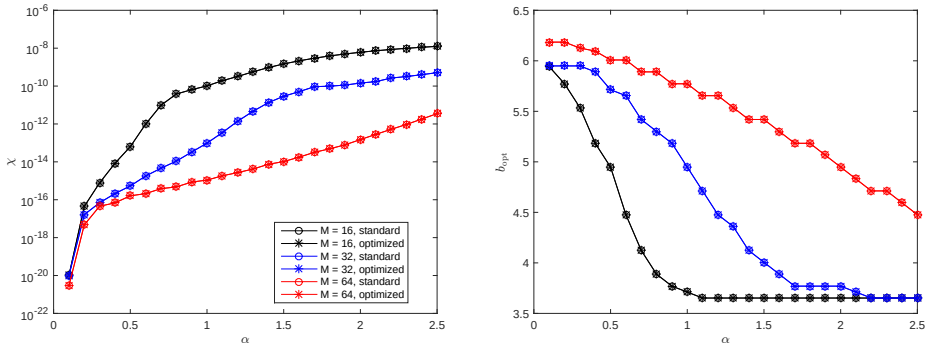

Figure 4.5: Approximate values for χ_{opt} (4.7) (∗) and χ_{standard} (4.6) (o) for different far field cutoffs M with respect to the splitting parameter α (left). Thereby we choose the support parameter $m = 6$ and the oversampling factor $\sigma = 1.25$. The corresponding optimal shape parameters b_{opt} are given in the plot on the right hand side. (window function: Bessel)

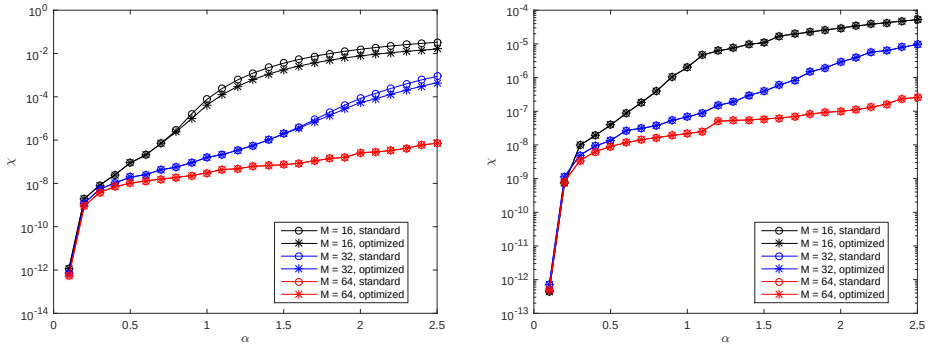

Figure 4.6: Approximate values for χ_{opt} (4.7) (∗) and χ_{standard} (4.6) (o) for different far field cutoffs M with respect to the splitting parameter α. Thereby we choose the support parameter $m = 3$ combined with the oversampling factors $\sigma = 1$ (left) and $\sigma = 1.25$ (right). For each configuration we use the optimal shape parameter b_{opt}. (window function: Bessel)

relatively large support parameters m. On the other hand, for relatively small values of α the B-spline window seems to be the better choice.

4.3. Parameter tuning and numerical examples

We recall Remarks 3.1 and 3.2, which show that the predicted errors in the Ewald summation are equal for particle systems with the same particle or rather charge density, if the same Ewald parameters r_{cut}, α and β are used.

Based on this, we hope to be able to tune all the parameters of our fast algorithm for a small particle system and apply the tuned sets of parameters also to larger systems while keeping the performance of the algorithm in terms of accuracy as well as efficiency, similar as done in [26].

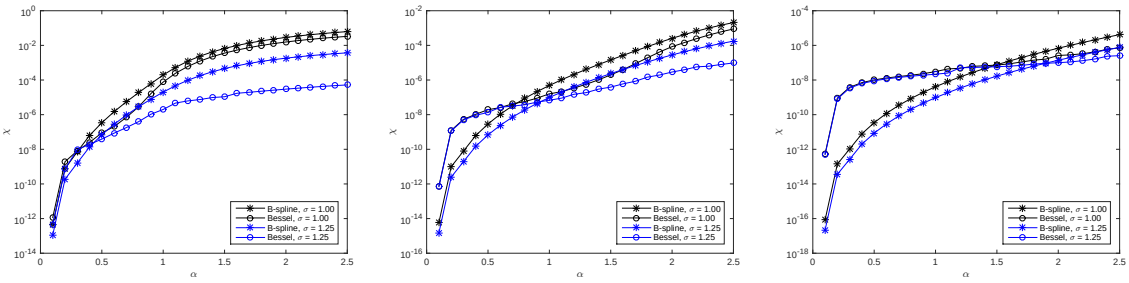

Figure 4.7: Comparison of the obtained errors χ_{standard} ($\approx \chi_{\text{opt}}$), approximated via (4.6), for the two window functions. We choose the support parameter $m = 3$ and use different meshes $\mathcal{I}_{(M,M,M)}$, where $M := 16$ (left), $M := 32$ (center) and $M := 64$ (right).

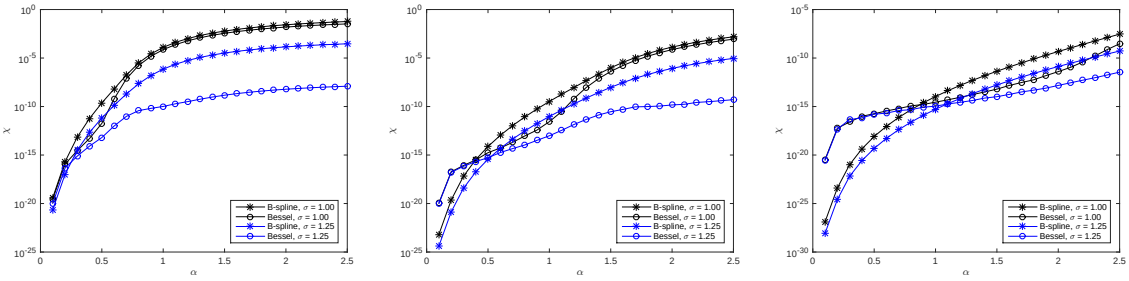

Figure 4.8: Comparison of the obtained errors χ_{standard} ($\approx \chi_{\text{opt}}$), approximated via (4.6), for the two window functions. We choose the support parameter $m = 6$ and use different meshes $\mathcal{I}_{(M,M,M)}$, where $M := 16$ (left), $M := 32$ (center) and $M := 64$ (right).

Choice of the oversampling factor

With the help of the error estimates (4.9), (4.10) and (4.12) we are able to tune the oversampling factor for a given mesh size $\boldsymbol{M}$ and support parameter m. For simplicity we again assume that approximately the same oversampling factor is applied in all three dimensions, i.e., we have $\boldsymbol{\sigma} \approx (\sigma, \sigma, \sigma)$ with $\sigma \geq 1$.

Assume that we have given some near field cutoff r_{cut} and a required accuracy ε. With the help of Algorithm 3.1 we are able to tune the Ewald summation parameters α and β appropriately.

Now, we aim to tune the NFFT parameters such that

$$\Delta F_{\text{fast}}^{\text{L}} \leq \frac{\varepsilon}{4} =: \varepsilon_2 \tag{4.13}$$

is fulfilled, i.e., we force the approximation error $\Delta F_{\text{fast}}^{\text{L}}$ as defined in (4.3) to be somewhat smaller that the Ewald type rms errors int the near field as well as in the far field. Our hope is that the the NFFT approximation errors are thus small enough such that the required accuracy ε can still be reached.

We apply a simple binary search algorithm in order to tune the required oversampling factor, see Algorithm 4.2.

Algorithm 4.2 (Tune the oversampling factor)**.**
Input: support parameter m, Ewald parameter α, NFFT mesh size $\boldsymbol{M}$, box length vector $\boldsymbol{L}$, required accuracy ε_2.

i) Define a maximum oversampling factor $\sigma_{\max}$, e.g., $\sigma_{\max} := 2$.

ii) Compute the resulting rms error $\Delta F^{\mathrm{L}}_{\mathrm{fast}}$ for $\sigma := 1$ (use the optimal shape parameter b_{opt} in case of the Bessel window). If already $\Delta F^{\mathrm{L}}_{\mathrm{fast}} \leq \varepsilon_2$, then set $\sigma_{\min} := 1$.

iii) Else: set $\sigma_{\min} := \sigma_{\max}$, $s := {}^1\!/_2(\sigma_{\max} - 1)$ and $\sigma := 1 + s$.
While $s \geq \frac{1}{M_{\max}}$:

a) Compute the actual oversampling factor: $M_{\mathrm{o},j} := 2\left\lceil \frac{\sigma}{2} M_j \right\rceil$, $\sigma_j := \frac{M_{\mathrm{o},j}}{M_j}$.

b) Compute the resulting rms error $\Delta F^{\mathrm{L}}_{\mathrm{fast}}$ for the current value of $\boldsymbol{\sigma}$.
If $\Delta F^{\mathrm{L}}_{\mathrm{fast}} \leq \varepsilon_2$, then set $\sigma_{\min} := \sigma$ and $\sigma := \sigma - s$ (try a smaller value).
Else, set $\sigma := \sigma + s$ (try a larger value).

c) Set $s := {}^s\!/_2$.

Output: required oversampling factor $\sigma_{\min}$. If the required accuracy ε_2 cannot be achieved, we simply return $\sigma_{\max}$. □

Example 4.6. We consider again the cloud wall system with $N = 600$ particles. For different values of r_{cut} we tune the Ewald parameters α and β with Algorithm 3.1 in order to reach a certain accuracy ε. In order to tune the NFFT parameters we claim (4.13) to be fulfilled.

For different values of the support parameter m we tune the oversampling factor σ by applying Algorithm 4.2 for the B-spline window as well as for the Bessel window function, where we set the maximum oversampling factor to $\sigma_{\max} := 2$.

In Tables 4.1 and 4.2 we list the tuned parameters for the two different window functions. We also list the achieved rms force errors ΔF as well as computation times t. For the given required accuracy ε there are several possible combinations of the parameters m and σ. Of course, one combination is supposed to be optimal regarding the required runtime, which may also depend on the used hardware, compiler and the like. The achieved runtimes presented in this paper include the computation of the potentials as well as the forces and have been measured on an Intel i5-2400 single core processor that runs on 3.10 GHz with 8 GB main memory. The software was built with the Gnu C Compiler at version 4.7.1 and optimization flags "-O3". For the repeated evaluation of the window function we use a third order interpolation scheme based on interpolation tables instead of evaluating the functions directly. Thus, the speed of the evaluation is independent from the used window function.

In our example, where we set $\varepsilon := 10^{-4}$, the support parameter $m = 4$ is in each case optimal with respect to runtime. Of course, for different required accuracies we expect other optimal parameter sets. For small values of the support parameter m the B-spline window requires less oversampling in order to reach the required accuracy. In contrast, for $m \geq 4$ the Bessel window function demands less or equal oversampling. This is reflected by the measured runtimes.

□

r_{cut}		$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$
3.0	$\sigma_{\min} \approx$	2.0000	1.3125	1.0625	1.0312	1.0000
3.0	$t =$	9.35e-02	4.14e-02	**3.18e-02**	3.40e-02	3.53e-02
3.0	$\Delta F =$	7.32e-05	4.24e-05	4.85e-05	4.69e-05	5.51e-05
4.0	$\sigma_{\min} \approx$	2.0000	1.3125	1.1250	1.0625	1.0625
4.0	$t =$	3.94e-02	2.26e-02	**2.03e-02**	2.28e-02	2.68e-02
4.0	$\Delta F =$	*1.81e-04*	4.43e-05	5.65e-05	4.79e-05	4.52e-05
5.0	$\sigma_{\min} \approx$	2.0000	1.3750	1.0625	1.0625	1.0625
5.0	$t =$	2.87e-02	2.23e-02	**2.11e-02**	2.41e-02	2.81e-02
5.0	$\Delta F =$	*1.03e-04*	5.17e-05	7.19e-05	5.56e-05	5.31e-05

Table 4.1: Tuned oversampling factors $\sigma_{\min}$, achieved computation times t in seconds and measured rms force errors ΔF for the B-spline window. The required absolute accuracy was set to $\varepsilon := 10^{-4}$.

r_{cut}		$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$
3.0	$\sigma_{\min} \approx$	2.0000	1.6250	1.0000	1.0000	1.0000
3.0	$t =$	9.36e-02	6.01e-02	**2.77e-02**	3.09e-02	3.53e-02
3.0	$\Delta F =$	*4.77e-04*	4.28e-05	4.93e-05	4.46e-05	4.40e-05
4.0	$\sigma_{\min} \approx$	2.0000	1.5625	1.0625	1.0000	1.0000
4.0	$t =$	3.99e-02	2.90e-02	**1.97e-02**	2.10e-02	2.50e-02
4.0	$\Delta F =$	*7.35e-04*	4.49e-05	4.41e-05	4.35e-05	4.36e-05
5.0	$\sigma_{\min} \approx$	2.0000	1.5000	1.0625	1.0625	1.0625
5.0	$t =$	2.85e-02	2.25e-02	**2.10e-02**	2.44e-02	2.80e-02
5.0	$\Delta F =$	*5.49e-04*	5.10e-05	5.30e-05	5.22e-05	5.21e-05

Table 4.2: Tuned oversampling factors $\sigma_{\min}$, achieved computation times t in seconds and measured rms force errors ΔF for the Bessel window. The required absolute accuracy was set to $\varepsilon := 10^{-4}$.

Runtime over r_{cut}

In addition to the above described parameter tuning, which needs the value of the near field cutoff r_{cut} as input, we also want to tune the parameter r_{cut} in order to achieve an (almost) optimal runtime.

For very small particle systems we may apply the above described tuning for different values of r_{cut} and compare the achieved runtimes. As an example, we again consider the (small) cloud wall system with $N = 600$ particles, see Example 4.7.

Example 4.7. For the cloud wall system consisting of $N = 600$ charges we tune the parameters as described in the previous considerations for different near field cutoffs r_{cut}, compare to Example 4.6. We consider two different required accuracies $\varepsilon \in \{10^{-4}, 10^{-7}\}$ and plot the measured runtimes over r_{cut} in Figure 4.9. Note the jumps in the runtime plots, which result from the increased computation time of the FFT in the long range part, if the oversampled grid size shows an unprofitable decomposition into prime factors.

The corresponding tuned parameters as well as the achieved rms force errors can be found in Appendix B. It can bee seen that the required accuracy is always achieved. In many cases

we obtain a more inconvenient parameter set for the B-spline window, which results in higher computation times compared to the Bessel window.

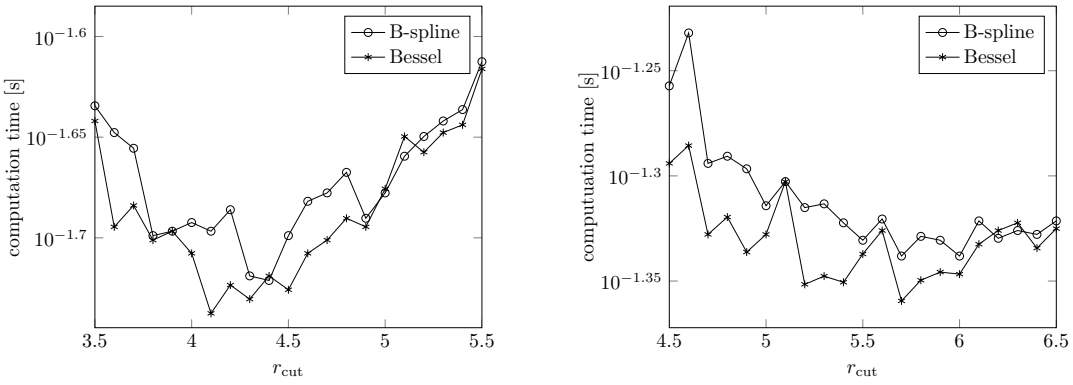

Figure 4.9: Measured runtimes for different values of r_{cut}. The required rms force accuracy was set to $\varepsilon := 10^{-4}$ (left) and to $\varepsilon := 10^{-7}$ (right). The parameters where chosen by applying Algorithms 3.1 and 4.2. For each r_{cut} we considered different combinations of m and σ, where we chose the one yielding the smallest computation time.

□

Scale parameters to larger particle systems

Based on the above described tuning algorithms we may tune all parameters for a small particle system, also with respect to runtime, in order to apply the obtained set of parameters also to larger systems. Provided that (3.8) is fulfilled, the Ewald type rms errors are supposed to be of a comparable size among a set of systems with increasing numbers of particles. Provided that the rms errors resulting from the NFFT approximations behave in an analog manner we expect to achieve also similar overall rms field errors.

Example 4.8. Based on Example 4.7 we choose the optimal parameters obtained for the small particle system ($N = 600$) and apply the same parameters (r_{cut}, α, β, m, σ, b) for computations with larger systems, for which (3.8) is fulfilled.

In the case $\varepsilon := 10^{-4}$ we choose $r_{\text{cut}} \in \{4.1, 4.4\}$ and for $\varepsilon := 10^{-7}$ we consider $r_{\text{cut}} \in \{5.7, 6.0\}$. We plot the obtained runtimes (scaled by the numbers of particles) in Figures 4.10 and 4.11. Again, the unexpected jumps in the runtimes concerning the long range parts result from the increased computation time of the FFT in case that the grid size shows an unprofitable decomposition into prime factors.

The achieved overall rms force errors can be found in Tables B.3–B.6 in Appendix B. We can see that the achieved errors are indeed of a comparable size among the considered particle systems. Note that for the Bessel window function the achieved errors are almost constant among all particle systems, whereas for the B-spline window the achieved rms force errors are in some cases somewhat larger than the required accuracy ε. In other words, the proposed approach seems to be more stable for the Bessel window function. □

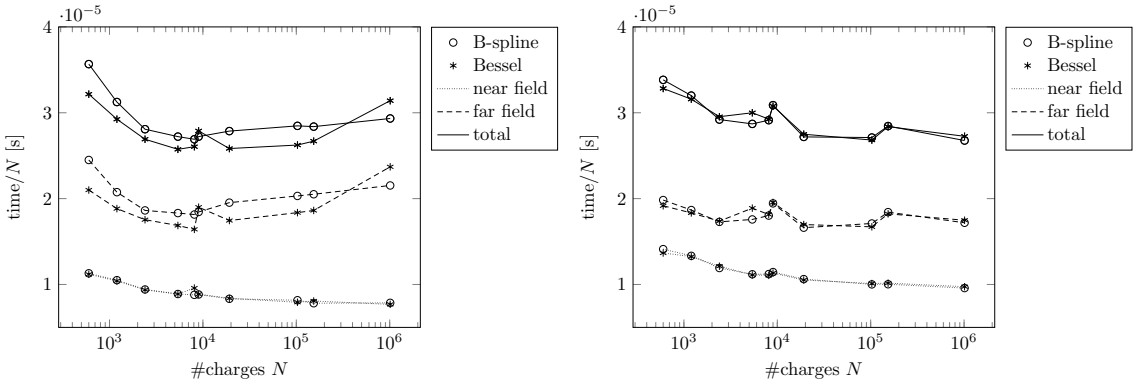

Figure 4.10: Achieved runtimes scaled by the numbers of particles. We set the required rms force accuracy to $\varepsilon := 10^{-4}$ and used the near field cutoffs $r_{\text{cut}} := 4.1$ (left) as well as $r_{\text{cut}} := 4.4$ (right), respectively.

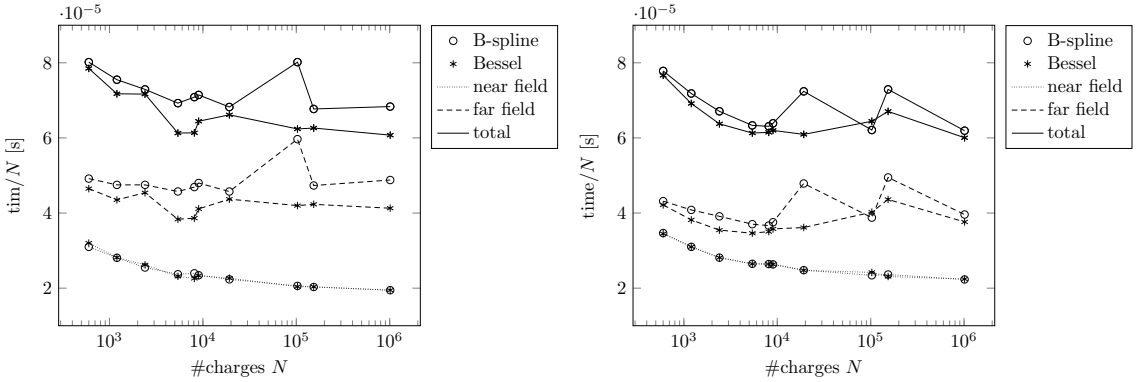

Figure 4.11: Achieved runtimes scaled by the numbers of particles. We set the required rms force accuracy to $\varepsilon := 10^{-7}$ and used the near field cutoffs $r_{\text{cut}} := 5.7$ (left) as well as $r_{\text{cut}} := 6.0$ (right), respectively.

5. Conclusion

In the present work we studied the error behavior of the P^{2}NFFT method and investigated the performance of the algorithm for the B-spline as well as the Bessel window function. We presented an approach to predict the occurrent rms errors in the forces precisely and efficiently. Based on this, we also suggest a method to tune all involved parameters automatically.

Given a required accuracy and an appropriate near field cutoff r_{cut} the splitting parameter α and the far field cutoff $\boldsymbol{M}$ can be computed easily by utilizing the error formulas for the Ewald sums. With the help of the stated error estimates for the errors caused by the NFFT approximations the NFFT parameters, as for instance the required oversampling factor and parameters describing the used window function, can also be tuned such that a certain accuracy is achieved. The presented numerical examples show that the error estimates are indeed very precise and that the proposed parameter tuning is adequate.

The results of the comparison between the two window functions can be summarized as follows. The applied combination of the near field cutoff r_{cut}, the splitting parameter α and the far field cutoff $\boldsymbol{M}$ very much influences which window function performs best in terms of accuracy. In order to achieve a required accuracy, different combinations of all involved parameters are possible. It will especially depend on the used hardware and software which set of parameters is optimal with respect to runtime. In order to optimize the method regarding runtime we claim that we can tune all involved parameters for a small particle system and apply the same parameters also to larger particle systems. We tested the described approach by considering a set of particle systems of increasing size. By applying the tuned parameters, we could achieve almost the same rms errors among all systems, which seems to be more stable if the Bessel window function is used. Furthermore, the Bessel window functions was in many cases also the better choice with respect to the required number of arithmetic operations or rather computation time.

Note that the derived representation of the rms error caused by the NFFT approximations shows that the P^{2}NFFT and the P^{3}M method are in principle equivalent. A relevant difference is that that the P^{2}NFFT approach enables the usage of different window functions and offers the possibility to use oversampling in the far field computations. The differences in the applied deconvolution schemes in terms of accuracy are negligible, as approved by some numerical examples. The tests also showed that spending some oversampling combined with a smaller support of the window function is in many cases more efficient than applying no oversampling, which requires the usage of a wider supported window function in order to achieve the same accuracy.

Appendix

A. Proof of Lemma 4.1

We can write the error $\boldsymbol{F}^{\mathrm{L}}_{\mathrm{t},j} - \tilde{\boldsymbol{F}}^{\mathrm{L}}_{\mathrm{nffth},j}$ as

$$\boldsymbol{F}^{\mathrm{L}}_{\mathrm{t},j} - \tilde{\boldsymbol{F}}^{\mathrm{L}}_{\mathrm{nffth},j} = \left(\boldsymbol{F}^{\mathrm{L}}_{\mathrm{t},j} - \boldsymbol{F}^{\mathrm{L}}_{\mathrm{nffth},j}\right) + \left(\boldsymbol{F}^{\mathrm{L}}_{\mathrm{nffth},j} - \tilde{\boldsymbol{F}}^{\mathrm{L}}_{\mathrm{nffth},j}\right),$$

where the first part can be expressed by $\boldsymbol{F}^{\mathrm{L}}_{\mathrm{t},j} - \boldsymbol{F}^{\mathrm{L}}_{\mathrm{nffth},j} =$

$$\frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k}) \left[\sum_{i=1}^{N} q_i[1-\hat{d}_{\boldsymbol{k}}c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})]\mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right] \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}$$

$$-\frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k}) \left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \sum_{i=1}^{N} q_i\hat{d}_{\boldsymbol{k}}c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})\mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right] \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})},$$

where we applied (2.11), and the second part of the error can be written as $\boldsymbol{F}^{\mathrm{L}}_{\mathrm{nffth},j} - \tilde{\boldsymbol{F}}^{\mathrm{L}}_{\mathrm{nffth},j}$

$$\frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\tilde{S}(\boldsymbol{k})[1-\hat{d}_{\boldsymbol{k}}c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})]\mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}$$

$$-\frac{2\mathrm{i}q_j}{V} \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\tilde{S}(\boldsymbol{k})\hat{d}_{\boldsymbol{k}}c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}(\tilde{\varphi}_{\mathrm{t}})\mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}})\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})},$$

cf. (2.8). The window function $\tilde{\varphi} : \mathbb{R}^3 \to \mathbb{R}$ is defined on $\mathbb{R}^3$ and is periodic with period 1 regarding each dimension. Inserting

$$\tilde{S}(\boldsymbol{k}) = \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \sum_{i=1}^{N} q_i \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})} + \sum_{i=1}^{N} q_i \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})},$$

which is obtained from (2.11), we can rewrite the error by

$$\boldsymbol{F}^\mathrm{L}_{\mathrm{t},j} - \tilde{\boldsymbol{F}}^\mathrm{L}_{\mathrm{nffth},j} = q_j \sum_{i=1}^{N} q_i \boldsymbol{\chi}_{ij},$$

where $\boldsymbol{\chi}_{ij} =$

$$\begin{aligned}
&\frac{2\mathrm{i}}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k}) \left[1 - \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})\right] \left[1 + \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})\right] \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})} \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})} \\
&-\frac{2\mathrm{i}}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right] \mathrm{e}^{-2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})} \\
&-\frac{2\mathrm{i}}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k})\hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}\boldsymbol{k}\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})} \left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}\right] \\
&-\frac{2\mathrm{i}}{V} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} (\boldsymbol{k}\odot\boldsymbol{L}^{-1})\hat{\psi}(\boldsymbol{k}) \left[\sum_{\boldsymbol{s}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{s}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{s}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})}\right] \times \\
&\qquad\qquad \left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{-2\pi\mathrm{i}(\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_j\odot\boldsymbol{L}^{-1})}\right].
\end{aligned}$$

If the window function φ is symmetric we obtain $\boldsymbol{\chi}_{jj} = \boldsymbol{0}$. In order to compute the rms error of our approximation we consider the quadratic mean

$$\chi^2 := \frac{1}{V}\int_{\mathcal{B}} \frac{1}{V}\int_{\mathcal{B}} \left|\boldsymbol{\chi}_{ij}\right|^2 \mathrm{d}\boldsymbol{x}_j \mathrm{d}\boldsymbol{x}_i,$$

where the integration domain $\mathcal{B}$ is defined as $\mathcal{B} := L_1\mathbb{T} \times L_2\mathbb{T} \times L_3\mathbb{T}$. We obtain

$$\begin{aligned}
\frac{1}{V}\int_{\mathcal{B}} \left|\boldsymbol{\chi}_{ij}\right|^2 \mathrm{d}\boldsymbol{x}_j = \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \Bigg| 1 - \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_\mathrm{t}) \\
- \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})} \Bigg|^2 \\
+ \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}^2(\tilde{\varphi}_\mathrm{t}) \times \\
\left| \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t}) + \sum_{\boldsymbol{s}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}} c_{\boldsymbol{k}+\boldsymbol{s}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t}) \mathrm{e}^{2\pi\mathrm{i}(\boldsymbol{s}\odot\boldsymbol{M}_\mathrm{o})\cdot(\boldsymbol{x}_i\odot\boldsymbol{L}^{-1})} \right|^2
\end{aligned}$$

and finally

$$\begin{aligned}\chi^2 = \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left|1-\hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}})\right|^2 \\ + \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}}) \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}}) \\ + \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \sum_{\boldsymbol{r}\in\mathbb{Z}^3\setminus\{\boldsymbol{0}\}} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}}) \sum_{\boldsymbol{r}\in\mathbb{Z}^3} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}}).\end{aligned}$$

Further simplifications give

$$\begin{aligned}\chi^2 = \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left|1-\hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}})\right|^2 - \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \hat{d}_{\boldsymbol{k}}^4 c_{\boldsymbol{k}}^4(\tilde{\varphi}_{\mathrm{t}}) \\ + \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left(\sum_{\boldsymbol{r}\in\mathbb{Z}^3} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})\right)^2.\end{aligned} \tag{A.1}$$

For $\hat{d}_{\boldsymbol{k}}, c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}}) \in \mathbb{R}$ we can compute the optimal coefficients $\hat{d}_{\boldsymbol{k}}$ as follows. Since

$$\begin{aligned}&\left(1-\hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}})\right)^2 - \hat{d}_{\boldsymbol{k}}^4 c_{\boldsymbol{k}}^4(\tilde{\varphi}_{\mathrm{t}}) + \left(\sum_{\boldsymbol{r}\in\mathbb{Z}^3} \hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})\right)^2 \\ &= 1 - 2\hat{d}_{\boldsymbol{k}}^2 c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}}) + \hat{d}_{\boldsymbol{k}}^4 \left(\sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})\right)^2 \\ &= \left(\hat{d}_{\boldsymbol{k}}^2 \sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}}) - \frac{c_{\boldsymbol{k}}^2(\tilde{\varphi}_{\mathrm{t}})}{\sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})}\right)^2 + 1 - \frac{c_{\boldsymbol{k}}^4(\tilde{\varphi}_{\mathrm{t}})}{\left(\sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})\right)^2}\end{aligned}$$

χ^2 is minimized by choosing

$$\hat{d}_{\boldsymbol{k}} := \frac{c_{\boldsymbol{k}}(\tilde{\varphi}_{\mathrm{t}})}{\sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})},$$

i.e., we have to optimize the NFFT algorithms with respect to the error in the $\mathcal{L}_2$-norm. The resulting optimal rms error in the forces is

$$\Delta F_{\mathrm{fast}}^{\mathrm{L}} \approx \frac{\chi_{\mathrm{opt}} Q}{\sqrt{N}},$$

where we obtain from (A.1)

$$\chi_{\mathrm{opt}}^2 := \frac{4}{V^2} \sum_{\boldsymbol{k}\in\mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k}\odot\boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left(1 - c_{\boldsymbol{k}}^4(\tilde{\varphi}_{\mathrm{t}}) \left[\sum_{\boldsymbol{r}\in\mathbb{Z}^3} c_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_{\mathrm{o}}}^2(\tilde{\varphi}_{\mathrm{t}})\right]^{-2}\right).$$

Setting $\hat{d}_{\boldsymbol{k}} := c_{\boldsymbol{k}}^{-1}(\tilde{\varphi}_\mathrm{t})$ we would end up with

$$\chi^2_{\text{standard}} = \frac{4}{V^2} \sum_{\boldsymbol{k} \in \mathcal{I}_{\boldsymbol{M}}} \|\boldsymbol{k} \odot \boldsymbol{L}^{-1}\|^2 \hat{\psi}(\boldsymbol{k})^2 \left(\left[\sum_{\boldsymbol{r} \in \mathbb{Z}^3} \frac{c^2_{\boldsymbol{k}+\boldsymbol{r}\odot\boldsymbol{M}_\mathrm{o}}(\tilde{\varphi}_\mathrm{t})}{c^2_{\boldsymbol{k}}(\tilde{\varphi}_\mathrm{t})} \right]^2 - 1 \right).$$

B. Tables

In Example 4.7 we considered the cloud wall system with $N = 600$ particles and tuned all involved parameters for different near field cutoffs r_cut. The corresponding tuned parameters as well as the achieved rms force errors are listed in Tables B.1 and B.2.

					B-spline			Bessel			
r_cut	α	M_1	M_2	M_3	m_opt	$\boldsymbol{\sigma}$	ΔF	m_opt	$\boldsymbol{\sigma}$	b_opt	ΔF
3.5	0.87	36	18	18	4	1.11,1.11,1.11	5.38e-05	4	1.06,1.11,1.11	4.05	4.26e-05
3.6	0.85	34	18	18	4	1.12,1.11,1.11	5.49e-05	4	1.00,1.00,1.00	3.93	4.71e-05
3.7	0.83	34	18	18	4	1.06,1.11,1.11	6.09e-05	4	1.00,1.00,1.00	3.93	5.42e-05
3.8	0.80	32	16	16	4	1.12,1.12,1.12	5.71e-05	4	1.06,1.12,1.12	4.05	4.33e-05
3.9	0.78	32	16	16	4	1.06,1.12,1.12	5.79e-05	4	1.06,1.12,1.12	4.05	4.48e-05
4.0	0.76	30	16	16	4	1.13,1.12,1.12	5.65e-05	4	1.07,1.12,1.12	3.74	4.41e-05
4.1	0.74	30	16	16	4	1.07,1.12,1.12	6.53e-05	4	1.00,1.00,1.00	3.93	5.41e-05
4.2	0.73	30	16	16	4	1.07,1.12,1.12	6.47e-05	4	1.00,1.00,1.00	3.93	5.82e-05
4.3	0.71	28	14	14	4	1.14,1.14,1.14	6.44e-05	4	1.07,1.14,1.14	3.74	4.65e-05
4.4	0.69	28	14	14	4	1.07,1.14,1.14	5.91e-05	4	1.07,1.14,1.14	4.16	4.68e-05
4.5	0.68	28	14	14	4	1.07,1.14,1.14	6.02e-05	4	1.00,1.00,1.00	3.93	4.92e-05
4.6	0.66	26	14	14	4	1.08,1.14,1.14	5.62e-05	4	1.00,1.00,1.00	3.93	5.23e-05
4.7	0.65	26	14	14	4	1.08,1.14,1.14	5.65e-05	4	1.00,1.00,1.00	3.93	5.28e-05
4.8	0.63	26	14	14	4	1.08,1.14,1.14	5.67e-05	4	1.00,1.00,1.00	3.93	5.39e-05
4.9	0.62	24	12	12	4	1.17,1.17,1.17	7.04e-05	4	1.08,1.17,1.17	3.74	5.26e-05
5.0	0.61	24	12	12	4	1.08,1.17,1.17	7.19e-05	4	1.08,1.17,1.17	4.16	5.30e-05
5.1	0.59	24	12	12	3	1.33,1.33,1.33	5.94e-05	4	1.08,1.17,1.17	4.16	4.89e-05
5.2	0.58	24	12	12	3	1.33,1.33,1.33	5.23e-05	4	1.00,1.00,1.00	3.93	4.91e-05
5.3	0.57	24	12	12	3	1.25,1.33,1.33	4.76e-05	4	1.00,1.00,1.00	3.93	4.36e-05
5.4	0.56	22	12	12	3	1.36,1.33,1.33	4.84e-05	4	1.00,1.00,1.00	3.93	4.50e-05
5.5	0.55	22	12	12	3	1.27,1.33,1.33	4.64e-05	4	1.00,1.00,1.00	3.93	4.25e-05

Table B.1: Tuned parameters and achieved rms force errors ΔF for the cloud wall system with $N = 600$ particles in a box of size $20 \times 10 \times 10$, where we started with different near field cutoffs r_cut. The required accuracy was set to $\varepsilon := 10^{-4}$.

In Example 4.8 we applied the tuned parameter sets for the small particle system ($N = 600$) also to larger systems. We list the particle sizes, the applied oversampled mesh sizes $\boldsymbol{M}_\mathrm{o}$ as well as the achieved rms force errors ΔF (by reference data) in the Tables B.3–B.6. It is easy to see that the different particle systems have the same particle or rather charge density, i.e., (3.8) is fulfilled. Furthermore, the obtained rms force errors are almost constant among all systems. In general, a slightly higher oversampling factor is needed in case of the B-spline window. Note that for the Bessel window function the achieved errors are almost constant among all particle systems, whereas for the B-spline window the achieved rms force errors are in some cases somewhat larger than the required accuracy ε.

					B-spline			Bessel			
r_{cut}	α	M_1	M_2	M_3	m_{opt}	$\boldsymbol{\sigma}$	ΔF	m_{opt}	$\boldsymbol{\sigma}$	b_{opt}	ΔF
4.5	0.89	46	24	24	6	1.13,1.17,1.17	4.34e-08	6	1.04,1.08,1.08	4.05	4.52e-08
4.6	0.87	46	24	24	7	1.04,1.08,1.08	4.76e-08	6	1.04,1.08,1.08	4.05	4.55e-08
4.7	0.85	44	22	22	6	1.14,1.18,1.18	4.72e-08	6	1.05,1.09,1.09	4.05	4.77e-08
4.8	0.84	44	22	22	6	1.14,1.18,1.18	4.81e-08	6	1.05,1.09,1.09	4.05	4.88e-08
4.9	0.82	42	22	22	6	1.14,1.18,1.18	4.74e-08	6	1.05,1.09,1.09	3.84	4.82e-08
5.0	0.80	42	22	22	6	1.10,1.09,1.09	5.59e-08	6	1.05,1.09,1.09	4.05	5.14e-08
5.1	0.79	42	22	22	6	1.10,1.09,1.09	4.89e-08	6	1.05,1.09,1.09	4.05	4.77e-08
5.2	0.77	40	20	20	6	1.15,1.20,1.20	4.74e-08	6	1.05,1.10,1.10	4.05	4.73e-08
5.3	0.76	40	20	20	5	1.25,1.30,1.30	4.53e-08	6	1.05,1.10,1.10	4.05	4.13e-08
5.4	0.74	38	20	20	5	1.26,1.30,1.30	4.23e-08	6	1.05,1.10,1.10	4.05	4.32e-08
5.5	0.73	38	20	20	6	1.11,1.10,1.10	4.38e-08	6	1.05,1.10,1.10	4.05	4.05e-08
5.6	0.72	38	20	20	6	1.11,1.10,1.10	4.28e-08	6	1.05,1.10,1.10	4.05	4.11e-08
5.7	0.70	36	18	18	5	1.28,1.33,1.33	4.97e-08	6	1.06,1.11,1.11	4.05	4.34e-08
5.8	0.69	36	18	18	5	1.28,1.33,1.33	4.55e-08	6	1.06,1.11,1.11	4.05	4.26e-08
5.9	0.68	36	18	18	6	1.11,1.11,1.11	6.56e-08	6	1.06,1.11,1.11	4.05	4.54e-08
6.0	0.67	34	18	18	6	1.12,1.11,1.11	5.38e-08	6	1.06,1.11,1.11	4.05	4.25e-08
6.1	0.66	34	18	18	6	1.12,1.11,1.11	5.07e-08	6	1.06,1.11,1.11	4.05	4.26e-08
6.2	0.64	34	18	18	5	1.24,1.22,1.22	4.47e-08	6	1.06,1.11,1.11	4.05	4.27e-08
6.3	0.63	34	18	18	5	1.24,1.22,1.22	4.29e-08	6	1.06,1.11,1.11	4.05	4.37e-08
6.4	0.62	32	16	16	5	1.31,1.38,1.38	4.70e-08	6	1.06,1.12,1.12	4.05	4.52e-08
6.5	0.61	32	16	16	6	1.12,1.12,1.12	5.98e-08	6	1.06,1.12,1.12	4.05	4.45e-08

Table B.2: Tuned parameters and achieved rms force errors ΔF for the cloud wall system with $N = 600$ particles in a box of size $20 \times 10 \times 10$, where we started with different near field cutoffs r_{cut}. The required accuracy was set to $\varepsilon := 10^{-7}$.

				Bessel				B-spline			
N	L_1	L_2	L_3	$M_{\text{o},1}$	$M_{\text{o},2}$	$M_{\text{o},3}$	ΔF	$M_{\text{o},1}$	$M_{\text{o},2}$	$M_{\text{o},3}$	ΔF
600	20	10	10	30	16	16	5.41e-05	32	18	18	6.53e-05
1200	20	10	20	30	16	30	4.91e-05	32	18	32	1.15e-04
2400	20	20	20	30	30	30	4.94e-05	32	32	32	1.17e-04
5400	30	30	20	44	44	30	4.92e-05	48	48	32	1.17e-04
8100	30	30	30	44	44	44	5.84e-05	48	48	48	1.17e-04
9000	20	30	50	30	44	74	5.65e-05	32	48	80	1.17e-04
19200	40	40	40	58	58	58	6.12e-05	62	62	62	7.10e-05
102900	70	70	70	102	102	102	6.08e-05	110	110	110	9.43e-05
153600	80	80	80	116	116	116	6.12e-05	124	124	124	7.10e-05
1012500	150	150	150	218	218	218	6.26e-05	232	232	232	1.10e-04

Table B.3: Oversampled mesh sizes and achieved rms force errors for the Bessel window function and the B-spline window applied to cloud wall systems of different size. We set $r_{\text{cut}} := 4.1$ and $\varepsilon := 10^{-4}$. See Table B.1 for the tuned parameters (α, m, $\boldsymbol{\sigma}$, b).

Acknowledgments

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

				Bessel				B-spline			
N	L_1	L_2	L_3	$M_{o,1}$	$M_{o,2}$	$M_{o,3}$	ΔF	$M_{o,1}$	$M_{o,2}$	$M_{o,3}$	ΔF
600	20	10	10	30	16	16	4.68e-05	30	16	16	5.91e-05
1200	20	10	20	30	16	30	4.61e-05	30	16	30	8.50e-05
2400	20	20	20	30	30	30	4.60e-05	30	30	30	8.57e-05
5400	30	30	20	46	46	30	4.61e-05	46	46	30	8.57e-05
8100	30	30	30	46	46	46	4.65e-05	46	46	46	6.37e-05
9000	20	30	50	30	46	74	4.88e-05	30	46	74	7.81e-05
19200	40	40	40	58	58	58	4.93e-05	58	58	58	6.55e-05
102900	70	70	70	100	100	100	4.96e-05	100	100	100	1.07e-04
153600	80	80	80	116	116	116	4.93e-05	116	116	116	6.55e-05
1012500	150	150	150	216	216	216	4.94e-05	216	216	216	9.93e-05

Table B.4: Oversampled mesh sizes and achieved rms force errors for the Bessel window function and the B-spline window applied to cloud wall systems of different size. We set $r_{\text{cut}} := 4.4$ and $\varepsilon := 10^{-4}$. See Table B.1 for the tuned parameters (α, m, $\boldsymbol{\sigma}$, b).

				Bessel				B-spline			
N	L_1	L_2	L_3	$M_{o,1}$	$M_{o,2}$	$M_{o,3}$	ΔF	$M_{o,1}$	$M_{o,2}$	$M_{o,3}$	ΔF
600	20	10	10	38	20	20	4.34e-08	46	24	24	4.97e-08
1200	20	10	20	38	20	38	4.34e-08	46	24	46	5.85e-08
2400	20	20	20	38	38	38	4.34e-08	46	46	46	5.87e-08
5400	30	30	20	56	56	38	4.40e-08	68	68	46	5.87e-08
8100	30	30	30	56	56	56	4.41e-08	68	68	68	5.24e-08
9000	20	30	50	38	56	94	4.38e-08	46	68	114	5.19e-08
19200	40	40	40	76	76	76	4.34e-08	90	90	90	4.96e-08
102900	70	70	70	130	130	130	4.44e-08	158	158	158	5.28e-08
153600	80	80	80	150	150	150	4.40e-08	180	180	180	4.96e-08
1012500	150	150	150	280	280	280	4.41e-08	338	338	338	5.29e-08

Table B.5: Oversampled mesh sizes and achieved rms force errors for the Bessel window function and the B-spline window applied to cloud wall systems of different size. We set $r_{\text{cut}} := 5.7$ and $\varepsilon := 10^{-7}$. See Table B.2 for the tuned parameters (α, m, $\boldsymbol{\sigma}$, b).

References

[1] *ScaFaCoS - Scalable Fast Coloumb Solvers*. http://www.scafacos.de.

[2] M. Abramowitz and I.A. Stegun (eds.): *Handbook of Mathematical Functions*. National Bureau of Standards, Washington, DC, USA, 1972.

[3] A. Arnold, F. Fahrenberger, C. Holm, O. Lenz, M. Bolten, H. Dachsel, R. Halver, I. Kabadshow, F. Gähler, F. Heber, J. Iseringhausen, M. Hofmann, M. Pippig, D. Potts, and G. Sutmann: *Comparison of scalable fast methods for long-range interactions*. Phys. Rev. E, 88:063308, 2013.

				Bessel				B-spline			
N	L_1	L_2	L_3	$M_{\mathrm{o},1}$	$M_{\mathrm{o},2}$	$M_{\mathrm{o},3}$	ΔF	$M_{\mathrm{o},1}$	$M_{\mathrm{o},2}$	$M_{\mathrm{o},3}$	ΔF
600	20	10	10	36	20	20	4.25e-08	38	20	20	5.38e-08
1200	20	10	20	36	20	36	4.29e-08	38	20	38	7.65e-08
2400	20	20	20	36	36	36	4.28e-08	38	38	38	7.68e-08
5400	30	30	20	54	54	36	4.29e-08	56	56	38	7.74e-08
8100	30	30	30	54	54	54	4.29e-08	56	56	56	9.06e-08
9000	20	30	50	36	54	90	4.29e-08	38	56	92	1.14e-07
19200	40	40	40	72	72	72	4.28e-08	74	74	74	5.73e-08
102900	70	70	70	124	124	124	4.25e-08	128	128	128	1.28e-07
153600	80	80	80	142	142	142	4.24e-08	146	146	146	1.33e-07
1012500	150	150	150	264	264	264	4.27e-08	272	272	272	1.49e-07

Table B.6: Oversampled mesh sizes and achieved rms force errors for the Bessel window function and the B-spline window applied to cloud wall systems of different size. We set $r_{\mathrm{cut}} := 6.0$ and $\varepsilon := 10^{-7}$. See Table B.2 for the tuned parameters (α, m, $\boldsymbol{\sigma}$, b).

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

[5] T. Darden, D. York, and L. Pedersen: *Particle mesh Ewald: An $N\log(N)$ method for Ewald sums in large systems.* J. Chem. Phys., 98:10089–10092, 1993.

[6] 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.

[7] M. Deserno and C. Holm: *How to mesh up Ewald sums. II. An accurate error estimate for the Particle-Particle-Particle-Mesh algorithm.* J. Chem. Phys., 109:7694 – 7701, 1998.

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

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

[10] U. Essmann, L. Perera, M.L. Berkowitz, T. Darden, H. Lee, and L.G. Pedersen: *A smooth particle mesh Ewald method.* J. Chem. Phys., 103:8577 – 8593, 1995.

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

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

[13] D. Frenkel and B. Smit: *Understanding molecular simulation: From algorithms to applications.* Academic Press, 2002.

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

[15] 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.

[16] F. Hedman and A. Laaksonen: *Ewald summation based on nonuniform fast Fourier transform.* Chem. Phys. Lett., 425:142 – 147, 2006.

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

[18] 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.

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

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

[21] 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.

[22] J. Kolafa and J.W. Perram: *Cutoff errors in the Ewald summation formulae for point charge systems.* Molecular Simulation, 9(5):351 – 368, 1992.

[23] S.W. de Leeuw, J.W. Perram, and E.R. Smith: *Simulation of electrostatic systems in periodic boundary conditions. I. Lattice sums and dielectric constants.* Proc. Roy. Soc. London Ser. A, 373:27 – 56, 1980.

[24] D. Lindbo and A.K. Tornberg: *Spectral accuracy in fast Ewald-based methods for particle simulations.* J. Comput. Phys., 230:8744 – 8761, 2011.

[25] F. Nestler: *Automated parameter tuning based on RMS errors for nonequispaced FFTs.* Preprint 2015-01, Faculty of Mathematics, Technische Universität Chemnitz, 2015.

[26] F. Nestler, M. Pippig, and D. Potts: *Fast Ewald summation based on NFFT with mixed periodicity.* J. Comput. Phys., 285:280 – 315, 2015.

[27] M. Pippig: *PFFT - An extension of FFTW to massively parallel architectures.* SIAM J. Sci. Comput., 35:C213 – C236, 2013.

[28] M. Pippig and D. Potts: *Particle simulation based on nonequispaced fast Fourier transforms.* In G. Sutmann, P. Gibbon, and T. Lippert (eds.): *Fast Methods for Long-Range Interactions in Complex Systems*, IAS-Series, pp. 131 – 158, Jülich, 2011. Forschungszentrum Jülich.

[29] 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.

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

[31] 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.

[32] D. Potts and M. Tasche: *Parameter estimation for nonincreasing exponential sums by Prony-like methods.* Linear Algebra Appl., 439:1024 – 1039, 2013.

[33] R. Roy and T. Kailath: *ESPRIT—estimation of signal parameters via rotational invariance techniques.* IEEE Trans. Acoustic speech and Signal Process., 37:984 – 994, 1989.

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

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

[36] Y.L. Wang, F. Hedman, M. Porcu, F. Mocci, and A. Laaksonen: *Non-uniform FFT and its applications in particle simulations.* App. Math., 5:520 – 541, 2014.

[37] Z.W. Wang and C. Holm: *Estimate of the cutoff errors in the Ewald summation for dipolar systems.* J. Chem. Phys., 115:6277–6798, 2001.

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