EUROREGIO
BNAM2022
Joint Acoustics Conference
9th-11th of May, Aalborg, Denmark

Mesh2HRTF / NumCalc: An Open-Source Project to Calculate HRTFs and wave scattering in 3D

W. Kreuzer[1,*], **K. Pollack**[1], **P. Majdak**[1], **F. Brinkmann**[2]

[1]Acoustics Research Institute, Austrian Academy of Sciences, Vienna, Austria,
[2]Audio Communication Group, Technical University of Berlin, Berlin, Germany.
[*]wolfgang.kreuzer@oeaw.ac.at

Abstract

Mesh2HRTF is an open source project originally developed to provide users with an easy-to-use platform for the calculation of the head-related transfer functions based on the boundary element method. However, its boundary element module NumCalc with its easy-to-use input file is flexible, and thus, it can be used as a standalone application for solving the Helmholtz equation in various other contexts such as calculating the sound field inside a car or a duct, or the calculation of the scattering of sound waves in an open field around objects in 3D. NumCalc is a combination of the collocation approach based on the Burton-Miller formulation of the boundary integral equation and the fast multipole method. This paper discusses novel features of NumCalc such as the definition of frequency-dependent admittance boundary conditions, and point to its current limitations with respect to the numerical accuracy and memory consumption.

Keywords: boundary element method, software, acoustic scattering

1 Introduction

In acoustics, the simulation of wave propagation around objects is an important task. Examples include the calculation of head-related transfer functions (HRTFs), which model the filtering effect of the human anthropometry on incoming sound, the simulation of noise mitigation measures, or the modelling of musical instruments [1, 2, 3]. Especially for wave propagation in open domains, the boundary element method (BEM) plays a special role, because compared to other numerical methods like the finite element method or the finite difference method, the BEM only needs a discretization of the surface of the scattering objects, but not the space inside or around the objects.

In this paper, we want to introduce the open source project Mesh2HRTF and its subproject NumCalc. Of course, one may ask “Why another BEM software project?”. First of all, Mesh2HRTF is open source, and second, it is relatively easy to use without the need of deep knowledge of BEM topics like boundary integral equations, singular integral operators or methods to deal with critical frequencies.

Mesh2HRTF was originally planned and designed as an easy-to-use tool for people working in 3D audio to calculate HRTFs. The idea behind the Mesh2HRTF-chain is, that users only need to load the geometry of the individual’s head and pinnae into the open source project Blender [4], mark a few elements there, and Mesh2HRTF takes care of the rest. To that end, Mesh2HRTF comes with an export script for Blender that generates a file called `NC.inp` that serves as an input for the BEM-code NumCalc. NumCalc then numerically solves the Helmholtz equation, that models the propagation of sound waves, and calculates the sound pressure on the head and at an evaluation grid around the head. Finally, Mesh2HRTF contains Matlab/Octave and Python scripts that calculate HRTFs using the BEM results and that store the filter functions in the SOFA format, a standardized format for HRTFs [5]. Interested readers are referred to [6] for some tutorials on how to use the pipeline provided with Mesh2HRTF. However, the structure of the input file for NumCalc is very simple, and therefore, the BEM code can be applied to many problems in acoustics. Also, as convenient as Blender and the export script may be, since `NC.inp` is a text file, there are multiple ways to create this file. E.g., for the sphere in Section 4, a simple Octave script was used, see [7] for the octave script used to create Fig. 3.

To be clear, among all the other free and open source BEM projects, Mesh2HRTF and NumCalc are probably not the most elaborate or the most accurate implementations of the Helmholtz BEM, but they provide an easy way to achieve sufficiently accurate simulation results for users not experienced in numerical mathematics or partial differential equations. Nevertheless, we also feel that users should know a bit about the software they are using, about possible caveats and the accuracy they can expect from the calculation.

The aim of this paper is twofold. First, we will use the opportunity to introduce a new feature of the soon-to-be-released version 1.0 of Mesh2HRTF: Frequency-dependent admittance boundary conditions to model sound-absorbing materials. Second, we will talk a bit about the input for NumCalc and its implementation, and provide results for a benchmark example to give users an impression of the accuracy they can expect when using Mesh2HRTF and/or NumCalc. We are fully aware that this paper can only give a little glimpse into some aspects of Mesh2HRTF and its application. A more detailed paper is currently in preparation, we also refer interested readers to the Mesh2HRTF Wiki [8] and to [9].

2 General

The BEM implementation in NumCalc is based on collocation with constant elements, i.e., on each element of the mesh (= discretization of the surface of the scatterer using quadrilaterals or triangles), the acoustic pressure and the particle velocity are assumed to be constant. As a general rule of thumb that works very well for most practical problems, the mesh should contain about 6 to 8 elements per wavelength [10]. See Fig. 1 for an example of a mesh of a human head and pinna that is fine enough to be used for frequencies up to 20000 Hz.

On one hand, it is known that the accuracy of collocation with constant elements is not high compared to other methods like, e.g, Galerkin with higher order elements. However, on the other hand, the implementation of the collocation is easier, the generation of the system matrix uses (in general) less numerical operations and the fact that we are using discontinuous elements allows us to subdivide elements without having to consider neighbouring elements. Also, for a lot of practical applications, a relative error in the range of 10^{-3} to 10^{-2}, which can be achieved by NumCalc, is more than enough.

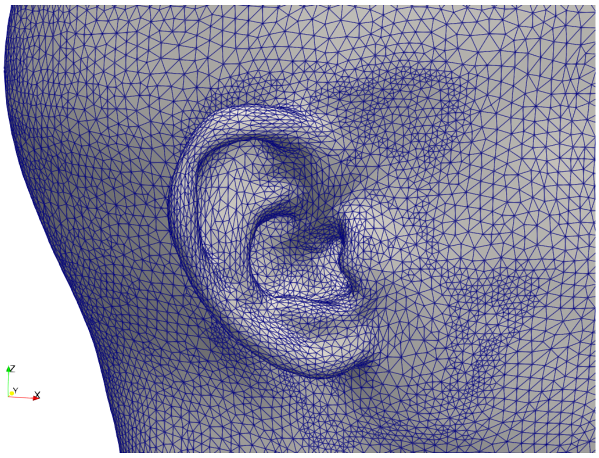

Figure 1: Mesh of the human pinna and parts of the head. The mesh (including parts of the torso) contains about 74000 triangular elements with an average edge length of 2.4 mm.

One drawback of the BEM is the fact the system matrix to be solved in connection with the BEM is densely populated and of size $N \times N$ where N denotes the number of elements. For the mesh in Fig. 1, for example, the system matrix would be a densely populated matrix with about 5.4 billion complex valued entries, which puts a strain on memory and calculation time. To deal with this fact, NumCalc uses the (multilevel) fast multipole method (FMM, cf. [11]) to speed up matrix-vector multiplication and to reduce the amount of memory needed to store the system matrix. In a nutshell, the FMM can be described using a man-in-the-middle principle, where parts of the mesh are grouped into different clusters, and point-to-point interactions are replaced by local expansions between point and center of the cluster, and expansion between cluster centers. This helps in reducing the number of operations from $O(N^2)$ to $O(N \log_2 N)$, also, the system matrix is never constructed in full, which reduces the time to set up the linear system of equations and the memory consumption. The linear system itself is solved using an iterative conjugated gradient solver, cf. [12, Chapter 7.4.1]. However, users should be aware that the FMM is just an approximation of the original system, and that its accuracy and efficiency is dependent on the clustering and the frequencies used.

On a side note: It is also known that the BEM for the Helmholtz equation has numerical difficulties at certain critical frequencies [13, 14]. To deal with these instabilities at critical frequencies, a formulation proposed by Burton and Miller [13] is used, however, this is handled automatically by NumCalc and users do not need to bother about the different integral equations.

3 Important input parameters for NumCalc

In order for NumCalc (and any other Helmholtz BEM code) to work, one needs to provide at least the following information:

- The frequencies for which the scattering problem should be solved,
- the geometry of the scatterer using small triangular or quadrilateral elements,
- boundary conditions for the elements,
- optional: Sound sources and evaluation points, that do not lie on the surface of the objects and where the sound field should be calculated too.

There are also general parameters containing, e.g., information about the speed of sound and the density of the medium, a flag if an interior or an exterior problem needs to be solved, and some advanced parameters that allow experienced users to change details with respect to the clustering used for the FMM. A detailed discussion of these parameters would be out of the scope of this manuscript, the default values set for these parameters cover a wide range of use cases, users do not need to worry about these settings in general. The interested reader is referred to [8] for an in-depth description of these parameters.

3.1. Frequency-defining curves

NumCalc numerically solves the Helmholtz equation in the frequency domain for a given set of frequencies provided by the user. Frequency-defining curves are piecewise linear functions that map a frequency step to a frequency or a frequency-dependent value of a boundary condition. The user has to provide the nodes of these curves. An example: If one would want to calculate the sound field for 10 frequencies between 1000 Hz and 10000 Hz in steps of 1000 Hz, a simple way to describe the curve is given in Lst. 1. A line in the input file that starts with the hash tag '#' is ignored by NumCalc and can be used for comments. The first non-comment line consists of a dummy parameter (`dummy = 1`) followed by the number of frequency steps to be calculated (`Number_of_freqsteps = 10`) followed by the distance h_i between two frequency steps (`stepsize = 1.0`). The last entry in this line is given by the start index i_0 of the curve (`Index0 = 0.0`). The next line contains the unique number of the curve (in case of a frequency definition curve the curve number is always `Curvenumber = 0`) and the number of nodes defining the piecewise linear function mapping the frequency step n to the respective frequency f_n. In Lst. 1 this map is defined as the line between the two nodes $(S_1, F_1) = (0.0, 0\,\text{Hz})$ and $(S_2, F_2) = (10.0, 10000\,\text{Hz})$. The frequency for the n-th frequency step is then given as

$$f_n = F_1 + \frac{F_2 - F_1}{S_2 - S_1}((i_0 + h_i \cdot n - S_1) = 0\,\text{Hz} + \frac{10000\,\text{Hz}}{10}(0.0 + 1.0 \cdot n) = 1000\,\text{Hz} \cdot n, n = 1, \ldots, 10. \quad (1)$$

```
# '#' denotes a comment line
# dummy_parameter_for_future_use Number_of_freqsteps stepsize Index0
1 10 1.0 0.0
# Curvenumber  Nr_of_nodes_defining_the_curve
0 2
# nodes of the curve (Si, Fi): freq-step vs frequency
0.0 0.0
10.0 10000.0
```

Listing 1: Definition of 10 uniform frequencies between 1000 Hz and 10000 Hz

A more elaborate example is given in Lst. 2. This code snippet contains the definition of 8 third-octave frequency bands (see Tab. 1) where three frequencies are chosen in each band.

```
#dummy n-steps stepsize
1      24   1.0 0.0
# Curvenr. Nr._of_Nodes
0        9
#define the curve
0.0   35.481
3.0   44.668
6.0   56.234
9.0   70.795
12.0 89.125
15.0 112.202
18.0 141.254
21.0 177.828
24.0 223.872
```

Listing 2: Definition of a frequency-defining curve for the third octave band example

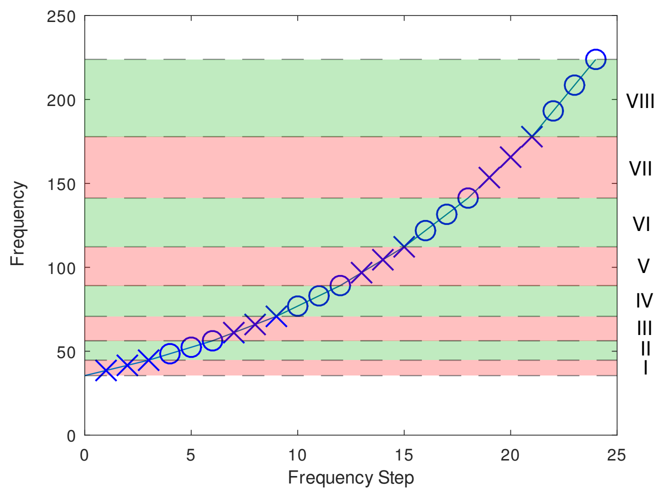

Figure 2: Frequency-defining curve for the third octave band described in Tab. 1.

Band:	I	II	III	IV	V	VI	VII	VIII
Low:	35.5	44.7	56.2	70.8	89.1	112	141	178
Center:	40	50	63	80	100	125	160	200
High:	44.7	56.2	70.8	89.1	112	141	178	224

Table 1: Lowest, center, and highest frequencies in Hz of the third-octave band used in the example.

3.2. Defining Geometry in NumCalc

To describe the geometry of the scattering objects, their surfaces need to be discretized using small triangles or plane quadrilaterals. To that end, users need to provide a list of the vertices of these elements, and a list which vertices make up which element. In this list, it is essential that the vertices of each element are ordered *counter-clockwise*. If users create the mesh using Blender, the order is correct if they ensure that the normal vector on each face points to the outside. If users are also interested in the sound field at (evaluation) points that do not lie on the surface of the objects, these points as well as an element list containing them need to be provided too. It may seem odd at first that evaluation points also need to have an element associated with them, but this has two reasons: First, the definition of the elements also contains the information about the type of the element, i.e., object element or evaluation element. Second, most of the time, users may want to display the results of their computations, and in this case, elements provide an easy way to interpolate and display results between nodal points.

For NumCalc this information needs to be provided using one or more text files containing the number and the coordinates of all the vertices/evaluation points, and one or more text files to describe the different elements, i.e. a list containing the number of its vertices and its type. Using multiple files to describe the geometry has the advantage that, e.g., BEM mesh and evaluation grid can be defined individually, and different combinations of object surface and evaluation grids can be used. If the export script provided by Mesh2HRTF is used in combination with one of the available evaluation grids, these text files are generated automatically.

The content of the files describing the nodes is as follows. The first line always needs to contain the number of nodes in the file, the following lines contain the node number and its x, y, and z coordinates. The files describing the elements have a similar format. The first line always needs to contain the number of elements in the file, the following lines are given by `element-number, node-number of each vertex, type, 0, groupnr`. It is possible to mix triangular and quadrilateral elements, NumCalc checks the length of a line to determine the form of the element. The `element-number` is later needed to assign a boundary condition to each element, the `type` of the element determines if the element is a BEM element on the surface of the scatterer (`type = 0`) or if it contains evaluation nodes outside or inside the scatterer (`type = 2`). Additionally, elements on object surfaces can be collected into different groups. However, as his grouping also affects the clustering of the FMM, users are advised to use only two groups. Group 0 for BEM elements, and Group 1 for evaluation elements. Again, if Mesh2HRTF is used to generate the input file, this is done automatically. In Lsts. 3 and 4, a few lines of an example node and element definition are given. The mesh contains 6602 vertices and evaluation points and 10464 elements. Element 1 is a triangle on the surface of the scatterer belonging to Group 0 , element 10464 is a quadrilateral evaluation element belonging to Group 1.

```
# number of nodes
6602
# node_number  x y z coordinate
1 -5.773503e-01 -5.773503e-01 -5.773503e-01
...
6602 3.000000e+00 0.000000e+00 1.200000e+00
```

Listing 3: First and last lines of an example node file.

```
# number of elements
10464
# elem_nr  vert1 vert2 vert3 (vert4) elemtyp 0 grpnumber
1 703 704 808 0 0 0
...
10464 6583 6584 6602 6601 2 0 1
```

Listing 4: First and last lines of an example element file.

3.3. Boundary conditions

Boundary conditions describe the acoustic behavior of the scattering objects. In NumCalc, three types of boundary conditions (`PRES,VELO,ADMI`) can be defined for each element. The general syntax for the boundary conditions is

```
ELEM e1 TO e2 bctype real(v0) curve1 imag(v0) curve2
```

This line means that all elements with element numbers in the interval [`e1`, `e2`] have the boundary condition `bctype` ∈ {`PRES,VELO, ADMI`} with value `v0`. Using `PRES` the sound pressure can be fixed for the element, if the pressure is set to zero, the element is sound soft. Using `VELO` the particle velocity can be fixed for the given element, if the velocity is set to zero, the element is sound hard. With `ADMI` an admittance can be assigned to the element to, e.g, model sound absorbing materials. In general, elements can only have one type of boundary condition, however, it is possible, that elements can have a `VELO` as well as an `ADMI` boundary condition. If no condition is assigned to an element, it is automatically assumed to be sound hard.

As sound-absorbing behavior can be frequency dependent, `ADMI` conditions can be combined with a piecewise linear curve (see Sec. 3.1) that maps each frequency step to a scaling factor for the admittance.. In case of the `ADMI` condition `curve1` and `curve2` can contain the number of the curves used to describe the frequency-dependence of the real and imaginary part of the admittance. The frequency steps themselves have already been defined for the frequencies (see Sec. 3.1). If no frequency-dependence is needed, the number for the curve is set to -1. The boundary condition section is closed using the keyword `RETURN`.

3.3.1 Example

In the following, we assume an example with different boundary conditions to illustrate the different options in `NC.inp`. Elements 0 to 10 are sound soft, element 11 has a velocity of 1.0 m/s, elements 50 to 52 have a constant admittance of $0.1 + 10^{-4}\mathrm{i}\ \mathrm{m}^3/(\mathrm{P}\cdot\mathrm{s})$ and element 65 has a frequency-dependent admittance that is defined by the frequency-defining curves 1 and 2. Per default, all other elements are sound hard.

```
BOUNDARY
ELEM 0 TO 10 PRES 0.0 -1 0.0 -1
ELEM 11 TO 11 VELO 1.0 -1 0.0 -1
#frequency independent admittance
ELEM 50 TO 52 ADMI 0.1 -1 1.0e-4 -1
#frequency-dependent admittance using curves 1 and 2
#the real part is scaled with 1.0, the imaginary part with 0.5
ELEM 65 TO 65 ADMI 100.0 1 10.0 2
# Definition of the boundary conditions is finished,
# denote this by the keyword 'RETURN'
RETURN
#
CURVES
# number_of_curves max_number_of_points_per_curve
2 4
# curve_number  number_of_nodes_defining_curve
1 4
0.0 0.0
1.0 1.0
3.0 4.0
11.0 8.0
# curve_number number_of_nodes_defining_curve
```

```
2 3
0.0 0.0
2.0 3.0
10.0 6.0
```

Listing 5: Definition of boundary conditions plus frequency-defining curves for the admittance.

For the admittance two curves are defined for real and imaginary part, respectively. Curve 1 consists of 3 linear segments with nodes (S_j, V_j) at (0.0, 0.0), (1.0,1.0), (3.0, 4.0) and (11.0,8.0). Curve 2 for the imaginary part consists of 2 linear segments with nodes (S_j, V_j) at (0.0,0.0), (2.0,3.0), and (10.0,6.0). Thus, different curves do not necessarily need to have the same number of nodes. The parameters for `Number_of_freqsteps`, `stepsize` and `Index0` are have already been defined in the frequency section Lst. 1. To provide an example we choose two arbitrary frequency steps ($n = 1$ and $n = 6$). Similar to Eq. (1) the real part of the admittance α_n is given by

$$\mathrm{Re}(\alpha_n) = \mathrm{Re}(v_0)\left(V_j + \frac{V_{j+1} - V_j}{S_{j+1} - S_j}(i_0 + n \cdot h_i - S_j)\right), \quad nh_i \in [S_j, S_{j+1}], \tag{2}$$

the imaginary part is defined in the same manner. The definition of the curves in Lst. 5 implies that at $f_1 = 1000\,\mathrm{Hz}$ (frequency-step $n = 1$) element 64 has an admittance of $\alpha_1 = 100.0 \cdot 1.0 + 10 \cdot \frac{3}{2}\mathrm{i} = 100 + 15\mathrm{i}$, at 6000 Hz ($n = 6$), for example, the element has an admittance of $\alpha_6 = 100.0\left(4 + \frac{8-4}{11-3}(6-3)\right) + 10.0\left(3 + \frac{6-3}{10-2}(6-2)\right)\mathrm{i} = 550 + 45\mathrm{i}$.

3.4. Sound Sources

External sound sources can be provided by a combination of point sources and plane waves. A point source $p_{\text{point}}(\mathbf{x}) := P_0 \frac{e^{ik||\mathbf{x}-\mathbf{x}_0||}}{4\pi||\mathbf{x}-\mathbf{x}_0||}$ at a source point $\mathbf{x}_0$ is defined by its number, the coordinates of its origin `(X0,Y0,Z0)` and its source strength `P0`:

```
POINT
Nr X0 Y0 Z0 Real(P0) -1 Imag(P0) -1
```

A plane wave $p_{\text{planewave}}(\mathbf{x}) := P_0 e^{ik\mathbf{d}\cdot\mathbf{x}}, ||\mathbf{d}|| = 1$ with direction `(DX,DY,DZ)` and strength `P0` is given by

```
PLANE
Nr DX DY DZ Real(P0) -1 Imag(P0) -1
```

Additionally, it is also possible to define arbitrary elements of the scatterer as a “vibrating” source by setting the appropriate `VELO` boundary condition; in this case the source strength is defined by `VELO = v0`.

4 Benchmark example

In our opinion, it is important that users of NumCalc are aware of the accuracy they can expect from the numerical solutions with NumCalc. In the following we look at the benchmark example of a scattered plane wave from a sound-hard sphere with radius $r = 1\,\mathrm{m}$. For this example, the acoustic field on the sphere and outside the sphere is known cf. [15, Chapter 6.10.3] and can be compared with the numerical BEM solution. In Fig. 3, the sound field for two frequencies ($f_1 = 200\,\mathrm{Hz}$ and $f_2 = 1000\,\mathrm{Hz}$) with speed of sound $c = 340\,\mathrm{m/s}$ is shown. The sphere was discretized using 20480 triangular elements, the average edge length of the elements was about 0.037 m. The triangularization itself is based on an icosphere, thus the triangles are almost equilateral.

On a computer with an Intel Core i5-4430 processor and 8 GB RAM, the calculation for $f_1 = 200\,\mathrm{Hz}$ took about 40 s, where 15 s were used for calculating the field on 24874 points outside the sphere, the iterative solver was finished after 26 iterations with an relative error of $2.3 \cdot 10^{-10}$. At $f_2 = 1000$ Hz the iterative solver was stopped after 17 iterations (rel error $5.8 \cdot 10^{-10}$), the total amount of calculation time was 68 s, where 41 s were used for calculating the sound field outside the sphere. In Fig. 4 the difference of the dB value of the sound pressure at

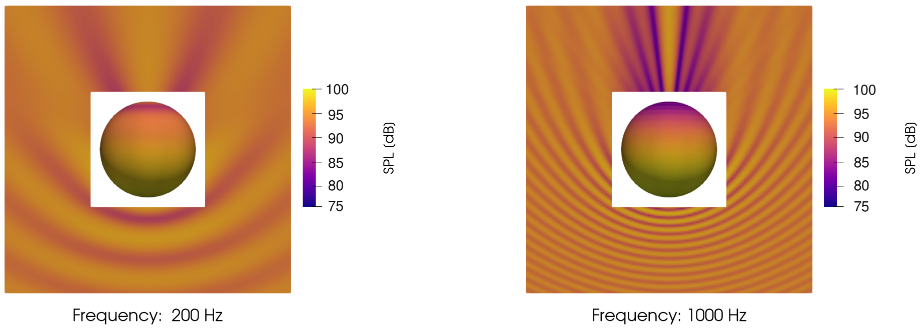

Figure 3: Acoustic field around a sound hard sphere at 200 Hz and 1000 Hz.

the sphere is depicted for two different types of discretizations at a frequency of $f_2 = 1000$ Hz. On the left and the middle panel the sphere was discretized using 5292 triangles that were constructed using a projection of a cube to the sphere, on the right panel the sphere was discretized with 5120 elements, where the triangularization was based on an icosphere. It is clearly visible that the regularity of the triangles has a lot of impact on the difference between BEM and analytic solution. In the case of the very regular icosphere triangularization a maximum difference of about 0.5dB could be observed on a few elements, the mean error over all elements of this discretization was about 0.114 dB. For the other triangularization the maximum difference was even in the range of 1.5 dB, the mean error was about 0.183 dB, and in contrast to the icosphere triangularization there was a clear distinction between "sunny" and "shadow" side, i.e. the side oriented towards the plane wave and the side oriented away from the plane wave.

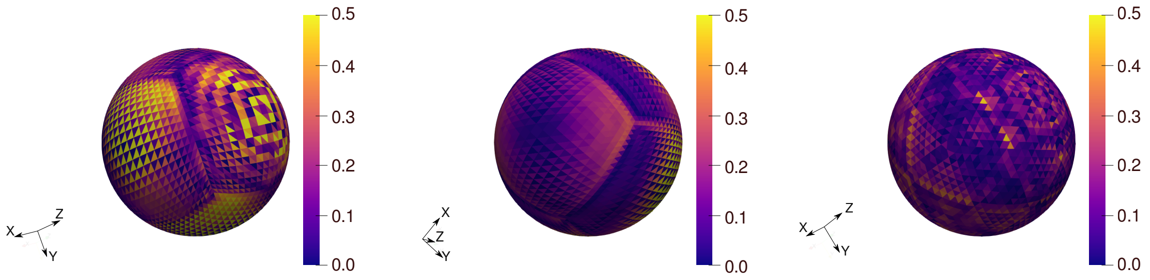

Figure 4: Difference between the dB sound pressures for the BEM calculation and the analytical solution at $f_2 = 1000$ Hz. Left: "Sunny"side, discretization based on a projected cube. Middle: "Shadow" side, discretization based on a projected cube. Right: Discretization based on an icosphere, "sunny" side.

Dependence on the number of elements

For most problems in practice, 10 elements per wavelength provide users with a solution that is sufficiently accurate. To illustrate the accuracy to be expected when using NumCalc, Fig.5 shows the mean relative error $\frac{|p-p_a|}{|p_a|}$ as a function of the number of triangular elements N. As a triangularization based on an icosphere is very restrictive in terms of number of elements, the discretization in this subsection is based on projecting a cube onto the sphere, which has the drawback, that the triangles are not as regular as for an icosphere. p is the sound pressure calculated with NumCalc, p_a is the sound pressure calculated using an analytic expression, cf. [15, Chapter 6.10.3]. The error is given as the mean error over all collocation nodes on the sphere (continuous line) and as the mean sound pressure over evaluation points around the sphere (dashed line). These points are the same 24874 points that were used for depicting the field outside the sphere in Fig. 3. The dotted lines depict the function $g = \frac{1}{\sqrt{N}}$. From Fig. 5 it becomes clear that the simplicity of the collocation method with constant

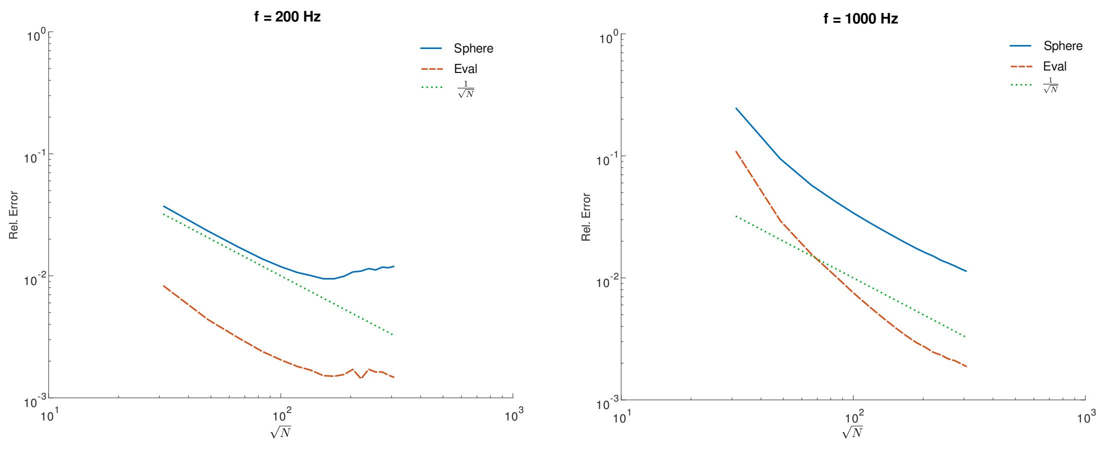

Figure 5: Mean relative error $\frac{|p-p_a|}{|p_a|}$ between the calculated sound pressure p and the analytic solution p_a on the surface of the sphere (continuous line) and on the evaluation grid (dashed line) as a function of the square root of the number of elements N. On the left side the frequency was set to $f_1 = 200$ Hz, on the right side $f_2 = 1000$ Hz. The average edge length of the elements used in the BEM are proportional to $\sqrt{N}$. The dotted line depicts the function $g = \frac{1}{\sqrt{N}}$.

elements has to be paid with a low convergence order of the error as a function of the number of elements. Also, the rise in the mean error in Fig. 5 for the case $f_1 = 200$ Hz shows the known fact, that the accuracy of the BEM solution suffers at low frequency when the mesh is too fine.

The rise in the error is also caused by the fact, that in the default setting, some numerical parameters, e.g. the truncation parameter of the multipole expansion or the accuracy of the quadrature methods for integrals over elements, are set rather small. This low setting on the one hand means, that the accuracy suffers, but on the other hand, the calculations become faster. Using a higher order in the FMM and a higher quadrature order for the integrals over the elements can stop this rise in the relative error.

5 Conclusion

In this paper, the open source software project Mesh2HRTF and the BEM code NumCalc have been presented briefly. Mesh2HRTF and NumCalc are aimed at providing a simple and easy-to-use method to solve acoustic wave propagation problems in 3D using the boundary element method. Input parameters have been described briefly and an example was given to illustrate the accuracy of the code. For in-depth tutorials and examples we refer to the homepage of Mesh2HRTF [16], where also the development version of the upcoming 1.0 release can be found, or the Wiki page for Mesh2HRTF [8]. There, also a list with the most common errors used in connection with Mesh2HRTF and NumCalc can be found.

Acknowledgements

We want to express our deepest thanks to Z.S. Chen, who was the original programmer behind NumCalc.

This work was supported by the European Union (EU) within the project SONICOM (grant number: 101017743, RIA action of Horizon 2020).

References

[1] C. H. Kasess, H. Waubke, M. Conter, C. Kirisits, R. Wehr, and H. Ziegelwanger. The effect of railway platforms and platform canopies on sound propagation. *Appl. Acoust.*, 151:137–152, 2019.

[2] W. Kreuzer, P. Majdak, and Z. Chen. Fast multipole boundary element method to calculate head-related transfer functions for a wide frequency range. *J. Acoust .Soc. Am.*, 126:1280–1290, 2009.

[3] N. H. Fletcher and T. D. Rossing. *The physics of musical instruments*. Springer, New York, 1991.

[4] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL `http://www.blender.org`.

[5] P. Majdak, F. Zotter, F. Brinkmann, J. De Muynke, M. Mihocic, and M. Noisternig. Spatially Oriented Format for Acoustics 2.0: Introduction and Recent Advances. *J Audio Eng Soc*, accepted, 2022. URL `https://projects.ari.oeaw.ac.at/research/Publications/Articles/2022/Majdak%202022%20SOFA%203A.pdf`.

[6] Tutorials for Mesh2HRTF. `https://sourceforge.net/p/mesh2hrtf/wiki/Tutorials`, last visited 19.04.2022.

[7] ARI Mesh2HRTF Homepage. `https://www.oeaw.ac.at/isf/das-institut/software/mesh2hrtf`, last visited 19.04.2022.

[8] Wikipage for Mesh2HRTF. `https://sourceforge.net/p/mesh2hrtf/wiki/Home`, last visited 19.04.2022.

[9] Z.-S. Chen, H. Waubke, and W. Kreuzer. A formulation of the fast multipole boundary element method (FMBEM) for acoustic radiation and scattering from three-dimensional structures. *J. Comput. Acoust.*, 16 (2):303–320, 2008.

[10] S. Marburg. Six boundary elements per wavelength: Is that enough? *J. Comput. Acoust.*, 10:25–51, 2002.

[11] R. Coifman, V. Rokhlin, and S. Wandzura. The fast multipole method for the wave equation: a pedestrian prescription. *IEEE Antennas Propag. Mag.*, 35(3):7–12, 1993.

[12] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, second edition, 2003.

[13] A. J. Burton and G. F. Miller. The application of integral equation methods to the solution of exterior boundary-value problems. *Proc. R. Soc. London Ser. A*, 323:201–210, 1971.

[14] H. A. Schenck. Improved integral formulation for acoustic radiation problems. *J. Acoust. Soc. Am.*, 44 (1):41–58, 1968.

[15] E.G. Williams. *Fourier Acoustics: Sound Radiation and Nearfield Acoustical Holography*. Academic Press, London, 1999.

[16] Any2HRTF/Mesh2HRTF. `https://github.com/Any2HRTF/Mesh2HRTF`, last visited 21.04.2022.