Approximation of scattered data by trigonometric polynomials on the torus and the 2–sphere

Daniel Potts *

Institute of Mathematics, University of Lübeck, D-23560 Lübeck, Germany.
email: potts@math.uni-luebeck.de

Abstract

Fast, efficient and reliable algorithms for the discrete least–square approximation of scattered points on the torus $\mathcal{T}^d$ and the sphere $\mathcal{S}^2$ by trigonometric polynomials are presented. The algorithms are based on iterative CG-type methods in combination with fast Fourier transforms for nonequispaced data. The emphasis is on numerical aspects, in order to solve large scale problems. Numerical examples show the efficiency of the new algorithms.

AMS Mathematics Subject Classification. 65D10, 65T50, 65F10

Key words and phrases. Trigonometric approximation, least square problems, torus, sphere, Vandermonde matrix, fast Fourier transforms for nonequispaced data, NFFT, FFT

1 Introduction

The paper discusses the approximation of data by d–variate periodic functions. In practical applications we are often confronted with the situation that experimental data or measured values of a function f are only known at prescribed points $\boldsymbol{x}_j$. At the end one wants to recover the original signal from its samples. But usually the functions possess some smoothness and decay properties and hence it is assumed that the given functions are band–limited. Because practical measured data are almost always noisy we prefer approximation methods rather than interpolation methods. In recent years nonuniform sampling sets appear in more and more applications. The aim of this paper is twofold. In the first part we assume that the data are lying on the torus $\mathcal{T}^d := \{\boldsymbol{x} = (x_1, \ldots, x_d)^{\mathrm{T}} \in \mathbb{R}^d; |x_j| \leq 1/2, j = 1, \ldots, d\}$. The discrete problem is formulated as a linear system of equations with a Vandermonde matrix. Our main tools are fast algorithms for the matrix times vector multiplication based on fast Fourier transforms for nonequispaced data (NFFT). We investigate the least–square approximation to scattered data by d–variate periodic trigonometric polynomials. In a special case our algorithm coincides with the well known ACT method (Adaptive weight, Conjugate gradient acceleration, Toeplitz matrices) [8] but avoids the normal equation, a very important fact in finite precision. However, the iterative approximation of band-limited functions was very successfully applied in a variety of applications (see e.g. [12]). In the second part, we generalize this method to the sphere $\mathcal{S}^2$, where $\mathcal{S}^2 := \{\boldsymbol{x} \in \mathbb{R}^3 : \|\boldsymbol{x}\|_2 = 1\}$. A straightforward generalization would suggest to expand a function into a Fourier series in terms of spherical

*Research supported in part by the EU Research Training Network MINGLE, RTN1-1999-00212.

harmonics. Working with spherical harmonics is in many respects computationally complex. There is a growing need for the fast summation of spherical harmonic expansions (see [3, 17, 14]). However the computational complexity of these algorithms is still higher than the complexity of the fast Fourier transform (FFT) or NFFT. Hence we generalize the methods described for the torus to the sphere $\mathcal{S}^2$ by blending two approximants together.

The paper is organized as follows: In Section 2 an efficient reconstruction algorithm for scattered data approximation on the torus by d–variate trigonometric functions is described. Section 3 generalizes these methods to the 2–sphere. Finally, Section 4 contains numerical examples.

2 Iterative approximation of scattered data on the torus by trigonometric polynomials

In this section we consider iterative solutions of large least squares problems. Our aim is to approximate scattered data on the torus $\mathcal{I}^d$ by a linear combination of trigonometric polynomials. The corresponding matrix times vector multiplications in the iterative process can be handled by fast Fourier transforms for nonequispaced data.

For a 1–periodic d–variate integrable function $f : \mathbb{R}^d \to \mathbb{R}$, we write

$$\mathcal{I}(f, K_N, \boldsymbol{v}) = \int_{\mathcal{T}^d} f(\boldsymbol{t}) K_N(\boldsymbol{v} - \boldsymbol{t}) \, \mathrm{d}\boldsymbol{t} \; = \; \int_{\mathcal{T}^d} f(\boldsymbol{v} - \boldsymbol{t}) K_N(\boldsymbol{t}) \, \mathrm{d}\boldsymbol{t} \, . \tag{2.1}$$

We are interested in kernels of the form

$$K_N(\boldsymbol{t}) := \sum_{\boldsymbol{n} \in I_N^d} c_{\boldsymbol{n}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{n} \boldsymbol{t}}$$

with $c_{\boldsymbol{n}} \in \mathbb{R}$ and $I_N^d := \{\boldsymbol{k} \in \mathbb{Z}^d : -\frac{N}{2} \le \boldsymbol{k} < \frac{N}{2}\}$ where the inequalities hold componentwise. Let $\chi_{[0,1)}$ denote the characteristic function of the interval $[0, 1)$. The *cardinal B–splines* N_m $(m \ge 1)$ *of order* m are defined by

$$N_1 := \chi_{[0,1)} \, , \; N_{m+1} := \int_0^1 N_1(t) \; N_m(\cdot - t) \; \mathrm{d}t$$

and their centered version by

$$\tilde{N}_m := N_m(\cdot + \frac{m}{2}) \, .$$

Note that $\tilde{N}_m$ is an even function with supp $\tilde{N}_m = [-\frac{m}{2}, \frac{m}{2}]$ and that

$$\int_{-\infty}^{\infty} \tilde{N}_m(t) \; \mathrm{e}^{-2\pi \mathrm{i} x t} \; \mathrm{d}t = (\operatorname{sinc}(\pi x))^m \, , \tag{2.2}$$

where

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

Let the *B–spline kernels* $B_{m,N}$ be defined by

$$B_{m,N}(x) := \frac{1}{\tilde{N}_m(0)} \sum_{k=-N+1}^{N-1} \tilde{N}_m \left(\frac{mk}{2N} \right) \mathrm{e}^{-2\pi \mathrm{i} k x}$$

Note that $B_{1,N}$ coincides with the Dirichlet kernel and $B_{2,N}$ with the Fejér kernel. It is important, that the Fourier coefficient $c_{N,k} = \tilde{N}_m(\frac{mk}{2N})/\tilde{N}_{2m}(0)$ can be computed in a simple way for example by applying a simplified version of de Boor's algorithm [1, p. 54].

It is a matter of fact, that there exists $f \in C(\mathcal{T}^1)$ for witch $\mathcal{I}(f, B_{1,N})$ does not converge uniformly to f, while the Fejér Weierstrass theorem asserts that $\mathcal{I}(f, B_{2,N})$ converges uniformly to f for every $f \in C(\mathcal{T}^1)$. From the approximation point of view not only Dirichlet kernels should be used [2]. In many applications the physical process that generates the signal provides further information about its nature. Thus in general a signal is not only essentially band–limited, but physics predicts the rate of decay of the spectrum of the signal [12]. Assume that the decay of the Fourier transform of f can be bounded, then the underlying approximation process should take care of this fact and special kernels should be used, e.g. B–spline kernels.

In order to get kernels in the multivariate case, we use the tensor product of univariate kernels

$$c_{\boldsymbol{n}} = \prod_{k=1}^{d} c_{N,n_k} \quad \boldsymbol{n} = (n_1, n_2, \ldots, n_d)^{\mathrm{T}} .$$

Given a uniform or nonuniform sampling set $\boldsymbol{v}_j$ ($j \in I_M^1$) the problem in many applications is to recover f from its samples $f(\boldsymbol{v}_j)$. Nonuniform sampling of signals leads very often to ill–conditioned systems. In general this occurs when the sampling set has large gaps, which is a common situation in some practical applications. The conjugate gradient type methods are probably the most efficient tool for solving large–scale discrete problems. In addition we use the iterative method as regularization method [13].

Let f be a function with uniformly convergent Fourier series such that

$$f(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in \mathbb{Z}^d} \hat{f}_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{v}} .$$

Then it follows that

$$\mathcal{I}(f, K_N, \boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_N^d} \hat{f}_{\boldsymbol{k}} c_{\boldsymbol{k}} \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{v}} . \tag{2.3}$$

Given a set of M arbitrary distinct nodes $\boldsymbol{v}_j$ ($j \in I_M^1$) with $\boldsymbol{v}_j \in \mathcal{T}^d$, and a function f whose values $f_j^1 = f(\boldsymbol{v}_j)$ at the nodes $\boldsymbol{v}_j$ are explicitly known. We are looking for a numerical method to compute $\hat{\boldsymbol{f}}_N^d$ such that $f(\boldsymbol{v}_j) \approx \mathcal{I}(f, K_N, \boldsymbol{v}_j)$ and

$$\boldsymbol{f}_M^1 = \boldsymbol{A}\boldsymbol{D}\hat{\boldsymbol{f}}_N^d \tag{2.4}$$

where the matrix $\boldsymbol{A}$ is given by

$$\boldsymbol{A} := \left(\mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{v}_j}\right)_{j \in I_M^1, \boldsymbol{k} \in I_N^d} \tag{2.5}$$

and where $\hat{\boldsymbol{f}}_N^d := (\hat{f}_{\boldsymbol{k}})_{\boldsymbol{k} \in I_N^d}$, $\boldsymbol{f}_M^1 := (\mathcal{I}(f, K_N, \boldsymbol{v}_j))_{j \in I_M^1}$ and $\boldsymbol{D} := \mathrm{diag}(c_{\boldsymbol{k}})_{\boldsymbol{k} \in I_N^d}$. Vandermonde systems are often extremely ill–conditioned because they are related to interpolation problems with a monomial basis in case of $d = 1$ and $M = N$. From a numerical point of view ill–conditioned systems behave similar as singular systems and additional information is need to obtain a satisfactory solution. This information is usually stated in terms of the "smoothness"

of the solution. To this end, a d-variate trigonometric polynomial $\mathcal{I}(f, K_N, \boldsymbol{v})$ of order N is sought that minimizes the discrete least-square error

$$\sqrt{\sum_{j \in I_M^1} |f_j - \mathcal{I}(f, K_N, \boldsymbol{v}_j)|^2 \; w_j^2}\,. \tag{2.6}$$

Here the $w_j > 0$ are user–defined weights, which can be chosen for instance to compensate for irregularities in the sampling geometry [8]. The reconstruction of f from the samples $f(\boldsymbol{v}_j)$ $(j \in I_M^1)$ amounts to the solution of the following standard least–squares problem: Minimize

$$\|\boldsymbol{W}\boldsymbol{A}\boldsymbol{D}\hat{\boldsymbol{f}}_N^d - \boldsymbol{W}\boldsymbol{f}_M^1\|_2\,, \tag{2.7}$$

where $\boldsymbol{W} := \mathrm{diag}(w_j)_{j \in I_M^1}$. It is well–known that the normal equation

$$\boldsymbol{D}^*\boldsymbol{A}^*\boldsymbol{W}^2\boldsymbol{A}\boldsymbol{D}\hat{\boldsymbol{f}}_N^d = \boldsymbol{D}^*\boldsymbol{A}^*\boldsymbol{W}^2\boldsymbol{f}_M^1 \tag{2.8}$$

is used to solve the least–squares problem (2.7) for over–determined systems, i.e. $M > N^d$.

We suggest to solve the equations (2.4) or

$$\boldsymbol{W}\tilde{\boldsymbol{f}}_M^1 = \boldsymbol{W}\boldsymbol{A}\boldsymbol{D}\hat{\boldsymbol{f}}_N^d$$

by a CG-type method applied to the normal equation, which can be viewed as least–squares projection method.

There are many ways, all mathematically equivalent, to implement the CG method for least–squares problems. In exact arithmetic they will all generate the same sequence of approximations, but in finite precision the achieved accuracy may differ substantially. It is important to notice that an implementation of the CG method for symmetric positive definite systems should *not* be applied directly to ill-conditioned normal equations (2.8) (see the discussion in [15, Sec. 7.1]). The system (2.8) and methods derived from it are often labeled with NR (N for "Normal" and R for "Residual"). Thus CGNR denotes the Conjugate Gradient method applied to (2.8) (see [21], CGNE in [13]). Algorithm 2.3 of [13, p. 17] reads with respect to our notation as follows.

Algorithm 2.1 *(**CGNR**)*
Input: $M, N \in \mathbb{N}$, $\boldsymbol{f}_M^1$, $\boldsymbol{W}$, $\boldsymbol{D}$
$\hat{\boldsymbol{f}}_N^{(0)}$ % *start vector*
$\boldsymbol{r}^{(0)} = \boldsymbol{W}\boldsymbol{f}_M^1 - \boldsymbol{W}\boldsymbol{A}\boldsymbol{D}\hat{\boldsymbol{f}}_N^{(0)}$
$\boldsymbol{q} = \boldsymbol{D}\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{r}^{(0)}$
$k = 0$
while (not stop) do
  $\alpha = \|\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{r}^{(k)}\|_2^2 / \|\boldsymbol{W}\boldsymbol{A}\boldsymbol{D}\boldsymbol{q}\|_2^2$
  $\hat{\boldsymbol{f}}_N^{(k+1)} = \hat{\boldsymbol{f}}_N^{(k)} + \alpha\boldsymbol{q}$
  $\boldsymbol{r}^{(k+1)} = \boldsymbol{r}^{(k)} - \alpha\boldsymbol{W}\boldsymbol{A}\boldsymbol{D}\boldsymbol{q}$
  $\beta = \|\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{r}^{(k+1)}\|_2^2 / \|\boldsymbol{D}\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{r}^{(k)}\|_2^2$
  $\boldsymbol{q} = \boldsymbol{D}\boldsymbol{A}^*\boldsymbol{W}\boldsymbol{r}^{(k+1)} + \beta\boldsymbol{q}$
  $k = k + 1$
end while
Output: Compute the CGNR solution $\hat{\boldsymbol{f}}_N^d = \hat{\boldsymbol{f}}_N^{(k)}$.

From the values $\hat{\boldsymbol{f}}_N^{(k)}$ we compute the Fourier coefficients of $\mathcal{I}(f, K_N, v)$ by $\boldsymbol{D}\hat{\boldsymbol{f}}_N^{(k)}$. Finally we are able to compute values on a regular grid by a d–variate FFT.

Straightforward matrix times vector multiplications with $\boldsymbol{A}$ and $\boldsymbol{A}^*$ requires $\mathcal{O}(N^d M)$ arithmetic operations, too much for the applications we have in mind. Only in case of equispaced nodes $\boldsymbol{l}/N$ ($\boldsymbol{l} \in I_N^d$), the above computation can be evaluated by the well–known *fast Fourier transform* (FFT) with only $\mathcal{O}(N^d \log N)$ arithmetic operations. In this case

$$\boldsymbol{A} = \boldsymbol{F} := (\mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{l}/N})_{\boldsymbol{l} \in I_N^d, \boldsymbol{k} \in I_N^d} \cdot$$

To speed up the matrix times vector multiplication with a general matrix $\boldsymbol{A}$ we use an approximative algorithm, which is known as NFFT. Fast and robust computation of the discrete Fourier transforms for nonequispaced data

$$f(\boldsymbol{v}_j) = \sum_{\boldsymbol{k} \in I_N^d} \hat{f}_{\boldsymbol{k}}\, \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{v}_j} \qquad (j \in I_M^1) \tag{2.9}$$

and

$$h(\boldsymbol{k}) := \sum_{j \in I_M^1} \hat{f}_j\, \mathrm{e}^{-2\pi \mathrm{i} \boldsymbol{k}\boldsymbol{v}_j} \qquad (\boldsymbol{k} \in I_N^d)\,, \tag{2.10}$$

are possible with the NFFT. For details we refer to the tutorial papers [18, 22]. In summary the NFFT, i.e. the fast computation of (2.9) or (2.10) requires only $\mathcal{O}(N^d \log N + (2m+1)^d M)$ arithmetic operations where m is a constant and depend only on the accuracy of the computation.

We denote by $\nu^d(N)$ the number of arithmetical operations for the d–variate FFT of length N. From Algorithm 2.1 and the NFFT follows immediately

Corollary 2.2 Each iteration step of CGNR requires $2\nu^d(2N) + \mathcal{O}(M + N)$ arithmetical operations. □

Fast and efficient algorithms for the solution of (2.7) for real valued function f were presented in the univariate case for Dirichlet kernels (i.e. $\boldsymbol{D} = \mathrm{diag}(1)_{\boldsymbol{k} \in I_N^1}$) in [5]. The algorithms are based on schemes for the solution of inverse unitary eigenproblems and require $\mathcal{O}(MN)$ arithmetical operations. In [20] an approach to compute the QR decomposition of $\boldsymbol{W}\boldsymbol{A}$ was presented in the univariate case for the Dirichlet kernel.

In [8] a new "second generation" reconstruction algorithm for irregular sampling was presented by Feichtinger, Gröchening and Strohmer. Their celebrated ACT method [8] is a combination of the adaptive weights method [7] and the method of conjugate gradients for the solution of positive definite linear systems. They have used the Dirichlet kernel and the observation that $\boldsymbol{T} := \boldsymbol{A}^*\boldsymbol{W}^2\boldsymbol{A}$ is a Toeplitz matrix (case $d = 1$, see also [4]). Moreover the condition number of $\boldsymbol{T}$ could be estimated in the 1D case and in some special 2D cases (see [12]). However there is no need to form the Toeplitz matrix explicitly. We stress again, that from the numerical point of view one should *avoid* the normal equation.

Iterative solution methods for Toeplitz systems, in particular the conjugate gradient method (CG–method), have attained much attention during the last years. The reason for this is that the essential computational effort per iteration step, namely the multiplication of a vector with $\boldsymbol{T} \in \mathbb{R}^{N,N}$, can be reduced to $2\nu^1(2N)$ arithmetical operations by embedding

the Toeplitz matrix in a circulant matrix. Note that the matrix–vector multiplication with $\boldsymbol{A}$ and $\boldsymbol{A}^*$ requires *also* $2\nu^1(2N)$ arithmetical operations (see [18]). The ACT method was generalized to the bivariate setting ($d = 2$), but again only by using the Dirichlet approach or heuristic methods [12, 19]. Note that the matrix $\boldsymbol{T} := \boldsymbol{A}^*\boldsymbol{W}^2\boldsymbol{A}$ is then a block–Toeplitz matrix with Toeplitz blocks. The weights w_j ($j \in I_M^1$) are chosen as the area (volume) of the Voronoi regions (see [12]). Numerical experiments show that for very ill–conditioned problems (large gaps) the choice of the weights is not so important.

Summarizing the results one should note, that each step of the CGNR method is as expensive as one step of the ACT method but has the advantage to avoid the normal equation. The matrix times vector multiplication is realized with the NFFT.

3 Iterative approximation of scattered data on the 2–sphere

Data fitting problems, where the underlying domain is the 2–sphere $\mathcal{S}^2$, arise in many areas including e.g. geophysics and meteorology. Over the last years several approaches were suggested to reconstruct a continuous function on the sphere from a finite number of discrete data (see [6, 11] and the references therein).

In this section we generalize the method described above to the sphere $\mathcal{S}^2$. Given M arbitrary points $\boldsymbol{x}_k \in \mathcal{S}^2$ ($k \in I_M^1$) and associated real values $f_k \in \mathbb{R}$ we address the problem of constructing a smooth function $F(\boldsymbol{x})$ on $\mathcal{S}^2$ which satisfies $F(\boldsymbol{x}_k) \approx f_k$ ($k \in I_M^1$). In order to avoid the problem at the poles we apply an idea from [10]: we map the sphere to two different rectangles, weight the given values, and blend the two parts together.

We work with two different rectangles, one of which leaves out regions surrounding the north ($\boldsymbol{x} = (0,0,1)^{\mathrm{T}}$) and south ($\boldsymbol{x} = (0,0,-1)^{\mathrm{T}}$) pole, and the other on leaves out regions surrounding the east ($\boldsymbol{x} = (0,1,0)^{\mathrm{T}}$) and west ($\boldsymbol{x} = (0,-1,0)^{\mathrm{T}}$) pole. Then the two fits can be blended together. For a discussion of some interpolation methods based on this idea, see [10]. In this section we develop a new blended approximation based on trigonometric polynomials. We construct $C^\infty(\mathcal{S}^2)$ functions on the sphere. Note that after mapping the sphere to a rectangle, we obtain a function which is periodic in one variable. Our blended functions take care about this fact. The standard latitude/longitude mapping from $[-\pi,\pi) \times [-\frac{\pi}{2},\frac{\pi}{2}]$ to the sphere $\mathcal{S}^2$ is given by

$$M_1(\phi,\theta) := (\cos(\phi)\cos(\theta), \sin(\phi)\cos(\theta), \sin(\theta))^{\mathrm{T}}.$$

The mapping M_1 is one–to–one if the domain is restricted to the interior of the rectangle and the image of this mapping is the sphere without the half circle $G_1 := \{M_1(-\pi,\theta) : -\pi/2 \leq \theta \leq \pi/2\}$. The inverse mapping of M_1 from $\boldsymbol{x} \in \mathcal{S}^2 \backslash G_1$ to the open rectangle is given by $\tilde{M}_1(\boldsymbol{x}) = (\phi,\theta)^{\mathrm{T}}$, where

$$\tilde{M}_1(\boldsymbol{x}) := \begin{cases} (\arccos\left(\frac{x_1}{\sqrt{1-x_3^2}}\right), \arcsin(x_3))^{\mathrm{T}} & \text{for } x_2 \geq 0\,, \\ (-\arccos\left(\frac{x_1}{\sqrt{1-x_3^2}}\right), \arcsin(x_3))^{\mathrm{T}} & \text{for } x_2 < 0\,. \end{cases}$$

To define $\tilde{M}_1$ over the entire sphere, we adopt the following convention for a point in G_1. If $\boldsymbol{x} \in G_1$, then $x_2 = 0, x_1 \leq 0$ and $x_1^2 = 1 - x_3^2$. If $\boldsymbol{x}$ is not the north or south pole, then $\tilde{M}_1(\boldsymbol{x}) = (-\pi, \arcsin(x_3))^{\mathrm{T}}$. For consistency, we define $\tilde{M}_1(0,0,1)^{\mathrm{T}} := (-\pi,\pi/2)^{\mathrm{T}}$ and $\tilde{M}_1(0,0,-1)^{\mathrm{T}} := (-\pi,-\pi/2)^{\mathrm{T}}$. In addition to the standard latitude/longitude mapping, we

also use a similar mapping based upon east pole and west pole. For $-\pi \le \phi < \pi$ and $-\pi/2 \le \theta \le \pi/2$ define the mapping to the sphere $\mathcal{S}^2$ by $M_2(\phi,\theta) = \boldsymbol{x} \in \mathcal{S}^2$, where

$$M_2(\phi,\theta) := (-\cos(\phi)\cos(\theta), \sin(\theta), \sin(\phi)\cos(\theta))^{\mathrm{T}}.$$

Restricted to the interior of the rectangle, M_2 is one–to–one mapping and the image of this mapping is the sphere without the half circle $G_2 := \{M_2(\pi,\theta) : -\pi/2 \le \theta \le \pi/2\}$. The inverse mapping of M_2 from $\boldsymbol{x} \in \mathcal{S}^2 \backslash G_2$ to the open rectangle is given by $\tilde{M}_2(\boldsymbol{x}) = (\phi,\theta)$, where

$$\tilde{M}_2(\boldsymbol{x}) := \begin{cases} (\arccos\left(\frac{x_1}{\sqrt{1-x_2^2}}\right), \arcsin(x_2))^{\mathrm{T}} & \text{for } x_3 \ge 0\,, \\ (-\arccos\left(\frac{x_1}{\sqrt{1-x_2^2}}\right), \arcsin(x_2))^{\mathrm{T}} & \text{for } x_3 < 0\,. \end{cases}$$

Following the convention used for $\tilde{M}_1$, we define $\tilde{M}_2$ at the east and west pole by $\tilde{M}_2(0,1,0)^{\mathrm{T}} := (-\pi,\pi/2)^{\mathrm{T}}$ and $\tilde{M}_2(0,-1,0)^{\mathrm{T}} := (-\pi,-\pi/2)^{\mathrm{T}}$.

The *map and blend approximation* to scattered data sampled on the sphere is the blend of two bivariate approximations f_l $(l = 1,2)$ defined over a planar domain given by

$$F(\boldsymbol{x}) = W_1(\boldsymbol{x}) f_1(\tilde{M}_1(\boldsymbol{x})) + W_2(\boldsymbol{x}) f_2(\tilde{M}_2(\boldsymbol{x})) \quad (\boldsymbol{x} \in \mathcal{S}^2)\,, \tag{3.1}$$

where the blending functions W_1 and W_2 satisfy certain continuity and periodicity conditions and

$$W_1(\boldsymbol{x}) + W_2(\boldsymbol{x}) = 1 \quad (\boldsymbol{x} \in \mathcal{S}^2)\,.$$

The functions $f_l(\boldsymbol{v})$ $(l = 1,2)$ are approximated by $\mathcal{I}(f_l, K_N, \boldsymbol{v})$. Now we are in the position to formulate the least–square problem on the sphere $\mathcal{S}^2$ more precisely. We are looking for two bivariate polynomials p and q of the form

$$p(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_N^2} \hat{p}_{\boldsymbol{k}} \mathrm{e}^{-\mathrm{i}(k_1 v_1 + 2\, k_2 v_2)} \tag{3.2}$$

and

$$q(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_N^2} \hat{q}_{\boldsymbol{k}} \mathrm{e}^{-\mathrm{i}(k_1 v_1 + 2\, k_2 v_2)} \tag{3.3}$$

that minimize the discrete least–square error

$$\sqrt{\sum_{j \in I_M^1} |f_j - W_1(\boldsymbol{x}_j)\, \mathcal{I}(f_1, K_N, \tilde{M}_1(\boldsymbol{x}_j)) - W_2(\boldsymbol{x}_j) \mathcal{I}(f_2, K_N, \tilde{M}_2(\boldsymbol{x}_j))|^2}\,. \tag{3.4}$$

In contrast to (2.6) we don't introduce further weights. Let $\hat{\boldsymbol{p}}_N^2$ and $\hat{\boldsymbol{q}}_N^2$ be the vectors of the unknown Fourier coefficients of p and q, respectively, and let $\boldsymbol{f}_M^1$ be the vector of the given values. Equation (3.4) can be rewritten as

$$\|\boldsymbol{W}_1 \boldsymbol{A}_1 \boldsymbol{D} \hat{\boldsymbol{p}}_N^2 + \boldsymbol{W}_2 \boldsymbol{A}_2 \boldsymbol{D} \hat{\boldsymbol{q}}_N^2 - \boldsymbol{f}_M^1\|_2\,, \tag{3.5}$$

where $\boldsymbol{W}_1 := \mathrm{diag}(W_1(\boldsymbol{x}_j))_{j \in I_M^1}$, $\boldsymbol{W}_2 := \mathrm{diag}(W_2(\boldsymbol{x}_j))_{j \in I_M^1}$, $\boldsymbol{D} := \mathrm{diag}(c_{\boldsymbol{k}})_{\boldsymbol{k} \in I_N^d}$, and

$$\boldsymbol{A}_l := \left(\mathrm{e}^{-\mathrm{i}(k_1 \phi_j + 2\, k_2 \theta_j)}\right)_{j \in I_M^1, \boldsymbol{k} \in I_N^2}\,, \quad (\phi_j, \theta_j)^{\mathrm{T}} := \tilde{M}_l(\boldsymbol{x}_j) \quad (l = 1,2)\,.$$

In order to solve (3.5), we apply the CGNR method to the equation

$$[\boldsymbol{W}_1\boldsymbol{A}_1\boldsymbol{D}, \boldsymbol{W}_2\boldsymbol{A}_2\boldsymbol{D}] \begin{bmatrix} \hat{\boldsymbol{p}}_N^2 \\ \hat{\boldsymbol{q}}_N^2 \end{bmatrix} = \boldsymbol{f}_M^1 . \tag{3.6}$$

We obtain the following CGNR algorithm. For the matrix times vector multiplications with the matrices $\boldsymbol{A}_1$, $\boldsymbol{A}_2$ and $\boldsymbol{A}_1^*$, $\boldsymbol{A}_2^*$ we apply the NFFT.

Algorithm 3.1 *(**CGNR** applied to (3.6))*
Input: $M, N \in \mathbb{N}$, $\boldsymbol{f}_M^1$, $\boldsymbol{W}$, $\boldsymbol{D}$
$\hat{\boldsymbol{p}}_N^{(0)}$, $\hat{\boldsymbol{q}}_N^{(0)}$ *% start vectors*
$\boldsymbol{r}^{(0)} = \boldsymbol{f}_M^1 - \boldsymbol{W}_1\boldsymbol{A}_1\boldsymbol{D}\hat{\boldsymbol{p}}_N^{(0)} - \boldsymbol{W}_2\boldsymbol{A}_2\boldsymbol{D}\hat{\boldsymbol{q}}_N^{(0)}$
$\boldsymbol{d}_1 = \boldsymbol{D}\boldsymbol{A}_1^*\boldsymbol{W}_1\boldsymbol{r}^{(0)}$
$\boldsymbol{d}_2 = \boldsymbol{D}\boldsymbol{A}_2^*\boldsymbol{W}_2\boldsymbol{r}^{(0)}$
$k = 0$
while (not stop) do
$\alpha = (\|\boldsymbol{D}\boldsymbol{A}_1^*\boldsymbol{W}_1\boldsymbol{r}^{(k)}\|_2^2 + \|\boldsymbol{D}\boldsymbol{A}_2^*\boldsymbol{W}_2\boldsymbol{r}^{(k)}\|_2^2)/\|\boldsymbol{W}_1\boldsymbol{A}_1\boldsymbol{D}\boldsymbol{d}_1 + \boldsymbol{W}_2\boldsymbol{A}_2\boldsymbol{D}\boldsymbol{d}_2\|_2^2$
$\hat{\boldsymbol{p}}_N^{(k+1)} = \hat{\boldsymbol{p}}_N^{(k)} + \alpha\boldsymbol{d}_1$
$\hat{\boldsymbol{q}}_N^{(k+1)} = \hat{\boldsymbol{q}}_N^{(k)} + \alpha\boldsymbol{d}_2$
$\boldsymbol{r}^{(k+1)} = \boldsymbol{r}^{(k)} - \alpha(\boldsymbol{W}_1\boldsymbol{A}_1\boldsymbol{D}\boldsymbol{d}_1 + \boldsymbol{W}_2\boldsymbol{A}_2\boldsymbol{D}\boldsymbol{d}_2)$
$\beta = (\|\boldsymbol{D}\boldsymbol{A}_1^*\boldsymbol{W}_1\boldsymbol{r}^{(k+1)}\|_2^2 + \|\boldsymbol{D}\boldsymbol{A}_2^*\boldsymbol{W}_2\boldsymbol{r}^{(k+1)}\|_2^2) \,/\, (\|\boldsymbol{D}\boldsymbol{A}_1^*\boldsymbol{W}_1\boldsymbol{r}^{(k)}\|_2^2 + \|\boldsymbol{D}\boldsymbol{A}_2^*\boldsymbol{W}_2\boldsymbol{r}^{(k)}\|_2^2)$
$\boldsymbol{d}_1 = \boldsymbol{D}_1\boldsymbol{A}_1^*\boldsymbol{W}_1\boldsymbol{r}^{(k+1)} + \beta\boldsymbol{d}_1$
$\boldsymbol{d}_2 = \boldsymbol{D}_2\boldsymbol{A}_2^*\boldsymbol{W}_2\boldsymbol{r}^{(k+1)} + \beta\boldsymbol{d}_2$
$k = k + 1$
end while
Output: Compute the CGNR solution $\hat{\boldsymbol{p}}_N^2 = \hat{\boldsymbol{p}}_N^{(k)}$ *and* $\hat{\boldsymbol{q}}_N^2 = \hat{\boldsymbol{q}}_N^{(k)}$.

From the values $\hat{\boldsymbol{p}}_N^{(k)}$ and $\hat{\boldsymbol{q}}_N^{(k)}$ we compute the Fourier coefficients of $\mathcal{I}(f_1, K_N, v)$ and $\mathcal{I}(f_2, K_N, v)$ defined in (3.2) and (3.3), by $\boldsymbol{D}\hat{\boldsymbol{p}}_N^{(k)}$ and $\boldsymbol{D}\hat{\boldsymbol{q}}_N^{(k)}$, respectively. Finally we compute approximating values of $F(\boldsymbol{x})$ by

$$F_N(\boldsymbol{x}) := W_1(\boldsymbol{x})\,\mathcal{I}(f_1, K_N, \tilde{M}_1(\boldsymbol{x})) + W_2(\boldsymbol{x})\,\mathcal{I}(f_2, K_N, \tilde{M}_2(\boldsymbol{x})) \quad (\boldsymbol{x} \in \mathcal{S}^2) \tag{3.7}$$

on an appropriate grid by applying the NFFT. We stress again that this reconstruction method is a very fast method, since each iteration step requires only $4\nu^2(2N) + \mathcal{O}(M+N)$ arithmetical operations, where $\nu^2(N)$ is the number of arithmetical operations of the bivariate FFT of length N.

Note that $\mathcal{I}(f_1, K_N, \boldsymbol{v})$ and $\mathcal{I}(f_2, K_N, \boldsymbol{v})$ are bivariate trigonometric polynomials and hence elements of C^∞ on the torus, but $p(\tilde{M}_1(\boldsymbol{x}))$ and $q(\tilde{M}_2(\boldsymbol{x}))$ are not necessary continuous on the sphere as $\boldsymbol{x}$ approaches the poles. In order to obtain $C^\infty(\mathcal{S}^2)$ functions we use the weight function

$$W(\phi,\theta)^{\mathrm{T}} := W(\theta) := \begin{cases} \mathrm{e}^{\frac{\pi^2}{4\theta^2-\pi^2}} & -\frac{\pi}{2} < \theta < \frac{\pi}{2}, \\ 0 & \theta = \pm\frac{\pi}{2} \end{cases}$$

and define the weight functions W_1 and W_2 by

$$W_l(\boldsymbol{x}) := \frac{W(\tilde{M}_l(\boldsymbol{x}))}{W(\tilde{M}_1(\boldsymbol{x})) + W(\tilde{M}_2(\boldsymbol{x}))} \quad (l = 1, 2).$$

Rewriting W_1 and W_2 with respect to $(\phi,\theta)^{\mathrm{T}} = \tilde{M}_1(\boldsymbol{x})$ we have $\boldsymbol{x} = M_1(\phi,\theta)$ and $(\tilde{\phi},\tilde{\theta})^{\mathrm{T}} = \tilde{M}_2(\boldsymbol{x}) = \tilde{M}_2(M_1(\phi,\theta))$, in particular $\tilde{\theta} = \arcsin(\sin(\phi)\cos(\theta))$. We conclude

$$\tilde{W}_1(\phi,\theta) := W_1(M_1(\phi,\theta)) = \frac{W(\theta)}{W(\theta) + W(\arcsin(\sin(\phi)\cos(\theta))},$$

$$\tilde{W}_2(\phi,\theta) := W_2(M_2(\phi,\theta)) = \frac{W(\arcsin(\sin(\phi)\cos(\theta))}{W(\theta) + W(\arcsin(\sin(\phi)\cos(\theta))}.$$

A plot of the functions $\tilde{W}_1(\phi,\theta)$ and $\tilde{W}_2(\phi,\theta)$ is shown in Figure 1. Clearly, $W_1(\boldsymbol{x})$ is zero

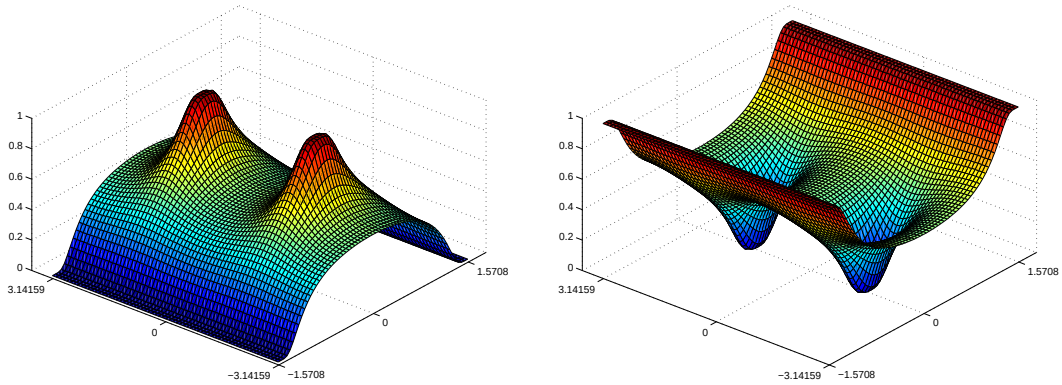

Figure 1: Weight function $\tilde{W}_1(\phi,\theta)$ (left) and $\tilde{W}_2(\phi,\theta)$ (right).

at the north and south pole, but one at the east and west pole. On the other hand $W_2(\boldsymbol{x})$ is zero at the east and west pole, but one at the north and east pole.

Lemma 3.2 *The functions $\tilde{W}_l$ $(l = 1, 2)$ have the following properties:*

i) $\tilde{W}_l$ are nonnegative functions,

ii) $\tilde{W}_1(\phi,\theta) + \tilde{W}_2(\phi,\theta) = 1$,

iii) $\tilde{W}_l(\phi,\theta)$ are 2π periodic with respect to the first variable ϕ,

iv) $\lim_{\theta\to-\pi/2+} \frac{d^n}{d\theta^n}\tilde{W}_1(\phi,\theta) = \lim_{\theta\to\pi/2-} \frac{d^n}{d\theta^n}\tilde{W}_1(\phi,\theta) = 0 \quad (n \in \mathbb{N}_0)$,

$$\lim_{\theta\to-\pi/2+} \tilde{W}_2(\phi,\theta) = \lim_{\theta\to\pi/2-} \tilde{W}_2(\phi,\theta) = 1\,,$$

$$\lim_{\theta\to-\pi/2+} \frac{d^n}{d\theta^n}\tilde{W}_2(\phi,\theta) = \lim_{\theta\to\pi/2-} \frac{d^n}{d\theta^n}\tilde{W}_2(\phi,\theta) = 0 \quad (n \in \mathbb{N}).$$

Proof: By definition of $\tilde{W}_l$ the properties i), ii), and iii) follow immediately. Furthermore we have

$$\lim_{\theta\to-\pi/2+} \frac{\mathrm{d}^n}{\mathrm{d}\theta^n} W(\theta) = \lim_{\theta\to\pi/2-} \frac{\mathrm{d}^n}{\mathrm{d}\theta^n} W(\theta) = 0 \quad (n \in \mathbb{N}_0),$$

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\arcsin(\sin(\phi)\cos(\theta)) = \frac{-\sin(\phi)\sin(\theta)}{\sqrt{1-(\sin(\phi))^2(\cos(\theta))^2}},$$

which implies iv). ∎

By Lemma 3.2 we have that $W_1(\boldsymbol{x})\,\mathcal{I}(f_1, K_N, (\tilde{M}_1(\boldsymbol{x})) \in C^\infty(\mathcal{S}^2)$ and similar $W_2(\boldsymbol{x})\,\mathcal{I}(f_2, K_N, (\tilde{M}_2(\boldsymbol{x})) \in C^\infty(\mathcal{S}^2)$. There are many other possibilities to choose the weight

functions W_1 and W_2. Foley has used $C^2(\mathcal{S}^2)$ piecewise biquintic patches in order to define the weight functions [10]. Then he has solved two scattered data interpolation problems by using the multiquadric method. Other weight functions constructed from $W(\theta) = 1 + \cos(2\theta)$ are possible and lead to functions in $C^2(\mathcal{S}^2)$.

Remark 3.3 Instead of using bivariate trigonometric polynomials 3.2 one can use the tensor product of trigonometric polynomials and Chebyshev polynomials

$$p(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_N^2} \hat{p}_{\boldsymbol{k}} \mathrm{e}^{-\mathrm{i}k_1 v_1} \, T_{k_2}(\cos(v_2)) \, .$$

Than one can apply the Discrete Cosine Transform on arbitrary nodes (see [16]), instead of the NFFT. □

4 Numerical Examples

We present numerical experiments to demonstrate the efficiency of our algorithms. All computations were done in C and tested on a INDIGO workstation with double precision arithmetic.

We start with a simple example on the torus $\mathcal{T}^1$. The knots x_j are randomly chosen in $[-1/2, -1/8] \cup [1/8, 1/2)$. We want to reconstruct the function $\tilde{N}_3$ and add Gaussian white noise. For simulation purposes, the blurred signal–to–noise ratio, is set to 5dB. We than apply Algorithm 2.1 with $N = 32$ and $M = 64$ and stop the iteration with the L–curve method. Figure 2 (left) shows the result after 35 iterations with the Dirichlet kernels $\mathcal{I}(f, B_{1,32})$ and Figure 2 (right) shows the result after 25 iterations with the B–spline kernels $\mathcal{I}(f, B_{3,32})$.

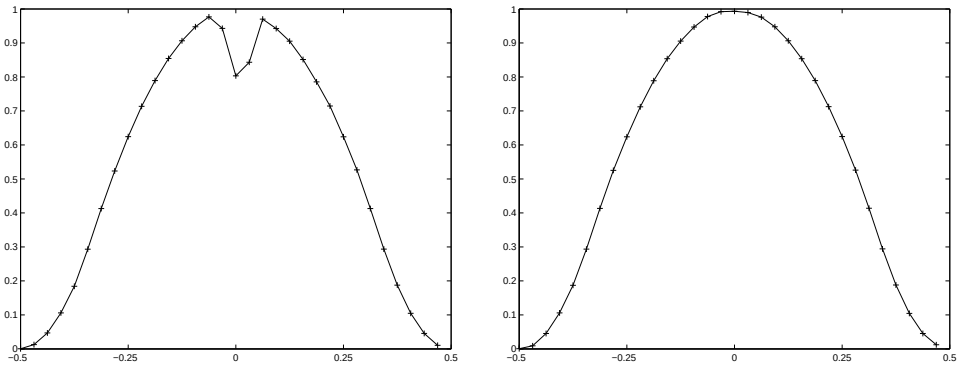

Figure 2: Iterative reconstruction on the torus with $B_{1,32}$ (left) and $B_{3,32}$ (right).

In our second example we take 10000 points shown in Figure 3 (left). (I wish to thank M. Floater for kind permission to use his data set, see [9]). The least squares method of Section 2 is then applied with $M = 10000$, $N = 128$ and $\boldsymbol{D}$ is given by the B–spline kernel $B_{3,128}$. The resulting surface is shown in Figure 3 (right).

Note that we solved an under–determined system, since $M \leq N^d$. In this case one should solve the dual least-square problem. Methods derived from it are often labeled with NE (N for "Normal" and E for "Error"). Thus CGNE denotes the Conjugate Gradient method applied to solve the system (2.4) if $M \leq N^d$ (see [21], CGME in [13], Craig's Method). The two methods (CGNR and CGNE) find approximations from the same affine Krylov subspace which achieve different optimality properties: Minimal residual for CGNR and minimal error

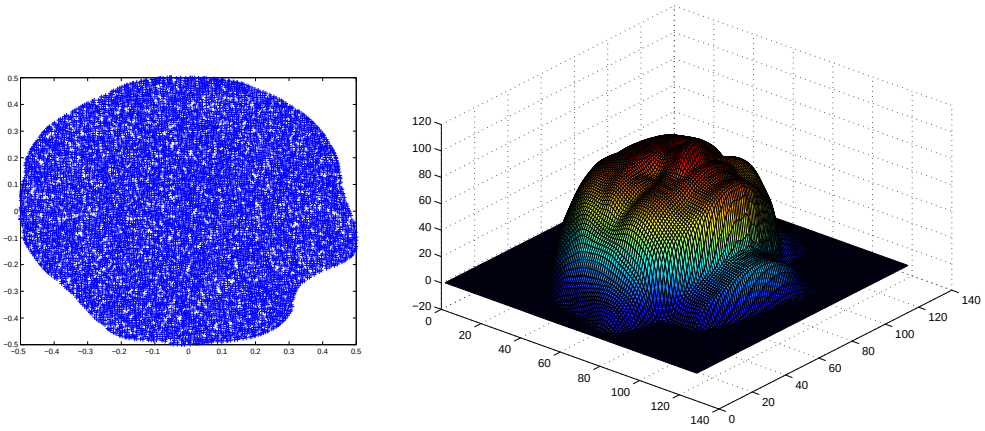

Figure 3: points on the torus $\mathcal{I}^2$ (left) and resulting surface (right).

for CGNE. We have observed that in our cases the CGNR gives slightly better results after fewer iterations. Hence we only used CGNR. We remark further that for equispaced data the matrix $\boldsymbol{A}$ is orthogonal and we obtain the solution after one CGNR step.

In our next example we take data points $\boldsymbol{a}_j = (a_1^j, a_2^j, a_3^j)^{\mathrm{T}} \in R^3$ $(j \in I^1_{9508})$ sampled from the head of Spock (see the Mingle Homepage http://www.cs.technion.ac.il/~vitus/mingle/ for the data) and solve the least–square problem (3.6) by Algorithm 3.1. We stop the iteration after 10 steps. First we define the points on the sphere as $\boldsymbol{x}_j := \frac{1}{r_j}\boldsymbol{a}_j$, where $r_j := \sqrt{a_1^2 + a_2^2 + a_3^2}$ and the values at the points $\boldsymbol{x}_j$ are given by r_j. Now we are looking for a smooth function $F(\boldsymbol{x})$ on the sphere such that $F(\boldsymbol{x}_j) \approx r_j$ $(j \in I^1_{9508})$. We show the 9508 given points in Figure 4 (left) as points on the sphere. Note that there are no points at the north and south pole. For the representation of the head we use the parametric plot of MATLAB. To this end, we compute values of the function $F_N(\boldsymbol{x})$ at the grid $M_1(j_1\pi/128, j_2\pi/256)$ $(j_1 = -128, \ldots, 128, j_2 = -128, \ldots, 128)$ by applying the NFFT. We remark that the small oscillations on the top of the head arise because no points are given there. Further, we have observed that trigonometric polynomials based on the Dirichlet kernels show much more oscillations (see also Figure 2).

In our second example we use the "earthmap" data from MATLAB. The data show the Earth's topography. We approximate the 64800 given data

$$f_{180j_1+j_2} = f(M_1(j_1\pi/180, (2j_2+1)\pi/360) \quad (j_1 = -180, \ldots, 179, j_2 = -90, \ldots, 89)$$

by functions of the form

$$F_N(\boldsymbol{x}) = W_1(\boldsymbol{x})\,\mathcal{I}(f_1, K_N, \tilde{M}_1(\boldsymbol{x})) + W_2(\boldsymbol{x})\,\mathcal{I}(f_2, K_N, \tilde{M}_2(\boldsymbol{x}))\,.$$

The polynomial degree is chosen as $N = 256$. Then we compute the function F_N on the grid $M_1(j_1\pi/128, j_2\pi/256)$ $(j_1 = -128, \ldots, 128, j_2 = -128, \ldots, 128)$ and plot the function in MATLAB by using the colored parametric plot, where the color represents the value of the function. Figure 6 (left) presents the result after 20 iterations, using the tensor-product of the B–spline kernel $B_{3,128}$.

One further advantage of representing functions on the sphere by using trigonometric polynomials is that we can simply compute the high frequency part on the sphere. More

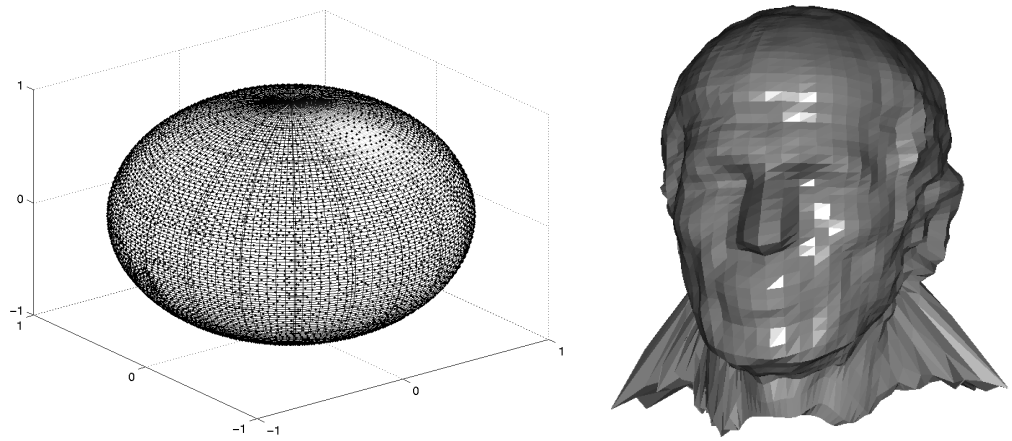

Figure 4: Given points on the sphere (left), reconstructed head for $N = 64$ using $B_{3,64}$ (right).

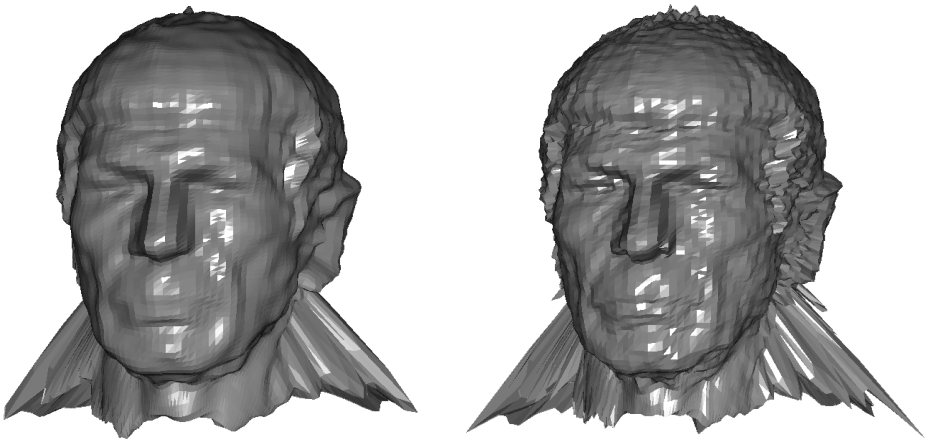

Figure 5: Reconstructed head for $N = 80$ using $B_{3,80}$ (left) and for $N = 128$ using $B_{3,128}$ (right).

precisely, we compute the high frequency part of the reconstructed function

$$F_w(\boldsymbol{x}) := W_1(\boldsymbol{x}) p_w(\tilde{M}_1(\boldsymbol{x})) + W_2(\boldsymbol{x}) q_w(\tilde{M}_2(\boldsymbol{x})),$$

where the trigonometric polynomials contain only the high frequency part

$$p_w(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_{256}^2 \setminus I_{64}^2} \hat{p}_{\boldsymbol{k}} c_{\boldsymbol{k}} \mathrm{e}^{-\mathrm{i}(k_1 v_1 + 2\, k_2 v_2)}$$

and

$$q_w(\boldsymbol{v}) = \sum_{\boldsymbol{k} \in I_{256}^2 \setminus I_{64}^2} \hat{q}_{\boldsymbol{k}} c_{\boldsymbol{k}} \mathrm{e}^{-\mathrm{i}(k_1 v_1 + 2\, k_2 v_2)}\,.$$

The function $|F_w(\boldsymbol{x})|$ on the grid $M_1(j_1\pi/128, j_2\pi/256)$ ($j_1 = -128, \ldots, 128, j_2 = -128, \ldots, 128$) is shown in Figure 6 (left).

Acknowledgment. I would like to thank the referees for their valuable suggestions.

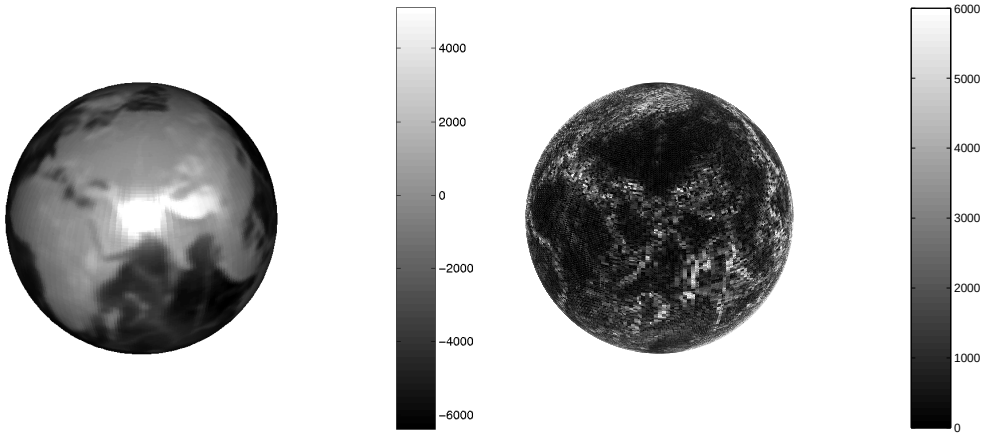

Figure 6: Reconstructed topography data for $N = 256$ using $B_{3,256}$ (left), high frequency part of the topography data for $N = 256$ using $B_{3,256}$ (right).

References

[1] C. de Boor. *Splinefunktionen.* Birkhäuser, Basel, 1990.

[2] R. DeVore and G. Lorentz. *Constructive Approximation.* Springer–Verlag, Berlin, 1993.

[3] J. Driscoll and D. Healy. Computing Fourier transforms and convolutions on the 2–sphere. *Adv. Appl. Math.*, 15:202 – 240, 1994.

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

[5] H. Faßbender. On numerical methods for discrete least-squares approximation by trigonometric polynomials. *Math. Comp.*, 66:719 – 741, 1997.

[6] G. E. Fasshauer and L. L. Schumaker. Scattered data fitting on the sphere. In M. Dahlen, T. Lyche, and L. L. Schumaker, editors, *Mathematical Methods for Curves and Surfaces II*, pages 117 – 166, Nashville, 1998. Vanderbilt University Press.

[7] H. Feichtinger and K. Gröchenig. Theory and practice of irregular sampling. In J. Benedetto and M. Frazier, editors, *Wavelets: Mathematics and Applications*, pages 305 – 363. CRC Press, 1993.

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

[9] M. S. Floater. How to approximate scattered data by least squares. Technical report, Sintef Applied Mathematics, 1998.

[10] T. Foley. The map and blend scattered data interpolation on a sphere. *Computers Math. Applic.*, 24:49 – 60, 1992.

[11] W. Freeden, T. Gervens, and M. Schreiner. *Constructive Approximation on the Sphere.* Oxford University Press, 1998.

[12] K. Gröchenig and T. Strohmer. Numerical and theoretical aspects of non-uniform sampling of band-limited images. In F. Marvasti, editor, *Theory and Practice of Nonuniform Sampling*. Kluwer/Plenum, 2001.

[13] M. Hanke. *Conjugate gradient type method for ill–posed problems.* Wiley, New York, 1995.

[14] M. J. Mohlenkamp. A fast transform for spherical harmonics. *Journal of Fourier Analysis and Applications*, 5:159 – 184, 1999.

[15] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8:43 – 71, 1982.

[16] D. Potts. Fast algorithms for discrete polynomial transforms on arbitrary grids. *Preprint,MU-Lübeck, A–01–16*, 2001. submitted.

[17] D. Potts, G. Steidl, and M. Tasche. Fast and stable algorithms for discrete spherical Fourier transforms. *Linear Algebra Appl.*, 275:433 – 450, 1998.

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

[19] M. Rauth and T. Strohmer. Smooth approximation of potential fields from noisy scattered data. *GEOPHYSICS*, 63:85 – 94, 1998.

[20] L. Reichel, G. Ammar, and W. Gragg. Discrete least squares approximation by trigonometric polynomials. *Math. Comp.*, 57:273 – 289, 1991.

[21] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS Publ., Boston, 1996.

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