SAMPLING THEORY IN SIGNAL AND IMAGE PROCESSING Vol. 1, No. 1, Jan. 2002, pp. 0-50

ISSN: 1530-6429

# Fast NFFT based summation of radial functions

Markus Fenn Gabriele Steidl
Dept. of Mathematics and Computer Science,
University of Mannheim,
D–68131 Mannheim, Germany
{mfenn, steidl}@math.uni-mannheim.de

*Dedicated to Professor Paul L. Butzer on the occasion of his 75$^{th}$ birthday.*

**Abstract**

This paper is concerned with the fast summation of radial functions by the fast Fourier transform for nonequispaced data. We enhance the fast summation algorithm proposed in [20] by introducing a new regularization procedure based on the two-point Taylor interpolation by algebraic polynomials and estimate the corresponding approximation error. Our error estimates are more sophisticated than those in [20]. Beyond the kernels $K_\beta(x) = 1/|x|^\beta$ ($\beta \in \mathbb{N}$) we are also interested in the generalized multiquadrics which play an important role in the approximation of functions by radial basis functions.

*Key words and phrases*: fast discrete summation, fast Fourier transform at nonequispaced knots, generalized multiquadric

*2000 AMS Mathematics Subject Classification* — 65T40, 65T50, 65F30

## 1 Introduction

The computation of sums of the form

$$\sum_{k=1}^{N} \alpha_k K(y_j - x_k) \qquad (x_k, y_j \in \mathbb{R}^d)$$

for $j = 1, \dots, N$ with $\mathcal{O}(N^2)$ arithmetic operations appears as bottleneck in many applications where the number of knots $N$ is large. Typical examples are the simulation of particle motion in potential fields [12], the approximation of curves and surfaces by linear combinations of radial basis functions (RBFs) [22] and, in a slightly different form, the solution of integral equations or partial differential equations via boundary integral methods [15]. The most famous algorithm for the fast evaluation of these sums with only $\mathcal{O}(N)$ arithmetic operations is the fast multipole method (FMM) introduced by Greengard and Rokhlin [12, 11], e.g. for the kernel $K(x) = \log|x|$ in $\mathbb{R}^2$. Here and in the following $|\cdot|$ denotes the Euclidean norm in $\mathbb{R}^d$.

The panel clustering method developed by Hackbusch et al. [15] at the same time in the context of the numerical solution of integral equations and its more recent generalization, the $\mathcal{H}$-matrix arithmetic [13, 14] as well as the mosaic-skeleton approach of Tyrtyschnikov et al. [23, 24] follow similar ideas as the FMM. During the last years the FMM was further adapted to various kernels, e. g. to various RBFs by Beatson et al. [3, 2]. Recently, Potts and Steidl [20, 19] have proposed a fast summation algorithm based on the fast Fourier transform for nonequispaced knots (NFFT) which requires $\mathcal{O}(N \log N)$ arithmetic operations and has the following advantages:

- it resembles the well-known algorithm for the fast multiplication of vectors with Toeplitz matrices based on the FFT,
- the incooperation of new kernels is very simple,
- it has a simple structure consisting of the blocks FFT – NFFT – fast summation.

The so-called NFFT and its relative, the NFFT$^{\mathrm{T}}$, are approximative algorithms. Let $I_n^d := \{k := (k_1, k_2, \dots, k_d) \in \mathbb{Z}^d \mid -\frac{n}{2} \le k \le \frac{n}{2}\}$ with componentwise inequalities, and $\epsilon_k := \epsilon_{k_1} \cdots \epsilon_{k_d}$, where

$$\epsilon_l := \begin{cases} \frac{1}{2} & \text{if } l = \pm\frac{n}{2}, \\ 1 & \text{otherwise.} \end{cases} \tag{1.1}$$

Then, for arbitrary $w_j$ in the torus $\mathbb{T}^d := [-\frac{1}{2}, \frac{1}{2})^d$, the NFFT$(n)$ computes sums of the form

$$f_j := \sum_{k \in I_n^d} \epsilon_k \hat{f}_k \, \mathrm{e}^{-2\pi \mathrm{i} k w_j} \qquad (j = 1, \dots, M),$$

and the $\mathrm{NFFT}^{\mathrm{T}}(n)$ sums of the form

$$\hat{h}_k := \epsilon_k \sum_{j=1}^{M} f_j \, \mathrm{e}^{2\pi \mathrm{i} k w_j} \qquad (k \in I_n^d)$$

with only $\mathcal{O}(n^d \log n + M)$ arithmetic operations. Meanwhile there exists a rich literature on NFFTs, where the algorithms are described in detail and where the reader can find estimates of the approximation error versus the complexity of the algorithm, see e. g. [8, 4, 21] and the references therein. Moreover, free NFFT software packages are available, e. g. [17, 9].

In this paper, we further develop the ideas from [20]. We introduce new regularization techniques with $B$-splines and algebraic polynomials. Based on the approach with algebraic polynomials we prove error estimates for our approximative summation algorithm. These error estimates are more sophisticated than those for the regularization with trigonometric polynomials in [20]. The later still involve numerical computations and consequently are only valid for a bounded number of parameters. In [20] only kernels of the form

$$K_0(x) = \log|x|, \; K_\beta(x) = \frac{1}{|x|^\beta} \qquad (\beta \in \mathbb{N}) \tag{1.2}$$

were considered. In this paper we add estimates for the parameter-dependent generalized multiquadrics

$$K_{-1}(x;c) = (|x|^2 + c^2)^{\frac{1}{2}}, \; K_\beta(x;c) = (|x|^2 + c^2)^{-\frac{\beta}{2}} \qquad (\beta \in \mathbb{N};\ \text{odd}) \tag{1.3}$$

which play an important role in the approximation of functions by linear combinations of RBFs [10].

Our paper is organized as follows: the next section describes our summation algorithm in 1D. One essential step of this algorithm consists in an appropriate kernel regularization which we consider in detail in Section 3. Error estimates for our algorithm with regularization by algebraic polynomials and the consequences for the choice of the parameters of the algorithm are derived in Section 4. Section 5 briefly sketches the generalization of the algorithm to the multivariate setting. Finally, Section 6 contains numerical results, mainly in 2D.

## 2 Fast Summation at One-dimensional Knots

In this section, we recall the idea of the fast summation algorithm introduced in [20]. Our aim consists in the fast evaluation of sums

$$f(x) := \sum_{k=1}^{N} \alpha_k K(x - x_k) \qquad (x_k \in \mathbb{R}), \tag{2.1}$$

at $M$ knots $y_j \in \mathbb{R}$ $(j = 1, \dots, M)$ for kernels $K(x) = K(|x|)$, i. e., in 1D for even kernels. The kernel function $K$ is in general a non-periodic function, while the use of Fourier methods requires to replace $K$ by a periodic version. Without loss of generality we may assume that the knots are scaled, such that $|x_k|,\ |y_j| < \frac{1}{4} - \frac{\varepsilon_B}{2}$ and consequently $|y_j - x_k| < \frac{1}{2} - \varepsilon_B$. The parameter $\varepsilon_B > 0$, which we specify later, guarantees that $K$ has to be evaluated only at points in the interval $[-\frac{1}{2} + \varepsilon_B, \frac{1}{2} - \varepsilon_B]$. This simplifies the later consideration of a 1-periodic version of $K$. Beyond a special treatment of $K$ near the boundary $\pm\frac{1}{2}$, we have to take care about properties of $K$ in the neighborhood of the origin. The kernels (1.2) considered in [20] are $C^\infty$ except of the origin, where they have a singularity. The parameter-dependent kernels $K = K_\beta(x; c)$ in (1.3), or its derivatives in case $\beta = -1$, have a singularity at zero if $c \to 0$.

To deduce a fast summation algorithm for (2.1) we replace the kernel $K$ by a 1-periodic smooth kernel $\tilde{K}$ by modifying $K$ near the boundary and near the origin:

$$\tilde{K}(x) := \begin{cases} K_I(x) & \text{for } x \in [-\varepsilon_I, \varepsilon_I], \\ K_B(x) & \text{for } x \in [-\frac{1}{2}, -\frac{1}{2} + \varepsilon_B] \cup [\frac{1}{2} - \varepsilon_B, \frac{1}{2}], \\ K(x) & \text{else,} \end{cases} \tag{2.2}$$

where $0 < \varepsilon_I < \frac{1}{2} - \varepsilon_B < \frac{1}{2}$. The functions $K_I$ and $K_B$ will be chosen such that $\tilde{K}$ is in the Sobolev space $H^p(\mathbb{T})$ for an appropriate parameter $p > 0$ which controls the smoothness of $\tilde{K}$. Various regularizations $\tilde{K}$ of $K$ are proposed in Section 3. If $p$ is large enough, then we may assume that $\tilde{K}$ can be approximated with sufficiently small error by the trigonometric polynomial

$$\mathcal{T}_n(\tilde{K})(x) := \sum_{l \in I_n^1} \epsilon_l b_l \, \mathrm{e}^{2\pi i l x}, \tag{2.3}$$

where

$$b_l := \frac{1}{n} \sum_{j \in I_n^1} \epsilon_j \, \tilde{K}\left(\frac{j}{n}\right) \mathrm{e}^{-2\pi \mathrm{i} j l / n} \qquad (l \in I_n^1).$$

Now the original kernel $K$ can be decomposed as

$$K = \left(K - \tilde{K}\right) + \left(\tilde{K} - \mathcal{T}_n(\tilde{K})\right) + \mathcal{T}_n(\tilde{K}), \tag{2.4}$$

where the summand in the middle becomes small for a sufficiently large parameter $n \in \mathbb{N}$ which we will specify later. We neglect this summand in (2.1) and approximate $f$ by

$$\tilde{f}(x) := \sum_{k=1}^{N} \alpha_k (K - \tilde{K})(x - x_k) + \sum_{k=1}^{N} \alpha_k \mathcal{T}_n(\tilde{K})(x - x_k). \tag{2.5}$$

Instead of $f$ we evaluate $\tilde{f}$ at the knots $y_j$ $(j = 1, \ldots, M)$. Indeed this can be done in a fast way by the following two steps:

**1) Near field computation** (first sum in (2.5))
To achieve the desired complexity of our algorithm we suppose that either the $N$ points $x_k$ or the $M$ points $y_j$ are "sufficiently uniformly distributed", i. e., we suppose that there exists a small constant $\nu \in \mathbb{N}$ such that each subinterval of $[-\frac{1}{4}, \frac{1}{4}]$ of length $2\varepsilon_I$ contains at most $\nu$ of the points $x_k$ or of the points $y_j$, respectively. This implies that $\varepsilon_I$ depends linearly on $1/N$, respectively $1/M$. In the following we restrict our attention to the case

$$\varepsilon_I \approx \frac{\nu}{2N}. \tag{2.6}$$

Then, since $|y_j - x_k| < \frac{1}{2} - \varepsilon_B$ and $\operatorname{supp}(K - \tilde{K}) \cap [-\frac{1}{2} + \varepsilon_B, \frac{1}{2} - \varepsilon_B] = [-\varepsilon_I, \varepsilon_I]$, the evaluation of

$$\sum_{k=1}^{N} \alpha_k (K - \tilde{K})(y_j - x_k) \qquad (j = 1, \ldots, M)$$

requires $\leq \nu M$, i. e. $\mathcal{O}(M)$ arithmetic operations.

**2) NFFT based summation** (second sum in (2.5))

By (2.3), the evaluation of the second sum in (2.5) can be rewritten as

$$\begin{aligned}\sum_{k=1}^{N} \alpha_k \mathcal{T}(\tilde{K})(y_j - x_k) &= \sum_{k=1}^{N} \alpha_k \sum_{l \in I_n^1} \epsilon_l b_l \, \mathrm{e}^{2\pi \mathrm{i} l (y_j - x_k)} \\ &= \sum_{l \in I_n^1} \epsilon_l b_l \left( \sum_{k=1}^{N} \alpha_k \, \mathrm{e}^{-2\pi \mathrm{i} l x_k} \right) \mathrm{e}^{2\pi \mathrm{i} l y_j} .\end{aligned}$$

This expression can be handled based on the NFFT as follows:

1. The sums
$$a_l = \sum_{k=1}^{N} \alpha_k \, \mathrm{e}^{-2\pi \mathrm{i} l x_k} \qquad (l \in I_n^1)$$
can be obtained by an $\mathrm{NFFT}^{\mathrm{T}}(n)$.

2. Then we compute the products
$$d_l = b_l a_l \qquad (l \in I_n^1).$$

3. Finally we use the $\mathrm{NFFT}(n)$ to compute
$$\sum_{l \in I_n^1} \epsilon_l \, d_l \, \mathrm{e}^{2\pi \mathrm{i} l y_j} \qquad (j = 1, \ldots, M).$$

These three steps require $\mathcal{O}(M + N + n \log n)$ arithmetic operations.

In summary, our summation algorithm requires

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

arithmetic operations. The relation between $M, N$ and $n$ determined by the approximation error of the algorithm will be specified in Section 4.

Once the basic idea of the algorithm is clear, it remains to specify the regularization procedure and to give estimates of the approximation error introduced by omitting $\tilde{K} - \mathcal{T}_n(\tilde{K})$ in the kernel approximation.

## 3 Kernel Regularization

Since $K$ is even, we have that $K^{(j)}(x) = (-1)^j K^{(j)}(-x)$. To ensure that

$$\tilde{K}(x) := \begin{cases} K_I(x) & \text{for } x \in [-\varepsilon_I, \varepsilon_I], \\ K_B(x) & \text{for } x \in [-\frac{1}{2}, -\frac{1}{2} + \varepsilon_B] \cup [\frac{1}{2} - \varepsilon_B, \frac{1}{2}], \\ K(x) & \text{else,} \end{cases}$$

is in $H^p(\mathbb{T})$, we need that the function $K_I$ fulfills the conditions

$$\begin{aligned} K_I^{(j)}(\varepsilon_I) &= K^{(j)}(\varepsilon_I), \\ K_I^{(j)}(-\varepsilon_I) &= K^{(j)}(-\varepsilon_I) = (-1)^j K^{(j)}(\varepsilon_I) \end{aligned} \tag{3.1}$$

and the function $K_B$ the conditions

$$\begin{aligned} K_B^{(j)}\left(\frac{1}{2} - \varepsilon_B\right) &= K^{(j)}\left(\frac{1}{2} - \varepsilon_B\right), \\ K_B^{(j)}\left(\frac{1}{2} + \varepsilon_B\right) &= K^{(j)}\left(-\frac{1}{2} + \varepsilon_B\right) = (-1)^j K^{(j)}\left(\frac{1}{2} - \varepsilon_B\right) \end{aligned} \tag{3.2}$$

for all $j = 0, \ldots, p-1$. Then, the periodicity of $\tilde{K}$ follows by setting

$$K_B\left(-\frac{1}{2} + x\right) := K_B\left(\frac{1}{2} + x\right) \qquad (x \in [0, \varepsilon_B]).$$

As simple regularizing functions $K_I$ and $K_B$ we propose

- algebraic polynomials,
- trigonometric polynomials,
- splines.

The regularization by trigonometric polynomials was considered in [20]. However the error estimates in [20] are not satisfactory since they involve numerical computations which can be done only up to a fixed number $p \in \mathbb{N}$. In this paper we briefly sketch the spline approach and consider the regularization by algebraic polynomials in more detail.

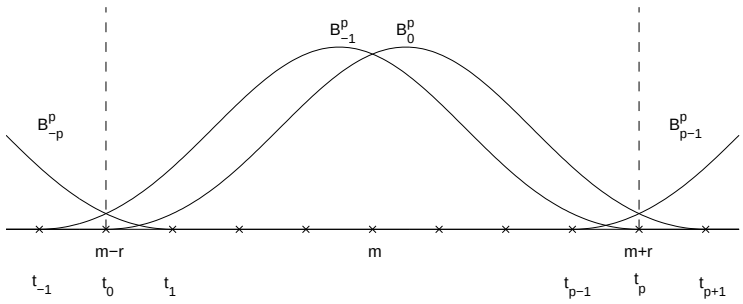


Figure 1: $B$-splines $B_k^p$.

### 3.1 Regularization by spline interpolation

The normalized cardinal $B$-splines $N_p$ of degree $p$ are recursively defined by

$$N_0(x) := \begin{cases} 1 & \text{for } x \in [0,1), \\ 0 & \text{otherwise} \end{cases}$$

and

$$N_p(x) := \frac{x}{k} N_{p-1}(x) + \frac{p+1-x}{k} N_{p-1}(x-1) \qquad (p \in \mathbb{N}).$$

Note that $\operatorname{supp} N_p = [0, p+1]$.

In our application we deal with intervals $[m-r, m+r]$ $(r > 0)$, more precisely with $[-\varepsilon_I, \varepsilon_I]$ and $[\frac{1}{2} - \varepsilon_B, \frac{1}{2} + \varepsilon_B]$. At the interval $[m-r, m+r]$ we choose the equispaced knots $\Delta := \{t_k = m - r + \frac{2r}{p} k : k = -p, \ldots, 2p\}$ and introduce the dilated and translated versions of $N_p$ with respect to these spline knots

$$B_k^p(x) := N_p\left(\frac{p(x-m+r)}{2r} - k\right),$$

see Figure 1.

The set of $B$-splines $\{B_k^p\}_{k=-p}^{p-1}$ forms a basis of the spline space

$$S_p(\Delta) := \{s \in C^{p-1}[m-r, m+r] : s|_{[t_k, t_{k+1}]} \in \Pi_p, k = 0, \ldots, p-1\}.$$

**Proposition 3.1 (Spline interpolation)** *For given $a_j$, $b_j$ $(j = 0, \ldots, p-1)$ there exists a unique spline $S \in S_p(\Delta)$ which satisfies the interpolation conditions*

$$S^{(j)}(m-r) = a_j, \qquad S^{(j)}(m+r) = b_j \qquad (j = 0, \ldots, p-1)$$

*at the endpoints of an interval* $[m-r, m+r]$ $(r > 0)$. *This spline can be written as*

$$S(x) = \sum_{k=-p}^{p-1} c_k B_k^p(x)$$

*where the coefficients* $c_k$ *are the solution of the two* $p \times p$ *linear systems*

$$\begin{aligned} \sum_{k=1}^{p} c_{-k} (B_{-k}^p)^{(j)}(m-r) &= a_j, \\ \sum_{k=1}^{p} c_{k-1} (B_{-k}^p)^{(j)}(m-r) &= (-1)^j b_j \qquad (j = 0, \ldots, p-1) \end{aligned}$$

*with the same coefficient matrix.*

The proposition is a direct consequence of [6, Theorem 1] and the fact that

$$(B_{-k}^p)^{(j)}(m-r) = (-1)^j (B_{k-1}^p)^{(j)}(m+r).$$

Since our kernels are even, we have by (3.1) and (3.2) for our application that $a_j = (-1)^j b_j$. Hence it remains to solve only one $p \times p$ system to obtain all coefficients $c_k$. Of course, for large $p \in \mathbb{N}$, this system is ill-conditioned. However, we will only need small values of $p$ in our algorithm, and, for $p \leq 16$, the corresponding systems can be solved without substantial errors.

Finally note that the fast evaluation of the spline $S(x)$ can be realized by the de Boor algorithm [7].

### 3.2 Regularization by polynomial interpolation

To construct polynomials $K_I$ and $K_B$ of degree $2p-1$ which fulfill the $2p$ Hermite interpolation conditions (3.1) and (3.2), respectively, we use the following two-point Taylor interpolation, see e. g. [1, Corollary 2.2.6]:

**Proposition 3.2 (Two-point Taylor interpolation)** *For given* $a_j$, $b_j$ $(j = 0, \ldots, p-1)$ *there exists a unique polynomial* $P$ *of degree* $2p-1$ *which satisfies the interpolation conditions*

$$P^{(j)}(m-r) = a_j, \qquad P^{(j)}(m+r) = b_j \qquad (j = 0, \ldots, p-1) \tag{3.3}$$

*at the endpoints of an interval* $[m-r, m+r]$ $(r>0)$. *This polynomial can be written as*

$$P(x) = \sum_{j=0}^{p-1} \sum_{k=0}^{p-1-j} \binom{p-1+k}{k} \left( \frac{(x-m+r)^j}{j!} \left(\frac{x-m-r}{-2r}\right)^p \left(\frac{x-m+r}{2r}\right)^k a_j + \frac{(x-m-r)^j}{j!} \left(\frac{x-m+r}{2r}\right)^p \left(\frac{x-m-r}{-2r}\right)^k b_j \right). \tag{3.4}$$

As in the spline case, the representation (3.4) can be further simplified if we have even kernels and (3.1), (3.2) in mind.

**Corollary 3.3** *For given* $a_j$ *and* $b_j = (-1)^j a_j$ $(j = 0, \dots, p-1)$ *the unique polynomial* $P$ *of degree* $2p-1$ *which satisfies* (3.3) *at the endpoints of an interval* $[m-r, m+r]$ $(r>0)$ *is given by*

$$P(x) = \frac{1}{2^p} \sum_{j=0}^{p-1} \gamma_j (1-y^2)^j \left( (1-y)^{p-j} + (1+y)^{p-j} \right), \tag{3.5}$$

*where* $y := \frac{x-m}{r}$ *and*

$$\gamma_j := \sum_{l=0}^{j} \binom{p-1+l}{l} \frac{r^{j-l}}{2^l (j-l)!}\, a_{j-l}.$$

**Proof.** By (3.4) we obtain for our special setting that

$$P(x) = \frac{1}{2^p} \sum_{j=0}^{p-1} \sum_{k=0}^{p-1-j} \binom{p-1+k}{k} \frac{r^j}{2^k} \frac{a_j}{j!} \left( (1+y)^{j+k} (1-y)^p + (1-y)^{j+k} (1+y)^p \right).$$

Now the change of the summation order results in the desired formula

$$P(x) = \frac{1}{2^p} \sum_{j=0}^{p-1} \sum_{l=0}^{j} \binom{p-1+l}{l} \frac{r^{j-l}}{2^l} \frac{a_{j-l}}{(j-l)!} \left( (1+y)^j (1-y)^p + (1-y)^j (1+y)^p \right). \square$$

In the next section we will estimate the approximation error introduced by our fast algorithm. For this purpose we will need an estimate for the $p$th derivative of $K_I$ and $K_B$, respectively.

**Theorem 3.4** *For $p \in \mathbb{N}$, the $p$th derivative of the polynomial $P$ in (3.5) can be estimated by*

$$\max_{x\in[m,m+r]} \left|P^{(p)}(x)\right| \le p! \left(\frac{3}{2}\right)^p r^{-p}\,\gamma,$$

*where*

$$\gamma := \sum_{l=0}^{p-2} \binom{p-1+l}{l} \frac{r^{p-1-l}}{2^l (p-1-l)!}\, \left|a_{p-1-l}\right|.$$

**Proof.** Since the two-point Taylor interpolation polynomial reproduces polynomials of degree at most $2p-1$, we obtain for the polynomial $\equiv 1$ by Corollary 3.3 that

$$\frac{1}{2^p} \sum_{j=0}^{p-1} \binom{p-1+j}{j} \frac{(1-y^2)^j}{2^j} \big((1-y)^{p-j} + (1+y)^{p-j}\big) = 1. \tag{3.6}$$

On the other hand, if we reorder the sum in (3.5) with respect to the coefficients $a_l$ $(l = 0, \ldots, p-1)$, then (3.6) is just the coefficient of $a_0$. Thus, $a_0$ does not appear in the $p$th derivative of any polynomial $P$ of the form (3.5).

Now, since $\frac{\mathrm{d}}{\mathrm{d}x} y = \frac{1}{r}$, the $p$th derivative of (3.5) can be written as

$$P^{(p)}(x) = \left(\frac{1}{2r}\right)^p \sum_{j=1}^{p-1} \tilde{\gamma}_j \frac{\mathrm{d}^p}{\mathrm{d}y^p} \Big[(1-y^2)^j \big((1-y)^{p-j} + (1+y)^{p-j}\big)\Big], \tag{3.7}$$

where

$$\tilde{\gamma}_j := \sum_{l=0}^{j-1} \binom{p-1+l}{l} \frac{r^{j-l}}{2^l (j-l)!}\, a_{j-l}.$$

We consider $Q_j(y) := \frac{\mathrm{d}^p}{\mathrm{d}y^p} \left[(1-y^2)^j\, 2R_j(y)\right]$ with

$$\begin{aligned} R_j(y) :=& \tfrac{1}{2} \big((1-y)^{p-j} + (1+y)^{p-j}\big) \\ =& 1 + \binom{p-j}{2} y^2 + \binom{p-j}{4} y^4 + \ldots \\ &+ \begin{cases} y^{p-j} & \text{for } p-j \text{ even}, \\ (p-j) y^{p-j-1} & \text{for } p-j \text{ odd}. \end{cases} \end{aligned}$$

Obviously $R_j(y)$ is an even polynomial in $y$ of degree at most $p-j$ with positive coefficients and therefore

$$R_j^{(l)}(y) \geq 0 \text{ for } y \geq 0 \quad \text{and} \quad \max_{y\in[0,1]} |R_j^{(l)}(y)| = R_j^{(l)}(1). \tag{3.8}$$

By applying the Leibniz rule we get

$$\begin{aligned} Q_j(y) &= 2\sum_{k=0}^{p} \binom{p}{k} \frac{\mathrm{d}^k}{\mathrm{d}y^k}\left[(1-y^2)^j\right] \frac{\mathrm{d}^{p-k}}{\mathrm{d}y^{p-k}}\left[R_j(y)\right] \\ &= 2\sum_{k=j}^{p} \binom{p}{k} \frac{\mathrm{d}^{k-j}}{\mathrm{d}y^{k-j}} \frac{\mathrm{d}^j}{\mathrm{d}y^j}\left[(1-y^2)^j\right] \frac{\mathrm{d}^{p-k}}{\mathrm{d}y^{p-k}}\left[R_j(y)\right] \end{aligned}$$

and further by the Rodrigues formula of the Legendre polynomials, i. e. $P_j(x) = (-1)^j \frac{1}{2^j j!} \frac{\mathrm{d}^j}{\mathrm{d}x^j}[(1-x^2)^j]$,

$$Q_j(y) = (-1)^j 2^{j+1} j! \sum_{k=j}^{p} \binom{p}{k} P_j^{(k-j)}(y) R_j^{(p-k)}(y).$$

We know that $\max_{y\in[0,1]} |P_j^{(k-j)}(y)| = P_j^{(k-j)}(1)$ (see, e. g. [18]). Consequently, we obtain together with (3.8) that

$$\max_{y\in[0,1]} |Q_j(y)| = 2^{j+1} j! \sum_{k=j}^{p} \binom{p}{k} P_j^{(k-j)}(1) R_j^{(p-k)}(1) = |Q_j(1)|. \tag{3.9}$$

On the other hand we conclude by the Leibniz rule that

$$\begin{aligned} Q_j(y) =& \frac{\mathrm{d}^p}{\mathrm{d}y^p}\Big[(1-y^2)^j\left[(1-y)^{p-j} + (1+y)^{p-j}\right]\Big] \\ =& \frac{\mathrm{d}^p}{\mathrm{d}y^p}\left[(1-y)^p(1+y)^j + (1-y)^j(1+y)^p\right] \\ =& p! \sum_{k=0}^{j} \binom{p}{k}\binom{j}{k}(-1)^k \Big((1-y)^k(1+y)^{j-k}(-1)^p \\ &+ (1+y)^k(1-y)^{j-k}\Big). \end{aligned}$$

Now $|Q_j(1)|$ can be easily estimated by

$$\begin{aligned}
|Q_j(1)| &= p! \left| \sum_{k=0}^{j} \binom{p}{k} \binom{j}{k} (-1)^k \left( \delta_{k,0} 2^{j-k} (-1)^p + 2^k \delta_{k,j} \right) \right| \\
&= p! \left| (-1)^p 2^j + \binom{p}{j} 2^j (-1)^j \right| \\
&= 2^j p! \left| (-1)^j \binom{p}{j} + (-1)^p \right| \\
&\leq 2^j p! \left( \binom{p}{j} + 1 \right).
\end{aligned}$$

Here $\delta_{k,j}$ denotes the Kronecker symbol. Combining this with (3.7) and (3.9), we obtain for $x \in [m, m+r]$ that

$$\begin{aligned}
|P^{(p)}(x)| &\leq \left(\frac{1}{2r}\right)^p \sum_{j=1}^{p-1} |\tilde{\gamma}_j| \, |Q_j(1)| \\
&\leq p! \left(\frac{1}{2r}\right)^p \left( \sum_{j=1}^{p-1} \binom{p}{j} 2^j + \sum_{j=1}^{p-1} 2^j \right) \max_{j=1,\ldots,p-1} |\tilde{\gamma}_j| \\
&= p! \left(\frac{1}{2r}\right)^p \left( (1+2)^p - 2^p + 2^p - 3 \right) \max_{j=1,\ldots,p-1} |\tilde{\gamma}_j| \\
&< p! \left(\frac{3}{2r}\right)^p \max_{j=1,\ldots,p-1} |\tilde{\gamma}_j| \, .
\end{aligned}$$

It remains to estimate $\max |\tilde{\gamma}_j|$. By definition of $\tilde{\gamma}_j$ it follows

$$\begin{aligned}
|\tilde{\gamma}_j| &= \left| \sum_{l=0}^{j-1} \binom{p-1+l}{l} \frac{r^{j-l}}{2^l (j-l)!} \, a_{j-l} \right| \\
&\leq \sum_{l=0}^{j-1} \binom{p-1+l}{l} \frac{r^{j-l}}{2^l (j-l)!} \, |a_{j-l}| \;=:\; s_j.
\end{aligned}$$

Now one can easily check that $s_j \leq s_{j+1}$ for $1 \leq j \leq p-2$. Thus, $\max_{j=1,\ldots,p-1} |\tilde{\gamma}_j| \leq s_{p-1} = \gamma$ and we are done. □

Now we apply Theorem 3.4 and Corollary 3.3 with respect to our special polynomials $K_I$ and $K_B$, i. e. we consider the intervals $[-\varepsilon_I, \varepsilon_I]$

and $[\frac{1}{2}-\varepsilon_B, \frac{1}{2}+\varepsilon_B]$ and set $a_j := K^{(j)}(-\varepsilon_I) = (-1)^j K^{(j)}(\varepsilon_I)$ and $a_j := K^{(j)}(\frac{1}{2}-\varepsilon_B)$, respectively. The result can be summarized as follows:

**Corollary 3.5** *The polynomials $K_I$ and $K_B$ which satisfy (3.1) and (3.2), respectively, are given by (3.5) with $y = \frac{x}{\varepsilon_I}$, $y = \frac{x-1/2}{\varepsilon_B}$ and $\gamma_j = \gamma_j^{I/B}$, respectively, where*

$$\begin{aligned}
\gamma_j^I &:= \sum_{l=0}^{j} \binom{p-1+l}{l} \frac{(-1)^{j-l}\varepsilon_I^{j-l}}{2^l (j-l)!} K^{(j-l)}(\varepsilon_I), \\
\gamma_j^B &:= \sum_{l=0}^{j} \binom{p-1+l}{l} \frac{(-1)^{j-l}\varepsilon_B^{j-l}}{2^l (j-l)!} K^{(j-l)}\left(-\frac{1}{2}+\varepsilon_B\right).
\end{aligned}$$

*The polynomials fulfill the estimates*

$$\max_{x\in[0,\varepsilon_I]} \left|K_I^{(p)}(x)\right| \leq p! \left(\frac{3}{2}\right)^p \varepsilon_I^{-p}\, \gamma^I, \tag{3.10}$$

$$\max_{x\in[\frac{1}{2}-\varepsilon_B,\frac{1}{2}]} \left|K_B^{(p)}(x)\right| \leq p! \left(\frac{3}{2}\right)^p \varepsilon_B^{-p}\, \gamma^B \tag{3.11}$$

*with*

$$\gamma^I := \sum_{l=0}^{p-2} \binom{p-1+l}{l} \frac{\varepsilon_I^{p-1-l}}{2^l (p-1-l)!} \left|K^{(p-1-l)}(\varepsilon_I)\right|, \tag{3.12}$$

$$\gamma^B := \sum_{l=0}^{p-2} \binom{p-1+l}{l} \frac{\varepsilon_B^{p-1-l}}{2^l (p-1-l)!} \left|K^{(p-1-l)}\left(\frac{1}{2}-\varepsilon_B\right)\right|. \tag{3.13}$$

# 4 Error Estimates

Beyond the well-known errors appearing in the NFFT computations which are discussed for example in [20], our algorithm introduces the errors $|f(y_j) - \tilde{f}(y_j)|$ $(j = 1, \ldots, M)$. By (2.4), (2.5) and (2.1), we ob-

tain for $|y| \leq \frac{1}{4} - \frac{\varepsilon_B}{2}$ that

$$\begin{aligned} \left| f(y) - \tilde{f}(y) \right| &= \left| \sum_{k=1}^{N} \alpha_k \left( \tilde{K}(y - x_k) - \mathcal{T}_n(\tilde{K})(y - x_k) \right) \right| \\ &\leq \sum_{k=1}^{N} |\alpha_k| \, \|K_{\mathrm{err}}\|_\infty, \end{aligned}$$

where

$$\|K_{\mathrm{err}}\|_\infty := \max_{|x| \leq \frac{1}{2}} |K_{\mathrm{err}}(x)|, \quad K_{\mathrm{err}}(x) := \tilde{K}(x) - \mathcal{T}_n(\tilde{K})(x). \tag{4.1}$$

**Lemma 4.1** *Let $K$ be an even kernel and let $\tilde{K} \in H^p(\mathbb{T})$ be defined by (2.2). Then, for $2 \leq p \ll n$, the following estimate holds true:*

$$\|K_{\mathrm{err}}\|_\infty \leq \frac{C}{(p-1)\pi^p n^{p-1}} \int_0^{\frac{1}{2}} |\tilde{K}^{(p)}(x)| \, \mathrm{d}x.$$

**Proof.** The proof follows by standard arguments. By Fourier expansion of $\tilde{K}$ and (2.3) we obtain for $x \in [-\frac{1}{2}, \frac{1}{2}]$ that

$$K_{\mathrm{err}}(x) = \sum_{k \in \mathbb{Z}} c_k(\tilde{K}) \, \mathrm{e}^{2\pi \mathrm{i} k x} - \sum_{l \in I_n^1} \epsilon_l b_l \, \mathrm{e}^{2\pi \mathrm{i} l x},$$

where the Fourier coefficients $c_k(\tilde{K})$ are defined in (A.1). Further, it follows by the aliasing formula (see Theorem Appendix A:.1) that

$$K_{\mathrm{err}}(x) = \sum_{k \in I_n^1} \epsilon_k \sum_{\substack{r \in \mathbb{Z} \\ r \neq 0}} c_{k+rn}(\tilde{K}) \, \mathrm{e}^{2\pi \mathrm{i} k x} (\mathrm{e}^{2\pi \mathrm{i} r n x} - 1).$$

Since $\tilde{K}$ is even, we can estimate

$$\|K_{\mathrm{err}}\|_\infty \leq 4 \sum_{k=\frac{n}{2}}^{\infty} \varepsilon_k \, |c_k(\tilde{K})|.$$

By construction we have that $\tilde{K} \in H^p(\mathbb{T})$ which implies that

$$c_k(\tilde{K}) = (2\pi \mathrm{i} k)^{-p} \, c_k(\tilde{K}^{(p)})$$

so that

$$\|K_{\mathrm{err}}\|_\infty \le 4 \left( \sum_{k=\frac{n}{2}}^{\infty} \varepsilon_k \, (2\pi k)^{-p} \right) \int_{-\frac{1}{2}}^{\frac{1}{2}} |\tilde{K}^{(p)}(x)| \, \mathrm{d}x.$$

For $p \ge 2$ the above sum can be estimated by an upper integral

$$\|K_{\mathrm{err}}\|_\infty \le \frac{2\left(1 + \frac{p-1}{n}\right)}{(p-1)\pi^p n^{p-1}} \int_{-\frac{1}{2}}^{\frac{1}{2}} |\tilde{K}^{(p)}(x)| \, \mathrm{d}x.$$

Since $p \ll n$, this implies the assertion with a constant $C \approx 4$. □

Now we obtain by the definition of $\tilde{K}$ that

$$\int_0^{\frac{1}{2}} |\tilde{K}^{(p)}(x)| \, \mathrm{d}x = \int_0^{\varepsilon_I} |K_I^{(p)}(x)| \, \mathrm{d}x + \int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} |K^{(p)}(x)| \, \mathrm{d}x + \int_{\frac{1}{2}-\varepsilon_B}^{\frac{1}{2}} |K_B^{(p)}(x)| \, \mathrm{d}x$$

and for the polynomials $K_I$ and $K_B$ in Corollary 3.5 by (3.10), (3.11)

$$\int_0^{\frac{1}{2}} |\tilde{K}^{(p)}(x)| \, \mathrm{d}x \le p! \left(\frac{3}{2}\right)^p \left( \varepsilon_I^{1-p} \gamma^I + \varepsilon_B^{1-p} \gamma^B \right) + \int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} |K^{(p)}(x)| \, \mathrm{d}x. \tag{4.2}$$

It remains to estimate $K^{(p)}$ and the values $\gamma^I$, $\gamma^B$ which depend on $K^{(j)}(\varepsilon_I)$ and $K^{(j)}(\frac{1}{2} - \varepsilon_B)$, respectively. Therefore we have to estimate the derivatives of $K$.

For the kernels (1.2) and $j \in \mathbb{N}$ we have

$$\left| K_\beta^{(j)}(x) \right| = \frac{(j+\beta-1)!}{(\beta-1)!} \, |x|^{-(j+\beta)} \qquad (x \ne 0; \beta \in \mathbb{N}_0), \tag{4.3}$$

where we set $(-1)! := 1$ in case $\beta = 0$.

**Theorem 4.2** *For $\beta \in \mathbb{N}_0$, let $K = K_\beta$ be defined by (1.2) and $\tilde{K}$ by (2.2) with $K_I$ and $K_B$ given by Corollary 3.5, where $\varepsilon_I \le \min\{\varepsilon_B, \frac{1}{2} - \varepsilon_B\}$. Then, for $2 \le p \ll n$, the error $\|K_{\mathrm{err}}\|_\infty$ in (4.1) can be estimated by*

$$\|K_{\mathrm{err}}\|_\infty \le C_\beta \, \frac{(p+\beta-2+\delta_{0,\beta})!}{\varepsilon_I^{p+\beta-1}} \, \frac{3^p}{\pi^p n^{p-1}} \tag{4.4}$$

*with a constant $C_\beta$ independent of $p, n$ and $\varepsilon_I$.*

**Proof.** We consider the summands in (4.2). By (4.3) we obtain that

$$\begin{aligned}\int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} |K^{(p)}(x)|\,\mathrm{d}x &= \frac{(p+\beta-1)!}{(\beta-1)!}\int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} |x|^{-(p+\beta)}\,\mathrm{d}x \\ &\le \frac{(p+\beta-2)!}{(\beta-1)!}\,\varepsilon_I^{-(p+\beta-1)}.\end{aligned}$$

Since $\varepsilon_I \le \min\{\varepsilon_B, \frac{1}{2}-\varepsilon_B\}$ it follows by (3.12), (3.13) and (4.3) that $\gamma^B \varepsilon_B^{1-p} \le \gamma^I \varepsilon_I^{1-p}$. Thus it remains to estimate $\gamma^I \varepsilon_I^{1-p}$. By (3.12) and (4.3) we get

$$\begin{aligned}\gamma^I \varepsilon_I^{1-p} &\le \frac{1}{\varepsilon_I^{p-1+\beta}} \sum_{l=0}^{p-2} \binom{p-1+l}{l} \frac{(p-2-l+\beta)!\,2^{-l}}{(\beta-1)!(p-1-l)!} \\ &\le \frac{1}{\varepsilon_I^{p-1+\beta}} \binom{p-2+\beta}{\beta-1} \sum_{l=0}^{p-1} \binom{p-1+l}{l} 2^{-l},\end{aligned}$$

where we set $\binom{n}{-1} := 1$ in case $\beta = 0$. Using $y = 0$ in (3.6) we see that the last sum equals $2^{p-1}$ so that

$$p! \left(\frac{3}{2}\right)^p \gamma^I \varepsilon_I^{1-p} \le \frac{p(p+\beta-2+\delta_{0,\beta})!\,3^p}{2(\beta-1)!}\,\varepsilon_I^{-(p+\beta-1)}.$$

Combining these estimates with (4.2) and Lemma 4.1 we obtain the assertion. □

Of course, for small $c$, the derivatives of the generalized multiquadrics $K_\beta(x;c)$ behave similar to those of $K_\beta(x)$. The following lemma estimates the derivatives of the generalized multiquadrics by taking $c$ into account.

**Lemma 4.3** *The derivatives of*

$$K(x) = K_\beta(x;c) := (x^2+c^2)^{-\frac{\beta}{2}} \qquad (\beta \in \mathbb{N};\ \textit{odd})$$

*can be estimated by*

$$\left|K_\beta^{(j)}(x;c)\right| \le \frac{\sqrt[4]{\pi(1+\frac{2c^2}{x^2})}\,(j+\beta-1)!\,\sqrt{2}^j}{\Gamma\left(\frac{\beta}{2}\right)(x^2+c^2)^{\frac{j+\beta}{2}}}.$$

**Proof.** We use the well-known formula [22]

$$K_\beta(x;c) = \frac{1}{c^\beta \Gamma\left(\frac{\beta}{2}\right)} \int_0^\infty e^{-t(x^2/c^2+1)} t^{(\beta-2)/2} \,\mathrm{d}t.$$

By differentiation we obtain

$$K_\beta^{(j)}(x;c) = \frac{1}{c^\beta \Gamma\left(\frac{\beta}{2}\right)} \int_0^\infty \frac{\mathrm{d}^j}{\mathrm{d}x^j} \left[e^{-tx^2/c^2}\right] e^{-t} t^{(\beta-2)/2} \,\mathrm{d}t.$$

Using the Rodrigues formula of Hermite polynomials, i. e. $H_j(x) = (-1)^j e^{x^2} \frac{\mathrm{d}^j}{\mathrm{d}x^j}\left[e^{-x^2}\right]$, we can rewrite this as

$$K_\beta^{(j)}(x;c) = \frac{1}{c^\beta \Gamma\left(\frac{\beta}{2}\right)} \int_0^\infty (-1)^j e^{-tx^2/c^2} H_j\left(x\tfrac{\sqrt{t}}{c}\right) \left(\tfrac{\sqrt{t}}{c}\right)^j e^{-t} t^{(\beta-2)/2} \,\mathrm{d}t.$$

Now we substitute $y = x\frac{\sqrt{t}}{c}$ and obtain

$$K_\beta^{(j)}(x;c) = \frac{2(-1)^j}{\Gamma\left(\frac{\beta}{2}\right) x^{j+\beta}} \int_0^\infty e^{-y^2} H_j(y) e^{-y^2c^2/x^2} y^{j+\beta-1} \,\mathrm{d}y.$$

Since the integrand is even, this is equal to

$$K_\beta^{(j)}(x;c) = \frac{(-1)^j}{\Gamma\left(\frac{\beta}{2}\right) x^{j+\beta}} \int_{-\infty}^\infty e^{-y^2} H_j(y) e^{-y^2c^2/x^2} y^{j+\beta-1} \,\mathrm{d}y.$$

By the Cauchy-Schwarz inequality we get

$$\begin{aligned} \left|K_\beta^{(j)}(x;c)\right| \quad \le \quad & \frac{1}{\Gamma\left(\frac{\beta}{2}\right) |x|^{j+\beta}} \left( \int_{-\infty}^\infty e^{-y^2} H_j^2(y) \,\mathrm{d}y \right)^{\frac{1}{2}} \\ & \left( \int_{-\infty}^\infty e^{-y^2(1+2c^2/x^2)} y^{2(j+\beta-1)} \,\mathrm{d}y \right)^{\frac{1}{2}}. \end{aligned}$$

By the normalization of the Hermite polynomials, i. e.

$$\int_{-\infty}^\infty e^{-x^2} H_j(x) H_m(x) \,\mathrm{d}y = \begin{cases} 0 & \text{for } j \neq m, \\ 2^j j! \sqrt{\pi} & \text{for } j = m, \end{cases}$$

the first integral is equal to $2^j j!\sqrt{\pi}$ . To evaluate the second integral we set $\alpha^2 := 1 + \frac{2c^2}{x^2}$ and use that

$$\int_{-\infty}^{\infty} e^{-\alpha^2 y^2} y^{2(j+\beta-1)} \,\mathrm{d}y = \frac{1}{\alpha^{2(j+\beta)-1}} \Gamma(j+\beta-\tfrac{1}{2}) \le \frac{1}{\alpha^{2(j+\beta)-1}} (j+\beta-1)! \,.$$

Combining these estimates we arrive at

$$\left| K_\beta^{(j)}(x;c) \right| \le \frac{\sqrt[4]{\pi(1+\frac{2c^2}{x^2})}\, \left( (j+\beta-1)!\, j!\, 2^j \right)^{1/2}}{\Gamma\left(\frac{\beta}{2}\right) (x^2+2c^2)^{\frac{j+\beta}{2}}} \,. \qquad \square$$

**Theorem 4.4** *For odd $\beta \in \mathbb{N} \cup \{-1\}$, let $K = K_\beta(\,\cdot\,;c)$ be defined by (1.3) and $\tilde{K}$ by (2.2) with $K_I$ and $K_B$ given by Corollary 3.5, where $\varepsilon_I \le \min\{\varepsilon_B, \frac{1}{2} - \varepsilon_B\}$. Further, let $0 < c \le \varepsilon_I$. Then the error $\|K_{\mathrm{err}}\|_\infty$ in (4.1) can be estimated by*

$$\|K_{\mathrm{err}}\|_\infty \le C_\beta \, \frac{(p+\beta-2+2\delta_{-1,\beta})!}{(\varepsilon_I^2+c^2)^{\frac{p+\beta-1}{2}}} \, \frac{(3\sqrt{2})^p}{\pi^p n^{p-1}}$$

*with a constant $C_\beta$ independent of $p, n$ and $\varepsilon_I$.*

**Proof.** The proof follows the same lines as the proof of Theorem 4.2.

First we obtain for $\beta \in \mathbb{N}$ by Lemma 4.3 and since $c^2 \le \varepsilon_I^2$ that

$$\begin{aligned} \int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} |K^{(p)}(x)| \,\mathrm{d}x &\le \frac{C\,(p+\beta-1)!\sqrt{2}^{\,p}}{\Gamma\left(\frac{\beta}{2}\right)} \int_{\varepsilon_I}^{\frac{1}{2}-\varepsilon_B} (x^2+c^2)^{-(p+\beta)/2} \,\mathrm{d}x \\ &\le \frac{C\,(p+\beta-2)!\sqrt{2}^{\,p+1}}{\Gamma\left(\frac{\beta}{2}\right)} \, (\varepsilon_I^2+c^2)^{-(p+\beta-1)/2}. \end{aligned}$$

Next we have for $\beta \in \mathbb{N}$ by (3.12) and Lemma 4.3 that

$$\begin{aligned} \gamma^I \varepsilon_I^{1-p} \le\ & \frac{C\,\sqrt{2}^{p-1}}{\Gamma\left(\frac{\beta}{2}\right)(\varepsilon_I^2+c^2)^{(p+\beta-1)/2}} \times \\ & \sum_{l=0}^{p-2} \binom{p-1+l}{l} \frac{(p-2-l+\beta)!}{(p-1-l)!} \left( \frac{\sqrt{\varepsilon_I^2+c^2}}{2\sqrt{2}\varepsilon_I} \right)^l \end{aligned} \tag{4.5}$$

and since $c^2 \leq \varepsilon_I^2$ further

$$\begin{aligned}\gamma^I \varepsilon_I^{1-p} &\leq C_\beta \frac{(p-2+\beta)!\ \sqrt{2}^{p-1}}{(p-1)!\ (\varepsilon_I^2+c^2)^{(p+\beta-1)/2}} \sum_{l=0}^{p-1} \binom{p-1+l}{l} 2^{-l} \\ &\leq C_\beta \frac{(p-2+\beta)!\ (2\sqrt{2})^{p-1}}{(p-1)!\ (\varepsilon_I^2+c^2)^{(p+\beta-1)/2}}\,.\end{aligned}$$

This results in

$$p!\left(\frac{3}{2}\right)^p \gamma^I \varepsilon_I^{1-p} \leq C_\beta \frac{p\,(p+\beta-2)!\ (3\sqrt{2})^p}{2\sqrt{2}}\,(\varepsilon_I^2+c^2)^{-(p+\beta-1)/2}\,.$$

Substituting of these estimates in (4.2) and applying Lemma 4.1 we obtain the assertion for $\beta \in \mathbb{N}$.

The case $\beta = -1$ follows similarly by using the fact that the Hardy multiquadric $K_{-1}(x;c) = (x^2+c^2)^{\frac{1}{2}}$ fulfills

$$K_{-1}^{(j)}(x;c) = c^2\, K_3^{(j-2)}(x;c) \qquad (j = 2,3,\ldots)\,. \qquad \square$$

Note that the right–hand side of (4.5) also converges under the weaker condition $c^2 < 7\varepsilon_I^2$ so that one can prove similar estimates with $d^p$, $d > 3\sqrt{2}$, instead of $(3\sqrt{2})^p$ assuming weaker conditions than $c^2 < \varepsilon_I^2$.

We will use the estimates in the Theorems 4.2 and 4.4 to specify the parameters $\varepsilon_I, p$ and $n$ of our algorithm. Since both cases can be handled in the same way, we restrict our attention to Theorem 4.2. Using the Stirling formula $p! \leq 1.1\,\sqrt{2\pi p}\,\left(\frac{p}{\mathrm{e}}\right)^p$ we can rewrite our error estimate as

$$\|K_{\mathrm{err}}\|_\infty \leq \tilde{C}_\beta\, \varepsilon_I^{-\beta} \left(\frac{3}{\mathrm{e}\,\pi}\,\frac{p-1}{\varepsilon_I\, n}\right)^{p-1} \frac{(p+\beta-2+\delta_{0,\beta})!\sqrt{2\pi(p-1)}}{(p-1)!}\,.$$

Thus, choosing $\varepsilon_I$ such that $\frac{3(p-1)}{\mathrm{e}\,\pi\,\varepsilon_I\, n} < 1$, our error decays exponentially in $p$. In our numerical examples we choose

$$\varepsilon_I = \frac{p}{n}. \tag{4.6}$$

While (4.6) steers the error, condition (2.6) on $\varepsilon_I$ is necessary to keep the near field computation linear in $M$. Now (4.6) and (2.6) together imply that

$$n \approx \frac{2Np}{\nu}. \tag{4.7}$$

If $M = N$, then the near field computation requires approximately

$$\nu N$$

and the NFFT computations

$$n \log n + \mathcal{O}(N) = \frac{2Np}{\nu} \log \left( \frac{2Np}{\nu} \right) + \mathcal{O}(N)$$

arithmetic operations. One should choose $\nu$ such that both operation counts are balanced. It seems that $\nu \approx 2\sqrt{p}$, respectively by (4.7),

$$n \approx \sqrt{p} N$$

is a good choice.

## 5 Fast Summation at Multidimensional Knots

In this section we briefly explain how to extend our one-dimensional scheme to higher dimensions $d \geq 2$ and rotation-invariant kernels $\mathcal{K}(x) = K(|x|)$. We focus on the fast computation of

$$f(y_j) := \sum_{k=1}^{N} \alpha_k \mathcal{K}(y_j - x_k) = \sum_{k=1}^{N} \alpha_k K(|y_j - x_k|) \qquad (x_k, y_j \in \mathbb{R}^d) \quad (5.1)$$

for $j = 1, \ldots, M$. Similar as in Section 3 we regularize $\mathcal{K}$ near 0 and near the boundary of $[-\frac{1}{2}, \frac{1}{2})^d$ to obtain a smooth periodic kernel $\tilde{\mathcal{K}}$:

$$\tilde{\mathcal{K}}(x) := \begin{cases} K_I(|x|) & \text{if } |x| \leq \varepsilon_I, \\ K_B(|x|) & \text{if } \frac{1}{2} - \varepsilon_B < |x| < \frac{1}{2}, \\ K_B\left(\frac{1}{2}\right) & \text{if } |x| \geq \frac{1}{2}, \\ K(|x|) & \text{otherwise.} \end{cases}$$

Here we choose $K_I$ as in Corollary 3.3. But instead of (3.2) we require that the polynomial $K_B$ fulfills the conditions

$$\begin{aligned} K_B^{(j)}\left(\frac{1}{2} - \varepsilon_B\right) &= K^{(j)}\left(\frac{1}{2} - \varepsilon_B\right) \quad (j = 0, \ldots, p-1), \\ K_B^{(j)}\left(\frac{1}{2}\right) &= \delta_{0,j} K\left(\frac{1}{2}\right), \qquad (j = 0, \ldots, p-1). \end{aligned} \quad (5.2)$$

The unique solution $K_B$ of (5.2) is given by Theorem 3.2, but now it does not have the symmetry of Corollary 3.3.

Then we approximate $\tilde{\mathcal{K}}$ by the Fourier series

$$\mathcal{T}_n(\tilde{\mathcal{K}})(x) := \sum_{l \in I_n^d} \epsilon_l b_l \, \mathrm{e}^{2\pi \mathrm{i} l x},$$

where

$$b_l := \frac{1}{n^d} \sum_{j \in I_n^d} \epsilon_j \tilde{\mathcal{K}}\left(\frac{j}{n}\right) \mathrm{e}^{-2\pi \mathrm{i} j l / n} \qquad (l \in I_n^d).$$

Now we can decompose the original kernel as

$$\mathcal{K} = (\mathcal{K} - \tilde{\mathcal{K}}) + \big(\tilde{\mathcal{K}} - \mathcal{T}_n(\tilde{\mathcal{K}})\big) + \mathcal{T}_n(\tilde{\mathcal{K}})$$

and, by neglecting the summand in the middle, we approximate $f$ by

$$\tilde{f}(x) := \sum_{k=1}^{N} \alpha_k (\mathcal{K} - \tilde{\mathcal{K}})(x - x_k) + \sum_{k=1}^{N} \alpha_k \mathcal{T}_n(\tilde{\mathcal{K}})(x - x_k). \tag{5.3}$$

Instead of $f$ we evaluate $\tilde{f}$ at the knots $y_j \in \mathbb{R}^d$ $(j = 1, \ldots, M)$ by the following two steps:

**1) Near field computation** (first sum in (5.3))

To achieve the desired complexity of our algorithm we suppose that either the $N$ points $x_k$ or the $M$ points $y_j$ are "sufficiently uniformly distributed" in the ball with radius $\frac{1}{2} - \varepsilon_B$, i. e., we suppose that there exists a small constant $\nu \in \mathbb{N}$ such that each ball with radius $\varepsilon_I$ contains at most $\nu$ of the points $x_k$ or of the points $y_j$, respectively. This implies that $\varepsilon_I$ depends linearly on $N^{-1/d}$, respectively $M^{-1/d}$. In the following we restrict our attention to the case

$$\varepsilon_I \approx \frac{1}{2} \left(\frac{\nu}{N}\right)^{1/d}. \tag{5.4}$$

Then, as in one dimension, the computation of the first sum requires only $\leq \nu M$ arithmetic operations.

**2) NFFT based summation** (second sum in (5.3))

The evaluation of the second sum in (5.3) is done exactly in the same way as in one dimension, but with $d$-dimensional NFFTs of size $n$ now,

which really involve a multidimensional setting. This computation part requires $\mathcal{O}(n^d \log n + N + M)$ arithmetic operations.

To obtain an exponential error decay in $p$, we have to choose again $\varepsilon_I \approx \frac{p}{n}$; see (4.6). On the other hand, we have to ensure (5.4) for an efficient near field computation. Thus,

$$n \approx 2p \left(\frac{N}{\nu}\right)^{1/d}.$$

To get a balanced arithmetic complexity of both parts of our algorithm one may choose $n \approx \sqrt{p}\, N^{1/d}$ if $N = M$.

## 6 Numerical Examples

Our algorithms were implemented in C using double precision arithmetic and tested on an AMD Athlon(tm) XP 1800+, 512MB RAM, SuSe-Linux 8.2.

Throughout our experiments we apply the NFFT/NFFT$^{\mathrm{T}}$ package [16] with Kaiser-Bessel functions and oversampling factor $\sigma = 2$.

For simplicity we have chosen $M = N$ in our summation algorithm and randomly distributed knots $y_j = x_j$ $(j = 1\ldots, N)$ in $\{x \mid |x| \leq \frac{7}{32}\}$, i. e. $\varepsilon_B = \frac{1}{16}$. The coefficients $\alpha_k$ were randomly distributed in $[0, 1]$. Moreover, we set $\varepsilon_I = \frac{p}{n}$.

We are interested in the error

$$E := \max_{j=1,\ldots,N} \frac{|f(x_j) - \tilde{f}(x_j)|}{|f(x_j)|}. \tag{6.1}$$

Figure 2 shows the behaviour of $E$ in 2D for various kernels in (1.2) and (1.3) with spline regularization (left) and regularization by algebraic polynomials (right). Here we have chosen $N = 512^2$ points, $n = \sqrt{N}$ and $c = 1/\sqrt{N}$ as parameter of the generalized multiquadrics. Further we use the truncation parameter $m = 8$ in the NFFT computations. First we observe that the error $E$ with spline regularization is slightly better than the error with regularization by algebraic polynomials. Further, the results confirm the exponential error decay with increasing $p$ proved in the Theorems 4.2 and 4.4. In the following we will always use regularization by polynomial interpolation.

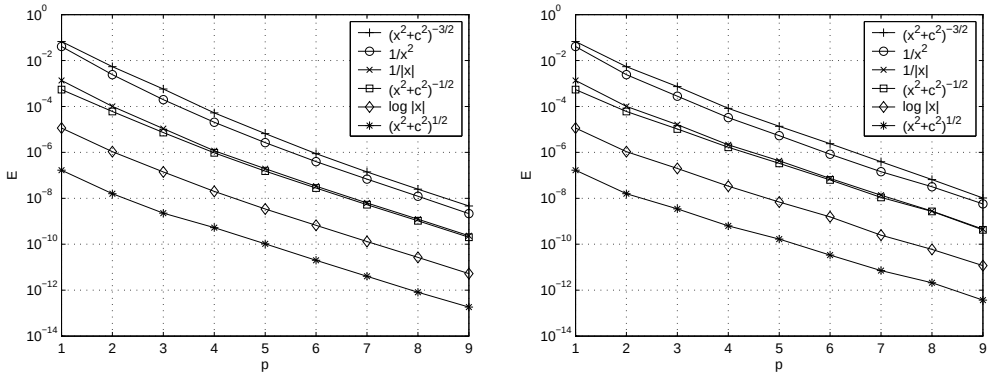


Figure 2: Error $E$ in dependence on $p$ for various kernels in 2D with $N = 512^2$, $n = 512$; regularization by spline interpolation (left) and by polynomial interpolation (right).

Figure 3 presents the 1D error $E$ in dependence on $p$ for the Hardy multiquadric (left) and the inverse Hardy multiquadric (right) with various scaling parameters $c$. Here we took $n = N = 1024$. Further we use the truncation parameter $m = 8$ in the NFFT computations. As expected, for decreasing $c$, the error increases until $c = \frac{1}{N}$, where it is approximately the same as for $c = 0$ in both cases. For $c = 1$, the error is about the same for both multiquadrics. In this case, we can also apply the algorithm without inner regularization, i. e. without near field computation. The corresponding curve is drawn with symbol $\triangle$. Note that without inner regularization $n$ does not depend on $N$ and the complexity of our algorithm becomes linear in $N$.

Figure 4 compares the computational time in dependence on the number $N$ of two-dimensional points for the direct computation of (5.1) and for our algorithm. As kernel function we have used $K(x) = \log|x|$. The parameters for our algorithm were $n = 2\sqrt{N}$ and $p = 4$ to achieve an accuracy of $E \leq 10^{-6}$. Further we use the truncation parameter $m = 4$ in the NFFT computations. Note that the computation time for the near field computation includes the time for the search of all points in the near field which requires $\mathcal{O}(\log N)$. The direct computation for $N = 2^{20}$ was only estimated based on the computational time and error for the first 1000 points, since the direct computation would take about 66 hours. Comparing this time with about 1.6 minutes required by our

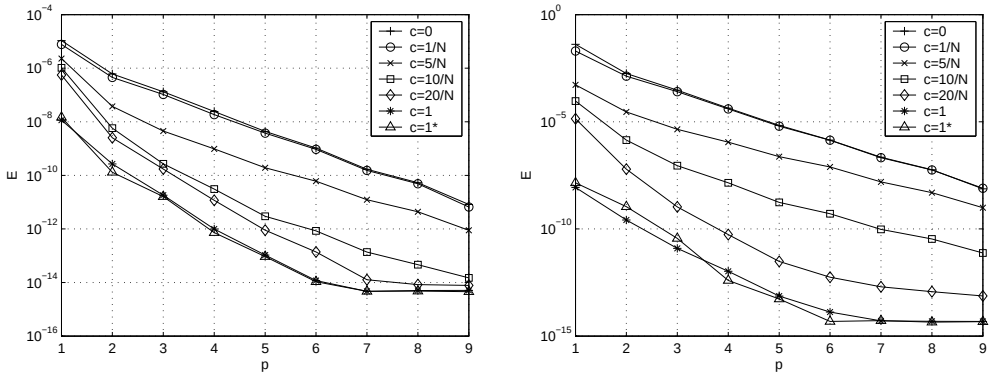


Figure 3: Error $E$ in dependence on $p$ for the Hardy multiquadric (left) and the inverse multiquadric (right) in 1D with various parameters $c$ and $n = N = 1024$. Here $c = 1^*$ denotes the algorithm without near field computation.

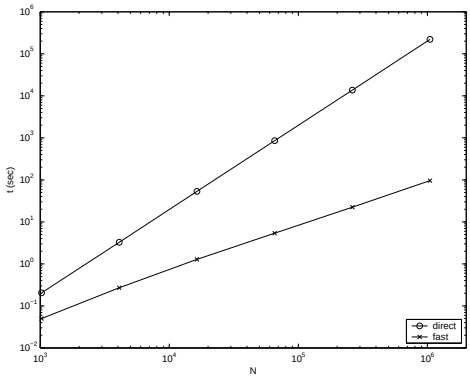


Figure 4: Computational time versus the number $N$ of points in 2D for the direct summation and our algorithm with $n = 2\sqrt{N}$ and $K(x) = \log|x|$.

algorithm, the time saving for large problem sizes $N$ becomes clear.

Finally, Table 1 compares the computational times required by our algorithm and by the algorithm proposed by Beatson et al. in [5]. In order to achieve an error $E \approx 10^{-6}$ in our algorithm, we have chosen $m = 4$ and $p = 3$. Further we have adapted the length $n \approx \sqrt{pN}$ of our NFFT such that the incorporated FFTs show a good performance.

| | | our algorithm | | Beatson et al. | |
|---|---|---|---|---|---|
| $N$ | $n$ | direct | fast | direct | fast |
| 2000 | 96 | $2.70 \times 10^{-1}$ | $6.0 \times 10^{-2}$ | $2.97 \times 10^{-1}$ | $7.8 \times 10^{-2}$ |
| 4000 | 144 | $1.02 \times 10^{+0}$ | $1.50 \times 10^{-1}$ | $1.19 \times 10^{+0}$ | $2.03 \times 10^{-1}$ |
| 8000 | 180 | $4.48 \times 10^{+0}$ | $3.10 \times 10^{-1}$ | $4.75 \times 10^{+0}$ | $4.84 \times 10^{-1}$ |
| 16000 | 216 | $2.32 \times 10^{+1}$ | $7.20 \times 10^{-1}$ | $2.50 \times 10^{+1}$ | $9.84 \times 10^{-1}$ |
| 32000 | 288 | $9.33 \times 10^{+1}$ | $1.83 \times 10^{+0}$ | $1.10 \times 10^{+2}$ | $2.23 \times 10^{+0}$ |

Table 1: Computational times (in seconds) of the algorithm of Beatson et al. in [5] and of our algorithm for $K(x) = \sqrt{x^2 + c^2}$ in $\mathbb{R}^2$.

As in [5] the multiquadric parameter was $c = \frac{1}{\sqrt{N}}$ and the coefficients were $\alpha_k = 1$ for all $k = 1, \ldots, N$. The computational times for the Beatson algorithm were taken fron Table 9.1 in [5]. Note that a different hardware was used for both algorithms so that the time for the direct computation may serve as a measure for comparison.

## References

[1] R. P. Agarwal and P. J. Y. Wong. *Error inequalities in polynomial interpolation and their applications*, volume 262 of *Mathematics and its Applications*. Kluwer Academic Publishers Group, Dordrecht, 1993.

[2] R. K. Beatson and W. A. Light. Fast evaluation of radial basis functions: methods for two-dimensional polyharmonic splines. *IMA J. Numer. Anal.*, 17:343–372, 1997.

[3] R. K. Beatson and G. N. Newsam. Fast evaluation of radial basis functions. I. *Comput. Math. Appl.*, 24:7–19, 1992.

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

[5] J. B. Cherrie, R. K. Beatson, and G. N. Newsam. Fast evaluation of radial basis functions: methods for generalized multiquadrics in $\mathbb{R}^n$. *SIAM J. Sci. Comput.*, 23:1549–1571, 2002.

[6] C. de Boor. Total positivity of the spline collocation matrix. *Indiana Univ. Math. J.*, 25(6):541–551, 1976.

[7] C. de Boor. *Splinefunktionen*. Lectures in Mathematics ETH Zürich. Birkhäuser Verlag, Basel, 1990. With a preface by Manfred R. Trummer.

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

[9] J. A. Fessler and B. P. Sutton. NUFFT – nonuniform FFT toolbox for Matlab.
`http://www.eecs.umich.edu/~bpsutton/MR/Code/NUFFT/`, 2002.

[10] R. Franke. Scattered data interpolation: tests of some methods. *Math. Comp.*, 38(157):181–200, 1982.

[11] L. Greengard. *The rapid evaluation of potential fields in particle systems*. MIT Press, Cambridge, MA, 1988.

[12] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325–348, 1987.

[13] W. Hackbusch. A sparse matrix arithmetic based on $\mathcal{H}$-matrices. I. Introduction to $\mathcal{H}$-matrices. *Computing*, 62:89–108, 1999.

[14] W. Hackbusch and B. N. Khoromskij. A sparse $\mathcal{H}$-matrix arithmetic. II. Application to multi-dimensional problems. *Computing*, 64:21–47, 2000.

[15] W. Hackbusch and Z. P. Nowak. On the fast matrix multiplication in the boundary element method by panel clustering. *Numer. Math.*, 54:463–491, 1989.

[16] S. Kunis and D. Potts. NFFT, software package, C subroutine library. `http://www.math.uni-luebeck.de/potts/nfft`, 2002.

[17] S. Kunis, D. Potts, and G. Steidl. Fast Fourier transform at nonequispaced knots - a user's guide to a C-library. Preprint, MU-Lübeck B-02-13, September 2002.

[18] C. Müller. Über die ganzen Lösungen der Wellengleichung. *Math. Annalen*, 124:235–264, 1952.

[19] A. Nieslony, D. Potts, and G. Steidl. Rapid evaluation of radial functions by fast Fourier transforms at nonequispaced knots. Preprint, MU-Lübeck A02-11, May 2002.

[20] D. Potts and G. Steidl. Fast summation at nonequispaced knots by NFFTs. *SIAM J. Sci. Comput.*, to appear.

[21] D. Potts, G. Steidl, and M. Tasche. Fast Fourier transforms for nonequispaced data: a tutorial. In *Modern sampling theory*, Appl. Numer. Harmon. Anal., pages 247–270. Birkhäuser Boston, Boston, MA, 2001.

[22] M. J. D. Powell. The theory of radial basis function approximation in 1990. In *Advances in numerical analysis, Vol. II (Lancaster, 1990)*, Oxford Sci. Publ., pages 105–210. Oxford Univ. Press, New York, 1992.

[23] E. E. Tyrtyshnikov. Mosaic-skeleton approximations. *Calcolo*, 33:47–57, 1996.

[24] E. E. Tyrtyshnikov. Mosaic ranks and skeletons. In L. G. Vulkov, J. Wasniewski, and P. Y. Yalamov, editors, *Numerical Analysis and its Applications (Rousse, 1996)*, volume 1196 of *Lecture Notes in Computer Science*, pages 505–516. Springer, Berlin, 1997.

## Appendix A: Aliasing Formula

**Theorem Appendix A:.1 (Aliasing formula)** *Let $g$ be a 1-periodic function with absolutely convergent Fourier series and Fourier coefficients*

$$c_k(g) := \int_{-\frac{1}{2}}^{\frac{1}{2}} g(x)\, \mathrm{e}^{-2\pi \mathrm{i} k x}\, \mathrm{d}x. \tag{A.1}$$

*For even $n \in \mathbb{N}$ and $\varepsilon_j$ given by (1.1) define an approximation*

$$\hat{g}_k := \frac{1}{n} \sum_{j=-\frac{n}{2}}^{\frac{n}{2}} \epsilon_j g\left(\frac{j}{n}\right) \mathrm{e}^{-2\pi \mathrm{i} j k/n}$$

*of $c_k(g)$ by using the trapezoidal quadrature rule. Then the following relation holds true:*

$$\hat{g}_k = c_k(g) + \sum_{\substack{r \in \mathbb{Z} \\ r \neq 0}} c_{k+rn}(g).$$