ACOUSTICS 2017 PERTH
SOUND, SCIENCE AND SOCIETY

Fast Spherical Filtering in the Broadband FMBEM using a non-equally spaced FFT

Daniel R. Wilkes (1) and Alec. J. Duncan (1)

(1) Centre for Marine Science and Technology, Department of Imaging and Applied Physics, Curtin University, Perth, Australia

ABSTRACT

The Fast Multipole Boundary Element Method (FMBEM) reduces the O(N^2) computational and memory complexities of the conventional BEM to O(Nlog$N$) and O(N) respectively, where N is the number of boundary unknowns. One critical feature of the fast multipole method in the high-frequency regime is the choice of the spherical filtering algorithm used to interpolate and/or filter the multipole expansions between octree levels. In this paper a fast spherical filtering algorithm based on the Non-equally spaced Fast Fourier Transform (NFFT) is presented. The filtering algorithm is implemented in MATLAB via a combination of sparse matrix-vector products and standard FFTs, and is shown to achieve the theoretical algorithmic complexity for this type of spherical filter. Furthermore, the cross-over point at which the present NFFT filter becomes faster than the direct method is shown to be up to a factor of 2 smaller compared to that of other fast filtering algorithms in the published literature. The NFFT algorithm has been incorporated into a broadband Helmholtz FMBEM model. Large-scale acoustic scattering problems are presented to demonstrate the method.

1 INTRODUCTION

The FMBEM reduces both the computational cost and memory requirements of the conventional Boundary Element Method (BEM) by employing multipole expansions to treat the interactions between well-separated groups of boundary elements. This alleviates the need to explicitly construct and store the full BEM coefficient matrices which represent the individual pair-wise interactions of unknowns on the discretized boundary surface. The FMBEM thus reduces the O(N^2) computational and memory complexities of the Helmholtz BEM for acoustic analysis problems to O(Nlog$N$) and O(N) respectively (Gumerov and Duraiswami 2004): a significant improvement for large-scale problems.

The FMBEM algorithms are typically classified as either 'low-frequency' or 'high-frequency' algorithms (Yasuda, et al. 2010, Cheng, et al. 2006), depending upon the type of multipole expansion utilised in the algorithm. The low-frequency FMBEMs use the spherical basis functions for the multipole expansions of the Helmholtz fundamental solution, while the multipole translations are applied using either the RCR (Gumerov and Duraiswami 2003) or plane-wave expansion (Greengard, Huang, et al. 1998) translation algorithms. In either case, the low-frequency translation algorithms have a computational cost proportional to O(p^3) for translating a truncated multipole expansion with p^2 coefficients, leading to a FMBEM algorithm whose computational cost scales as O($N^{1.5}$) (Gumerov and Duraiswami 2004). Conversely, in the high-frequency regime the FMBEM algorithms are based on the far-field signature functions and diagonal translation methods (Rokhlin 1993) which provide an O(p^2) translation method. Both the low and high frequency FMBEM algorithms cannot in the first instance be applied in the opposite domain, due to either numerical instabilities and/or the computational cost of the translation algorithms when applied outside of the optimal frequency range. This has led to the development of various 'broadband' FMBEM algorithms which incorporate both low and high-frequency multipole expansions/translations and automatically switch between these representations based on the problem frequency (Gumerov and Duraiswami 2009, Cheng, et al. 2006). Alternatively, modifications can be made to either the low or high frequency FMBEMs to make them applicable at all frequencies (Darve and Have 2004, Chaillat and Collino 2015).

One issue for the high-frequency FMBEMs are the requisite interpolation and filtering operations required to change the truncation number (i.e. the number of coefficients in the truncated multipole expansions) between octree levels. The direct implementation of these spherical filtering algorithms results in an O(p^3) method, and so the formal algorithmic complexity of the high-frequency FMBEM becomes O($N^{1.5}$) (Gumerov and Duraiswami 2004). To achieve the O(Nlog$N$) scaling, so-called 'fast' spherical filtering algorithms must be employed to further reduce the O(p^3) cost of the direct filtering method. Different algorithms based on the 1D Fast Multipole Method (FMM) (Jakob-Chien and Alpert 1997), NFFT (Bohme and Potts 2003), fast Legendre transform (Press, et al. 2007) and Cauchy matrix (Gumerov and Duraiswami 2004) have been proposed to accelerate the spherical filter; in all cases leading to an O($p^2 \log p$) filtering method.

In the present work, a fast spherical filter based on the NFFT (Bohme and Potts 2003) is implemented using fast Gaussian gridding (Greengard and Lee 2004) and the performance is compared to the direct spherical filtering method. The algorithmic complexity of the present implementation is shown to agree with the theoretical estimates, while the memory complexity for the NFFT spherical filter algorithm appears to be reported for the first time in the published literature. The observed 'cross over' point where the NFFT filter becomes faster than the direct method, while slower in the present implementation compared to some other models (Bohme and Potts 2003), is shown to be up to a factor of 2 smaller compared to the cross over point reported for the fast algorithm based on the Legendre transform (Press, et al. 2007). Results from a broadband Helmholtz FMBEM utilising the fast spherical filter are presented to demonstrate the NFFT filtering algorithm for large-scale and high-frequency acoustic scattering problems.

The paper is organised as follows: Section 2 briefly presents the theory for the low and high-frequency multipole expansions of the Helmholtz Green's function used in the broadband FMBEM. The direct and NFFT fast spherical filtering algorithms are then presented in Section 3. Section 4 presents the algorithmic and memory complexities of each filtering algorithm and demonstrates the application of the NFFT spherical filter to a Broadband Helmholtz FMBEM for acoustic scattering problems. Finally, Section 5 concludes the paper.

2 THE BROADBAND HELMHOLTZ FMBEM

2.1 The Low-Frequency Fast Multipole Method

The Fast Multipole Method (FMM) for the acoustic BEM relies on the decomposition of the Helmholtz Green's function to allow for the localised grouping and corresponding interactions between well separated groups of boundary elements in the discretised problem. The Helmholtz Green's function

$$G(\boldsymbol{x},\boldsymbol{y}) = \frac{e^{\mathrm{i}kr}}{4\pi r}, \qquad r = |\boldsymbol{x}-\boldsymbol{y}|, \tag{1}$$

has been shown to admit the following decomposition using the S and R type spherical basis functions in the low-frequency regime of the FMM (Gumerov and Duraiswami 2004):

$$G(\boldsymbol{x},\boldsymbol{y}) \approx \mathrm{i}k \sum_{n=0}^{p} \sum_{m=-n}^{n} R_n^{-m}(\boldsymbol{y}-\boldsymbol{c})\, S_n^m(\boldsymbol{x}-\boldsymbol{c}). \tag{2}$$

In the above equations, $\boldsymbol{x}$ and $\boldsymbol{y}$ are well separated points on the boundary surface, $\boldsymbol{c}$ is the expansion centre for the multipole expansions, k is the acoustic wavenumber of the problem and i is the complex number. The spherical basis functions are defined in terms of the spherical Bessel, Hankel and harmonic functions; see (Gumerov and Duraiswami 2004) for details. In the present context, the pertinent information is that the expansions are truncated after a certain number of terms p corresponding to the maximum degree n in the truncated expansion. Choosing p such that $\max n = p-1$ yields p^2 coefficients in the truncated spherical basis function.

2.2 Conversion Between the Low and High Frequency Fast Multipole Methods

In the broadband FMBEM, the low-frequency spherical basis functions are converted to the high-frequency far field signature functions above a particular octree level, corresponding to when a certain number of acoustic wavelengths spans an octree box on that level of the octree structure. The spherical basis functions ($F = R, S$) are converted to the discretised far field signature functions, $f(\theta_j, \varphi_k)$, as follows (Cheng, et al. 2006):

$$f(\theta_j,\varphi_k) = \sum_{n=0}^{p} \sum_{m=-n}^{n} F_n^m Y_n^m(\theta_j,\varphi_k), \tag{3}$$

where Y_n^m is the spherical harmonic function

$$Y_n^m(\theta_j,\varphi_k) = \bar{P}_n^{|m|}(\cos\theta_j) e^{\mathrm{i}m\varphi_k}, \quad \bar{P}_n^m(z) = (-1)^m \sqrt{\frac{2n+1}{4\pi}\frac{(n-|m|)!}{(n+|m|)!}} P_n^m(z), \tag{4}$$

P_n^m is the associated Legendre function, ! denotes a factorial, and (θ_j, φ_k) is the set of sampling points for the far field signature function, defined in terms of the polar (θ) and azimuthal (φ) directions on the unit sphere. Selecting $\theta_j = \cos^{-1} t_j$ where t_j are the nodes ($j = 1{:}p$) for the p-point Gauss-Legendre quadrature rule, and φ_k as equally spaced points over the $0{:}2\pi$ azimuth angular range ($k = 1{:}2p$), provides a $[p \times 2p]$ grid of points which (with the corresponding quadrature weights) can exactly integrate products of the spherical harmonics up to a

summed degree of $2p$ (Cheng, et al. 2006). Conversely, the discretised far field signature functions can be converted to the truncated spherical basis function $F_{\hat{n}}^{\hat{m}}$ with coefficients $\hat{n} = 0{:}\hat{p}$, $\hat{m}$ =-$\hat{n}{:}\hat{n}$ as follows:

$$F_{\hat{n}}^{\hat{m}} = \frac{2\pi}{\hat{p}} \sum_{\hat{j}=1}^{\hat{p}} w_{\hat{j}} \sum_{\hat{k}=1}^{2\hat{p}} f\left(\theta_{\hat{j}}, \varphi_{\hat{k}}\right) Y_{\hat{n}}^{-\hat{m}}(\theta_{\hat{j}}, \varphi_{\hat{k}}), \tag{5}$$

where $w_{\hat{j}}$ are the Gauss-Legendre weights for the $\hat{p}$-point quadrature rule ($\hat{j} = 1{:}\hat{p}$). Equations (3) and (5) thus provide a mechanism to convert a truncated spherical basis function with p^2 (or $\hat{p}^2$) coefficients into a discretised far field signature function sampled at $2p^2$ (or $2\hat{p}^2$) coefficients, or vice versa.

3 THE DIRECT AND NFFT SPHERICAL FILTERS

3.1 The Direct Spherical Filter

Spherical filtering operations are required in the high-frequency FMBEM when interpolating or filtering a discretised far-field signature function sampled on one grid of spherical angular points $\left(\theta_{\hat{j}}, \varphi_{\hat{k}}\right)$ specified by the truncation number $\hat{p}$, to a different set of points $\left(\theta_j, \varphi_k\right)$ specified by the truncation number p, where $\hat{p} \neq p$. The direct spherical filter algorithm applies this resampling by converting from the far field signature functions to the spherical basis functions (for the same truncation number $\hat{p}$) via Eq. (5), followed by resampling these spherical basis function at the new grid of finer or coarser far field signature functions (specified by the new truncation number p) via Eq. (3). The direct spherical filter is applied in the following steps to reduce the overall computational cost and leverage the use of the FFT. By first substituting for $Y_{\hat{n}}^{-\hat{m}}(\theta_{\hat{j}}, \varphi_{\hat{k}})$, Eq. (5) is applied in two steps:

$$\beta_{\hat{j}}^{\hat{m}} = \frac{2\pi}{\hat{p}} \sum_{\hat{k}=1}^{2\hat{p}} f\left(\theta_{\hat{j}}, \varphi_{\hat{k}}\right) e^{-i\hat{m}\varphi_{\hat{k}}}, \tag{6}$$

$$F_{\hat{n}}^{\hat{m}} = \sum_{\hat{j}=1}^{\hat{p}} w_{\hat{j}} \beta_{\hat{j}}^{\hat{m}} \bar{P}_{\hat{n}}^{\hat{m}}(\cos\theta_{\hat{j}}), \tag{7}$$

where the temporary set of $\beta_{\hat{j}}^{\hat{m}}$ coefficients in Eq. (6) are calculated for $\hat{j} = 1{:}\hat{p}$ and $\hat{m}$ =-$\hat{p}{:}\hat{p}$. As the azimuthal sampling points $\varphi_{\hat{k}}$ are equally spaced, the $\hat{j}^{th}$ set of $\beta_{\hat{j}}^{\hat{m}}$ coefficients can be calculated with the FFT; totalling $O(\hat{p}^2 \log \hat{p})$ operations for the calculation of $\hat{p}$ FFTs in Eq. (6). Equation (7) requires $O(\hat{p}^3)$ operations: a summation over $\hat{p}$ for each of the $2\hat{p}^2$ grid points. Similarly, Eq. (3) can be applied in two steps after initially substituting for $Y_n^{-m}(\theta_j, \varphi_k)$:

$$\alpha_j^m = \sum_{\hat{n}=|\hat{m}|}^{\hat{p}} F_{\hat{n}}^{\hat{m}} \bar{P}_{\hat{n}}^{\hat{m}}(\cos\theta_j), \tag{8}$$

$$f\left(\theta_j, \varphi_k\right) = \sum_{m=-n}^{n} \alpha_j^m e^{im\varphi_k}, \tag{9}$$

where the temporary set of α_j^m coefficients in Eq. (8) are calculated for $j = 1{:}p$ and m =-$j{:}j$ ($O(\hat{p}^2 p)$ operations) and Eq. (9) can similarly be applied via p calls of the FFT ($(p^2 \log p)$ operations). It can be seen that Eq. (8) applies the $\hat{p} \rightarrow p$ filtering of the $F_{\hat{n}}^{\hat{m}}$ spherical basis function coefficients ($\hat{n} = 0{:}\hat{p}, \hat{m} = -\hat{n}{:}\hat{n}$) to yield the temporary set of coefficients α_j^m, defined for the new truncation number p at the polar sampling points θ_j. Equation (9) then calculates the far field signature function on the corresponding azimuth grid points φ_k via the FFT.

When applying Eqs. (6)-(9) for the express purpose of filtering a far field signature function to a coarser or finer grid of sample points, the explicit conversion to and from the spherical basis functions is not required. In this case, Eqs. (7) and (8) can be combined into a single step (Bohme and Potts 2003), as follows:

$$\alpha_j^m = \sum_{\hat{j}=1}^{\hat{p}} w_{\hat{j}} \beta_{\hat{j}}^{\hat{m}} S\left(x_j, x_{\hat{j}}\right), \quad S\left(x_j, x_{\hat{j}}\right) = \sum_{\hat{n}=|\hat{m}|}^{\hat{p}} \bar{P}_{\hat{n}}^{\hat{m}}(\cos\theta_j) \bar{P}_{\hat{n}}^{\hat{m}}(\cos\theta_{\hat{j}}) \tag{10}$$

The inner *summation* appearing in Eq. (10) for $S\left(x_j, x_{\hat{j}}\right)$, has a closed form (Jakob-Chien and Alpert 1997) known as the Christoffel-Darboux formula. This sum can be evaluated, assuming $\hat{p} \neq p$ (as is the case when filtering or interpolating the signature functions) as follows:

$$S\left(x_j, x_{\hat{j}}\right) = \frac{1}{x_{\hat{j}} - x_j} \varepsilon_{\hat{p}+1}^{\hat{m}} \left[\bar{P}_{\hat{p}+1}^{\hat{m}}(x_j) \bar{P}_{\hat{p}}^{\hat{m}}(x_{\hat{j}}) - \bar{P}_{\hat{p}}^{\hat{m}}(x_j) \bar{P}_{\hat{p}+1}^{\hat{m}}(x_{\hat{j}})\right] \tag{11}$$

where $\varepsilon_n^m = \sqrt{(n^2 - m^2)/(4n^2 - 1)}$. Substituting Eq. (11) into Eq. (10) thus allows the middle steps of the direct spherical filter to be applied in a single summation over $\hat{p}$. Note that the algorithmic complexity of the combined

middle steps is not reduced: the summation over $\hat{p}$ for each of the $2p^2$ far-field grid points still has a computational cost of O(p^3). However, the calculation of Eq. (11) can now be accelerated using fast methods.

3.1.1 Numerical implementation of the direct spherical filter algorithm

The direct spherical filtering algorithm has been implemented in MATLAB as follows:

1. The $\hat{p}$ FFTs in Eq. (6) are applied to the columns of the input set of far field signature function sample points, $f(\theta_j, \varphi_{\hat{k}})$, which are stored as a $[\hat{p} \times 2\hat{p}]$ array.
2. The columns of the $[\hat{p} \times 2\hat{p}]$ array $\beta_j^{\hat{m}}$ output from step 1 are reshaped into a single column vector, and Eq. (10) is applied using the Christoffel-Darboux formula (Eq. (11)) as a sparse matrix-vector product. The sparse matrix has dimensions of $[2p^2 \times 2\hat{p}^2]$ with the individual matrix coefficients of the form $w_{\hat{j}} S(x_j, x_{\hat{j}})$ arranged on each row such that the matrix-vector product applies the summation in Eq. (10). The row dimension of the matrix, of size $2p^2$, automatically applies the truncation or 0-padding required for the second FFT applied in Eq. (9) and any reordering for zero-frequency shifting of the initial/final FFTs is also incorporated into the sparse matrix by reordering the rows/columns.
3. The $[2p^2 \times 1]$ column vector output from step 2 is reshaped back into a $[p \times 2p]$ array and Eq. (9) is applied via p invocations of the FFT, yielding the filtered far-field signature function $f(\theta_j, \varphi_k)$.

The direct spherical filter is applied in the present FMBEM model by precalculating and storing the sparse matrix, and so at run-time the filter is efficiently applied as FFTs, array reshaping (which does not require the array data to be rewritten) and a sparse matrix-vector product.

3.2 The NFFT Spherical Filter

The fast spherical filtering methods in the literature (Jakob-Chien and Alpert 1997, Bohme and Potts 2003, Gumerov and Duraiswami 2004) seek to further reduce the algorithmic complexity of the middle steps of the direct spherical filter by approximating the O(N^2)-type interaction involved in evaluating the $1/(x_{\hat{j}} - x_j)$ term appearing in Eq. (11). Substituting Eq. (11) into Eq. (10) and rearranging yields (Bohme and Potts 2003):

$$\alpha_j^m = \varepsilon_{\hat{p}+1}^{\hat{m}} \left(\bar{P}_{\hat{p}}^{\hat{m}}(x_j) \sum_{\hat{j}=1}^{\hat{p}} \frac{w_{\hat{j}} \beta_{\hat{j}}^{\hat{m}} \bar{P}_{\hat{p}+1}^{\hat{m}}(x_{\hat{j}})}{x_{\hat{j}} - x_j} - \bar{P}_{\hat{p}+1}^{\hat{m}}(x_j) \sum_{\hat{j}=1}^{\hat{p}} \frac{w_{\hat{j}} \beta_{\hat{j}}^{\hat{m}} \bar{P}_{\hat{p}}^{\hat{m}}(x_{\hat{j}})}{x_{\hat{j}} - x_j} \right). \tag{12}$$

Each of the sums in the above equation can be applied using an NFFT by first regularising the kernel near to the singularity ($x_{\hat{j}} = x_j$) and then smoothing and scaling the resulting function to provide a continuous periodic function over the $(x_{\hat{j}} - x_j) = \pm\frac{1}{2}$ range. The resulting function can then be well approximated by a Fourier series sampled on a non-uniform grid of points (recall that $x_{\hat{j}}$ and x_j correspond to the nodes of the $\hat{p}$ and p-point Gauss-Legendre rules, respectively) and so the summations in Eq. (12) can be accelerated using an NFFT.

3.2.1 Smoothing and scaling the NFFT kernel

Assuming a kernel of the form:

$$f(x_j) = \sum_{\hat{j}=1}^{\hat{p}} \beta_{\hat{j}} K(x_{\hat{j}} - x_j), \tag{13}$$

for each summation term appearing in Eq. (12), the kernel $K(y) = 1/y$ can be smoothed and scaled to yield a 1-periodic version of the function, K_R, over the range $[-1/2 : y : 1/2]$ as follows. Firstly $K(y)$ is set to 0 at $y = 0$ to remove the singularity there and the function is then parameterised by a constant a, which determines the range near $y = 0$ and $y = \pm 1/2$ where the kernel will be replaced by a smooth sine series. $K_R(y)$ is thus defined as:

$$K_R(y) = \begin{cases} T_I(y), & y \in \left(-\frac{a}{n}, \frac{a}{n}\right), \\ T_B(y), & y \in \left[-\frac{1}{2}, -\frac{1}{2} + \frac{a}{n}\right) \cup \left(\frac{1}{2} - \frac{a}{n}, \frac{1}{2}\right], \\ K(y), & \text{otherwise}, \end{cases} \tag{14}$$

where n is the number of terms later used in the Fourier series representation of $K_R(y)$ (Bohme and Potts 2003). The sine series used to regularise the kernel are defined by a smoothing parameter q to ensure continuity of the kernel and its first $q - 1$ derivatives at the joins: $\pm\frac{a}{n}$ and $\pm\left(\frac{1}{2} \mp \frac{a}{n}\right)$. The sine series are (Bohme and Potts 2003):

$$T_I(y) = \sum_{l=1}^{q} a_l^I \sin\frac{\pi n l}{2a} y, \tag{15}$$

$$T_B(y) = \begin{cases} \sum_{l=1}^{q} a_l^B \sin\frac{\pi n l}{2a}\left(y-\frac{1}{2}\right), & y \in \left(\frac{1}{2}-\frac{a}{n}, \frac{1}{2}\right], \\ \sum_{l=1}^{q} a_l^B \sin\frac{\pi n l}{2a}\left(y+\frac{1}{2}\right), & y \in \left[-\frac{1}{2}, -\frac{1}{2}+\frac{a}{n}\right), \end{cases} \tag{16}$$

where the coefficients a_l^I and a_l^B are calculated by enforcing the previously mentioned continuity requirements at the joins: see (Bohme and Potts 2003) for details. The kernel values $x_{\hat{\jmath}} - x_j$ must also be scaled within the $\pm 1/2$ range for the periodic function. It is advantageous to apply the scaling such that $x_{\hat{\jmath}} - x_j \in \left[-\frac{1}{2}+\frac{a}{n}, \frac{1}{2}+\frac{a}{n}\right]$ as this minimises the number of correction terms required over the smoothed regions of K_R (discussed next). To satisfy this condition, $x_{\hat{\jmath}} - x_j$ is replaced with $s * (x_{\hat{\jmath}} - x_j)$ where the scaling factor s is defined as (Bohme and Potts 2003):

$$s = \frac{\frac{1}{4}-\frac{a}{2n}}{\max\{|x_{\hat{\jmath}}|, |x_j|\}}, \tag{17}$$

and $K(y)$ is replaced with $K\left(\frac{y}{s}\right)$ to correct for the scaling of the magnitude introduced by s. The smoothing and scaling operations required for the NFFT spherical filter are demonstrated in Figure 1 below for the following settings: $\hat{p} = 150$, $p = 100$, $a = 6$, $q = 6$, $n = 256$, $\hat{\jmath} = 1{:}150$, $j = 15$.

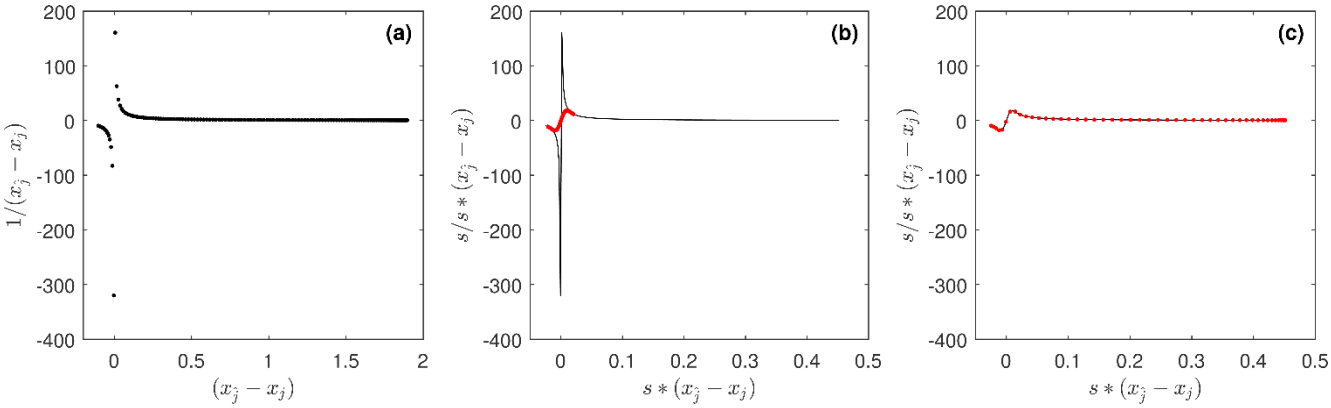

Figure 1. Example of the smoothing and scaling operations applied to the $K(y)$ kernel to apply the NFFT spherical filter. Figure 1(a) shows the discrete points for the unaltered $K(y)$ kernel for $\hat{\jmath} = 1{:}150$ and $j = 15$, where the corresponding $x_{\hat{\jmath}}$ and x_j values are the Gauss-Legendre points for the $\hat{p} = 150$ and $p = 100$ rules. Figure 1(b) shows the continuous scaled $K(y)$ function in black, and the section of the scaled function which is replaced with the smoothed function, $K_R(y)$, in red. The complete smoothed function $K_R(y)$ is shown in Figure 1(c) (continuous line) along with the corresponding Fourier series approximation of the smoothed function, $K_{RF}(y)$, shown as sparsely sampled circular markers for clarity (note the non-equal spacing for the Gauss-Legendre sample points).

It can be seen from Figure 1(b) that by scaling the y-range of the $K(y)$ kernel by s, only a small number of points lie within $y = \pm a/n$, and so the smoothing function $K_R(y)$ must only be applied to small subset of points: outside of this range $K_R(y) = K(y)$ (see Eq. (14)) and so no correction term is required to account for the smoothing.

3.2.2 Fourier series approximation of the smoothed kernel

The smoothed function $K_R(y)$ can thus be approximated by the following Fourier series **(Bohme and Potts 2003)**

$$K_{RF}(y) = \sum_{l=-n/2}^{n/2-1} b_l e^{\mathrm{i}2\pi l y}, \tag{18}$$

where n is the number of Fourier terms (chosen here to be the next largest power of 2 for $\max(\hat{p}, p)$) and the Fourier coefficients b_l are defined as follows:

$$b_l = \frac{1}{n}\sum_{k=-n/2}^{n/2-1} K_{RF}\left(\frac{k}{n}\right) e^{-\mathrm{i}2\pi k l/n}, \quad l = -\frac{n}{2}:\frac{n}{2}-1. \tag{19}$$

An example of the Fourier series approximation for $K_R(y)$ is shown in Figure 1(c), where it can be seen that $K_{RF}(y)$ (markers) is in good agreement with the smoothed function $K_R(y)$ (continuous line). The Fourier series approximation of the smoothed function can thus be efficiently calculated via an NFFT (discussed next).

3.2.3 Fast evaluation of the kernel using the NFFT

The function kernel $K(y)$ is thus evaluated in two parts (assuming the error from the Fourier series approximation of the smoothed function $K_R(y)$ is negligibly small):

$$K(y) = \big(K(y) - K_R(y)\big) + K_{RF}(y), \tag{20}$$

where the non-equally spaced FFT is used to accelerate the calculation of Eq. (18) for the Fourier series approximation of the smoothed kernel, while a 'near-field' correction factor $\big(K(y) - K_R(y)\big)$ is directly calculated for the small number of smoothed function terms within $y = \pm a/n$, to give the complete $K(y)$ function. The correction factor is equal to the difference between the black and red lines in Figure 1(b): applying this correction to $K_{RF}(y)$ will thus yield the original function (Figure 1(a)) for the scaled y-coordinates. Denoting the near-field correction as $K_{NE}(y) = \big(K(y) - K_R(y)\big)$, Eq. (13) is thus calculated as (Bohme and Potts 2003):

$$f(x_j) = \sum_{\hat{\jmath}=1}^{\hat{p}} \beta_{\hat{\jmath}} K_{NE}(x_{\hat{\jmath}} - x_j) + \sum_{\hat{\jmath}=1}^{\hat{p}} \beta_{\hat{\jmath}} K_{RF}(x_{\hat{\jmath}} - x_j). \tag{21}$$

The first term in Eq. (21) is evaluated in the same way as Eq. (11) for the direct spherical filtering algorithm, but must only be evaluated for the small number of smoothed points in $K_R(y)$. The dominant computational cost in Eq. (21) is thus for the evaluation of the $K_{RF}(y)$ term, which can be evaluated using the NFFT as follows:

$$\begin{aligned} f_{RF}(x_j) &= \sum_{\hat{\jmath}=1}^{\hat{p}} \beta_{\hat{\jmath}} K_{RF}(x_{\hat{\jmath}} - x_j), \\ &= \sum_{\hat{\jmath}=1}^{\hat{p}} \beta_{\hat{\jmath}} \sum_{l=-n/2}^{n/2-1} b_l e^{\mathrm{i}2\pi l(x_{\hat{\jmath}} - x_j)}, \\ &= \sum_{l=-n/2}^{n/2-1} b_l \Big(\sum_{\hat{\jmath}=1}^{\hat{p}} \beta_{\hat{\jmath}} e^{\mathrm{i}2\pi l x_{\hat{\jmath}}}\Big) e^{-\mathrm{i}2\pi l x_j}. \end{aligned} \tag{22}$$

The inner summation in Eq. (22) can be approximately calculated (with a controllable accuracy) using a 'Type 1' NFFT (Greengard and Lee 2004), which is an FFT calculated for a non-equally spaced set of sample points $x_{\hat{\jmath}}$. The result of this operation can then be multiplied with b_l and the outer summation can be similarly evaluated with a 'Type 2' NFFT (Greengard and Lee 2004): an FFT for a non-equally spaced set of evaluation points x_j. Each of these NFFTs can be applied in $O(p \log p)$ operations for p sample/evaluation points. In the present work, an NFFT algorithm which uses fast Gaussian gridding (Greengard and Lee 2004) has been developed by the authors in MATLAB and incorporated into the NFFT spherical filtering algorithm. The underlying NFFT algorithm essentially 'smears' the non-uniformly distributed sample or evaluation points onto a finely sampled regular grid and then applies the regular FFT on this finer grid: see (Greengard and Lee 2004) for details.

The NFFT spherical filter algorithm thus reduces the algorithmic complexity for the direct evaluation of Eq. (12) from $O(p^3)$ operations to $O(p^2 \log p)$ operations: the NFFT summation is called $O(p)$ times (for $m = -j{:}j$), each costing $O(\hat{p} \log \hat{p})$ operations. Assuming $p \approx \hat{p}$, and that the number of Fourier terms used in Eq. (19) is chosen as $n \approx (p, \hat{p})$, the NFFT spherical filter algorithm can be shown to have an $O(p^2 \log p)$ complexity: see (Bohme and Potts 2003) for details. Equations (6) and (9) are similarly applied as in the direct spherical filter using the FFT: each costing $O(p^2 \log p)$ operations. Thus the total algorithmic complexity for the NFFT spherical filter is $O(p^2 \log p)$ + some lower complexity terms resulting from the near-field correction and error control of the NFFT.

3.2.4 Numerical implementation of the NFFT spherical filter algorithm

The NFFT spherical filtering algorithm has been implemented in MATLAB as follows:

1. The $\hat{p}$ FFTs in Eq. (6) are applied to the columns of the input set of far field signature function sample points, $f(\theta_j, \varphi_{\hat{k}})$, which are stored as a $[\hat{p} \times 2\hat{p}]$ array.
2. The columns of the $[\hat{p} \times 2\hat{p}]$ array $\beta_j^{\hat{m}}$ output from step 1 are reshaped into a single $[2\hat{p}^2 \times 1]$ column vector and the corresponding sets of $\beta_{\hat{\jmath}}$ coefficients (Eq. (13)) for each of the summations appearing in Eq. (12) are calculated. The columns of $\beta_{\hat{\jmath}}$ coefficients are reshaped back to $[\hat{p} \times 2\hat{p}]$ arrays.

3. The Type 1 and Type 2 NFFTs are then applied in turn to each set of β_j coefficients via 3 substeps. Firstly, a matrix-vector product with a $[n \times \hat{p}]$ sparse matrix applies the 'smearing' operation for the NFFT to the regular spaced grid with n points. Then, $2\hat{p}$ invocations of the n-point FFT are applied to the columns of the $[n \times 2\hat{p}]$ array. Finally, a second sparse matrix-vector product with a $[p \times n]$ sparse matrix is applied to sum the smeared coefficients over the regular grid to the p non-equally spaced points. Between applying the Type 1 and 2 NFFTs, the coefficients must be multiplied with the b_l coefficients according to Eq. (22). In the present algorithm the sparse matrix for the 3rd substep of the Type 1 NFFT, the multiplication with the b_l coefficients and the sparse matrix for the 1st substep of the Type 2 NFFT are combined into a single sparse $[n \times n]$ matrix and applied in one matrix-vector product operation.
4. The near field correction term $K_{NE}(y)$ is then applied to each $[\hat{p} \times 2\hat{p}]$ array of β_j coefficients output from step 2 via sparse matrix-vector multiplications with $[p \times \hat{p}]$ sparse matrices and the resulting arrays added to the $[p \times 2\hat{p}]$ arrays output from the NFFTs applied in step 3.
5. The arrays resulting from the evaluation of Eq. (21) (step 4) are then combined according to Eq. (12) to give the equivalent α_j^m coefficients from the NFFT algorithm. The $[p \times 2\hat{p}]$ array is reshaped into a $[2\hat{p}p \times 1]$ vector and multiplied with a sparse $[2p^2 \times 2\hat{p}p]$ array to apply the truncation or 0-padding required for the second FFT (Eq. (9)), as well as any element reordering for zero-frequency shifting of the final FFT. Similarly, the other sparse matrices in the previous steps incorporate any element reordering required between each of the FFT operations.
6. Finally, the $[2p^2 \times 1]$ column vector output from step 2 is reshaped back into a $[p \times 2p]$ array and Eq. (9) is applied via p invocations of the FFT, yielding the filtered far-field signature function $f(\theta_j, \varphi_k)$.

The NFFT spherical filter is applied in the present FMBEM model by precalculating and storing the sparse matrices for the NFFT/filtering algorithms, and so at run-time the filter is applied as a series of array reshaping operations (which does not require the array data to be rewritten), sparse matrix-vector products and FFTs.

3.3 Comparison of Spherical Filtering Algorithms

Steps 2-5 of the NFFT spherical filter algorithm (Section 3.2.4) must be performed faster than step 2 for the direct spherical filter (Section 3.1.1) for the NFFT algorithm to provide a computational advantage over the direct method. The overhead in performing the more numerous computational steps in the NFFT algorithm (each of which has a lower algorithmic complexity than step 2 of the direct filter) will determine this 'cross over' point with respect to the truncation number of the far field signature function, above which the NFFT will provide faster performance. Böhme and Potts reported a cross over point of $p = 64$ and an algorithmic scaling for the NFFT spherical filter which is slightly larger than O(p^2) (Bohme and Potts 2003). Comparatively, the cross over point for the fast spherical filter based on the 1D FMM was reported as $p = 106$ (Jakob-Chien and Alpert 1997), while for the fast Legendre transform filter, a cross over point of $p \approx 500$ was cited in Press et al. (Press, et al. 2007). Clearly the exact cross over point for each algorithm will strongly depend on the implementation and programming language employed, as well as the contributions from the lower-complexity terms. Finally, it should be noted that the computational cost of the filtering operations in the high-frequency FMBEM can be small compared to the other parts of the algorithm, and so in practice the high-frequency FMBEM can still exhibit an O(*N*log*N*) complexity when utilizing the direct filtering algorithm: see, for example, (Chaillat, Bonnet and Semblat 2008).

4 NUMERICAL RESULTS

The direct and NFFT spherical filters have both been implemented in MATLAB, and incorporated into a broadband Helmholtz FMBEM model. The following subsections compare the algorithmic and memory complexities of the different spherical filtering algorithms, and demonstrate the effect of the spherical filtering functions on the Helmholtz FMBEM for large-scale acoustic scattering problems. All calculations were performed on a desktop workstation equipped with an Intel i7 hexacore CPU (running at 3.30 GHz) and 80 GB of RAM.

4.1 Comparison of the Direct and NFFT Algorithmic and Memory Complexities

The algorithmic and memory complexities for the present implementations of the direct and NFFT spherical filtering algorithms are presented in Figure 2 below, for both the interpolation and 'anterpolation' (adjoint interpolation, or filtering) operations. The truncation numbers are selected as $p_2 = 1.5 * p_1$ and $p_1 = 1.5 * p_2$ for the interpolation and anterpolation cases respectively, where p_1 and p_2 denote the start and end truncation numbers, respectively. These ratios for the start/end truncation numbers represent typical values for the difference between the truncation numbers on successive levels of the octree structure for large-scale Helmholtz FMBEM problems. The complexity results are shown for the spherical filtering of 250 sets of far field signature functions, which might be a typical number of occupied boxes for an arbitrary boundary element mesh on the higher octree levels. At lower levels there would be both more sets of expansions and smaller truncation numbers.

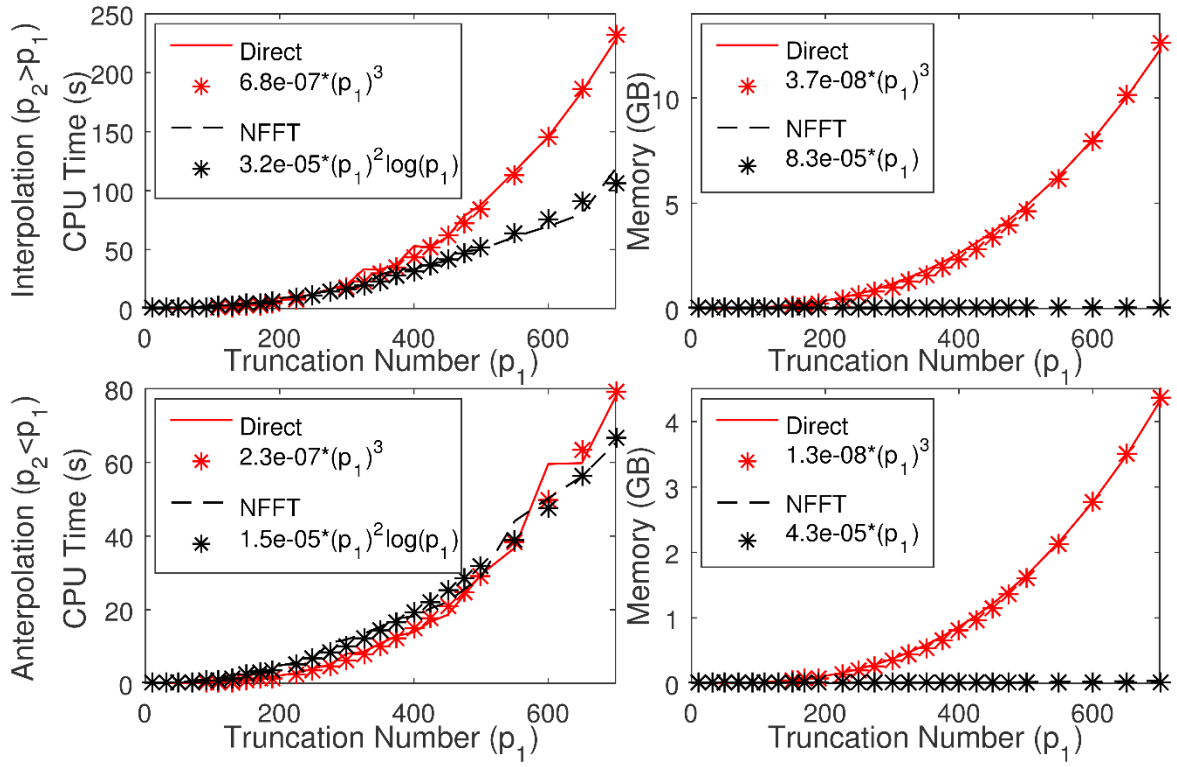

Figure 2. Comparison of the calculation times (left column) and storage requirements (right column) of the direct and NFFT spherical filtering methods for the interpolation (top row) and anterpolation (bottom row) operations. The truncation numbers are specified as $p_2 = 1.5 * p_1$ and $p_1 = 1.5 * p_2$ respectively for the interpolation and anterpolation cases, and the results are shown for the filtering of 250 sets of far field signature functions.

It can be seen from Figure 2 that both the direct and NFFT algorithms exhibit the expected algorithmic complexities: being proportional to O(p^3) for the direct method and O($p^2 \log p$) for the NFFT filter. In the present implementation the NFFT algorithm becomes faster than the direct method for $p_1 \approx 250$ for the interpolation case and $p_1 \approx 500$ for the anterpolation case. These cross over points are several times larger than those reported in Böhme and Potts (Bohme and Potts 2003) for their implementation of the NFFT filter, while the present results for the interpolation case are twice as fast as the $p \approx 500$ cross over point reported in Press et al. for the fast Legendre filter (Press, et al. 2007), and are approximately equivalent for the anterpolation case. It should also be noted that different settings have been used for the smoothing and smearing operations, while the results reported in (Bohme and Potts 2003) are only for the case where $p = \hat{p}$. These differences, as well as the different programming languages and implementations of the algorithms, likely account for the larger cross over point in the present work compared to Böhme and Potts (Bohme and Potts 2003).

The memory complexity of the NFFT spherical filtering algorithm appears to have not yet been reported in the published literature. The memory storage requirements (that is, the memory required to store the precomputed data) for the direct and NFFT spherical filters is presented in Figure 2. The memory footprint of the direct method scales as O(p^3): similar to the computational cost for the direct method. Conversely, the memory footprint for the NFFT filtering algorithm is shown to scale as O(p)—a negligible cost—while the maximum instantaneous memory used in the algorithm is determined by the refined sampling grid for the Type 1 and Type 2 NFFTs (Greengard and Lee 2004). This step requires a temporary array proportional to O(np) to be created for the smeared NFFT operations which, depending of the choice of the oversampling ratio used for the NFFT, can result in an array which has a larger memory footprint than the sparse matrix for the direct filtering method. However, this calculation can be split into blocks (i.e. filtering the far field signature functions in subsets) to reduce the instantaneous memory footprint of the NFFT, while the sparse matrix for the direct spherical filtering algorithm must be explicitly constructed and stored for the efficient implementation of the direct filter. Furthermore, in a parallel implementation of the FMBEM, the precomputed data for the spherical filters must be replicated across the worker threads. It can be seen in Figure 2 that this could amount to several GB of data per thread for the direct filter algorithm, while the stored data for the NFFT algorithm is negligibly small. Therefore, in the parallel implementation of the FMBEM it may become advantageous to switch from the direct to the NFFT filtering algorithm at mid-truncation numbers ($p = 200$ or so) to minimise the memory footprint of the filtering operations.

4.2 Numerical Results for the Broadband Helmholtz FMBEM Utilising the Direct and NFFT Filters

The direct and NFFT spherical filtering algorithms have both been incorporated into a broadband Helmholtz FMBEM model and are demonstrated for two large-scale acoustic scattering problems. Firstly, the acoustic scattering of a $k = 150$ plane wave from a unit radius sphere under a rigid (Neumann) boundary condition is considered. The sphere mesh is discretised with 1547088 plane triangular elements with constant unknowns, giving approximately 10 elements per wavelength. Selecting the 'switching' truncation number as $p \approx 200$ results in the filtering operations for the 2 highest octree levels (out of 8 levels for the current FMBEM settings) being treated with the NFFT algorithm, while the lower levels are treated with the direct spherical filter. Figure 3. compares the real and imaginary components of the total surface pressure as a function of angle from the direction of the incident field ($0°$ is the backscatter direction), with and without utilising the NFFT spherical filtering function to apply the filtering between the upper octree levels.

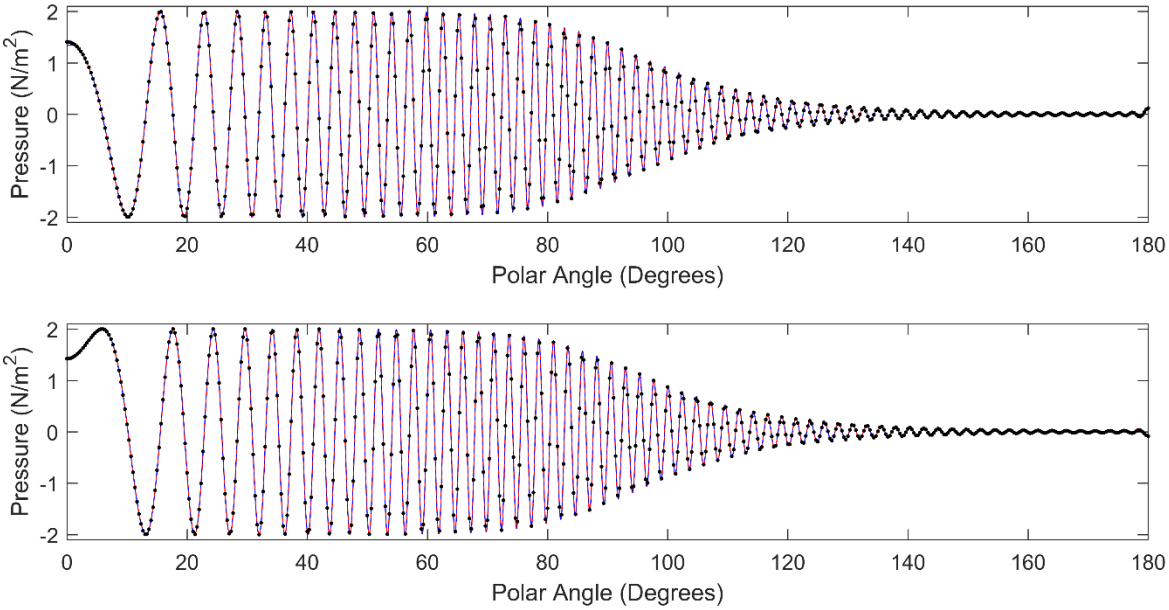

Figure 3. Comparison of the real (top) and imaginary (bottom) components of the total surface pressure for a $k = 150$ plane wave scattered from a unit radius sphere under a rigid (Neumann) boundary condition. The continuous red line, dashed blue line and black markers correspond to the combined NFFT/Direct filter FMBEM, Direct-filter only FMBEM and analytic solutions, respectively.

Figure 3. indicates good agreement between the two FMBEM solutions and the analytic solution: the relative error norm with respect to the analytic solution for the NFFT FMBEM was 0.59469%, compared to 0.59466% for the FMBEM solution which only used the direct spherical filtering algorithm. The NFFT FMBEM solution required 1.52hrs to set up and 10.39hrs to solve, while the direct FMBEM solution required 1.73hrs to set up and 10.51hrs to solve. The total memory storage footprint of the NFFT FMBEM was 15.68GB, compared to 16.05GB for the direct-filter only FMBEM solution. The NFFT algorithm thus provides a reduction in both the memory storage and computation time for the FMBEM, and will continue to provide more pronounced savings as the problem size and truncation numbers increase, i.e. such that more octree levels are treated with the NFFT filter.

Finally, as an application example, the acoustic scattering of a 3kHz plane wave impinging the BeTSSi II generic submarine pressure hull (Gilroy, et al. 2014) at broadside incidence for a rigid boundary condition is calculated using the NFFT-accelerated FMBEM: see Figure 4. The outer hull of the BeTSSi II submarine model has a total length of 62 m, a main hull diameter of 7 m and is discretised in the present BEM mesh via 1538580 elements. The NFFT-accelerated FMBEM model required 71 minutes to set up and 6.86hrs to solve using a single CPU core. The pressure solution from this FMBEM calculation can be used, for example, to calculate the monostatic target strength of large-scale objects at high frequencies to compare to the various approximate high-frequency TS models (i.e. based on Kirchhoff methods or raytracing).

5 CONCLUSIONS

This paper has presented a theoretical and computational analysis of the direct and NFFT spherical filtering algorithms used to filter the far field signature functions in the high-frequency FMBEM. Numerical results indicate that the present implementation conforms to the algorithmic complexities for each method, while the numerically demonstrated memory complexity for the NFFT filter appears to be analysed for the first time in the literature. The cross over point at which the NFFT filter becomes faster than the direct method is shown to be slower in the

present implementation compared to that for other models in the published literature, while still outperforming a fast filtering algorithm based on the Legendre transform by up to a factor of 2. Numerical examples from a broadband Helmholtz FMBEM model have demonstrated the advantages of the NFFT spherical filtering algorithm for large-scale problems, while the small memory storage requirements of the NFFT filter indicate that this filter may also be beneficial for minimising the memory footprint for parallel implementations of the FMBEM where the filtering data must be replicated amongst the workers. Further optimisation of the present implementation to minimise the cross over point for the NFFT filter will reduce the problem size at which the fast filtering algorithm can be employed to accelerate the filtering operations in the broadband FMBEM.

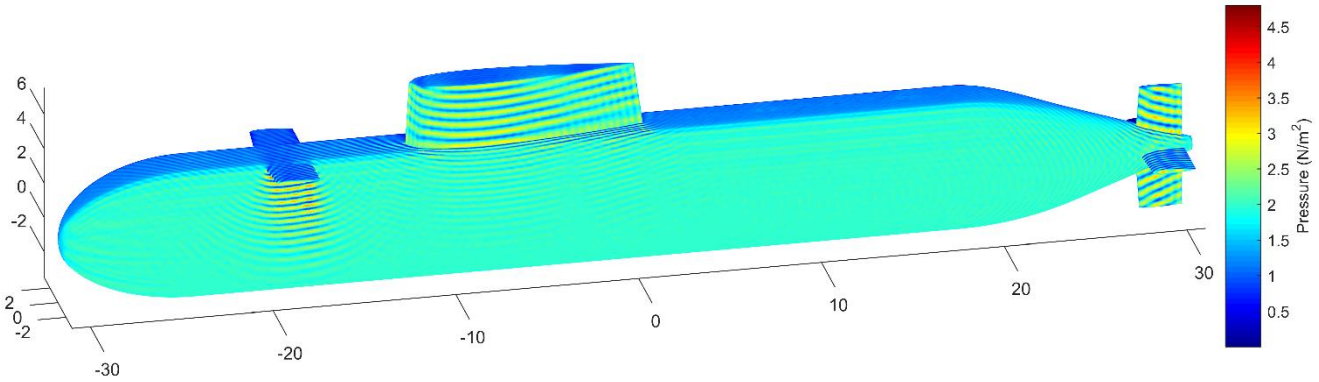

Figure 4. Acoustic scattering of a 3kHz plane wave from the BeTSSi II generic submarine pressure hull at broadside incidence.

REFERENCES

Bohme, M., and D. Potts. 2003. “A Fast Algorithm for Filtering and Wavelet Decomposition on the Sphere.” *Electronic Transactions on Numerical Analysis* 16: 70-93.

Chaillat, S., and F. Collino. 2015. “A Wideband Fast Multipole Method for the Helmholtz Kernal: Theoretical Developments.” *Mathematical Modelling and Numerical Analysis* 660-678.

Chaillat, S., M. Bonnet, and J.-F. Semblat. 2008. “A multi-level fast multipole BEM for 3-D elastodynamics in the frequency domain.” *Computer Methods in Applied Mechanics and Engineering* 4233-4249.

Cheng, H., W. Y. Crutchfield, Z. Gimbutas, L. F. Greengard, J. F. Ethridge, J. Huang, V. Rokhlin, Y. Norman, and J. Zhao. 2006. “A Wideband Fast Multipole Method for the Helmholtz Equation in Three Dimensions.” *Journal of Computational Physics* 216: 300-325.

Darve, E., and P. Have. 2004. “A Fast Multipole Methodfor Maxwell Equations Stable at all Frequencies.” *Philosophical Transactions of the Royal Society of London A* 362: 603-628.

Gilroy, L., C. De Jong, B. Nolte, and I. Schafer. 2014. “BeTSSi II: Benchmark Target Strength Simulation.” *Anlage zu WTD71 0029/2013 WB.* Accessed August 9, 2017.

Greengard, L., and J.-Y. Lee. 2004. “Accelerating the Nonuniform Fast Fourier Transform.” *SIAM Review* 46 (3): 443-454.

Greengard, L., V. Huang, V. Rokhlin, and S. Wandzura. 1998. “Accelerating Fast Multipole Methods for the Helmholtz Equation at Low Frequencies.” *IEEE Computing in Science & Engineering* 25: 32-38.

Gumerov, N. A., and R. Duraiswami. 2009. “A Broadband Fast Multipole Accelerated Boundary Element Method for the 3D Helmholtz Equation.” *The Journal of the Acoustical Society of America* 125: 191-205.

Gumerov, N. A., and R. Duraiswami. 2004. *Fast Multipole Methods for the Helmholtz Equation in Three Dimensions.* Oxford: Elsevier.

Gumerov, N. A., and R. Duraiswami. 2003. “Recursions for the Computation of Multipole Translation and Rotation Coefficients for the 3-D Helmholtz Equation.” *SIAM Journal on Scientific Computing* 25 (4): 1344-1381.

Jakob-Chien, R., and B. K. Alpert. 1997. “A Fast Spherical Filter with Uniform Resolution.” *Journal of Computational Physics* 136: 580-584.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 2007. *Numerical Recipes.* 3rd. New York: Cambridge University Press.

Rokhlin, V. 1993. “Diagonal Forms of Translation Operators for the Helmholtz Equation in Three Dimensions.” *Applied and Computational Harmonic Analysis* 1: 82-93.

Yasuda, Y., T. Oshima, T. Sakuma, A. Gunawan, and T. Matsumoto. 2010. “Fast Multipole Boundary Element Method for Low-Frequency Acoustic Peoblems Based on a Variety of Formulations.” *Journal of Computational Acoustics* 18 (4): 363-395.